NanoPower Flexible Battery Pack
BPx series
V2.0

High capacity battery pack for nano-satellite

NanoPower BPx series battery pack

High capacity lithium ion battery pack for nano satellites. Allows a wide range of parallel/series combinations and up to 16 cells with a maximum of 8 cells in series. Several BPx battery packs can be coupled in parallel if even greater capacity packs are needed.

Feature Overview

- Lithium ion battery back for space applications
- Utilizes 18650 lithium ion cells.
- Several different cell types to choose from
- Can be configured for nominal voltages ranging up to 28.8V
- Provides telemetry over I²C
    - voltage
    - current
    - temperature
- Autonomous heater system

Applications

- CubeSat satellites
- Nano satellites
- ISS launch qualified

Functional Description

The NanoPower BPx battery pack is designed to be compact and light, while providing high battery capacity at low cost.

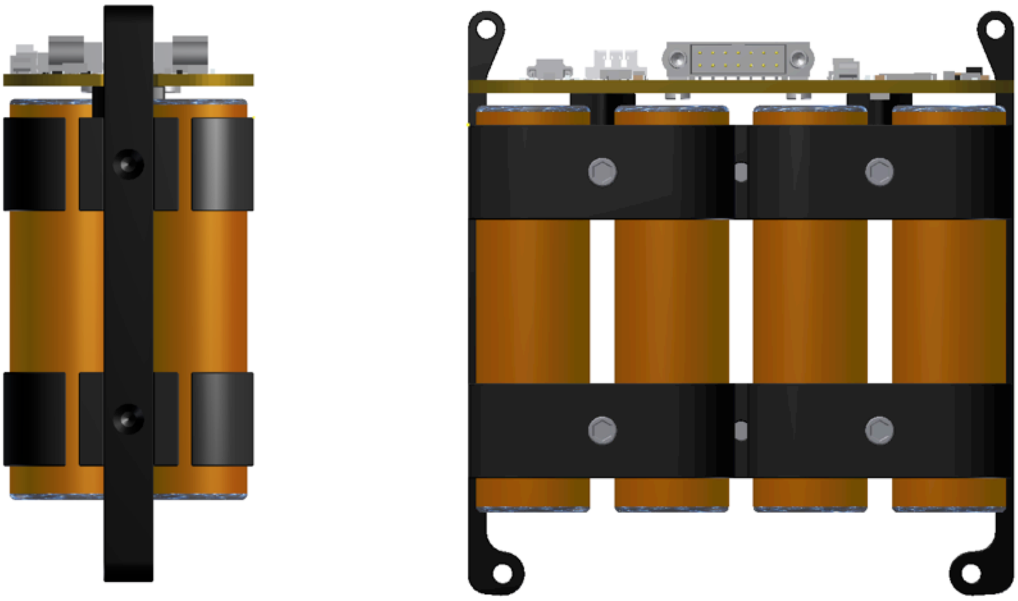
Models

NanoPower BPx can be configured with a wide range of different battery options.

Configurations	Number of cells	2600mAh cell (Nominal)	1800mAh cell (Nominal)	1100mAh cell (Nominal)
2S-3P	6	58Wh 6 - 8.4V 7.8Ah	40Wh 6 - 8.4V 5.4Ah	21.8Wh 4 - 7.2V 3.3Ah
3S-2P	6	58Wh 9 - 12.6V 5.2Ah	40Wh 9 - 12.6V 3.6Ah	21.8Wh 6 - 10.8V 2.2Ah
2S-4P	8	77Wh 6 - 8.4V 10.4Ah	53Wh 6 - 8.4V 7.2Ah	29.0Wh 4 - 7.2V 4.4Ah
4S-2P	8	77Wh 12 - 16.8V 5.2Ah	53Wh 12 - 16.8V 3.6Ah	29.0Wh 8 - 14.4V 2.2Ah
8S-1P	8	77Wh 24 - 33.6V 2.6Ah	53Wh 24 - 33.6V 1.8Ah	29.0Wh 16 - 28.8V 1.1Ah
4S-3P	12	115Wh 12 - 16.8V 7.8Ah	80Wh 12 - 16.8V 5.4Ah	44Wh 8 - 14.4V 3.3Ah
3S-4P	12	115Wh 9 - 12.6V 10.4Ah	80Wh 9 - 12.6V 7.2Ah	44Wh 6 - 10.8V 4.4Ah

