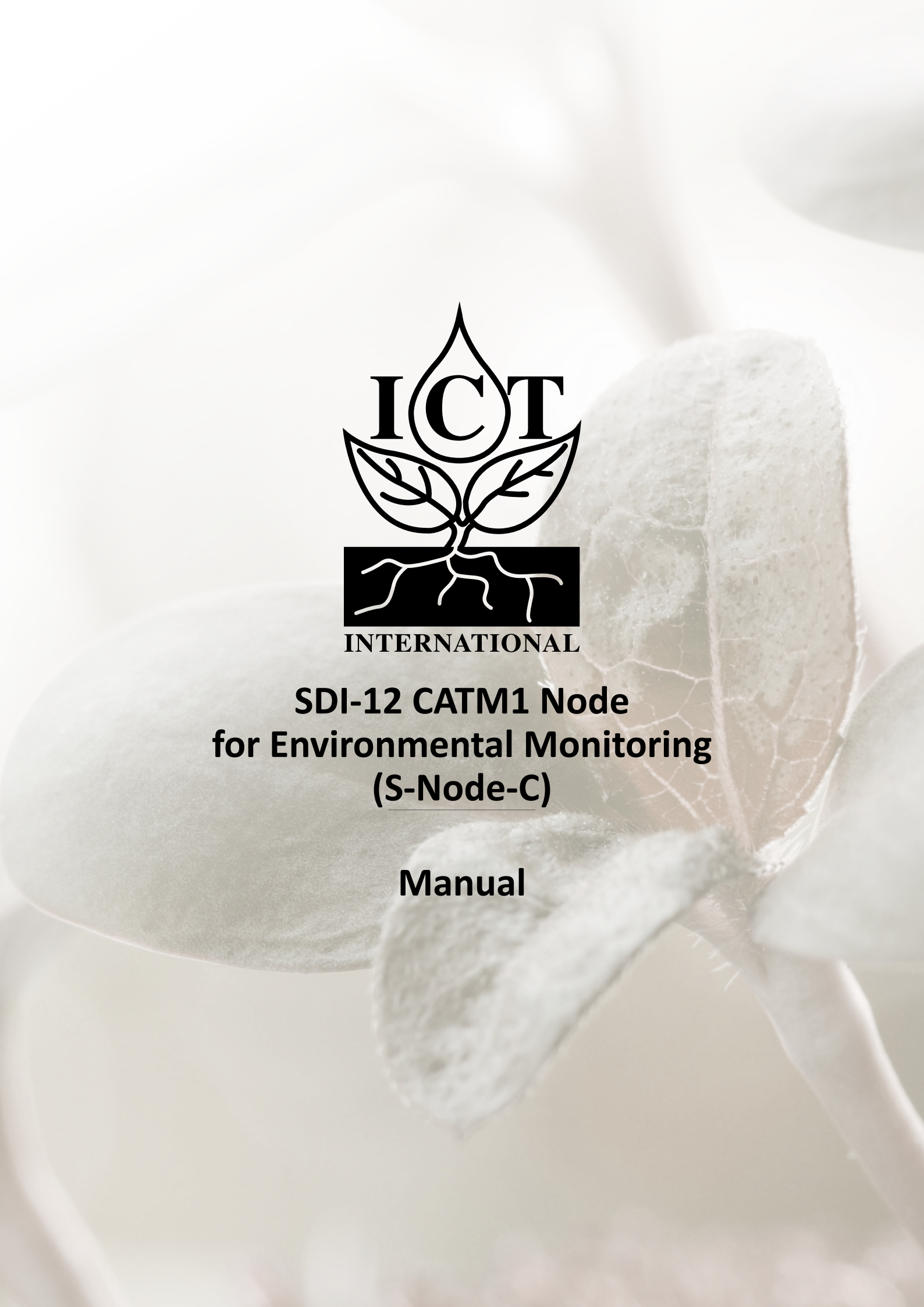


ICT INTERNATIONAL

SDI-12 CATM1 Node for Environmental Monitoring (S-Node-C)

Manual

Contents

1. Introduction

The ICT International S-Node is a low power SDI-12 LoRaWAN data transmission device.

