THINGSLOG

Command line guide

Low power mobile data logger

ThingsLog LPMDL-1103 LoRa© / LoRaWAN© Data logger

11 Magnaurska shkola str., fl. 3, office 315 (Hightech Business Park, BIC IZOT),

1784, Sofia, Bulgaria

Phone (+359) 875 32 80 70• Email: info@thingslog.com

Access to CLI console port

To access the command line interface you have to connect with an USBtoTTL cable to the logger console (UART) port.

To access the console port on v.3.0 loggers you have to take out the logger from the enclosure turn it upside down and connect to the 3 pins on the other side of the PCB.

The inputs are GND, TX and RX as it is shown on the picture.

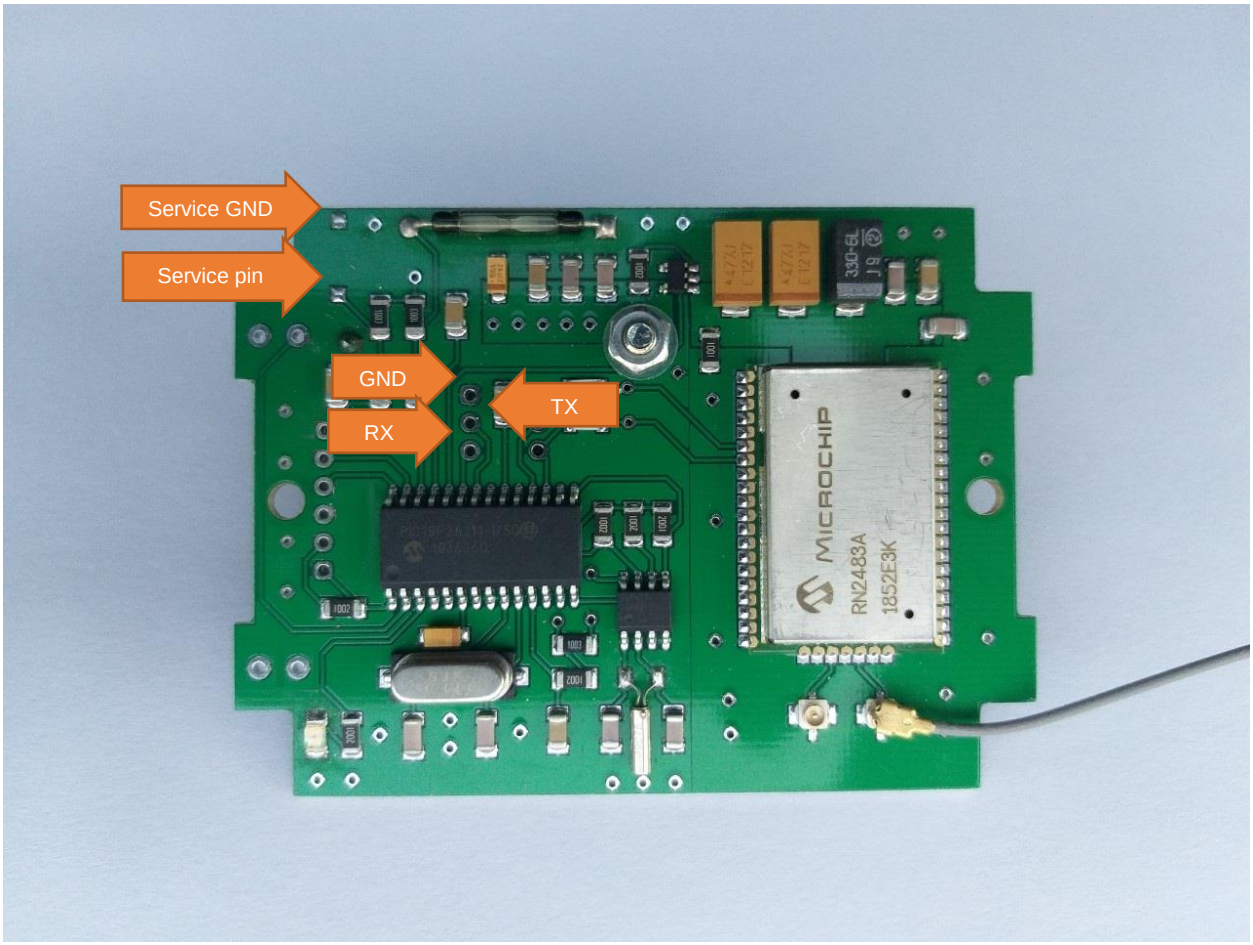

Figure 1 UART & service mode pins in 4 channel data loggers (v.3.0)

To put the device into service mode you need to short circuit the two pins close to the reed switch, see on the picture the “Service pin” and “Service GND”.

If you are using ThingsLog UART cable the color scheme is the following:

Black	GND
White	RX
Green	TX
Red	5V (not used)

Note: RX on the cable must be connected to TX on the board and RX on the cable must be connected with TX on the board.

CLI commands

There are different commands that can be used to configure the LoRa device when it is in service mode. In order to do this configuration it is necessary to put device in service mode using the service button.

Enter in base service mode

1. Connect to the debug UART port using baud rate 9600
2. Press the service button
3. Verify that the following message appear in UART console: "service # ". This indicates that the device is in the base service mode

Example

```
service #
```

Enter in lora_service mode

1. Execute `extension` command. This will enter in device (LoRA, GPRS, NBIoT) specific sub-menu
2. Verify " lora_service # " prompt appears.

Example

```
service # extension
lora_service #
```

Exit the lora_service mode and enter in base service mode

1. Verify you are in lora_service mode - the prompt is "lora_service # "
2. Execute `exit` command to exit "lora_service" mode and to enter the base service mode
