Proxidyne 360LR

# Proxidyne 360LR Asset Detector Setup Guide

The Proxidyne 360LR Asset Detector Setup Guide is used to detect nearby assets that transmit iBeacon signals. It can be used to detect nearby iBeacon devices such as iBeacon stickers, tags, lanyards, or any device that transmits an iBeacon signal. The Proxidyne LoRaWAN Asset Detector scans at regular intervals for a predetermined period of time and send any detected iBeacons to the LoRaWAN bridge. The Asset Detector is configured with the LoRaWAN settings and iBeacon settings prior to deployment.

## Setting up the 360LR

The 360LR is configured via a terminal interface. You can use any terminal program such as Serial on macOS or HyperTerm in Windows to connect and configure the unit. The 360LR has a mini USB port that is used for both power and communication.

### Connecting the 360LR to your computer

With a USB Type A to Mini USB cable, connect the USB type A end to the computer and the USB mini end to the 360LR. Set the terminal with the following settings:

| | |
|---|---|
| Baud | 9600 |
| Data Bits | 8 |
| Parity | none |
| Stop Bits | 1 |
| Return Key | CR+LF |

When the return key is pressed in the terminal program, the message “{"error":"json_format"}” should appear. Commands are sent in JSON format so a blank line returns an error.

# LoRaWAN settings

To configure the 360LR to communicate to the LoRaWAN gateway, specific settings must be provided. To configure LoRaWAN settings, put the 360LR into LoRaWAN passthrough mode by entering the following command:

```
{"set_mode":"radio_command"}
```

## LoRaWAN Network Settings

The 360LR will reboot and enter passthrough mode for configuring LoRaWAN settings. When the return key is pressed, the message “invalid_param” will be printed. This means you are successfully in passthrough mode. Enter your LoRaWAN settings, substituting your values as needed below:

```
mac set deveui BE7A00000000027B
mac set nwkskey BAEFE5DB3BB60A0A130128BA20E3D0FC
mac set appskey 2C9150E165073B7E57E8165A67AA999B
mac set appkey 025BFD76C44D190B7EC8239D538F67E5
mac set appeui BE7A00000000027B
mac set devaddr 01DA0B0E
```

## Enabling and Disabling Channels

If you are using the 360LR with a gateway that does not use all 64 channels, you need to disable the channels that are unused. It is common for US gateways that use 8 to channels to use the first 8 channels (0-7) and channel 64. To adjust if a channel is used, use the following command:

```
mac set ch status <channel number> <off|on>
```

See Appendix D for a common settings for the US 8 channel gateway.

## Saving Settings

Once the values are set, enter in the following command to save the settings:

```
mac save
```

## Joining LoRaWAN network

To test the setting, join the LoRaWAN network:

```
mac join abp
```

### Sending Test Data

Send some data:

```
mac tx uncnf 1 3212
mac tx cnf 1 ABAEFF
```

Send data a few times to verify that the gateway is receiving the data reliably. If the 360LR is set to send data on all frequencies and the gateway is set to only receive on a subset of the frequencies, it is possible that data is only received sometimes.

If the radio says that it is pause, resume it:

```
mac resume
```

### Verify Data on LoRaWAN Portal

Log in to your LoRaWAN portal or gateway and verify all data was successfully received. If not, check your settings and try again. Note all commands in passthrough mode are documented in the “RN2903 LoRa Technology Module Command Reference User’s Guide” from Microchip.

Reboot the 360LR back to command mode by entering the following command:

```
<escape key><return key>
```

The 360LR will return to JSON command mode. Enter the following commands to configure the beacon settings:

# Beacon Scan Configuration

## UUID and Major number

All iBeacons transmit 3 values: A UUID, a major value, and a minor value. All 360LR LoRaWAN Asset Detectors filter on specific UUID and Major keys, and report the minor values of any discovered iBeacons. To set the UUID and Major keys, enter the following commands:

```
{"asset":"set","uuid_key":"E2C56DB5DFFB48D2B060D0F5A71096E0"}
{"asset":"set","major_key":"2"}
```

## Scan Interval, duration, and reporting interval

The 360LR Asset Detector will scan bluetooth for the number of seconds defined in scan_duration. Once scanning is complete, any discovered iBeacons minor numbers will be reported to the LoRaWAN gateway. Once the minor numbers have been reported, the 360LR will wait the number of seconds defined in scan_interval. To set scan_interval and scan_duration, enter the following commands

```
{"asset":"set","scan_interval":"10"}
{"asset":"set","scan_duration":"5"}
```

## RSSI Threshold

Only iBeacons that are above a specific threshold will be reported. You can adjust the RSSI threshold by setting the rssi_thresh. The rssi_thresh ranges from -103 to -30. A value of -103 will report any beacons that match the UUID and Major numbers defined, and a value of -30 will only report beacons that are very close to the 360LR Asset Detector. By default, the RSSI threshold is set to -103. You can use a bluetooth scanner to determine the RSSI values at the different distances to correctly set the RSSI threshold. To set the RSSI threshold, enter the following command:

```
{"asset":"set","rssi_thresh":"-103"}
{"asset":"set","rssi_thresh":"-65"}
```