The device is powered by a lithium-ion battery pack (6.7Ah or 13.4Ah) and is charged by external 12-24V DC input – typically a 10W or 20W solar panel.

Long-range low-power communications are achieved through use of LoRaWAN™ networks via an onboard LoRa® radio.

It is designed to easily connect to 4 x SDI-12 sensors. The Node can be customised to accommodate more sensors upon request.

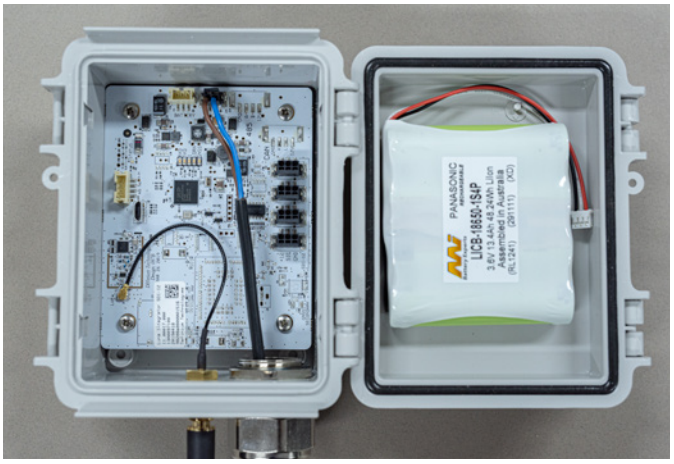


Figure 1. Inside The S-Node Box - The S-Node Board

Figure 2. The S-Node Box - With Antenna

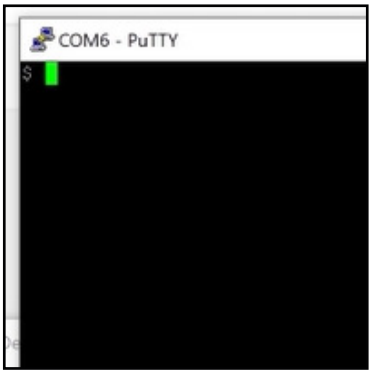
2. Device Operation

2.1 SDI-12 Configuration

The S-Node's SDI-12 communications can be customised to allow for interfacing with any SDI-12 sensor. See section 3.5 for a detailed description of the configuration interface of the device.

Adding or modifying an SDI-12 command for taking a measurement from a sensor uses the `sdi12 add` command. Commands can be disabled/enabled using the `sdi12 activate` command – by default, commands are enabled when they are added or modified. Other commands such as diagnostic and calibration requests can be sent directly to the sensor using the `sdi12 send` command.

For more information on SDI-12 related commands, see section 3.5.



Configuration Program Example

3. Device Configuration

The ICT International S-Node is configured (i.e. identified and authenticated) over USB serial console using a terminal or terminal emulator. It is compatible with Windows 10, Mac OS and Linux.