3. Verify "service # " prompt for the base service mode appear

Example

```
lora_service # exit
service #
```

Set configuration application

1. Verify you are in lora_service mode - the prompt is "lora_service # "
2. Execute command `set_app_config`

3. Verify " appkey=" prompt appear
4. Enter the new app key
5. Verify "appeui=" prompt appear
6. Enter the new appeui and press enter
7. Verify “deveui=” prompt appear
8. Enter the new deveui key
9. Verify "done" appear

Example:

```
lora_service # extension

set_app_config

lora_service # set_app_config

appkey=AAAAAAAAAAAAAAAAA

appeui=BBBBBBBBBBBBB

deveui=XXXXXXXXXXXXX
```

Get collecting config application

1. Verify you are in lora_service mode - the prompt is "lora_service # "
2. Execute command `get_app_config`
3. Verify "`appkey` = `AAAAAAAAAAAAAAAAA`" appears
4. Verify "`appeui` = `BBBBBBBBBBBBB`" appears
5. Verify "`deveui` = `XXXXXXXXXXXXX`" appears
6. Verify that done appears in the end

Example:

```
lora_service # get_app_config

appkey= AAAAAAAAAAAAAAAAA

appeui= BBBBBBBBBBBBB

deveui= XXXXXXXXXXXXX

done
```

Set Collecting application

1. Execute command `set_app_collect`
2. Verify " appskey=" prompt appears

3. Enter the new APPKEY and press enter
4. Verify “nwkskey=” prompt appears
5. Enter the new NWKKEY and press enter
6. Verify that the “devaddr=” prompt appears
7. Enter the devaddr and press enter
8. Verify "done" appear

Example:

```
lora_service # set_app_collect
appskey=AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
nwkskey=BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
devaddr=12345468
done
```

Get collecting app collect

1. Execute command `get_app_collect`
2. Verify "appkey= AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" appear
3. Verify “nwkskey= BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB” appear
4. Verify “devaddr=12345468” appear
5. Verify “done” appear

Example:

```
lora_service # get_app_collect[\n]

appskey= AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

nwkskey= BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

devaddr=12345468

done
```

