

## NanoPower P-series Datasheet P31u / P31us V9.0

Electric Power Systems for mission critical space  
applications with limited resources

### NanoPower P-series

The power supply is the heart of a satellite. It is therefore very important not to compromise on quality and reliability of it. The NanoPower is designed as a reliable and flight proven system with digital interface and advanced features like maximum power point tracking and latchup-protection.

## Feature Overview

The P31 has 3 input channels with independent power-point setting giving an input power capacity of 30W.

The standard series are optimized for panels with up to 2 solar cells in series. The S-series is optimized for up to 7 solar cells in series.

Maximum power point tracking.

Battery under-voltage and over-voltage protection

Can operate after end of battery lifetime

Two regulated power buses: 3.3V@5A and 5V@4A(user selectable).

6 configurable and controlled output switches w. latching current limiter

Discrete control of output switches

Onboard housekeeping measurements

Interface to battery board NanoPower QuadBat BP-4 or NanoPower BP-X

Separation-switch interface with latching mechanism

Remove-Before-Flight-pin interface

Onboard lithium ion battery pack

*(S series has no batteries but requires an external battery pack)*

I<sup>2</sup>C interface

Operational temperature: -40 to +85 °C

Dimensions: 96mm x 90mm x (16 to 26) mm

PCB: glass/polyamide from ESA approved producer

IPC-A-610 Class 3 assembly

## Applications

1U, 2U, 3U CubeSat satellites

Nano Satellites

Solar-powered autonomous systems

## Compatibility

GomSpace products

Cubesat Space Protocol

CubeSat Kit products

Innovative Solutions in Space products

ClydeSpace products

## Flight Heritage

Flight heritage from multiple missions including GomX-1, Funcube, Triton-1.


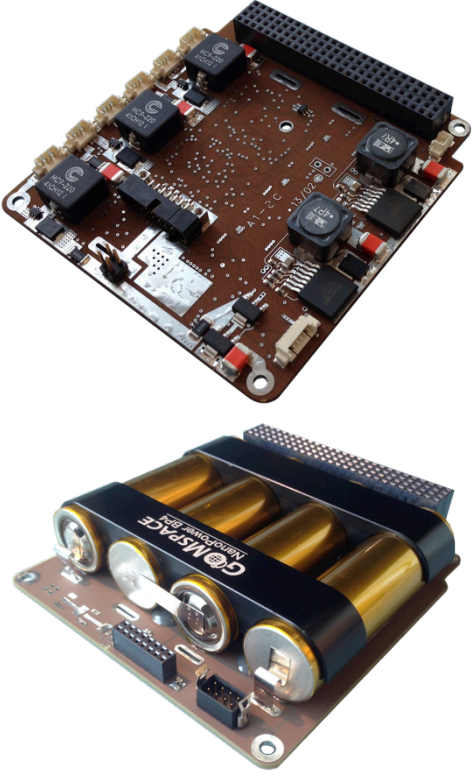
## Functional Description

### General Overview

The NanoPower P-series power supplies are designed for small, low-cost satellites with power demands from 1-30W. Employing a strictly KISS design philosophy, the NanoPower interfaces to triple junction photo-voltaic cells and uses a highly efficient boost-converter to condition their output power in order to charge the provided lithium-ion batteries. The incoming power along with the energy stored in the batteries is used to feed two buck-converters supplying a 3.3V@5A and a 5V@4A (configurable) output bus. Six individually controllable output switches with over-current shut-down and latch-up protection, each separately configurable to either 3.3 or 5.0V output.

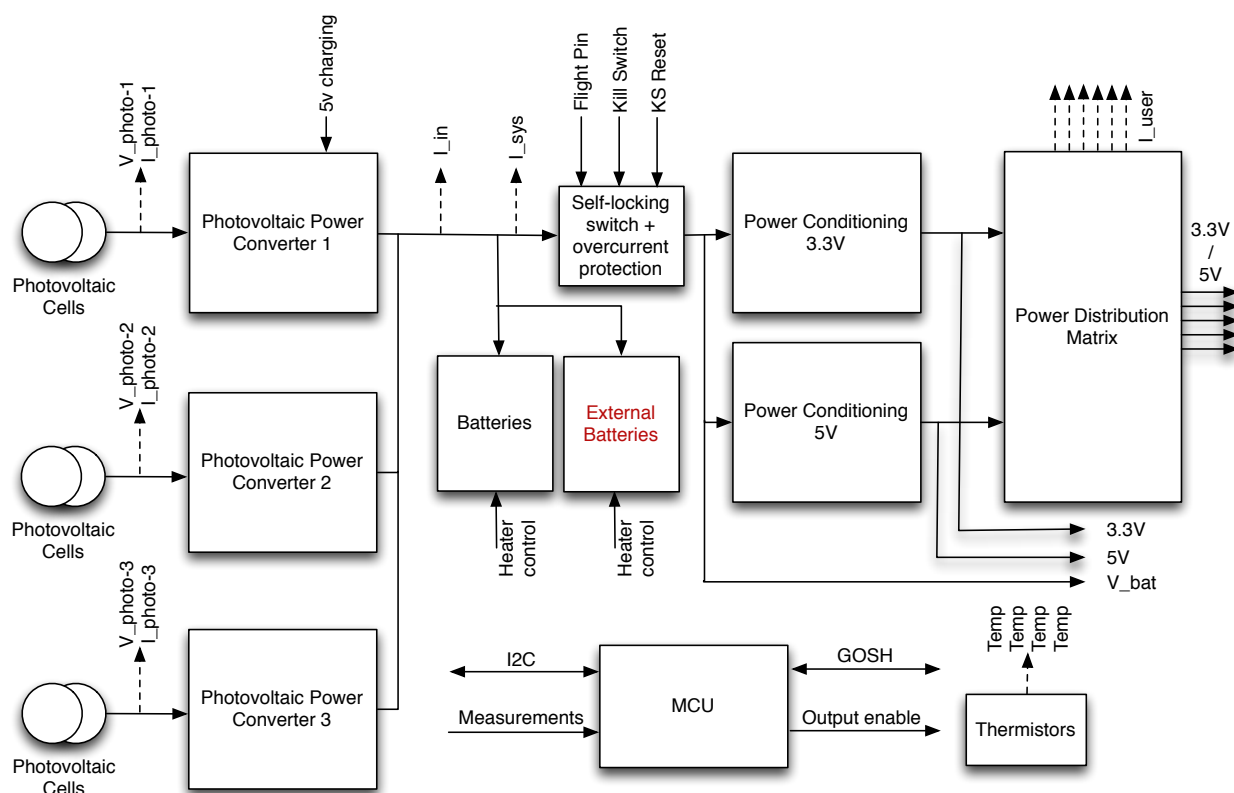
## Models

NanoPower is available in two different configurations, the normal version P31u and the high voltage version P31us.

<b>P31u</b>	<p>NanoPower with three photovoltaic input converters          (6 - 8.4 V battery)</p>	
<b>P31us</b>	<p>NanoPower optimized for up to 4 solar cell string panels and 4-battery string power pack.          (12 – 16.8 V battery)</p>	

The 'S' models are optimized for use with solar panels from 1 to 4 cells in string and requires an external battery pack for operation. The models without the 'S' suffix supports either 1 or 2 cells in string.

## Block Diagram



## Microcontroller

NanoPower features a microcontroller that provides maximum power-point tracking (MPPT) capability, measures and logs voltages, currents and temperatures of the system, enables user control etc. Using an I<sup>2</sup>C interface, it is possible to read out measurements, control the on/off-state of 3.3 V and 5 V busses, switch on/off the MPPT and to set/read various parameters.

## Multiple Photo-Voltaic Inputs

The P31u have three individual photo-voltaic input channels each having its own power-point setting. On satellites with up to three solar panels in the sunlight, this enables the voltage to be set independently on all panels thus capturing the exact maximum power-point at all illuminated cells when MPPT is employed. If used on a “box” satellite such as a CubeSat, simply connecting pairs of opposite mounted solar panels in parallel to each of the three inputs will allow individual conversion of the power from all cells in sunlight.

The standard NanoPower versions are optimized for 2 GaAs solar cells in a string for each panel, but on request it can also handle a single cell as a panel. If more cells in a string is required the ‘S’-series support up to 4 GaAs solar cells in a string.

The photo-voltaic input converter is designed to handle up to 2 A input current. The inputs are designed for two triple junction cells in series and a number of such in parallel. Each series string must have a protection diode in series in order to avoid non-illuminated cells from drawing current from illuminated ones.

## Power-Point Tracking

The NanoPower models features two different choices of setting the solar cell power points, selectable through the I<sup>2</sup>C interface (the default mode is configurable).

- Constant voltage power point on solar cells (level adjustable over I<sup>2</sup>C interface, default value configurable).
- Maximum power point tracking.

## Battery Protection

The lithium-ion battery is charged with all the available power from the photo-voltaic inputs which is not drained by the loads on the external power busses. To ensure safe operation and long battery life, the battery is protected against too low and too high voltage.

The low voltage protection is implemented in both hardware and software.

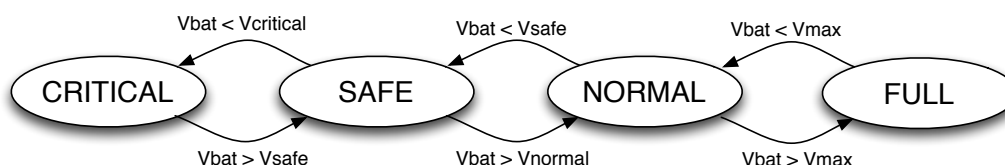
The hardware low voltage protection will switch off the kill-switch, thereby disabling all outputs from the Nanopower and most electronics on the Nanopower itself leaving only the ability to charge the battery from the solar cells. The hardware high voltage protection will set the input voltage on the solar cells to zero.