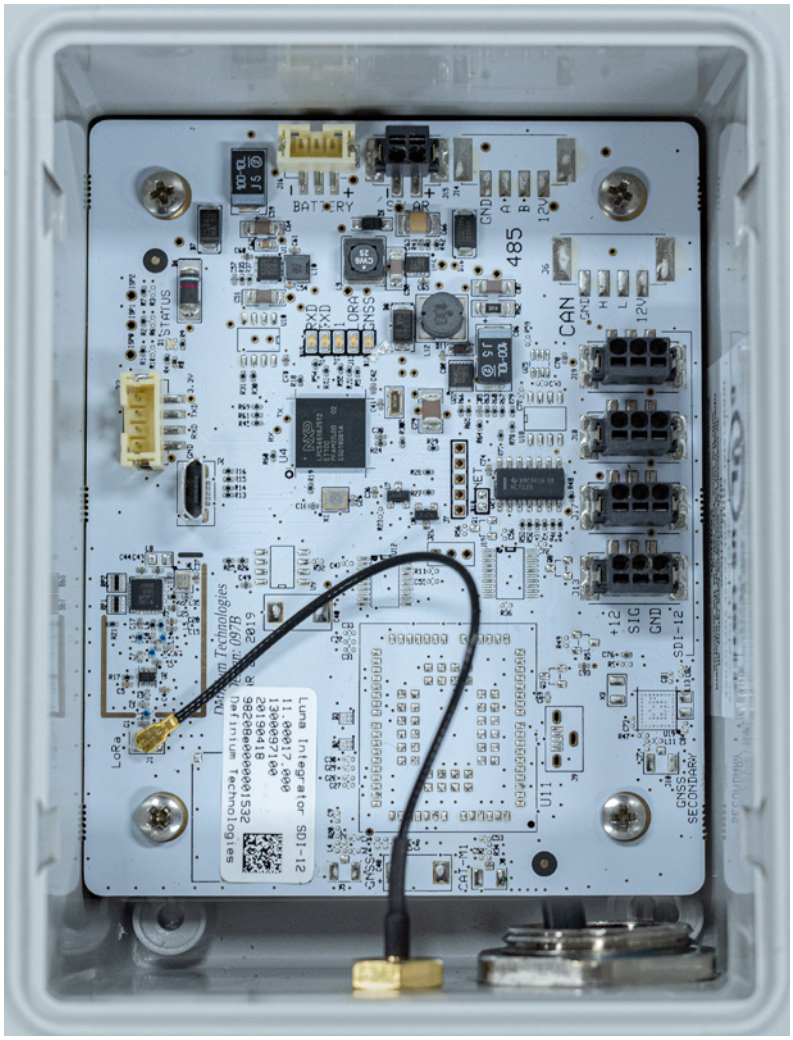
One terminal emulator we recommend is Putty, which can be downloaded from https://www.putty.org/.

All commands are entered as ASCII text and will return any response as ASCII text.

Connecting a S-Node to a computer via the Micro USB port will provide a serial port for configuration.

Recommended settings are as follows:

- Baud Rate: 115200 baud
- Bits: 8
- Parity: None
- Stop Bits: 1
- Flow Control: Disabled.



3.1 Configuration Commands

These commands are entered into a terminal or terminal emulator such as Putty, https://www.putty.org/, to action several types of commands to the S-Node.

3.1.1 Firmware Version – version

Command Input: `version`

Compatible: Device Firmware Versions > 1.2

Command Description: Returns information about the device firmware version and configured frequency.

Command Type	Syntax Used	Response Type	Example Result of Command
Get	`version`	<String>	`Definium Technologies Pty Ltd LoRaWAN Class-A Sensor 235c0e89-dirty Luna Station 4000096001-"AS923"`

3.1.2 Save Current Device Configuration – save

Command Input: `save`

Compatible: All Device Firmware Versions

Command Description: Saves the running configuration to permanent storage.

Command Type	Syntax Used	Response Type	Example Result of Command
Action	`save`	Saved config	`Saved config`

3.1.3 Reload Saved Configuration – load

Command Input: `load`

Compatible: All Device Firmware Versions

Command Description: Saves the running configuration to permanent storage.

Command Type	Syntax Used	Response Type	Example Result of Command
Action	`load`	Loaded config	`Loaded config`

3.1.4 Reset to Factory Defaults – config reset

Command Input: `config reset`

Compatible: Device Firmware Versions > 1.2

Command Description: Resets the running configuration factory defaults.

Command Type	Syntax Used	Response Type	Example Result of Command
Get	`config reset`	Reset app config to defaults	`Reset app config to defaults`

3.2 General LtE™ Configuration - Modem Settings

These commands are entered into a terminal or terminal emulator such as Putty, https://www.putty.org/, to action several types of commands to the MFR-Node.