Store configuration

1. Verify you are in base service mode - the prompt is "service # "
2. Execute "store" command to save the configurations
3. Verify "service # " prompt appear

Example:

```
service # store
service #
```

Load configuration

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `load` command to load the configurations

3. Verify "service # " prompt appear

Example:

```
service # load
service #
```

Get device number

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `number` command to get the device number
3. Verify "number=00000001" hex device number appear

Example:

```
service # number
number=00000001
service #
```

Set device number

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `set_n` command to set the device number
3. Verify "number=" prompt appear
4. Enter the new device number 0x + (8 hex digits) and press enter
5. Verify that "done" appears

Example:

```
service # number
number=0x00000001
Done
service #
```

Get firmware version

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `version` command to get the firmware version
3. Verify "version=<some_version>" appear

Example:

```
service # version
version=0x0604bbc1
service #
```

Get date

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `date` command to get the current date

3. Verify "date=<some_date>" appear

Example:

```
service # date
date=2017-04-01 12:40:00
service #
```

Set date

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `set_d` command to set the device number
3. Verify "date=" prompt appear
4. Enter the new date in format YYYY-MM-dd HH:mm:ss
5. Verify "done" appear

Example:

```
service # set_d
Date=2017-04-01 12:40:00
Done
service #
```

Get current initial config

1. Verify you are in base service mode - the prompt is "service # "
2. Get current config by executing `configure`.

Example:

```
*** Config ***
2018-09-03 12:03:04
-----------
digits1=8
digits2=8
count_init1=1
count_init2=1
record_period=0
every=1
counts_threshold=300
sensors=01: CNT1
-----------
```

Description of configuration parameters

Name	Default Value	Possible values	Description
digits1	8	• Min value: 1 • Max value: 9	Total number of countable digits on the display of the meter for the first pulse input

digits2	8	• Min value: 1 • Max value: 9	Total number of countable digits on the display of the meter for the second pulse input
count_init1	1	Max value depends on the digits1 value	The value of the meter display of the first pulse input
count_init2	1	Max value depends on the digits1 value	The value of the meter display of the second pulse input
record_period	0	• 0 – MINUTES • 1 – HOURS	The value of the record period 0 mean minutes
every	1	• Min value: 1 • Max value: 127	The value in record period (1 - every 1 minute
counts_threshold	300	• Min value: 1 • Max value: 256	How many counters to keep in memory
sensors	01	Mask of 4 bits • 0x000001 – Pulse sensor 1 • 0x000010 – Pulse sensor 2 • 0x000100 – Current sensor 1 • 0x001000 – Current sensor 2 • 0x010000 – On/Off sensor 1 • 0x100000 – On/Off sensor 2	Which input is active, 01 means the first pulse input

Set initial config

1. Verify you are in base service mode - the prompt is "service # "
2. Execute `set_conf` and enter the configuration menu

```
service # set_conf
config # help
commands:
 help
 set_count_init1
 set_count_init2
 set_digits1
 set_digits2
 set_rec_period
 set_every
 set_counts_threshold
 set_sensors
 save
 exit
```

3. Set inputs reading frequency. Execute `set_every`

```
config # set_every
```

```
set_every= 3
```

4. Enable sensor input port

```
config # set_sensors
sensors= 1
```

5. Set counts threshold (the number of counters to be gathered prior transmission attempt). Execute `set_counts_threshold` If the transmission is successful all counters will be deleted from memory. If transmission fails counters will be kept in memory and will be re-transmitted on next attempt. If memory got full the oldest counters will be deleted first.

```
config # set_counts_threshold
counts_threshold= 100
```

6. Set counter value. Has to be set all meaningful digits up to the magnet pointer. Execute `set_count_init1` for the first input and `set_count_init2` for the second.

```
config # set_count_init1
count_init1= 123
config # set_count_init2
count_init2= 234
```

7. Save and apply the configuration. Execute `save`

```
config # save
Save config
Applying config
Config Counters
rec_conf, size=102, rec_size=2, buff_size=206
Alarm enabled
service #
```