Hardware level set points	Voltage level (V)
Switch off	6,0 (*12.0)
Switch on	6,4 (*12.8)

- Only on Pxx-S

The software high voltage protection (battery FULL mode) implements a constant voltage charge scheme that will keep the battery at its maximum voltage while at the same time supping the users with the necessary power. The full mode regulation works by lowering the voltage on the solar panel inputs, thereby only taking in the power needed.

The software low voltage protection is a three state system with a CRITICAL, a SAFE and a NORMAL mode.



In normal mode everything is nominal but should the battery voltage drop below Vsafe, the Nanopower will change its output switch configuration to a safe mode configuration (user configurable). This allows the switch off of all non-essential systems and leave a simple low power beacon or similar running.

Should the battery voltage continue to drop below Vcritical, the Nanopower will switch off all user outputs. Note that “always-on” outputs will only be affected by the hardware protection.

Mode	Software level set points	Voltage level (V)
FULL (4)	Vmax	8.3 (*16.6)
NORMAL (3)	Vnormal	7.4 (*14.8)
SAFE (2)	Vsafe	7.2 (*14.4)
CRITICAL (1)	Vcritical	6.5 (*13.2)

\* Only on Pxx-S

The set points and safe mode configurations are user configurable through configuration 2.

### Cycle life

The cycle life of any rechargeable battery depends on a number of factors, but most importantly the Depth-Of-Discharge (DOD) of the cycles, temperature, and end of charge voltage (EOCV) (see battery data sheet for more information, *gs-ds-batteries.pdf*). The NanoPower P31u allows a user configurable EOCV to extend the cycle life of the battery. By default the NanoPower will not charge the batteries fully to 4.2 V per cell, but only to 4.15 V. This can be configured in the range 3.95 V to 4.2 V.

### Battery Heaters

Both the NanoPower P31u and the BP-4 QuatBat comes with an option of battery heaters which can be controlled from the NanoPower. The heaters can either be controlled directly from the output side by switch on or off the heater through commands, or by an autonomous heater controller with the heater on and off temperatures settable through the configuration system. The heaters onboard the NanoPower and on Quadbat can be enabled/disabled through 'board' commands 'board heateronboard' and 'board heaterquadbat'.

A single heater element (of either 20 or 40 Ohm) is used pr. 2 batteries

### Separation-Switch and Flight-Pin Operation

Most launch-providers will require that the satellite is unpowered during launch. NanoPower has a built-in switch (kill-switch / separation-switch), which disconnects the batteries from the buck-converters and thereby turning off the satellite. In order to close this switch, and activate the satellite, the NanoPower has three separation switch inputs. If one of these pins is connected to ground, the satellite will start up, and remain turned on. The advantage of the NanoPower separation-switch input is that no current ( $\ll 1$  mA) is sent through the small separation switches, which increases overall reliability.

A Remove-Before-Flight-pin input is also provided which cuts off the function of the separation-switch, which is used to ensure the satellite is turned off even when the separation-switch is released (active). This is convenient for handling purposes.

Furthermore an optional battery ground-break connector is available (P14). This connection disconnects the battery negative terminal from the NanoPower ground. Connect a normally closed separation-switch to this connector. Note that all current through the batteries will also run through this connector and thereby through the switch.

**KS** (Connector P11, P10, P8): The kill-switch input, short this input to ground and the NanoPower will switch ON

**KS\_RESET** (Connector P8): The kill-switch reset, short this input to ground and the NanoPower will switch OFF



**RBF** (Connector P8): The Remove-Before-Flight, if this input is shorted to ground, the NanoPower CANNOT be switched ON.

**GND\_BRK** (Connector P14): Break ground connection on batteries. Short this connector to connect batteries.

NanoPower can be delivered with different configurations of the separation-switch: Self-locking (recommended) or without self-locking.

### Self-locking

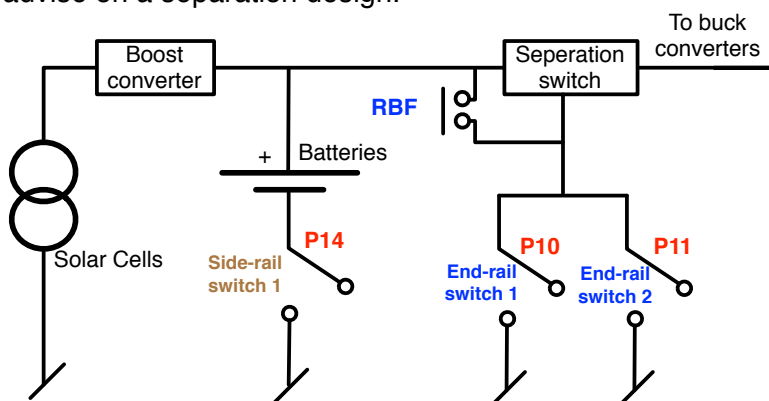
The self-locking switch locks into the ON state as soon as the separation-switch is released and ensures the system remains ON even if a failure in the separation-switch due to the space environment causes the separation-switch to disconnect. In order to switch the system OFF when the separation-switch is depressed, the separation-switch reset input is shorted for a moment, which unlocks the self-locking switch.

### Non Self-locking

When separation switch is closed, NanoPower is on. When separation switch is open, NanoPower is off.

### Separation switch example

Below is shown an example of how the separation switches can be used. The number and arrangement of separation switches required depends on the launch provider. Please enquire your launch provider for advise on a separation design.



### Battery overcurrent protection

The battery of the NanoPower P31u is protected against short circuit. The circuitry monitors the current  $I_{sys}$  (see block diagram) and if it surpasses the threshold the kill-switch is switched off for 100ms and then on again. This will switch off power to all systems, both internal and external for 100ms.

### Hard reset

It is possible to command the Nanopower to perform a hard-reset. This will switch off the kill-switch for 100ms and then on again. This will switch off power to all systems, both internal and external for 100ms.

## External Watch Dog Timers

The NanoPower provides four optional watch dog timer functions for external users (all user configurable through serial shell).

### I<sup>2</sup>C WDT

Dedicated WDT

Active CSP ping WDT 1 and 2

To setup the watch dog timers, it is necessary to use the GOSH interface (command 'board').

### I<sup>2</sup>C WDT

If NanoPower has received no meaningful I<sup>2</sup>C communication for a period of time (length of period configurable by customer), the I<sup>2</sup>C WDT can be configured to switch off all outputs and do a reset, which will return the system to its original state. This can be used to solve a potential software fault in an onboard computer or similar.

Any valid I<sup>2</sup>C communication to the EPS will kick (reset) this WDT.

The I<sup>2</sup>C wdt timeout action can be configured with 'board i2cwdt\_rst 0' to only cycle the 6 switchable outputs.

The I<sup>2</sup>C wdt timeout action can be configured with 'board i2cwdt\_rst 1' to do a hard-reset and thereby reset the NanoPower itself as well as all outputs including the permanent outputs.

The I<sup>2</sup>C watchdog does not run when NanoPower is in critical power mode.

### Dedicated WDT

The dedicated WDT is reset by a dedicated command to the NanoPower. This can for example be used as a ground communication watch dog, i.e. this command is issued to NanoPower on each connection with the ground station. If no communication has been received for a long period of time (configurable by customer), NanoPower will switch off all outputs and do a reset.



The dedicated WDT is **ONLY** intended for long timeouts and low interval kicks. Every time the WDT is kicked, an EEPROM value is written and this value can only be written roughly 2-4 million times.

Recommended settings could be a timeout of 48 hours and a kick from the ground station on all passes.

The WDT counts down in seconds, but every integer hour it stores its hour value in persistent storage. On reboot it starts from the hour value last stored. On reboot the hour count is also decremented by one (to penalize reboot-loops). This means that if less than one hour remains on the WDT and a reboot occurs, a timeout will occur immediately after the restart.

The value of the WDT cannot be set lower than 1 hour.



*Note that if the dedicated WDT times out, the default config is restored on NanoPower.*

### Active CSP ping WDT 1 and 2

If any CSP address is configured for CSP WDT1 or 2, the EPS will actively monitor the connectivity to each of these systems with a ping packet every 60 seconds. If 5 consecutive pings are lost



(maximum allowed response time is 30 ms), the corresponding power channel is power cycled with a 5 second off time.

Use GOSH to configure the CSP WDT, for example to setup CSP\_WDT1 up to monitor CSP address 3 on a subsystem connection to power output 5 do 'board wdtdsp 1 3 7'.

## Housekeeping

The NanoPower provides a number of measurement points that enables monitoring of the condition of the system. These measurements are available as buffered voltages to be sampled by an external system or as digital readings retrievable through the I<sup>2</sup>C interface. Measurements include:

- Four temperatures
- Current into and out of photovoltaic power converters
- Photovoltaic input voltage for each input converter
- Battery voltage
- Total current into the output bus converters.
- Current out of all power output channels
- Number of latch-up event detected for each power output channel

Parameter	Range (non-S)	Resolution (non-S)	Range (S)	Resolution (S)
Temperature	-40 to +125 °C	1 °C	-40 to +125 °C	1 °C
I_photo	0 to 3 A	3 mA	0 to 3 A	3 mA
I_in	0 to 6 A	6 mA	0 to 6 A	6 mA
I_sys	0 to 12 A	12 mA	0 to 12 A	12 mA
I_switch	0 to 2.4 A	3 mA	0 to 2.4 A	3 mA
V_photo	0 to 9.5 V	10 mV	0 to 19 V	20 mV
V_bat	0 to 9.5 V	10 mV	0 to 19 V	20 mV

## Bootstrapping

The NanoPower system is designed to be operational even once the battery has reached end-of-life and have zero charge capacity. The photo-voltaic input converter is powered by the battery, but even if the battery cannot supply the current to start the boost-converter, the unconditioned current from the PV inputs alone will supply the boost regulator and start up the conversion.