Configurations	Number of cells	2600mAh cell (Nominal)	1800mAh cell (Nominal)	1100mAh cell (Nominal)
4S-4P	16	154Wh 12 - 16.8V 10.4Ah	107Wh 12 - 16.8V 7.2Ah	58Wh 8 - 14.4V 4.4Ah
2S-8P	16	154Wh 6 - 8.4V 20.8Ah	107Wh 6 - 8.4V 14.4Ah	58Wh 16 - 28.8V 8.8Ah
8S-2P	16	154Wh 24 - 33.6V 5.4Ah	107Wh 24 - 33.6V 3.6Ah	58Wh 16 - 28.8V 2.2Ah

For information on battery specifications (temperatures, charge and discharge current etc), please see the GomSpace battery datasheet (gs-ds-battery).



Example of a 8-cell bpx

Housekeeping

The NanoPower BPx 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 I²C interface. Measurements include:

- Two temperatures
- Current into and out of batteries
- Battery voltage

Parameter	Range	Resolution
Temperature	-40 to +125 deg C	1 deg C
Currents	0 to 6A / 0 to 10A	6mA / 10mA
Battery voltage	0 to 20V / 0 to 40V	20mV / 40mV

Note that in some cases the current measurements are not capable of measuring all the way to the zero.

Specifications

Parameter	Condition	Min	Typ	Max	Unit
Heater output	*Optional*				
- Voltage	Battery voltage		Vbat		V
- Current				2	A
Enable input (active high)	Onboard 300k pull down				
- input low		0		0.6	V
- input high		2		Vbat	V
Enable input (active low)	*Optional* Onboard 300k pull up				
- active	Use open-collector output to control	0		Vbat - 3	V
Battery - Voltage - Discharge - Charge	Raw battery voltage (depends on battery configuration) *Please see battery datasheet for more information*				
Power consumption	Power consumed by onboard electronics (at battery voltage)		4		mA
Off current	Current consumed with enable pin OFF		15		uA

Short circuit protection

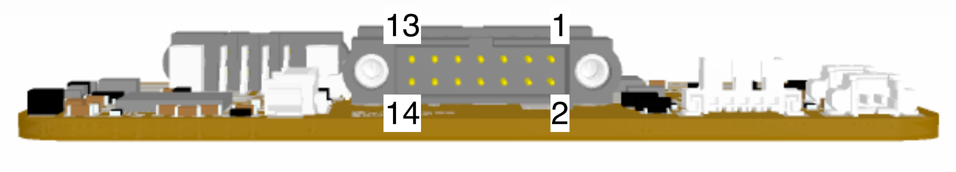
The raw battery output on connector P2 can optionally be short circuit protected and subject to activation through the enable pin. When the short circuit protection is chosen, it is advisable to use the optional active low enable pin.

