BuWizz 2.0 API

Version: 1.3

Introduction

BuWizz is a smart connected power brick device compatible with Lego elements. It features a built-in rechargeable battery, 4 power outputs (e.g. for motors) and a wireless connection using Bluetooth Low Energy (BLE, Bluetooth Smart).

This document describes the communication protocol with the BuWizz device in bootloader and application mode. The document provides information on Bluetooth BLE specific device structure, interfaces and packet exchange protocol.

BLE is a wireless communication standard, introduced in Bluetooth 4. It is intended for low-power accessory devices, has lower transfer speeds and lower overhead than classic Bluetooth technology. It features a unique software model, introducing GATT (Generic Attribute Profile) as the primary communication mean.

BLE devices operate either as central (running GATT client) or device (running GATT server) and exchange data via reading and writing to characteristics values, which are exposed via service and characteristic descriptors.

Notable operations are (supported operations are shown in **bold**):

- **advertisement**: the device advertises special message so that the central can scan and find active devices
- **services list discovery** (central can discover the list of services, supported by the device)
- **write data** to the device with or without device acknowledge
- read (poll) the data from the device
- **notifications**: device notifies the central with the new data (without acknowledging)
- indications: device indicates to the central with the new data (same as notify, but uses acknowledgement)

Note
BLE supports pairing, but is not required. This allows the streamlined user experience, where the application (iOS or Android) can scan for devices and connect to one it finds (without requesting the user to pair the device in system settings first).

Operation modes

The device has two operation modes:

- Bootloader mode: enables transferring of the new firmware
- Application mode: user application running to support BuWizz device functions
- Application sleep mode: the device is in low-power mode, no Bluetooth function is active, motor power is disabled

By default, the device starts in application mode if memory contents verification succeeds. Otherwise or if application requests it, the device starts in bootloader mode (bootloader has a timeout of 2 minutes - if no data is received for this amount of time, the device is automatically restarted).

BuWizz Bluetooth BLE interface

The BuWizz device is a standard Bluetooth BLE device and is configured to not use the pairing with the central in order to speed up the connection process and allow multiple users to use the device (one at a time).

Discovering BuWizz device

When active, BuWizz device advertises main advertisement data and optional scan response data (shown in Figure 1). Main advertisement data contains device name ('BuWizz') and short manufacturer information (sequence of 6 bytes - 05:4E:'B':'o':'o':'t' in bootloader and 05:4E:'B':'W':'x':'y' in the application, where x and y are replaced with firmware version). Scan response data contains the 128-bit UUID of the device's main (BuWizz application) service.

The data in main advertisement packet and scan response packet can be used to identify the device.

If required, version bytes x and y can be used for other purposes (e.g. identifying a specific BuWizz device by its serial number).

The advertisement interval is 100-250 ms to facilitate fast device discovery and connection.

Description	Value	Index
AD Data 0: <<Flags>>		
AD Data 1: <<Complete Local Name>>		
Length of this data	0x07	[3]
<<Complete Local Name>>	0x09	[4]
B	0x42	[5]
u	0x75	[6]
W	0x57	[7]
i	0x69	[8]
z	0x7A	[9]
z	0x7A	[10]
AD Data 2: <<Manufacturer Specific Data>>		
Length of this data	0x07	[11]
<<Manufacturer Specific Data>>	0xFF	[12]
0x4E	0x4E	[13]
0x05	0x05	[14]
0x42	0x42	[15]
0x6F	0x6F	[16]
0x6F	0x6F	[17]
0x74	0x74	[18]

Description	Value	Index
AD Data 0: <<Incomplete List of 128-bit Service Class UUIDs>>		
Length of this data	0x11	[0]
<<Incomplete List of 128-bit Service Class UUIDs>>	0x06	[1]
Service:		
[0]	0x93	[2]
[1]	0x6E	[3]
[2]	0x67	[4]
[3]	0xB1	[5]
[4]	0x19	[6]
[5]	0x99	[7]
[6]	0xB3	[8]
[7]	0x88	[9]
[8]	0x81	[10]
[9]	0x44	[11]
[10]	0xFB	[12]
[11]	0x74	[13]
[12]	0x00	[14]
[13]	0x00	[15]
[14]	0x05	[16]
[15]	0x4E	[17]

Figure 1: Main advertisement data (left) and scan response data (right)

Connecting to BuWizz device

The device uses standard Bluetooth BLE mechanisms for establishing a connection.