## Charging Interface

All NanoPower products are equipped with an external charging interface that allows the user to charge the batteries using a single 5Vdc power supply. The interface will draw 0.8-0.9 A until the batteries are fully charged. When fully charged the reaction depends on whether the EPS is ON or OFF:

- If charging in OFF mode, the NanoPower will stop drawing power when fully charged.
- If charging in ON mode, the NanoPower will start lowering the voltage on the input when fully charged. This will correspond to a short as seen from the charging-powersupply.

Use a **5V@1A** power supply to charge. It is possible to change the charge current by changing the charge voltage, if the voltage is lowered to 4.8 V the charge current is lowered to around 0.7-0.8 A.



**Warning:** It is advised never to exceed **1 A** on the charging interface.



**Warning:** Always use a current limited charging power supply and never ever set the current limit higher than **1 A**

The charging interface works both when NanoPower is OFF and when it is ON. However, the charging uses the photo-voltaic Input converter 3 and therefore, if the NanoPower is ON and the maximum power point tracker is running, it will affect the charging. Therefore, it is recommended to charge either with NanoPower OFF, or with Nanopower ON but it fixed power mode (3.7V power point).

## Power Outputs

NanoPower has the following power outputs:

- Six latch-up protected and user controllable output
- Two “always on” 3 and two “always on” 5 V bus outputs
- Always on direct battery voltage output

The latch-up protection is implemented such that for short time (<28 ms) over currents the respective output is power cycled. If the problem persists after the power cycle the NanoPower will attempt to power cycle the respective output with longer intervals (approx. 500 ms).



**Warning:** Use the “always on” outputs with care. Shortening these will shorten the respective power bus and rely on the power conditioning converter current limiter.

All power outputs are user configurable as default ON or OFF with a configurable switch on time.

Parameter	Condition	Min	Typ	Max	Unit
<b>Battery</b>	Battery connection				
- Voltage		6.0 (*12.0)	7.40 (*14.9)	8.40 (*16.80)	V V
- Current, charge	(Depends on battery configuration)			6.00	A
- Current, discharge	Overcurrent protection threshold ***		6.8***		A
<b>PV inputs</b>	Photo-voltaic inputs (Customer selectable)				
- Voltage		0 (*0)	4.2 (*8.4)	8.5 (*17)	V V
- Current, charge		0.00		2.00	A
<b>5V_in</b>	Battery charge input (beware of inrush) 5 V => 0.9 A charge, 4 V => 0 A charge @5 V				
- Voltage		4.10	5.00	5.00	V
- Current, cont.			0.9	1.1	A
<b>OUT-1,2,3,4,5,6</b>	Latch-up protected outputs Configurable Current cut-off limit (Cust. select)				
- Voltage			3.3/4.98		V
- Current limit		0.5	Select	3.0	A
<b>+5 V</b>	5 V regulated output (always on)				
- Voltage		4.89	4.98	5.05	V
- Current, cont. **	Total current including output channels	0.005		4.00	A
<b>+3.3 V</b>	3.3 V regulated output (always on) From -40 to +85 °C				
- Voltage		3.29	3.34	3.39	V
- Current, cont.	Total current including output channels	0		5.00	A
<b>V_BAT</b>	Raw battery voltage (Depends on battery configuration)				
- Voltage		6.0 (*12.15)		8.40 (*16.80)	V
- Current out			12		A
<b>Power consumption</b>	Power consumed by NanoPower		115 (*210)		mW
<b>Off current</b>	Current consumed with separation switch OFF		35	60	uA
<b>Shelf-life</b>	Period until batteries are fully discharged when separation switch is OFF. (Depends on battery configuration)	700	1400		Days

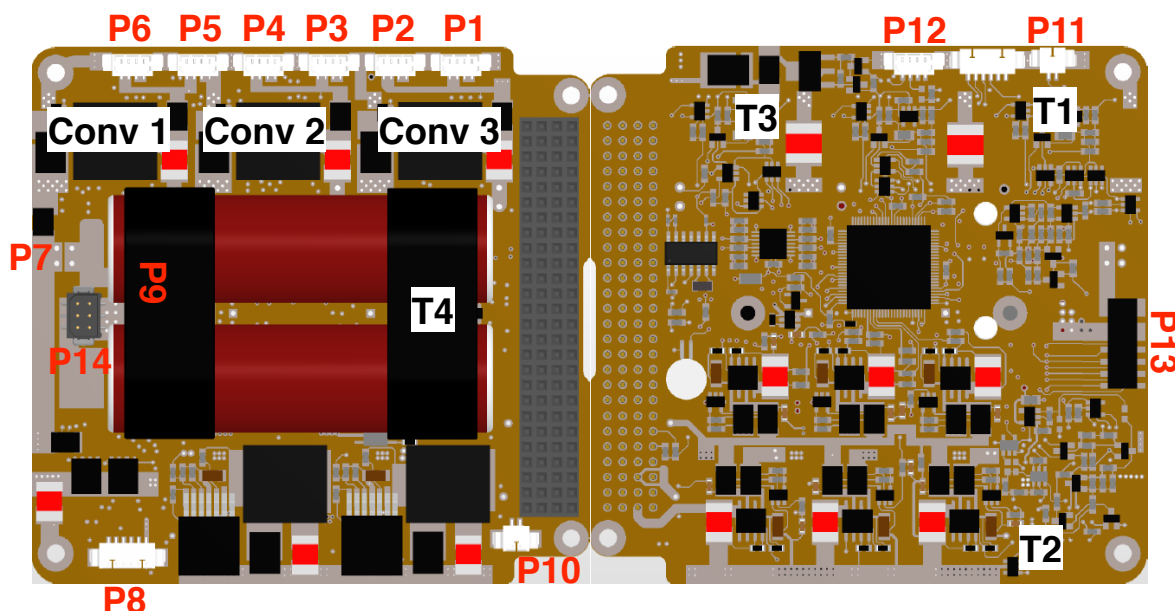
\* Only on Pxx-S

\*\* A completely unloaded 5 V channel may show oscillations.

\*\*\* For higher threshold, please enquire at info@GomSpace.com.

## Connections

The NanoPower P-series power supplies are equipped with the 104 pin CubeSat Kit connector. The drawing below shows different connection on the top and bottom side of the circuit board. T1-T4 are the four temperature sensors.



P1-P6: (Picoblade 4 pin) Solar-panel input connections

P7: (2x2 2.54mm male-header) Battery ARM connector: This connects the batteries to the NanoPower circuitry.

P8: (Picoblade 6 pin) Flight preparation panel connector

P9: (Harwin M80-5421442) Battery board extension connector (for BP-X, only when no batteries are mounted)

P10 and P11: (Picoblade 2 pin) Kill switch connectors

P12: (Picoblade 4 pin) Serial connector for GOSH interface.

P13: Battery board extension connector (for BP-4 QuadBat)

P14: (Harwin M80) Optional battery ground break connector

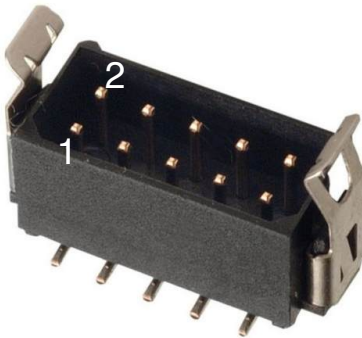
**Note:** The BPX battery board extension connector (P9) cannot co-exist with onboard batteries.

P8 Flight Preparation Panel connector	
Pin	Usage
1	GND
2	RBF <sup>1</sup>
3	KS <sup>2</sup>
4	KS_RESET <sup>3</sup>
5	Vbat (through 100kOhm) <sup>4</sup>
6	CHARGE

P1 to P6 Solar panel input connectors	
Pin	Usage
1	GND
2	GND
3	Vsc
4	Vsc

P10 and P11 Kill Switch Connectors	
Pin	Usage
1	KS- (GND) <sup>4</sup>
2	KS+ <sup>4</sup>

P12 Serial connector	
Pin	Usage
1	GND
2	Not connected
3	RxD
4	TxD

P14 (Harwin M80-8280642) Battery ground break connector		
Pin	Usage	Pinout
1,3,5 2,4,6	GND Battery minus (BAT GND)	

<sup>1</sup> RBF pin on P8 must be shorted to ground to engage

<sup>2</sup> KS on P8 must be shorted to ground to override kill switch (switch on NanoPower).

<sup>3</sup> KS\_RESET on P8 must be shorted to ground to engage reset.

<sup>4</sup> KS on P10 and P11 is a two pin connector that must be shorted to switch on NanoPower, alternatively KS+ can be shorted to any ground common with NanoPower.