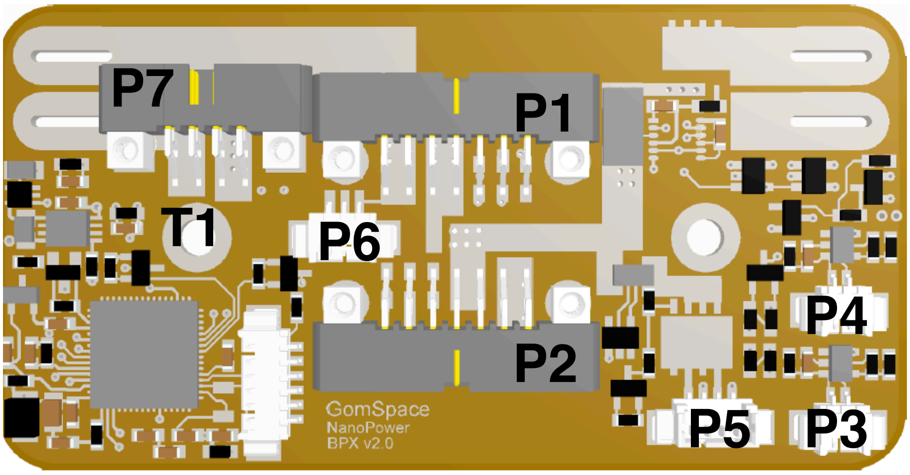
Parallel connection

It is possible to connect several BPx in parallel using the loop connector P1. Be aware that it is very important to balance out charge between battery packs before connecting them in parallel.

Connections

The drawing below shows different connection of the circuit board.





- P1: Battery Connector 1 (Loop Connector)
- P2: Battery Connector 2 (Output Connector)
- P3: Battery temperature sensor 1 interface
- P4: Battery temperature sensor 2 interface
- P5: Heater connector
- P6: GOSH serial port
- P7: ISS launch compatible ground break connector.
- T1: Temperature sensor 1

P1 / P2 Battery Connector 1 and 2 *Harwin M80*	
Pin	Usage
1	GND
2	GND
3	GND
4	GND
5	Vbat
6	Vbat
7	Vbat
8	Vbat
9	SCL
10	Enable (Active high)
11	SDA
12	GND
13	RESERVED
14	Enable (Active low) (Optional)

P7 Ground break connector *(optional)* *Harwin M80*	
Pin	Usage
1	GND_BAT
2	GND_BAT
3	GND_BAT
4	GND_BAT
5	GND_SYS
6	GND_SYS
7	GND_SYS
8	GND_SYS

Terminal (GOSH)

As an extension of the BPx interface, there is also a command line utility available. This command system works by interpreting commands given on the serial port of the BPx and calling the BPx 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 BPx.
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 BPx 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 BPx and back again.

Serial port P6	
Pin 0	BPx Tx
Pin 1	BPx Rx
Pin 2	GND
Voltage	0 - 3.3V
Baud	38400

⚠ When the BPx is off, it is advised to remove the serial cable. Leak current on the Rx line can in some case keep powering the electronics.

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 BPx type ‘cmp ident <csp add> <timeout>’. Example:

```
eps # cmp ident 7 1000
  Hostname: bpx
  Model:    BPx
  Revision: v2.0
  Date:     Dec 19 2013
  Time:     09:51:22
```

To see the UID of the unit type ‘board getuid’.

The standard BPx commands are all prefixed with the word ‘bpx’. In order to list the BPx commands type ‘bpx <tab>‘

Command and Data Interface

The NanoPower BPx 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 command

If you have a nanomind OBC in your satellite, a driver for the all nanopower systems are already included. If you do not, there is also the option of writing a custom driver, based on the CSP interface specification.

1. **Using Lib-IO (C-interface)**
   All the functions and commands in GOSH are using the C-interface in <io/bpx.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 bpx_get_hk(). The C-interface will take care of generating the correct request and sending the request over the CSP network to the BPx, wait for the reply, decode the data and represent it nicely in a C data structure called 'bpx_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.

2. **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 nanopower drivers to your own CPU and architecture. Please contact gomspace for more information about this option.

CSP Network interface

Applies to latest version of BPx - Your system 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	9	empty	0	Send empty packet to request housekeeping struct.
		struct bpx_hk_t;	struct	Reply: Data-structure (see below)
RESET_COUNTERS	15	uint8 magic=0x42;	1	Send this command to reset boot counter magic = 0x42
		none	x	
HEATER CONTROL	16	uint8 cmd;	1	Manual control of heater (will be overruled if heater is in auto mode). Send cmd = 0 to switch heater off, cmd = 1 to switch heater on, and any other value of cmd to read out state of heater.
		uint8 heater_state	1	Reply: heater_state = 0 (OFF), heater_state = 1 (ON)
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 BPx 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 BPx and save it.
		none	x	

Information like uptime, identification etc can be accessed through the standard CSP interface, please see libcsp.org for more information.