3.2.1 Modem Access Point (APN) – modem apn

Command Input: `modem apn`

Command Description: Sets the modem access point (APN) specific to the SIM card provider.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`modem apn`	`modem apn`	<string>	`modem apn telstra.m2m`
Set	`modem apn <string>`	`modem apn telstra.m2m`	<string>	`modem apn telstra.m2m`

3.2.2 APN Username – modem usr

Command Input: `modem usr`

Command Description: Set the Modem username if required by the SIM card provider you are using.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`modem usr`	`modem usr`	<string>	`Modem username:`
Set	`modem usr <string>`	`modem usr yourMSID@uscc.net`	<string>	`Modem username: yourMSID@uscc.net`

3.2.3 APN Password – modem pwd

Command Input: `modem pwd`
Command Description: Set the Modem password if required by the SIM card provider you are using.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`modem pwd`	`modem pwd`	<string>	`Modem password:;`
Set	`modem pwd <string>`	`modem pwd your MSID`	<string>	`Modem password: your MSID`

3.2.4 APN Authentication – modem auth

Command Input: `modem auth`
Command Description: Enable Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP).

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`modem auth`	`modem auth`	<status>	`Modem auth method: 0`
Set	`modem auth <bool>`	`modem auth 1`	<status>	`Modem auth method: 1`

Parameter	Type	Description
<bool>	Number	Password Authentication Protocol (PAP) and/or Challenge Handshake Authentication Protocol (CHAP) status to be set: `0`: none `1`: pap `2`: chap `3`: pap or chap

3.3 General MQTT Configuration - Settings

These commands are entered into a terminal or terminal emulator such as Putty, https://www.putty.org/, to action several types of commands to the MFR-Node.

3.3.1 MQTT Hostname – mqtt host

Command Input: `mqtt host`
Command Description: Enter in the hostname of your MQTT broker.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt host`	`mqtt host`	<string>	`MQTT Hostname:`
Set	`mqtt host <string>`	`mqtt host ictcatm1.com`	<string>	`MQTT Hostname: ictcatm1.com`

3.3.2 MQTT Port – mqtt port

Command Input: `mqtt port`
Command Description: Set network port to connect to (optional, 0 = default based on TLS setting).

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt port`	`mqtt port`	<int>	`mqtt port 0`
Set	`mqtt port <int>`	`mqtt port 1883`	<int>	`mqtt port 1883`

3.3.3 MQTT Username – mqtt user

Command Input: `mqtt user`
Command Description: The username to use with your MQTT broker.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt user`	`mqtt user`	<string>	`MQTT User:`
Set	`mqtt user <string>`	`mqtt user ictinternational`	<string>	`MQTT User: ictinternational`

3.3.4 MQTT Password – mqtt pass

Command Input: `mqtt pass`
Command Description: The corresponding password for the username to use with your MQTT broker.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt pass`	`mqtt pass`	<string>	`MQTT Password:`
Set	`mqtt pass <string>`	`mqtt pass 1234`	<string>	`MQTT Password: 1234`

3.3.5 MQTT Topic – mqtt topic

Command Input: `mqtt topic`
Command Description: UTF-8 string that the broker uses to filter messages for each connected client. The topic consists of one or more topic levels. Each topic level is separated by a forward slash (topic level separator).

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt topic`	`mqtt topic`	<string>	`MQTT Topic:`
Set	`mqtt topic <string>`	`mqtt topic`	<string>	`MQTT Topic: ict/data/mfr/1K704`

3.3.6 MQTT QoS Level – mqtt qos

Command Input: `mqtt user`
Command Description: Quality of Service (QoS) level is an agreement between the sender of a message and the rceiver of a message that defines the guarantee of delivery for a specific message. There are 2 QoS levels: At most once (0); At least once (1).

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt qos`	`mqtt qos`	<bool>	`0`
Set	`mqtt qos <bool>`	`mqtt qos 1`	<bool>	`1`

3.3.7 Use MQTT over TLS (MQTTS) – mqtt tls

Command Input: `mqtt tls`
Command Description: Use MQTT over TLS (MQTTS): disabled (0); enabled (1).