### Solar panel input converters

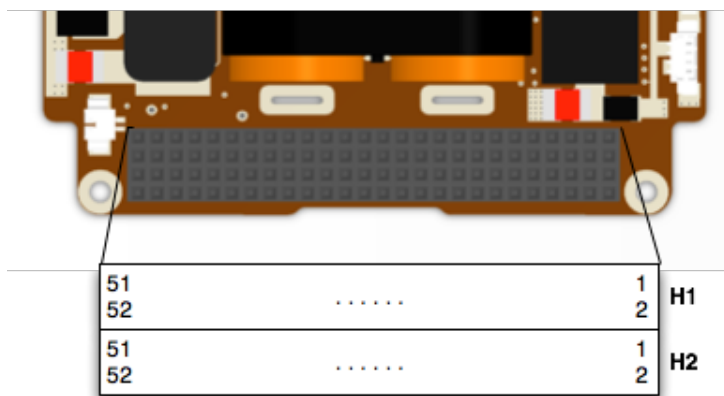
Converter 1: P6 and P5

Converter 2: P4 and P3

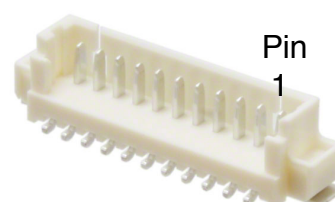
Converter 3: P2 and P1

## Stack Connector

Pin#	Mnemonic	Dir	Description
H1-32	5 V_in	I	5 V battery charge input (Same as P8 pin 6)
H1-41	I <sup>2</sup> C-SDA	I/O	I <sup>2</sup> C serial data
H1-43	I <sup>2</sup> C-SCL	I/O	I <sup>2</sup> C serial clock
H1-47	OUT-1	O	latch-up protected output
H1-49	OUT-2	O	latch-up protected output
H1-51	OUT-3	O	latch-up protected output
H1-48	OUT-4	O	latch-up protected output
H1-50	OUT-5	O	latch-up protected output
H1-52	OUT-6	O	latch-up protected output
H2-25	+5 V	O	Permanent 5 V output
H2-26	+5 V	O	Permanent 5 V output
H2-27	+3.3 V	O	Permanent 3.3 V output
H2-28	+3.3 V	O	Permanent 3.3 V output
H2-29	GND	O	Power ground
H2-30	GND	O	Power ground
H2-31	AGND	O	Analogue ground
H2-32	GND	O	Power ground
H2-36	EPS RX	I	Serial port Rx (optional)
H2-38	EPX TX	O	Serial port Tx (optional)
H2-45	V_BAT	O	Battery voltage
H2-46	V_BAT	O	Battery voltage



Stack connector pinout



Picoblade pinout



## Batteries

NanoPower P31u products include a pack of two cylindrical lithium-ion cells in a series configuration with a nominal voltage of 7.4 V (8.4 V maximum charge). The two cells are re-packed with Kapton insulation and (optionally) fitted to the board with Scotch-Weld 2216 epoxy. In addition, aluminum brackets are glued to the batteries and screwed to the PCB for added mechanical and thermal stability.

*For information on battery specifications, please see the GomSpace battery datasheet (gs-ds-battery).*

### Configurations

The NanoPower can be used with with two onboard batteries for a total capacity of 2600 mAh @ nom. 7.4 V.

**BP-4 QuadBat:** For increased battery capacity GomSpace offer the external NanoPower Quad-Battery board that is fully compatible with the P31u power supply products. Please look at [www.GomSpace.com](http://www.GomSpace.com) for data sheets and pricing. This allows a capacity of 5200 mAh @ nom. 7.4 V, or in special cases with both external and internal batteries for a total capacity of up to 7800 mAh @ nom. 7.4 V.

**BP-X:** For unlimited battery capacity GomSpace offer the external NanoPower BP-X battery pack that is fully compatible with the P31u power supply products. Please look at [www.GomSpace.com](http://www.GomSpace.com) for data sheets and pricing.

**P31u-S:** The P31u is configured to operated with four 3.7 V batteries in series and has no onboard batteries - it needs to be connected to either a QuadBat BP-4-S battery pack 2600 mAh @ nom. 14.8 or a BP-X.

### Connecting the batteries

For onboard batteries, the NanoPower comes with the batteries soldered to the board. To connect the batteries to the rest of the circuit, connect jumpers to the battery ARM connector (P7). Battery voltage is on the two pins towards the batteries and NanoPower circuit is on to two pins towards the edge. The two jumper connections are redundant in the way that it is enough to short one to connect the batteries to the circuit.

*For flight configuration in is recommended to solder the header connection securely together.*



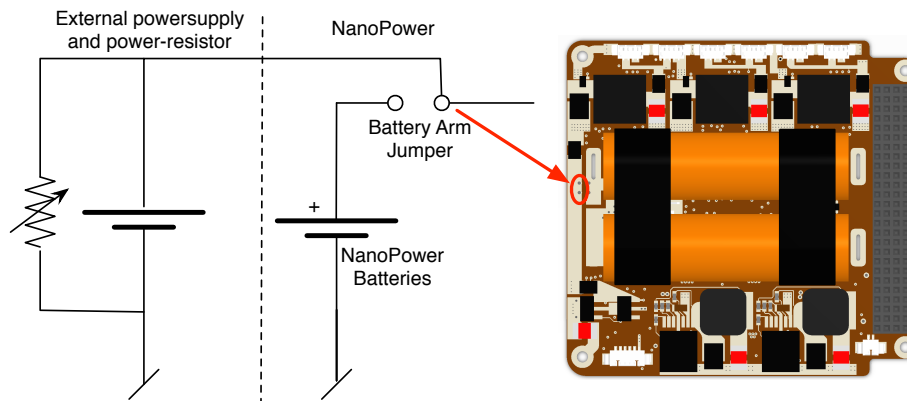
**Warning:** When using QuadBat together with a NanoPower unit with onboard batteries it is important to ensure that batteries on NanoPower and QuadBat have exactly the same charge before connecting them together. This can be done by connecting them with a sliding resistor and an ampere-meter to let them equal out their charge state at a controlled pace. Start with a high resistance and gradually lower it as the two battery sets equalize their charge (voltage difference must be smaller than <100 mV before direct connection). Do not let the current go above 1A. As standard batteries are shipped from GomSpace with same charge, but it is important to verify the voltage of both battery packs before connecting them.

Parameter	Condition	Min	Typ	Max	Unit
<b>Space height</b>	Spacer size between QuadBat and NanoPower	6.3	6.5	6.7	mm

## Operating without batteries

During testing it can sometimes be beneficial to operate without batteries connected. Instead a bench powersupply and a power resistor can be used to simulate the batteries. The power supply must be set to a voltage that corresponds to the battery voltage range of NanoPower, e.g. 7.4V. The power resistor is used to sink current coming in from battery charging and must be sized accordingly both in terms of resistance value and in terms of power rating.

Example: With a voltage of 7.4 V and a 5 Ohm resistor the simulated battery can sink  $7.4 \text{ V} / 5 \text{ Ohm} = 1.48 \text{ A}$  of current and the resistor must be able to dissipate  $7.4 \text{ V} * 1.48 \text{ A} = 10.9 \text{ W}$ .



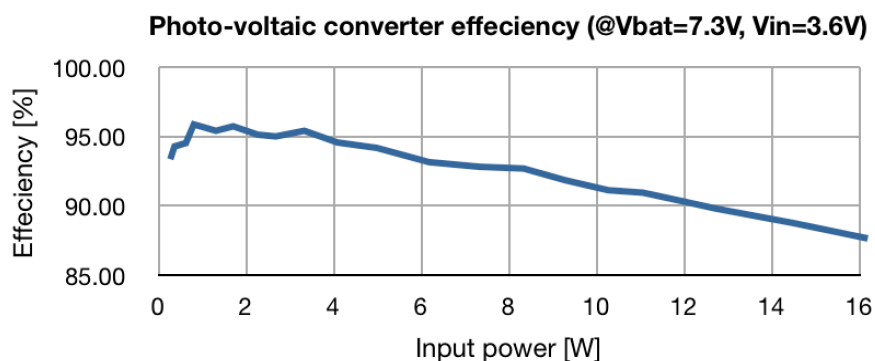
**Warning:** NanoPower power supplies are capable of operation with batteries that have lost all capacity or even completely without batteries. However, operating without batteries should only be considered as a failure backup mode. If you consider using the NanoPower entirely without batteries, please consult GomSpace first.

## Performance

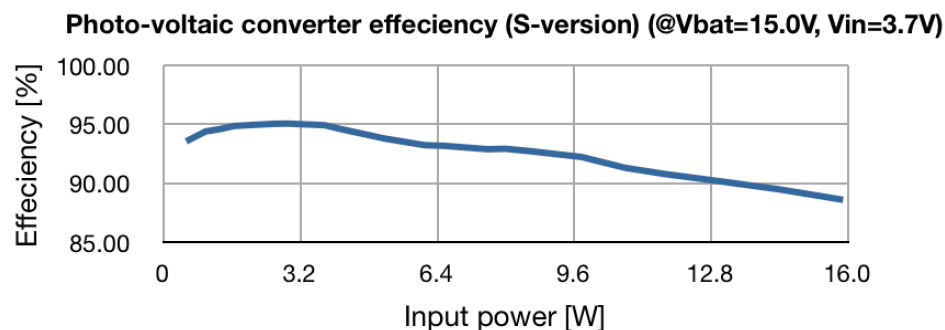
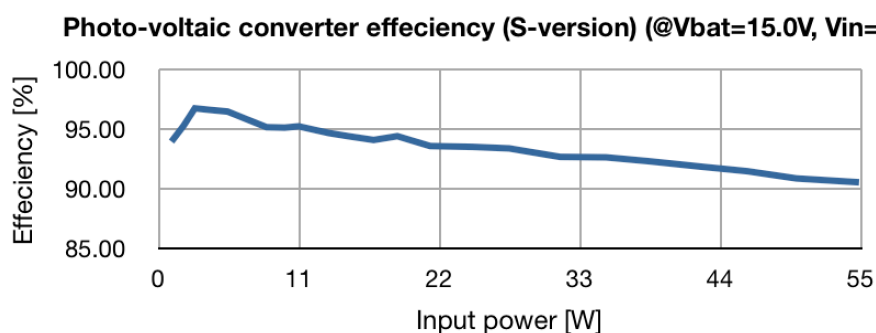
The NanoPower P-series is designed with efficiency and reliability in mind. One of the most critical current paths in a satellite system is the Photo-Voltaic (PV) input that supplies energy for the batteries. On the NanoPower P-series, the PV input path consists of a boost-converter that converts the approximately 4.2 V (8.4 V for Pxx-S) from two series connected triple-junction cells up to 8.4 V (16.8 V for Pxx-S) on the batteries. The component-count in this path is kept to a minimum and are all power- and current-derated by 70%. Also, the component-count in the controller circuitry for the photovoltaic and power conditioning converters is kept extremely low and employs only analogue components.