Housekeeping format:

The housekeeping data is a structure as specified below:

```
typedef struct {
       uint16_t cur_charge;          //! Charge current in mA
       uint16_t cur_discharge;       //! Discharge current in mA
       uint16_t cur_heater;          //! Heater current in mA
       uint16_t vbatt;               //! Battery voltage in mV
       int16_t bat_t1;               //! Battery temperature 1 in degC
       int16_t bat_t2;               //! Battery temperature 2 in degC
       int16_t board_t;              //! BPX board temperature in degC
       uint16_t bootcount;           //! Number of reboots
       uint8_t  reset;               //! Cause of last reset
} bpx_hk_t;
```

Reset cause can be 0=Power On Reset, 1=External Reset, 2=Brown Out Reset, 4=WDT reset, 8=JTAG reset

I^{2}C bus specification

The I^{2}C interface is used for commanding the NanoPower and for receiving housekeeping and status messages. NanoPower operates on the I^{2}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^{2}C bus addressing and framing format. The CSP/I^{2}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.

Parameter	Condition	Min	Typ	Max	Unit
I^{2}C speed					
transmit			400		kbit
receive		0		400	kbit
I^{2}C / CSP address	Default (can be changed through GOSH)		7		

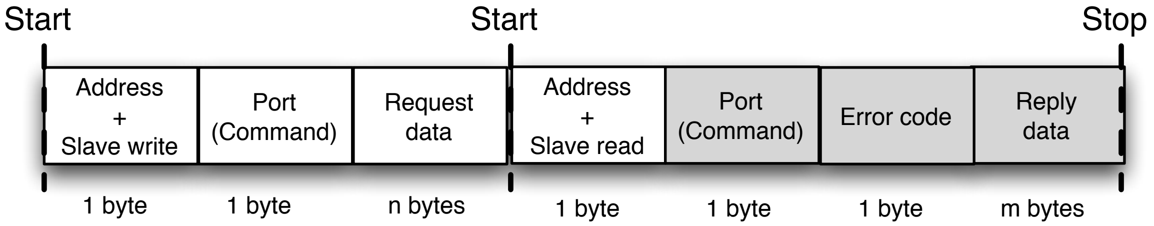
I2C slave mode

For users which do not wish to support the CSP protocol, or the multi-master I2C interface. The BPx comes with a separate slave-mode I2C interface. This interface can be enabled by setting the ‘board i2cslave’ to 1.

NOTE: When the BPx is set to slave mode, the CSP commands like “bpx hk”, “cmp ident”, etc in GOSH does not work. Furthermore the slave interface, ie. “bpxslave” 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 BPx 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 BPx 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 I2C master, while grey boxes marks data read from the BPx system:



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

Additional Slave Mode Commands

Since the CSP stack is not used, the 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 BPx 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 BPx is rebooted, so no reply is generated. A stop condition should be sent after the request, instead of the repeated START.

Configuration

The BPx allows the user to setup a number of configurations like auto-heater parameters. This is done in two different config structs through the standard command system through the ‘bpx conf’ commands:

```
bpx # bpx conf
  get                   Conf get
  set                   Conf set
  edit                  Edit local config
  print                 Print local config
  restore              Restore config from default
```

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

```
bpx # conf
  restore                      Configuration restore
  store_default                Store configuration as default
```

To view the current config of the BPx type ‘bpx conf get’ in GOSH and to edit the config type ‘bpx conf edit’. To send and save the configuration on BPx type ‘bpx conf set’.

On the BPx, a configuration has two instances:, a working config and a default config:

- The working config is the one currently used by BPx and it can be edited through the ‘conf get’, ‘conf edit’ and ‘conf set’ commands.
- The default config can be seen as a backup config which can only be set before launch through GOSH by by ‘conf store_default’. There are three ways of restoring to the default config:
  1. By typing ‘conf restore’ in BPx GOSH
  2. By using the restore command over I2C (‘BPx conf restore’ in GOSH)
  3. If a checksum error is found in the working config, the default config is restored

The config struct is defined as

```
typedef struct __attribute__((packed)) {
        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]
} bpx_config_t;
```

Operation and Handling

Warnings:

This high-performance battery contains batteries capable of delivering very high currents. Be very careful in avoiding shorts.