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt tls`	`mqtt tls`	<bool>	`disabled`
Set	`mqtt tls <bool>`	`mqtt tls 1`	<bool>	`enabled`

3.3.8 Report in JSON rather than CSV – mqtt json

Command Input: `mqtt json`
Command Description: Send data in json format rather than csv format (default): disabled (0); enabled (1).

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt json`	`mqtt json`	<bool>	`disabled`
Set	`mqtt json <bool>`	`mqtt json 1`	<bool>	`enabled`

3.3.9 MQTT Connection is for Azure IoT Hub– mqtt azure

Command Input: `mqtt azure`
Command Description: Azure Mode settings reporting data in .json, and will expect a Azure string as per section ?.?: Azure Configuration String..

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt azure`	`mqtt azure`	<status>	`disabled`
Set	`mqtt azure <bool>`	`mqtt azure`	<status>	`mqtt azure 1 enabled`

Parameter	Type	Description
<bool>	Number	mqtt azure to be set: `0`: Unjoined `1`: Joined
<status>	String	mqtt azure status string. Options: `enabled`: mqtt azure enabled `disabled`: mqtt azure disabled

3.3.10 MQTT Azure Configuration String – mqtt azureconn

Command Input: `mqtt azureconn`
Command Description: To configure the node to communicate to the Azure Hub it must contain in a string organised below to include the following: HostName, Device ID and Shared Access Key, please note the Shared Access Key will end in an equals (=) sign:.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command

Get	`mqtt azureconn`	`mqtt azureconn`	<status>	`Azure IoT Hub Hostname:`
				`Azure IoT Hub Device ID:`
				`Azure IoT Hub Device Key:`

Set	`mqtt azureconn <string>`	`see parameter description below`	<status>	`Azure IoT Hub Hostname: ictcatm1.com`
				`Azure IoT Hub Device ID:`
				`Azure IoT Hub Device Key:`

Parameter	Type	Description
<status>	String	Example Layout of the Command String to enter will be: mqtt azureconn HostName=xxxxxxxx.azure-devices.net; DeviceId=yyyyyyyyyyyyy;SharedAccessKey=Base64EncodedAccessKeyGoesHere=

3.3.11 Enable MQTT Last Will Testament (LWT) – mqtt will enable

Command Input: `mqtt will enable`
Command Description: Enable a LWT message to be sent to other clients if device doesn't connect to broker.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt will enable`	`mqtt will enable`	<status>	`disabled`
Set	`mqtt will enable <bool>`	`mqtt will enable 1`	<status>	`enabled`

Parameter	Type	Description
<bool>	Number	Enable a LWT message to be sent to other clients if device doesn't connect to broker: `0`: disable `1`: enable

3.3.12 Enable MQTT Last Will Testament (LWT) Topic – mqtt will topic

Command Input: `mqtt will topic`
Command Description: The LWT topic that the broker will publish to.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt will topic`	`mqtt will topic`	<string>	`MQTT LWT Topic:`
Set	`mqtt will topic <string>`	`mqtt will topic device mlna1902/ status`	<string>	`MQTT LWT Topic: device mlna1902/status`

3.3.13 Enable MQTT Last Will Testament (LWT) Message– mqtt will msg

Command Input: `mqtt will msg`
Command Description: LWT Message posted to the designated LWT topic.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`mqtt will msg`	`mqtt will msg`	<string>	`MQTT LWT Message:`
Set	`mqtt will msg <string>`	`mqtt will msg device mlna1902 offline`	<string>	`MQTT LWT Message: device mlna1902 offline`

3.4 Logging and Data Transmission

Commands for managing the device's state transition timings.

3.4.1 Logging interval / Period between reports – report period

Command Input: `report period`

Compatible: Device Firmware Versions > 1.2

Command Description: Initial wait time on failed communications before retrying. Doubles each failure until it reaches backoff max.