### Converters efficiency

Efficiency of a single normal version photovoltaic converter is shown in the graph below for up to 16 W.

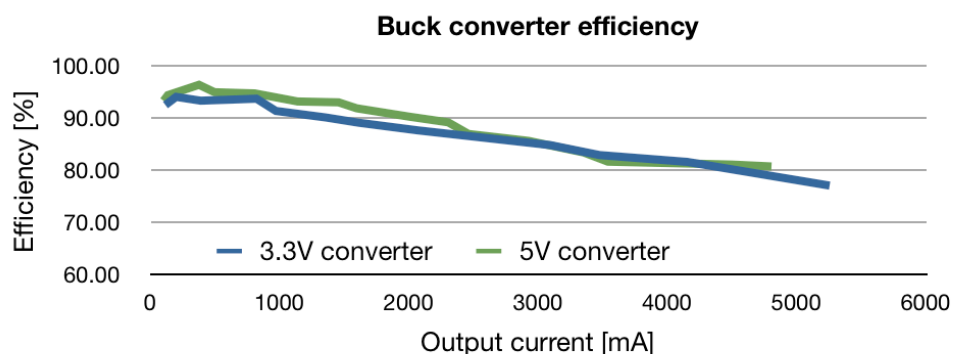


Efficiency of a single S-version photovoltaic input converter is show below for input powers up to 55 W with an input voltage of 10.5 V and up to 16 W for an input voltage of 3.7 V.



**NOTE:** Efficiency of input converters depends very much on the actual current through the converter. Therefore, in general if input voltage is higher, then efficiency is higher at the same power input.

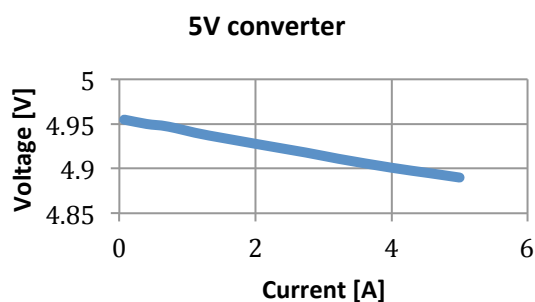
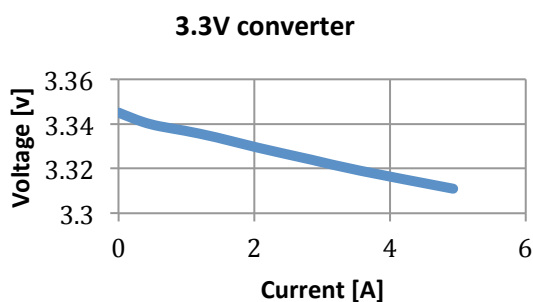
Efficiency of the power-conditioning converter, measured from battery to user outputs, can be seen on the figure below:



### Line loss / Voltage drop

From output converters to the users there is a certain resistance that will result in loss. The output switch, the PCB tracks and the stack connector all have resistance. A total resistance from power converters to users is typically <50 mOhm.

A measurement of the voltage drop on the 3.3 V and 5 V bus is show below. The current is drawn from a single pin in the stack connector (H2-26 and H2-27 respectively).



## Command and Data Interface

The NanoPower EPS is an independent network node designed for the CSP protocol and I2C bus communication. The CSP protocol is implemented in all GomSpace products and provides a highly capable and integrated networking architecture that can be utilized across multiple physical link implementations to cover both the space and ground segment.

### How to send a NanoPower command

If you have a NanoMind OBC in your satellite, a driver for the NanoPower system is already included. If you do not, there is still the option of opening the serial port of the EPS and executing commands directly. Finally there is also the option of writing a custom driver, based on the CSP interface specification.

### Using GOSH (GomSpace Shell)

The GomSpace shell is included on all GomSpace products, even on the EPS itself. The shell has a complete interface with all the commands needed in order to configure and operate the power supply. See the next chapter for a list of commands.

### Using Lib-IO (C-interface)

All the functions and commands in GOSH are using the C-interface in `<io/nanopower.h>`. These functions are also available to be called from any part of your own C-code. In order to get eps housekeeping data just issue a C-call to the function `eps_get_hk()`. The C-interface will take care of generating the correct request and sending the request over the CSP network to the EPS, wait for the reply, decode the data and represent it nicely in a C data structure called 'eps\_hk\_t'. This makes it very easy to write a housekeeping collector or send commands to power subsystems on and off from anywhere on the satellite.

### Write your own driver

If you do not have a GomSpace OBC, or a GomSpace ground station with Lib-IO or GOSH, GomSpace is happy to share with you the source-code for the CSP network protocol and the C-library. This way you can port the EPS driver to your own CPU and architecture. Please contact GomSpace for more information about this option or see [github.com/GomSpace/libcsp](https://github.com/GomSpace/libcsp).

### CRC

After firmware version 2.5, the NanoPower supports the CSP option flag "CRC" (only on CSP communication, no in slave mode). If this flag is set on incoming packages, NanoPower expects a 32 bit CRC at the end of the package and will respond with a 32 bit CRC added to the data-package. Please see the CSP source code for more information about this

## CSP Network interface

*Applies to latest version of NanoPower - Your NanoPower may have other commands or may not support some of these commands. If you are in doubt please contact GomSpace for further help*

Mnemonic	Port	Request/Reply Data	Bytes	Description
GET_HK_1 (backwards compatible)	8	empty	0	Send empty packet to request backwards compatible housekeeping struct.
		struct hkparam_t;	struct	Reply: Data-structure (see below)
GET_HK_2 (p31u-8 format)	8	uint8_t type = 0	1	Send packet of length = 1 with type = 0 to request p31u-8 housekeeping struct.
		struct eps_hk_t;	struct	Reply: Data-structure (see below)
GET_HK_2_VI	8	uint8_t type = 1	1	Send packet of length = 1 with type = 1 to request voltage and current subset of HK_2
		struct eps_hk_vi_t;	struct	Reply: Data-structure (see below)
GET_HK_2_OUT	8	uint8_t type = 2	1	Send packet of length = 1 with type = 2 to request output switch data subset of HK_2
		struct eps_hk_out_t;	struct	Reply: Data-structure (see below)
GET_HK_2_WDT	8	uint8_t type = 3	1	Send packet of length = 1 with type = 3 to request wdt data subset of HK_2
		struct eps_hk_wdt_t;	struct	Reply: Data-structure (see below)
GET_HK_2_BASIC	8	uint8_t type = 4	1	Send packet of length = 1 with type = 4 to request the basic data subset of HK_2
		struct eps_hk_basic_t;	struct	Reply: Data-structure (see below)
SET_OUTPUT	9	uint8 output_byte;	1	Set output switch states by a bitmask where "1" means the channel is switched on and "0" means it is switched off. LSB is channel 1, next bit is channel 2 etc. (Quadbat switch and heater cannot be controlled through this command) [NC NC 3.3V3 3.3V2 3.3V1 5V3 5V2 5V1]
		none	x	No reply to this command
SET_SINGLE_OUTPUT	10	uint8 channel, uint8 value; int16 delay	4	Set output %channel% to value %value% with delay %delay%, Channel (0-5), Quadbat heater (6), Quadbat switch (7) Value 0 = Off, 1 = On Delay in seconds.
		none	x	No reply to this command
SET_PV_VOLT	11	uint16 voltage1, uint16 voltage2, uint16 voltage3;	6	Set the voltage on the photo-voltaic inputs V1, V2, V3 in mV. Takes effect when MODE = 2, See SET_PV_AUTO. Transmit voltage1 first and voltage3 last.
		none	x	No reply to this command
SET_PV_AUTO	12	uint8 mode;	1	Sets the solar cell power tracking mode: MODE = 0: Hardware default power point MODE = 1: Maximum power point tracking MODE = 2: Fixed software powerpoint, value set with SET_PV_VOLT, default 4V
		none	x	No reply to this command
SET_HEATER	13	uint8 cmd, uint8 heater, mode;	3	Cmd = 0: Set heater on/off Heater: 0 = BP4, 1 = Onboard, 2 = Both Mode: 0 = OFF, 1 = ON



Mnemonic	Port	Request/Reply Data	Bytes	Description
		uint8 bp4_heater, uint8 onboard_heater	2	Command replies with heater modes. 0=OFF, 1=ON. To do only query, simple send an empty message.
RESET_COUNTERS	15	uint8 magic=0x42;	1	Send this command to reset boot counter and WDT counters. magic = 0x42
		none	x	
RESET_WDT	16	uint8 magic=0x78;	1	Send this command to reset (kick) dedicated WDT. magic = 0x78
		none	x	
CONFIG_CMD	17	uint8 cmd;	1	Use this command to control the config system. cmd=1: Restore default config
		none	x	
CONFIG_GET	18	none	x	Use this command to request the NanoPower config.
		struct eps_config_t	struct	Reply: Data-structure (see below)
CONFIG_SET	19	struct eps_config_t	struct	Use this command to send a config to the NanoPower and save it.
		none	x	
HARD_RESET	20	none	x	Send this command to perform a hard reset of NanoPower, including cycling permanent 5V and 3.3V and battery outputs.
		none	x	
CONFIG2_CMD	21	uint8 cmd;	1	Use this command to control the config 2 system. cmd=1: Restore default config cmd=2: Confirm current config
		none	x	
CONFIG2_GET	22	none	x	Use this command to request the NanoPower config 2.
		struct eps_config2_t	struct	Reply: Data-structure (see below)
CONFIG2_SET	23	struct eps_config2_t	struct	Use this command to send config 2 to the NanoPower and save it (remember to also confirm it)
		none	x	

## Examples

For example, to set the photo-voltaic references, a request to address 2 port 11 must be made with 6 bytes of data, specifying the values. No reply will be given from the EPS on this command.