Balance out charge between NanoPower BPx battery packs when connecting them in parallel.

The NanoPower BPx 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!

Technical Support

Upon delivery of the product the costumers 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 BPx Form* to indicated desired options upon ordering - it can be downloaded from GomSpace's homepage.

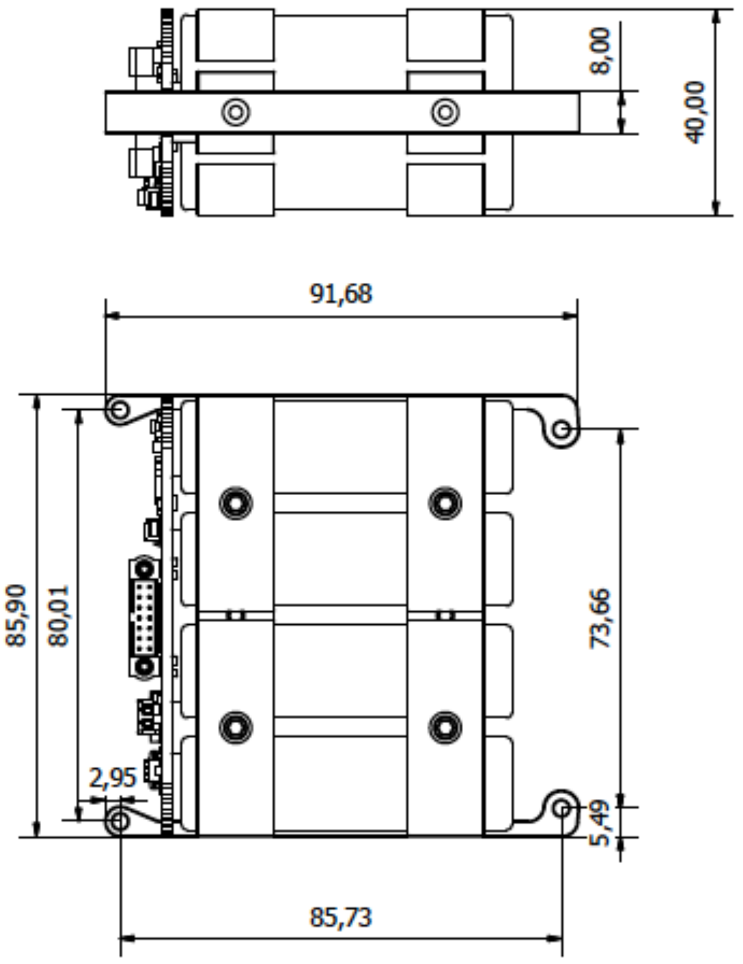
Quality Assembly

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.

Mechanical specifications

PCB-type is Glass-Polyimide from ESA approved producer , 4 layer (2 pairs), tin-lead HAL surface.
Aluminum is hard-anodized 7075-T6.
Adhesive is Scotch-Weld epoxy 2216.
Wire is PTFE insulated.

Dimensions vary depending on configuration. All dimensions are given in mm.



Masses will vary depending on configuration and cell choices. Below are a number of example configurations shown.

Model	Mass
BPx 6 cell 2600mAh	370 g
BPx 8 cell 2600mAh	470 g
BPx 12 cell 2600mAh	700 g
BPx 16 cell 2600mAh	900 g

Revision Date	Changes
22/04-2013	First document version
23/01-2014	Updated with V2.0 pcb information. Updated housekeeping Added slave cmd information Added information on config system Added GOSH information

GomSpace ApS • Niels Jernes Vej 10 • 9220 Aalborg E • Denmark
Tel: +45 9635 6111 • Fax: +45 9635 4599 • Web: 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