## Reporting

The 360LR will report to the LoRaWAN gateway whenever there is a change in nearby iBeacons that are greater than the RSSI threshold and match the UUID and major number. When the 360LR starts up, it does an initial scan and reports all beacons detected. After the initial scan, any changes between scans are reported. On a regular basis, the current state will be reported to the LoRaWAN gateway. This can be adjusted with the scan_max_skipped value and is the number of scan_intervals that can occur before a full status is reported. To set the scan_max_skipped value, enter in this command:

```
{"asset":"set","scan_max_skipped":"10000"}
```

## Data format

The data that is sent to LoRaWAN gateway is split into groups and one-hot encoded. If there were no changes between scans, no data is reported until the next reporting interval. The 360LR is able to scan for iBeacons that have minor numbers from 0 to 255 and can report the status of all 512 beacons in 8 packets. Each packet contains 10 bytes, and the first two bytes is the group offset followed by the data. All data is binary data encoded as strings.

```
<GROUP OFFSET BYTE 1><GROUP OFFSET BYTE 2><Data Byte 1><Data Byte
2>...<Data Byte 8>
```

To calculate the assets detected, swap the first two bytes and convert to the an integer. This is the group offset. Then convert each data byte to an integer, subtract 1 since it is zero indexed, add in 8 times the byte position and add to the group offset. For example:

```
c0 00 00 01 00 00 00 00 00 00
```

Group offset is 0x00c0->192
The 2nd byte is 0x01, so the value is (8 * 1)+1-1, or 8

The group offset plus the value would be 192 + 8, or 200.  So the iBeacon minor number would be 200.

Peer to Peer Settings

The 360LR can also communicate over the LoRa protocol to another 360LR. The transmitting 360LR must be put into peer asset mode.  The receiving 360LR must be put into scanning mode and will output the discovered asset packets received from any transmitting 360LR.  All 360LR must share the same symmetric keys that are configured prior to deployment.

Set up the scanning 360LR, set a 16 byte session key (each 360LR should have the exact same session key:

```
{"set_sesskey":"517765727479756 96f70617364666768"}
```

NOTE: DO NOT USE THE SESSION KEY IN THE EXAMPLE ABOVE.  GENERATE YOUR OWN.

The 360LR is in scanning mode by default, but if the 360LR is in a different mode, set scanning mode by entering the command below:

{“set_mode":"scanner"}

On all the 360LR that are scanning for beacons, set a session key and enter into peer asset mode:

```
{"set_sesskey":"517765727479756 96f70617364666768"}
{"set_mode":"peer_asset"}
```

The scanner 360LR should now start receiving the assets scanned from each of the peer asset scanners. The output is JSON with the asset group and the serial number of the the peer asset 360LR.

## Appendix A: Script for decoding data

In Python, converting c00101000F0000000001 gives [448.0, 464.0, 465.0, 466.0, 467.0, 504.0]:

```
import math
def bits(n):
    while n:
        b = n & (~n+1)
        yield b
        n ^= b

input_string="c00101000F0000000001"
group_string1=input_string[0:2]
group_string2=input_string[2:4]
group_string3=group_string2+group_string1
group_int=int(group_string3,16)
data_string=input_string[4:]
final_array=[]
for n in range(0,16,2):
    curr_data=data_string[n:n+2]
    print curr_data
    curr_int=int(curr_data,16)
    for b in bits(curr_int):
        pos=math.log(b,2)+8*n/2
        final=pos+group_int
        final_array.append(final)
output={"assets":final_array}
print output
```

## Appendix B: 360LR Command Reference

Read commands

| Input | Response | Description |
|---|---|---|
| {"read":"info"} | JSON formatted configuration values | Dump of configuration settings. |
| {"read":"serial_id"} | {"serial_id":"0004a30b001f88c9"} | Returns the UID of the device. |
| {"read":"fw_version"} | {"version":"1.0.0_RC1"} | Returns the firmware version number. |
| {"read":"api_version"} | {"version":"3.0.0"} | Returns the JSON API version number. |
| {"read":"node"} | {"node":"255"} | Returns the node number. |
| {"read":"lane"} | {"lane":"7"} | Returns the lane number. |
| {"read":"mode"} | `{"mode":"scanner" | "peer_asset" |
| {"read":"radio_config"} | `{"radio_config":"mode_1" | "mode_2" |
| {"read":"lora_sesskey"} | {"sesskey":"51776572747975696f70617364666768"} | Returns the lora peer network key. |