Data exchange with BuWizz device

Commands and data are exchanged with the BuWizz device with the use of BuWizz application and bootloader services, depending on the operating mode of the device. When writing to inactive service, the device will discard the data.

The services have the following 128-bit UUIDs:

	Service UUID
Application	93:6E:67:B1:19:99:B3:88:81:44:FB:74:00:00:05:4E
Bootloader	0F:DC:A4:95:E6:CD:0E:90:BA:46:98:AC:C1:A4:05:4E

These services both have a data exchange characteristic with a data exchange descriptor and a Client Characteristic Configuration Descriptor (CCCD) controlling the characteristic behavior.

	Data descriptor UUID	Data descriptor handle	CCCD handle
Application	0x92D1	0x03	0x05
Bootloader	0x0001	0x09	0x09

Data is sent to BuWizz device by writing to the data descriptor of the target service, while notifications are used to receive data from the BuWizz device. A value of 1 must be written to CCCD descriptor to enable the notifications. Packet size that is sent to device or received from the device can be between 1 and 183 bytes long. No error checking is included at the packet level (data integrity is guaranteed by the lower levels of the BLE protocol).

The device supports MTU size exchange to increase the MTU size in bootloader mode for improved transfer speed.

Handle	UUID	UUID Description	Value	Properties
Primary Service Declaration				
0x0001	0x2800	Primary Service Declaration	93:6E:67:B1:19:99:B3:88:81:44:FB:74:00:00:05:4E	
Characteristic Declaration				
0x0002	0x2803	Characteristic Declaration	1C:03:00:D1:92	
0x0003	0x92D1			0x1C
0x0004	0x2901	Characteristic User Description	43:6F:6D:6D:75:6E:69:63:61:74:69:6F:6E:20:69:6E:74:65:72:66:61:63	
0x0005	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration				
0x0006	0x2800	Primary Service Declaration	0F:DC:A4:95:E6:CD:0E:90:BA:46:98:AC:C1:A4:05:4E	
Characteristic Declaration				
0x0007	0x2803	Characteristic Declaration	18:08:00:01:00	
0x0008	0x0001			0x18
0x0009	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration: Generic Access				
0x000A	0x2800	Primary Service Declaration	00:18 (Generic Access)	
Characteristic Declaration: Device Name				
0x000B	0x2803	Characteristic Declaration	02:0C:00:00:2A	
0x000C	0x2A00	Device Name	42:75:57:69:7A:7A	0x02
Characteristic Declaration: Appearance				
0x000D	0x2803	Characteristic Declaration	02:0E:00:01:2A	
0x000E	0x2A01	Appearance	80:01	0x02
Characteristic Declaration: Peripheral Preferred Connection Parameters				
0x000F	0x2803	Characteristic Declaration	02:10:00:04:2A	
0x0010	0x2A04	Peripheral Preferred Connection Parameters	28:00:28:00:00:00:C8:00	0x02
Primary Service Declaration: Generic Attribute				
0x0011	0x2800	Primary Service Declaration	01:18 (Generic Attribute)	
Characteristic Declaration: Service Changed				
0x0012	0x2803	Characteristic Declaration	20:13:00:05:2A	
0x0013	0x2A05	Service Changed		0x20
0x0014	0x2902	Client Characteristic Configuration	00:00	
Primary Service Declaration: Device Information				
0x0015	0x2800	Primary Service Declaration	0A:18 (Device Information)	
Characteristic Declaration: Manufacturer Name String				
0x0016	0x2803	Characteristic Declaration	02:17:00:29:2A	
0x0017	0x2A29	Manufacturer Name String	46:4F:52:54:52:4F:4E:49:4B	0x02
Characteristic Declaration: Model Number String				
0x0018	0x2803	Characteristic Declaration	02:19:00:24:2A	
0x0019	0x2A24	Model Number String	42:57:31:37:2E:30	0x02
Characteristic Declaration: Serial Number String				
0x001A	0x2803	Characteristic Declaration	02:1B:00:25:2A	
0x001B	0x2A25	Serial Number String	78:78:78:78:78:78:78:78:78:78:78:78	0x02
Characteristic Declaration: Firmware Revision String				
0x001C	0x2803	Characteristic Declaration	02:1D:00:26:2A	
0x001D	0x2A26	Firmware Revision String	78:2E:78:2E:78:78:00:00:00:00	0x02
Characteristic Declaration: Software Revision String				
0x001E	0x2803	Characteristic Declaration	02:1F:00:28:2A	
0x001F	0x2A28	Software Revision String	31:2E:30:2E:30:30:00:00:00:00	0x02

Figure 2: BuWizz device services and characteristics

Complete set of services of BuWizz device is shown in Figure 2. Device information service is included due to the requirements of the iOS devices.