Another example, if you wish to request the housekeeping data, a packet to address 2 port 8 with a single data byte (=0) must be sent as a request. The reply will be generated to the source-address and the source-port of the request, and contain the eps\_hk\_t data-structure.

## Housekeeping format (P31u-6):

The housekeeping data is a structure as specified below:

```
typedef struct {  
    uint16_t pv[3];           //Photo-voltaic input voltage [mV]  
    uint16_t pc;              //Total photo current [mA]  
    uint16_t bv;              //Battery voltage [mV]  
    uint16_t sc;              //Total system current [mA]  
    int16_t temp[4];          //Temp. of boost converters (1,2,3) and onboard battery [degC]  
    int16_t batt_temp[2];     //External board battery temperatures [degC];  
    uint16_t latchup[6];      //Number of latch-ups on each output 5V and +3V3 channel  
                                //Order[5V1 5V2 5V3 3.3V1 3.3V2 3.3V3]  
                                //Transmit as 5V1 first and 3.3V3 last  
    uint8_t reset;            //Cause of last EPS reset  
    uint16_t bootcount;       //Number of EPS reboots  
    uint16_t sw_errors;       //Number of errors in the eps software  
    uint8_t ppt_mode;         //0 = Hardware, 1 = MPPT, 2 = Fixed SW PPT.  
    uint8_t channel_status;   //Mask of output channel status, 1=on, 0=off  
                                //MSB - [QH QS 3.3V3 3.3V2 3.3V1 5V3 5V2 5V1] - LSB  
                                // QH = Quadbat heater, QS = Quadbat switch  
}  
hkparam_t;
```

*EPS reset cause can be*

0. *Unknown reset*
1. *Dedicated WDT reset*
2. *I2C WDT reset*
3. *Hard reset*
4. *Soft reset\**
5. *Stack overflow*
6. *Timer overflow*
7. *Brownout or power-on reset*
8. *Internal WDT reset*

*\* Not all soft reset commands are captured and some of these will therefore register as reset nr. 7 or 8.*

*Temperatures are ordered as [converter 1, converter 2, converter 3, onboard battery], see picture under Connections for location of converters.*

## Housekeeping format (P31u-8 and P31u-9):

The housekeeping data is based in structures as specified below:

```
typedef struct __attribute__((packed)) {  
    uint16_t vboost[3];       //!< Voltage of boost converters [mV] [PV1, PV2, PV3]  
    uint16_t vbatt;           //!< Voltage of battery [mV]  
    uint16_t curin[3];        //!< Current in [mA]  
    uint16_t cursun;          //!< Current from boost converters [mA]  
    uint16_t cursys;          //!< Current out of battery [mA]  
    uint16_t reserved1;       //!< Reserved for future use  
    uint16_t curout[6];       //!< Current out (switchable outputs) [mA]  
    uint8_t output[8];        //!< Status of outputs**  
    uint16_t output_on_delta[8]; //!< Time till power on** [s]  
    uint16_t output_off_delta[8]; //!< Time till power off** [s]  
    uint16_t latchup[6];      //!< Number of latch-ups  
    uint32_t wdt_i2c_time_left; //!< Time left on I2C wdt [s]  
    uint32_t wdt_gnd_time_left; //!< Time left on I2C wdt [s]  
    uint8_t wdt_csp_pings_left[2]; //!< Pings left on CSP wdt  
    uint32_t counter_wdt_i2c;  //!< Number of WDT I2C reboots  
    uint32_t counter_wdt_gnd;  //!< Number of WDT GND reboots  
    uint32_t counter_wdt_csp[2]; //!< Number of WDT CSP reboots  
    uint32_t counter_boot;     //!< Number of EPS reboots  
    int16_t temp[6];           //!< Temperatures [degC] [0 = TEMP1, TEMP2, TEMP3, TEMP4, BP4a, BP4b]*  
}
```

```

uint8_t bootcause;    //!< Cause of last EPS reset
uint8_t battmode;    //!< Mode for battery [0 = initial, 1 = undervoltage,
                        2 = safemode, 3 = nominal, 4=full]

uint8_t pptmode;      //!< Mode of PPT tracker [1=MPPT, 2=FIXED]
uint16_t reserved2;

} eps_hk_t;
*BP4a and BP4b are temperatures on Quadbat BP4.
**[6] is BP4 heater, [7] is BP4 switch

typedef struct __attribute__((packed)) {
    uint16_t vboost[3];    //!< Voltage of boost converters [mV] [PV1, PV2, PV3]
    uint16_t vbatt;        //!< Voltage of battery [mV]
    uint16_t curin[3];     //!< Current in [mA]
    uint16_t cursun;       //!< Current from boost converters [mA]
    uint16_t cursys;       //!< Current out of battery [mA]
    uint16_t reserved1;    //!< Reserved for future use
} eps_hk_vi_t;

typedef struct __attribute__((packed)) {
    uint16_t curout[6];    //!< Current out (switchable outputs) [mA]
    uint8_t output[8];     //!< Status of outputs**
    uint16_t output_on_delta[8]; //!< Time till power on** [s]
    uint16_t output_off_delta[8]; //!< Time till power off** [s]
    uint16_t latchup[6];   //!< Number of latch-ups
} eps_hk_out_t;
**[6] is BP4 heater, [7] is BP4 switch
typedef struct __attribute__((packed)) {
    uint32_t wdt_i2c_time_left;    //!< Time left on I2C wdt [s]
    uint32_t wdt_gnd_time_left;    //!< Time left on I2C wdt [s]
    uint8_t wdt_csp_pings_left[2]; //!< Pings left on CSP wdt
    uint32_t counter_wdt_i2c;      //!< Number of WDT I2C reboots
    uint32_t counter_wdt_gnd;      //!< Number of WDT GND reboots
    uint32_t counter_wdt_csp[2];   //!< Number of WDT CSP reboots
} eps_hk_wdt_t;

typedef struct __attribute__((packed)) {
    uint32_t counter_boot;    //!< Number of EPS reboots
    int16_t temp[6];          //!< Temperatures [degC] [0 = TEMP1, TEMP2, TEMP3, TEMP4, BATT0, BATT1]
    uint8_t bootcause;        //!< Cause of last EPS reset
    uint8_t battmode;         //!< Mode for battery [0 = initial, 1 = undervoltage,
                                2 = safemode, 3 = nominal, 4=full]

    uint8_t pptmode;          //!< Mode of PPT tracker [1=MPPT, 2=FIXED]
    uint16_t reserved2;

} eps_hk_basic_t;

```

## I<sup>2</sup>C bus specification

The I<sup>2</sup>C interface is used for commanding the NanoPower and for receiving housekeeping and status messages. NanoPower operates on the I<sup>2</sup>C bus as multi-master node that is either as *slave receiver* or as *master transmitter*. NanoPower transmits at 400 kbit and can receive at anything from up to 400 kbit. The CSP network stack takes care of the I<sup>2</sup>C bus addressing and framing format. The CSP/I<sup>2</sup>C frame looks like this:

`<start><write><csp-header><data><stop>`

Where the CSP header is 4 bytes big endian (For more information on this, see libcsp.org and read the specification for the csp\_if\_i2c interface), and the data field contains the request/reply as specified in the command and data handling section of this datasheet.

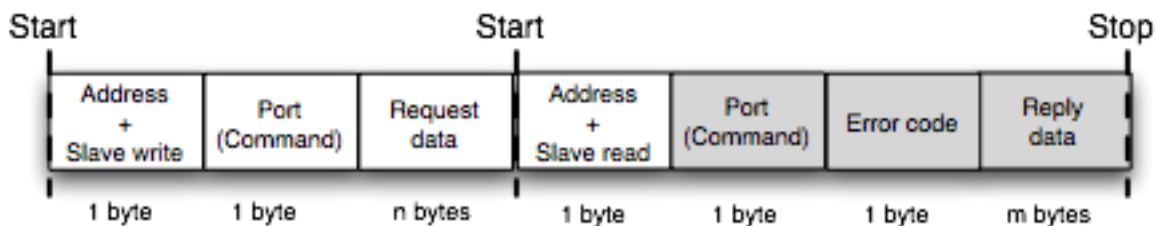
## I<sup>2</sup>C slave mode (legacy interface)

For users who do not wish to support the CSP protocol, or the multi-master I<sup>2</sup>C interface. The NanoPower comes with a separate slave-mode I2C interface. This interface can be enabled by setting the 'board i2cslave' to 1.

NOTE: When the NanoPower is set to slave mode, the CSP commands like "eps hk", "cmp ident", etc in GOSH does not work. Furthermore the slave interface, ie. "epsslave" commands, does not work either as the unit cannot be both slave and master at the same time.

This mode of operation disables the use of the CSP stack, and uses a slave-mode only protocol instead. A limited set of the CSP commands are available in this mode. An I2C master wishing to communicate with the NanoPower device should send a single byte specifying the command number, followed by any command arguments. The command number should match the CSP port number from multi-master mode.

The NanoPower returns the same 1-byte command number, followed by an error code. A successful command will return an error code of 0. Errors are marked by returning a non-zero error code. If a command is marked erroneous, the remaining bytes of the reply will be undefined and should be discarded. The following figure shows the command sequence on the I2C bus. White boxes are data sent from the I<sup>2</sup>C master, while grey boxes marks data read from the NanoPower system:



As in CSP mode, all values of more than 1 byte must be transmitted in big endian byte order.