## Set commands

| Input | Response | Description |
|---|---|---|
| {"set_sesskey":"51776572747975696f70617364666768"} | {"sesskey":"51776572747975696f70617364666768"} | Sets session key for lora peer network |
| {"set_node":"3"} | {"set_node":"success"} | Sets the node number of the device |
| {"set_lane":"9"} | {"set_lane":"success" | Sets the lane number of the device |

| | | |
|---|---|---|
| {"set_mode":"scanner"} | Device resets on success for changes to take affect | Changes the mode of the device |
| {"set_mode":"radio_command"} | Device resets on success for changes to take affect | Changes the mode of the device |
| {"set_mode":"peer_asset"} | Device resets on success for changes to take affect | Changes the mode of the device |
| {"set_mode":"lorawan_asset"} | Device resets on success for changes to take affect | Changes the mode of the device |
| `{"set_radio_config":"mode_1" | "mode_2" | "mode_3" |
| {"set_radio_burst_number":"3"} | {"set_radio_burst_number":"done"} | Sets the number of times the LoRa radio repeats an event packet (1,255) |
| {"asset":"set","uuid_key":"E2C56DB5DFFB48D2B060D0F5A71096E0"} | {"set_asset_uuid_key":"success"} | Sets the asset UUID key to determine which asset tags are monitored |
| {"asset":"set","major_key":"1"} | {"set_asset_major_key":"success"} | Sets the asset MAJOR key to determine which asset tags are monitored |
| {"asset":"set","scan_interval":"15"} | {"set_asset_scan_interval":"success"} | Sets the time between asset scans. Units of seconds, range is 1 to 65535. |
| {"asset":"set","scan_duration":"3"} | {"set_asset_scan_duration":"success"} | Sets the time duration of an asset scan. Units of seconds, range is 2 to 65535. |
| {"asset":"set","scan_max_skipped":"10"} | {"scan_max_skipped":"success"} | Sets the number of consecutive scan events that can go unreported if no changes are detected. Range is 0 to 1000. |

| | | |
|---|---|---|
| {"asset":"set","rssi_thresh":"-65"} | {"set_asset_rssi_thresh":"success"} | Sets the asset tag RSSI detection limit. Packets below this limit are ignored. |

## Appendix C: LoRaWAN Radio Commands

### Get and set the globally unique device identifier for the module

mac set deveui <DEVEUI>
mac get deveui

### Get and set the Network Session Key

mac set nwkskey <NWKSKEY>
mac get nwkskey

### Get and set the Application Session Key

mac set appskey <APPSKEY>
mac get appskey

### Get and set the Application Key

mac set appkey <APPKEY>
mac get appkey

### Get and set the Application Identifier

mac set appeui <APPEUI>
mac get appeui

### Get and set the Device Address

mac set devaddr <DEVICE ADDRESS>

### Store settings in NVRAM (persistent across reboots)

mac save

### Join using Activation by Personalization

mac join abp

Send data
mac tx cnf <SLOT> <BINARY DATA>
Example: mac tx cnf 4 101012

### Reset settings

mac reset

## Appendix D: Common 8 Channel Gateway Configuration

```
mac set ch status 0 on
mac set ch status 1 on
mac set ch status 2 on
mac set ch status 3 on
mac set ch status 4 on
mac set ch status 5 on
mac set ch status 6 on
mac set ch status 7 on
mac set ch status 8 off
mac set ch status 9 off
mac set ch status 10 off
mac set ch status 11 off
mac set ch status 12 off
mac set ch status 13 off
mac set ch status 14 off
mac set ch status 15 off
mac set ch status 16 off
mac set ch status 17 off
mac set ch status 18 off
mac set ch status 19 off
mac set ch status 20 off
mac set ch status 21 off
mac set ch status 22 off
mac set ch status 23 off
mac set ch status 24 off
mac set ch status 25 off
mac set ch status 26 off
mac set ch status 27 off
mac set ch status 28 off
mac set ch status 29 off
mac set ch status 30 off
mac set ch status 31 off
mac set ch status 32 off
mac set ch status 33 off
mac set ch status 34 off
mac set ch status 35 off
mac set ch status 36 off
mac set ch status 37 off
mac set ch status 38 off
mac set ch status 39 off
mac set ch status 40 off
mac set ch status 41 off
mac set ch status 42 off
mac set ch status 43 off
mac set ch status 44 off
```

```
mac set ch status 45 off
mac set ch status 46 off
mac set ch status 47 off
mac set ch status 48 off
mac set ch status 49 off
mac set ch status 50 off
mac set ch status 51 off
mac set ch status 52 off
mac set ch status 53 off
mac set ch status 54 off
mac set ch status 55 off
mac set ch status 56 off
mac set ch status 57 off
mac set ch status 58 off
mac set ch status 59 off
mac set ch status 60 off
mac set ch status 61 off
mac set ch status 62 off
mac set ch status 63 off
mac set ch status 64 on
mac set ch status 65 off
mac set ch status 66 off
mac set ch status 67 off
mac set ch status 68 off
mac set ch status 69 off
mac set ch status 70 off
mac set ch status 71 off
```