BuWizz API

Control data for the motors, device's status data and OTA (Over-The-Air) data need to be transferred via the BLE wireless link. This chapter defines the protocol that is used to construct the packets.

Application commands

The following list of commands is supported by the BuWizz device in the application mode with an active BLE connection.

Device status report

Once enabled by writing 1 to data CCCD, the BuWizz device will periodically send device status report with the approximate frequency of 25 Hz.

<table>
<tr><th>Byte</th><th>Function / value</th></tr>
<tr><td>0 (command)</td><td>0x00</td></tr>
<tr><td>1</td><td>Status flags - bit mapped to the following functions:
<table>
<tr><th>Bit</th><th>Function</th></tr>
<tr><td>7</td><td>unused</td></tr>
<tr><td>6</td><td>USB connection status (1 - cable connected)</td></tr>
<tr><td>5</td><td>Battery charging status (1 - battery is charging, 0 - battery is full or not charging)</td></tr>
<tr><td>3-4</td><td>Battery level status (0 - empty, motors disabled; 1 - low; 2 - medium; 3 - full)</td></tr>
<tr><td>2-1</td><td>unused</td></tr>
<tr><td>0</td><td>error (overcurrent, overtemperature…)</td></tr>
</table></td></tr>
<tr><td>2</td><td>Battery voltage (3 V + value * 0,01 V) - range 3,00 V - 4,27 V
Example: 0x00 => 3,00 V, 0x7F => 4,27 V</td></tr>
<tr><td>3</td><td>Output (motor) voltage (4 V + value * 0,05 V) - range 4,00 V - 16,75 V</td></tr>
<tr><td>4-7</td><td>Motor currents, 8-bit value for each motor output (value * 0,033 A) - range 0 - 8,5 A</td></tr>
<tr><td>8</td><td>Current power level
<table>
<tr><th>Value</th><th>Function</th></tr>
<tr><td>0</td><td>Power is disabled (default value after start or BLE disconnect)</td></tr>
<tr><td>1</td><td>Slow</td></tr>
<tr><td>2</td><td>Normal</td></tr>
<tr><td>3</td><td>Fast</td></tr>
<tr><td>4</td><td>LDCRS</td></tr>
</table></td></tr>
<tr><td>9</td><td>Microcontroller temperature (value in °C)</td></tr>
<tr><td>10-11</td><td>Accelerometer x-axis value (left-aligned 12-bit signed value, 12 mg/digit)</td></tr>
<tr><td>12-13</td><td>Accelerometer y-axis value (left-aligned 12-bit signed value, 12 mg/digit)</td></tr>
<tr><td>14-15</td><td>Accelerometer z-axis value (left-aligned 12-bit signed value, 12 mg/digit)</td></tr>
</table>

0x10 Set motor data

Transfers motor data to the device.

Byte	Function / value
0 (command)	0x10 - Set motor data
1-4	Motor data (signed 8-bit value for each motor output) 0x81 (-127): Full backwards 0x00 (0): Stop 0x7F (127): Full forwards
5	Brake flags - bit mapped to bits 3-0 (1 bit per each motor, bit 0 for first motor, bit 3 for the last) If brake flag is set for a target motor, slow-decay control mode will be used (shortcircuiting the motor armature over the inactive phase). If flag is not set, the corresponding motor will be controlled in fast-decay control method (coasting the motor during the inactive phase).

No response is generated

0x11 Set power level

Changes the power level.

Byte	Function / value
0 (command)	0x11 - Set power level
1	Power level index

Value	Function
0	Power is disabled (default value after start or BLE disconnect)
1	Slow
2	Normal
3	Fast
4	LDCRS

No response is sent.

0x20 Set current limits

Setups current limits for the motors. By default, the current limits are configured to default value of 750 mA on every BLE connection start.

If the motor current is above the current limit, the PWM duty cycle of the affected motor is reduced until the current falls below the limit. PWM duty cycle will increase if the motor load is reduced.

Byte	Function / value
0 (command)	0x20 - Set current limit data
1-4	Current limit in steps of 33 mA

No response is generated