Command Type	Syntax Used	Example Command	Response Type	Example Result of Command
Get	`report period`	`report period`	<string>	`Report Period: 900 sec Current: 12 Last: 0 Next:0`
Set	`report period <period>`	`report period 600`	<string>	`report period 600 Report Period: 600 sec Current: 26 Last: 0 Next:0`

Parameter	Type	Description
<period>	Number	Time in seconds between reports.
<time>	Number	Current device timestamp
<last>	Number	Timestamp that state last triggered at
<next>	Number	Timestamp of next state trigger

3.5 SDI-12

3.5.1 SDI-12 add command – sdi12 add

Command Input: `sdi12 add`

Compatible: Device Firmware Versions

Command Description: Add or modify SDI-12 command in slot. Available SDI-12 commands are: Measurement (M!), Concurrent (C!) and Result (R!).

Measurement and Concurrent must use the Measure type command. Result must use the Data command

Examples:

```
sdi12 add 0 M 0C0! 3 3 0D0! 111
```

Use SDI-12 command id/slot 0 to send a Concurrent measurement command to SDI-12 sensor address 0, delay 3 seconds, value length 3, send a data command to sensor address 0, get 3 parameters.

```
sdi12 add 1 D 4R0! 1101
```

Use SDI-12 command slot 1 to send a Result command to sdi-12 sensor address 4, returning 4 parameters and discarding parameter 3.

Command Type	Syntax Used	Response Type
Measure	`sdi12 add <id> M <command> <delay length> <value length> <result command> <mask>`	<string>
Data	`sdi12 add <id> D <command> <mask>`	<string>

Parameter	Type	Description
<id>	Number	0-9 Slot ID to add or modify.
<command>	String	SDI-12 command to execute.
<delay length>	Number	0-9 Number of digits returned as delay by Measure command.
<values length>	Number	0-9 Number of digits returned as reading by Measure command.
<mask>	Binary	Mask for selecting data from Read command, length of the mask can be equal to the number of readings. E.g: the mask to select the first and fourth reading for an 8 reading data command is: 10010000

3.5.2 Send SDI-12 Command – sdi12 send

Command Input: `sdi12 send`

Compatible: All Device Firmware Versions
Command Description: Send an SDI-12 command. Can be used for identifying sensors on a bus, configuring SDI-12 addresses, or any other sensor specific functions.

Type	Syntax Used	Example Command	Response Type	Example Result of Command
Action	`sdi12 send <command>`	`sdi12 send`	<response>	`sdi12 send 0A1! 1`

Parameter	Type	Description
<command>	String	SDI-12 command to execute.
<response>	String	Response from SDI-12 command, command dependant

Commands	Description
`aI!`	Sends sensor identification request for the sensor at address `a`
`aAb!`	Change sensor address from `a` to `b`
`?!`	Query sensor address, can only be done when a single sensor is connected

3.5.3 Activate/Deactivate SDI-12 Command Slot – sdi12 activate

Command Input: `sdi12 activate`

Compatible: All Device Firmware Versions
Command Description: Enable or disable the sending of a defined SDI-12 command.

Command Type	Syntax Used	Response Type	Example Result of Command
Action	`sdi12 activate <id> <enable>`	<response>	`sdi12 activate 0 1 Success`

Parameter	Type	Description
<id>	Number	0 – 9 slot ID to add or modify
<enable>	Number	Enable (1)/Disable (0)

3.5.4 Delete all SDI-12 Commands – sdi12 remove all commands

Command Input: `sdi12 remove all commands`

Compatible: All Device Firmware Versions
Command Description: Removes all configured SDI-12 command slots.

Type	Syntax Used	Response Type	Example
Action	`sdi12 remove all commands`	<string>	`sdi12 remove all commands` `SDI12 Commands Erased`

3.5.5 Set low battery mode threshold – battery threshold

Command Input: `battery threshold`

Compatible: All Device Firmware Versions

Command Description: Sets the battery threshold, below which the device will enter low power mode and cease regular transmission until the battery has charged above the threshold. By default, this is set to 3.4 volts.