NOTE: The NanoPower will clock-stretch up to 10ms until reply is ready.

## Additional Slave Mode Commands

Since the CSP stack is not used, the NanoMind command set has been extended with a ping and reboot command. These commands are normally handled by the CSP service handler.

Mnemonic	Port	Request/Reply Data	Bytes	Description
PING	1	uint8_t value	1	One byte ping value. Any value can be used.
		uint8_t value	1	The NanoPower replies with the same value as in the ping request.
REBOOT	4	uint8_t magic[4]	4	Magic sequence must be: 0x80,0x07,0x80,0x07
		none		The NanoPower is rebooted, so no reply is generated. A stop condition should be sent after the request, instead of the repeated START.

## NanoPower Terminal (GOSH)

As an extension of the NanoPower interface, there is also a command line utility available. This command system works by interpreting commands given on the serial port of the NanoPower and calling the NanoPower C-library functions. This means that they are available directly on the serial port by typing commands. This is great for debugging, configuring and getting started with the NanoPower.

The GomSpace Shell (GOSH) currently runs on almost all GomSpace products and even on the ground-station PC in a small application called CSP-Term, which is freely available. This means that the NanoPower commands can be executed from several different sources, depending on where you have access to a GOSH shell. When issuing a command from the Ground-station, TNC or OBC, the underlying CSP-protocol will ensure to route the commands correctly to the NanoPower and back again.

NanoPower Serial Port	
Voltage	0 - 3.3V
Baud	500000

### Commands

By typing 'help', in any GOSH shell, a list of commands will be printed. Most commands available are not subsystem specific, please see the GOSH manual for more information about these. To see the version of the NanoPower type 'cmp ident <csp add> <timeout>'. Example:

```
eps # cmp ident 2 1000
  Hostname: nanopower
  Model:    NanoPower P31u
  Revision: v2.0-20-g2b197c4
  Date:     Mar 21 2013
  Time:     09:51:22
```

To see the UID of the unit type 'board getuid'.

The standard NanoHub commands are all prefixed with the word 'eps'. In order to list the NanoPower commands type 'eps <tab>':

```
eps # eps
node          Set EPS address in OBC host table
hk            Get HK
hksub         Get HK sub structs
conf          Configuration
outputmask    Set on/off, argument hex value of output char
output        Set channel on/off, argument (channel) to (1 or 0)<channel> <mode> <delay>
resetcounters Reset all counters to zero
resetwdt      Resets the WDT GND
hardreset     Completely power cycle the EPS
ppt           Change current PPT mode
vboost        Change current VB00ST voltage
```

These commands are also found on other GOSH enable GomSpace subsystems like NanoMind.

A number of commands to setup up the NanoPower is available, type 'board <tab>' to see the options:

```
eps # board
getuid          Get product UID
heaterquadbat   Set/get heater on quadbat
heateronboard   Set/get heater onboard
wdti2c_rst      Set/get wdt i2c reset type
wdtgnd          Set/get wdt gnd
wdtcsp          Set/get wdt csp
i2cslave        Set/get i2c slave
i2cspeed        Set/get i2c speed (khz)
```

i2caddr	Set/get i2c addr
cspaddr	Set/get csp addr
csp_route_save	Save routing table to flash
csp_route_load	Load routing table from flash
csp_route_restore	Restore to factory routing table in flash

To help determining the characteristics of the attached solar panels, a power sweep function is included.

eps # epsdebug  
 powersweep Measure power at different vboost values

An example of requesting housekeeping is show below:

eps # eps hk

```

      +-----+
      |               |
      |               | I(mA),  lup,Ton(s),Toff(s)
      +-----+-----+ 1 (H1-47) --> EN:1 [ 245,    0,    0,    0]
1:    |               |
375 mV -> | Voltage   | 2 (H1-49) --> EN:1 [ 233,    0,    0,    0]
  0 mA -> | 07759 mV  |
  0 mW -> |               | 3 (H1-51) --> EN:1 [ 229,    0,    0,    0]
2:    | Input       |
3662 mV -> | 00173 mA 00713 mW | 4 (H1-48) --> EN:1 [ 229,    0,    0,    0]
357 mA -> |               |
1307 mW -> | Output   | 5 (H1-50) --> EN:1 [ 238,    0,    0,    0]
3:    | 00772 mA 05989 mW |
3662 mV -> |               | 6 (H1-52) --> EN:1 [ 245,    0,    0,    0]
  41 mA -> | Efficiency: |
 150 mW -> | In: 91 %   | 7          --> EN:0
      | Normal       | 8          --> EN:0
      +-----+-----+

```

	1	2	3	4	5	6
Temp:	+25	+28	+26	+25	+0	+0

	Boot	Cause	PPTm
Count:	5	2	2

	WDTi2c	WDTgnd	WDTcsp0	WDTcsp1
Count:	0	0	0	0
Left:	0	0	5	5



## Configuration

The NanoPower allows the user to setup a number of configurations. This is done in two different config structs (config and config2) through the standard command system through the 'eps conf' commands:

```
eps # eps conf
get          Conf get
set          Conf set to RAM
edit         Edit local config
print        Print local config
restore      Restore config from default
confirm      Confirm new config (save to EEPROM)
```

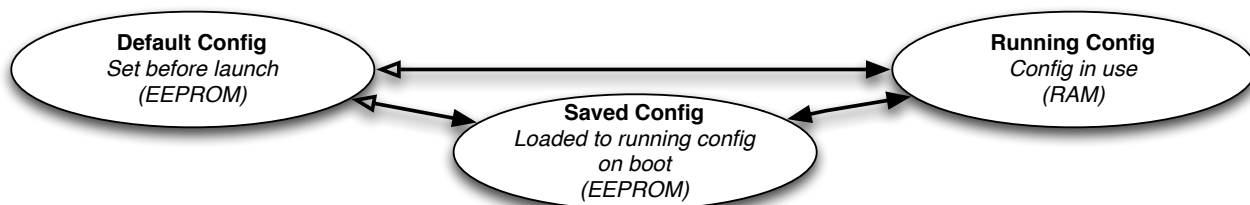
and the 'eps conf2' commands:

```
eps # eps conf2
get          Conf2 get
set          Conf2 set to RAM
edit         Edit local config2
print        Print local config2
restore      Restore config2 from default
confirm      Confirm new config2 (save to EEPROM)
```

Furthermore, there are dedicated GOSH commands to handle the default configs:

```
eps # conf
restore      Configuration restore
store_default Store configuration as default
eps # conf2
restore      Configuration2 restore
store_default Store configuration2 as default
```

The NanoPower configuration has three instances: *a running config*, *a saved config* and *a default config*.



The running config is the one currently used by NanoPower and it can be edited through the 'conf get', 'conf edit' and 'conf set' commands.

The running config must be saved to the saved config through the 'conf confirm' command. On boot the saved config is loaded to the running config.

The default config can be seen as a backup config, which can only be set before launch through GOSH by 'conf store\_default'. There are four ways of restoring to the default config:

- By typing 'conf restore' in NanoPower GOSH
- By using the restore command over I2C ('eps conf restore' in GOSH)
- If a checksum error is found in the working config, the default config is restored
- If the Dedicated WDT times out, the default config is restored.

All these restore events will restore to both the saved and the running config.

The idea behind having a default config that can only be written before launch is that any potential erroneous config sent to NanoPower will be reverted if it threatens the satellite mission. An

example could be a config with all outputs set to off, this will mean that no subsystems are switched on and the mission is threatened. However, after the timeout of the dedicated WDT used and a ground communication WDT, the default config is restored and the normal subsystem are switched on.

To view the current config of the NanoPower type 'eps conf get' in GOSH. To edit the config type 'eps conf edit'. To send and set the running config on NanoPower type 'eps conf set'. If this is not followed by a 'eps conf confirm' within 30 seconds the previous running config is restored. If a 'eps conf confirm' is received, the restore is prevented and the config is furthermore save to the 'saved config'.

Config2 can be used in a similar way using the 'eps conf2' commands.

The config struct is defined as

```
typedef struct __attribute__((packed))
{
    uint8_t ppt_mode;                //!< Mode for PPT [1 = AUTO, 2 = FIXED]
    uint8_t battheater_mode;         //!< Mode for battheater [0 = Manual, 1 = Auto]
    int8_t battheater_low;           //!< Turn heater on at [degC]
    int8_t battheater_high;          //!< Turn heater off at [degC]
    uint8_t output_normal_value[8];  //!< Nominal mode output value
    uint8_t output_safe_value[8];    //!< Safe mode output value
    uint16_t output_initial_on_delay[8]; //!< Output switches: init with these on delays [s]
    uint16_t output_initial_off_delay[8]; //!< Output switches: init with these off delays [s]
    uint16_t vboost[3];              //!< Fixed PPT point for boost converters [mV]
} eps_config_t;
```

The config2 struct is defined as

```
typedef struct __attribute__((packed)) {
    uint16_t batt_maxvoltage;
    uint16_t batt_safevoltage;
    uint16_t batt_criticalvoltage;
    uint16_t batt_normalvoltage;
    uint32_t reserved1[2];
    uint8_t reserved2[4];
} eps_config2_t;
```



**Warning:** Before integrating NanoPower into a satellite, make sure that the configs and default configs are set to appropriate values.

## Operation and Handling

### Warnings:



The high-performance battery-charge system is designed to operate only when the battery is connected to act as a power sink. *Before applying current to any solar panel input, make sure that the battery is properly connected!!* If the battery is not connected, internal voltage transients may cause degradation of the expected life time of the system.



Balance out charge between NanoPower batteries and external batteries before connecting them together.