Command Type	Syntax Used	Response Type	Example Result of Command
Get	`battery threshold`	<voltage>	`battery threshold` `3400`
Set	`battery threshold <voltage>`	<voltage>	`battery threshold 3600` `3600`

Parameter	Type	Description
<voltage>	Number	Low power cutoff in mV.

3.5.6 Command List – help

Command Input: `help`

Compatible: All Device Firmware Versions

Command Description: Lists all available commands with brief descriptions of their functions.

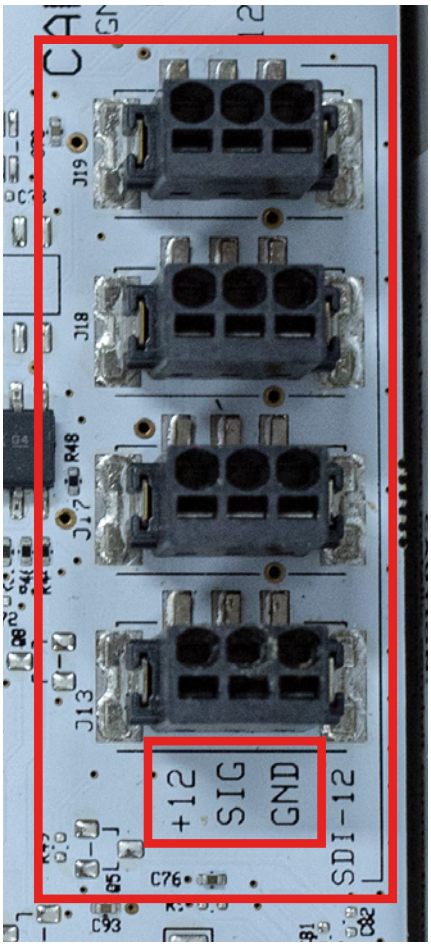
Command Type	Syntax Used	Response Type
Get	`help`	List of commands

4. Connecting SDI-12 Sensors

A connector is available on the right-hand side of the board for each type of sensor. See instructions below for removing and inserting sensor wires to these sockets.

For all four SDI-12 Sensor ports, the left sockets are labelled +12 (Excitation Power), the middle sockets are SIG (SDI-12 Signal) and the right sockets are labelled GND (Ground). Match each wire core to their correct respective sockets accordingly.

Please refer to each Sensor Manufacturers Manual for technical information on wire colouring and identification.



4.1 Connecting and Removing The Wires

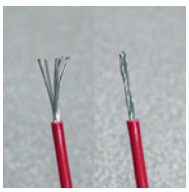
For each device and their sensor wires follow the instructions below. The following steps also apply to wires connecting to an external solar power source. The images below and right also correspond to their respective steps.

4.1.1 Insert and Connect The Wires

1. Prepare the wire - Strip the outer insulation of the inner conductors to reveal the inner conductor exposed wire. If stranded, lightly twist before inserting in (1.a). Make sure the final amount of exposed wire is a maximum length of 7mm (1. b).
2. Use a pair of tweezers or suitable flathead screwdriver to push into the square hole beside the desired socket to temporarily release the spring clamp.
3. Hold the tweezers down and insert the prepared wire (3.a). Push the wire in, down all the way (3.b).
4. Remove the tweezers and the wire should be clamped in.
5. After removing tweezers, do a simple gentle tug test to each wire to be sure it is clamped and secured correctly.

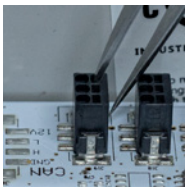
4.1.2 Remove or Reposition Wires

2. Use a pair of tweezers or suitable flathead screwdriver to push into the square hole beside the desired socket to temporarily release the spring clamp.
3. Hold the tweezers down and gently pull out the inner wire to remove it - (3.b) then (3.a). Remove the tweezers. The wire is now free to remove or reposition. To reinsert, follow the order of instructions above in 4.1.1.

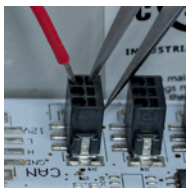


1.a

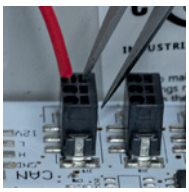
1.b



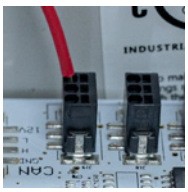
2.



3.a



3.b



4.

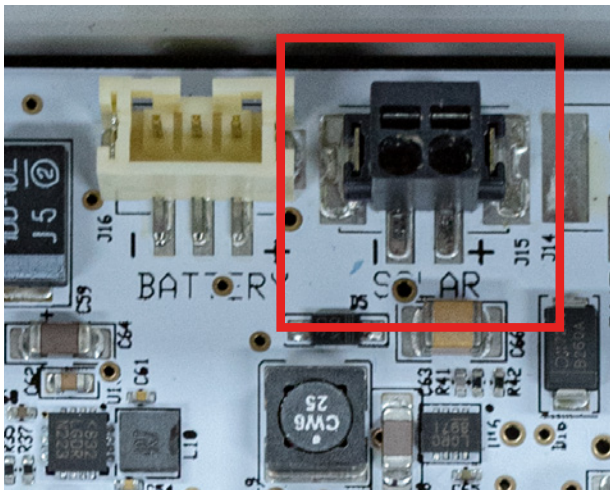
5. External Power

The input for external power is located at the top, on the right of the board, labelled SOLAR.

This input is polarised, please ensure that positive (+) is inserted in the (+) terminal and negative (-) in the (-) terminal. It's advised to use a multi-meter to test the polarity before connecting.

The S-Node has an on-board solar charge controller and can be directly connected to a 12V or 24V solar panel. Alternatively, a 12V to 24V mains DC power supply can be connected for indoor use.

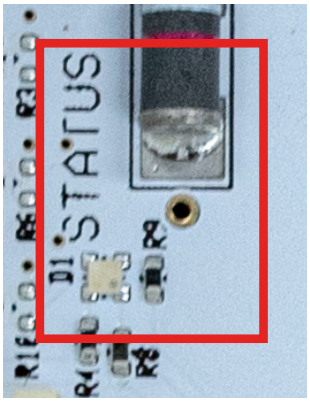
Please ensure that the S-Node's internal battery is plugged in when using solar power.



6. LED Status

The status led is used to indicate the following:

- **LIGHT BLUE**: Joining Network
- **DARK BLUE**: Network Joined / Taking measurement
- **ORANGE**: Transmitting sensor data
- **PURPLE**: Measurement Complete
- **GREEN**: USB Idle
- **RED**: Failed to Join Network



7. MQTT Packet Structure

MQTT packets are sent by the device in JSON or CSV format in the following structure using the example of a multi-parameter weather station below:

- Date & Time <ddmmyyyy hh:mm>
- Battery Mv <BattmV>
- External supply mV <ExtmV>
- Charge State <1>
- SDI12 Slot 0 Result 0 <AverageWindDirection>
- SDI12 Slot 0 Result 1 <AverageWindSpeed>
- SDI12 Slot 0 Result 2 <AirTemperature>
- SDI12 Slot 0 Result 3 <RelativeHumidity>
- SDI12 Slot 0 Result 4 <AtmosphericPressure>
- SDI12 Slot 1 Result 0 <PM2.5><PM10>
- SDI12 Slot 1 Result 1 <Noise>

CSV Format:

<Record Time><slotno><BattmV><ExtmV><AverageWindDirection><AverageWindSpeed>
<AirTemperature><RelativeHumidity><AtmosphericPressure><PM2.5><PM10><Noise

7. Decoder Notes

Please contact ICT International for the S-NODE decoder applicable to your order and suitable for TTN (https://www.thethingsnetwork.org/):

Enabling better global research outcomes in soil, plant & environmental monitoring.

www.ictinternational.com sales@ictinternational.com.au
Ph: +61 2 6772 6770 | 211 Mann St., Armidale, NSW 2350, Australia