The NanoPower system employs components based on FETs and therefore requires anti-static handling precautions to be observed. Do not touch or handle the product without proper grounding!

## Getting Started

In order to get the NanoPower up and running attach a serial cable (eg. a TTL-232RG-VSW3V3-WE) to connector P12 (3.3 V levels) and setup terminal program (eg Minicom in Linux or RealTerm in Windows) to a baudrate of 500.000 baud. Switch ON the NanoPower by using one of the kill-switch inputs. You are now connected to the GOSH terminal on the NanoPower.

To see it charge the batteries, the following setups are suggested:

### Setup A: Charging through charge-circuit

Connect the battery with jumper.

Make sure no serial cable is inserted.

Connected a 5V/1A laboratory power supply to the charge input (either Connector P8 pin 6, CHARGE, or Stackconnector 5V\_in) 5V\_in pin and ground.

Watch the voltage and current on the power supply - should read approximately 5V/0.9A.

At end of charge NanoPower will stop drawing power and the power supply should read 5V/0A

### Setup B: Charging through photo-voltaic input

Connect the battery with jumper.

Connected a current-limited 5 V/1A laboratory power supply to one of the photovoltaic inputs.

Activate the system by briefly shorting the kill-switch connection.

Watch the voltage and current on the lab power supply - should read 3.7 V/1 A. This means that it has set the power point to 3.7 V and draws as much current as limited by the lab supply.

For latching kill-switch

To switch on NanoPower (for latching kill-switch): Short the kill-switch connection to ground once.

To switch off NanoPower: Short the reset connection to ground once

To ensure a switched off NanoPower: Keep RBF connection shorted (inhibits a switch on)

## Technical Support

Upon delivery of the product the costumer is assigned to a support engineer at GomSpace, who will assist the costumer in starting to use the product and who will provide support and answers to technical questions. Support communication will be by means of e-mail and occasional phone calls.

For products bought by the costumer under the conditions and prices listed on our homepage only limited and reasonable support can be expected from GomSpace and no additional technical documentation other than this datasheet can be expected. If additional support is required please discuss options and obtain a quote prior to ordering.

It is important that the costumer assigns a single person (for each GomSpace product), who is responsible for all communication in relation to technical support.

## Customization Options

As GomSpace realizes that different applications place different requirements to a power system, the NanoPower products present a variety of options for customization. Options are to be agreed upon time of order placement.

**Please use *NanoPower Options Form* to indicated desired options upon ordering - it can be downloaded from GomSpace homepage.**

## Quality Assembly

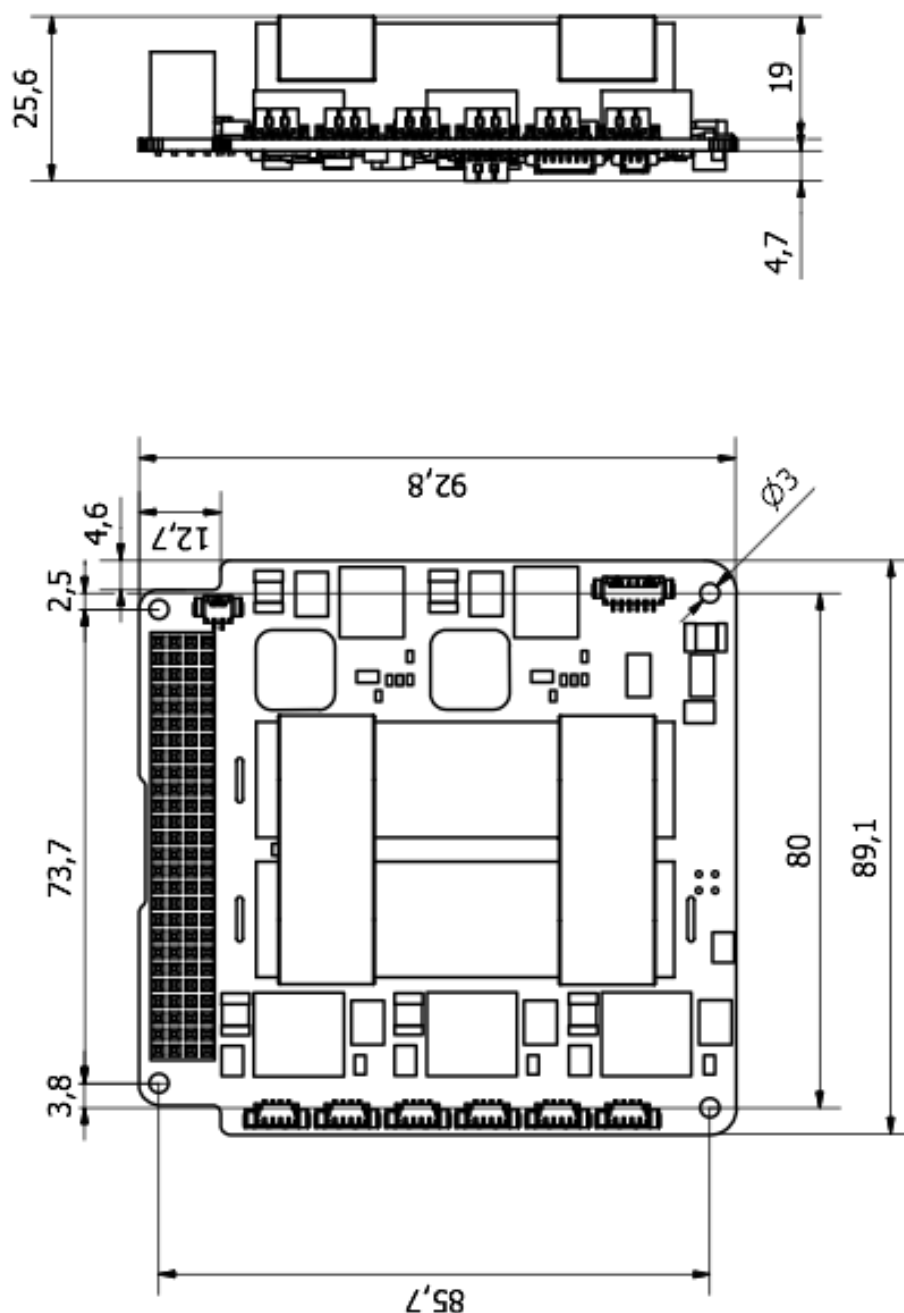
GomSpace space hardware is assembled in a procedure where all parts are cleaned with IPA and then soldered in an anti-static environment to "IPC-A-610 Class 3" specifications. All solder-work is done with tin-lead 63/37 using rosin flux. All solder joints are re-checked for class 3 compliance and the PCB is finally cleaned with IPA and tested.

## Physical Dimensions

PCB-type is Glass-Polyimide from ESA approved producer, 6 layer (3 pairs), tin-lead HAL surface, 1.60 mm thick. Masses will vary depending on customer choices.

Model	Mass
NanoPower P31u without batteries. With low stack connector	100 g
NanoPower P31u. With 2600mAh batteries. With high stack connector	200 g
NanoPower P31u-s	270 g

Dimensions are given in mm.





## Changes from version 8 to version 9

The version 9 of NanoPower has received a number of upgrades:

- Added a battery ground break connector for ISS launch compatibility.
- Short circuit protection of batteries
- Dedicated channel for onboard battery heater
- Diode protection on serial port to allow charging with NanoPower off and serial cable inserted.
- Improved internal latch-up protection and power dissipation

Revision Date	Changes	Valid for firmware version
05/05-2014	First version of datasheet for NanoPower P31u v9 Updated information on: serial port, heater, connections.	2.11
11/06-2014	Rearranged sections in datasheet. Fixed spelling errors. Updated block diagram.	2.11
05/08-2014	Added section on operating without battery Added option for I <sup>2</sup> C wdt to perform hardreset Updated 'board' command information Updated photo of BP4	2.12
17/10-2014	Updated information on charging. Fixed battery mode error in housekeeping	2.12
12/02-2015	Updated for firmware 2.16. Updated Dedicated WDT description.	2.16
18/03-2015	Added s-version power consumption to table.	2.16
19/08-2015	Fixed textual problem on page 9	2.16
27/08-2015	Updated config and config2 operation as pr firmware 2.17.	2.17

Firmware Revision	Changes
2.11	Update to P31u-9 board Added new onboard heater channel Fixed issue with GND wdt not resetting when set to 1 hour
2.12	Fixed issue where I <sup>2</sup> C wdt could do reset when disabled. Added option for choosing I <sup>2</sup> C timeout action to hardreset.
2.16	Fixed issue on firmware 2.12 where EPS would do random resets when communication in slave I <sup>2</sup> C mode. Updated GND WDT functionality. It will now timeout on reboots as well. See updated description.



Date: 27/08-2015  
Doc. ref: gs-ds-nanopower-p31u-9.0  
Doc. Type: Datasheet  
Name: NanoPower P31u-9.0

Firmware Revision	Changes
2.17	Operation of config and config2 are changed to both function in the same manner. 'eps conf set' sets new config to RAM and 'eps conf confirm' saves to RAM. If new confirm command is send within 30 seconds the original RAM config is restored. MPPT algorithm now resets to the user set vboost value.

GomSpace ApS • Niels Jernes Vej 10 • 9220 Aalborg E • Denmark  
Tel: +45 9635 6111 • Fax: +45 9635 4599 • Web: [www.GomSpace.com](http://www.GomSpace.com)

The information furnished by GomSpace in this data sheet is believed to be accurate and reliable. However, no responsibility is assumed by GomSpace for its use. GomSpace reserves the right to change circuitry and specifications at any time without notification to the customer. Current sale and delivery conditions are available on the homepage of GomSpace

© 2014 GomSpace ApS.