TECHNICAL INFORMATION MANUAL

Revision 2 – 22 March 2019

ion R4301P

UHF Long Range Reader with GPRS/WIFI

easy2read

CAENRFID

Visit the Ion R4301P web page, you will find the latest revision of data sheets, manuals, certifications, technical drawings, software and firmware. All you need to start using your reader in a few clicks!

Scope of Manual

The goal of this manual is to provide the basic information to work with the Ion R4301P UHF Long Range Reader with GPRS/WIFI.

Change Document Record

Date	Revision	Changes	Pages
17 Mar 2016	00	Preliminary release	-
04 Jul 2016	01	Added *CE Compliance, CE Declaration of Conformity FCC Grant part B, FCC Grant part C.*	54, 55, 56, 57
		Modified *RoHS EU Directive*	54
22 Mar 2019	02	Added warning in the *SIM card placement* paragraph	44
		Modified *Setting up the GSM module* and *Surfing the Internet* paragraphs	45, 46
		Modified *Development virtual machine image* paragraph.	35
		Modified *CE Compliance* and *CE Declaration of Conformity*	54, 55
		Modified *New and standard API* paragraph	37
		Modified *Development Kit* paragraph	9

Reference Document

[RD1] EPCglobal: EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz – 960 MHz, Version 2.0.1 (April, 2015).

[RD2] G.S.D. s.r.l. - Report Federal Communication Commission (FCC) – R4301P – Ion UHF Long Range Reader. Test report n. FCC-16574B Rev. 00 – 20 April 2014.

[RD3] G.S.D. s.r.l. - Report Federal Communication Commission (FCC) – R4301P – Ion UHF Long Range Reader. Test report n. FCC-16547 Rev. 00 – 20 April 2014

CAEN RFID srl

Via Vetraia, 11 55049 Viareggio (LU) - ITALY
Tel. +39.0584.388.398 Fax +39.0584.388.959
info@caenrfid.com
www.caenrfid.com

© CAEN RFID srl – 2019

Disclaimer

No part of this manual may be reproduced in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of CAEN RFID.

The information contained herein has been carefully checked and is believed to be accurate; however, no responsibility is assumed for inaccuracies. CAEN RFID reserves the right to modify its products specifications without giving any notice; for up to date information please visit www.caenrfid.com.

Federal Communications Commission (FCC) Notice

This device was tested and found to comply with the limits set forth in Part 15 of the FCC Rules. Operation is subject to the following conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received including interference that may cause undesired operation. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment.

This device generates, uses, and can radiate radio frequency energy. If not installed and used in accordance with the instruction manual, the product may cause harmful interference to radio communications. Operation of this product in a residential area is likely to cause harmful interference, in which case, the user is required to correct the interference at their own expense. The authority to operate this product is conditioned by the requirements that no modifications be made to the equipment unless the changes or modifications are expressly approved by CAEN RFID.

Disposal of the product

Do not dispose the product in municipal or household waste. Please check your local regulations for disposal/recycle of electronic products.

Index

Scope of Manual 2
Change Document Record 2
Reference Document 2
Index 4
List of Figures 5
List of Tables 5
1 INTRODUCTION 7
Product Description 7
Installation Notice 8
Development Kit 9
Ordering Options 10
2 GETTING STARTED 11
Introduction 11
Connecting to the Ion R4301P Reader 11
Connecting to the Ion using the Easy Controller Software 11
Connecting to the Ion using the serial port 11
3 EXTERNAL INTERFACE DESCRIPTION 12
Connectors 12
Power supply 12
Ethernet port 13
GPIO ports 14
USB host ports 15
GSM/GPRS antenna port (only in Mod. WR4301PXGPRS) 16
WIFI antenna port (only in Mod. WR4301PXWIFI) 16
RS232 17
RFID antenna ports 17
LEDs 18
4 READER CONFIGURATION 19
Introduction 19
Reader Configuration using the Web Interface 19
Configure Ethernet settings 20
Set Date and Time 21
Reader Configuration using the serial port 22
Configure Ethernet settings 22
Set Date and Time 23
Configure the serial port to communicate with the *caenrfidd* daemon 23
How to restore the R4301P default factory settings 24
5 GPIO 30
Using Ion GPIO Interface 30
6 *CAENRFIDD* DAEMON 31
Introduction 31
Configuring the daemon 31
The system wide configuration file 31
The main configuration file 31
The reader configuration file 32
The CAENRFID Ion 33
7 DEVELOPING APPLICATION 35
Introduction 35
Development virtual machine image 35
Installing the development virtual machine 35
Using the development virtual machine 36
8 GETTING STARTED WITH C 37
Development code installation 37
New and standard API 37
Standard API example 37
New API usage example 39
9 GETTING STARTED WITH JAVA 41
Introduction 41
Setup the development environment 41
Testing the development environment 41
Running Java applications on the Ion reader 42

Optional: using the Eclipse IDE 43
10 USING THE GSM 44
SIM card placement 44
Setting up the GSM module 45
Sending a SMS message 45
C language 45
Surfing the Internet 46
11 UPGRADE PROCEDURE 48
Upgrade the Ion packages 48
Install new Debian software 48
12 TECHNICAL SPECIFICATIONS 50
Technical Specifications Table 50
Reader – Tag Link Profiles 51
Power Supply Connector Electrical Characteristics 51
RFID Antenna Ports Electrical Characteristics 51
GSM/GPRS Antenna Port Electrical Characteristics 51
WIFI Antenna Port Electrical Characteristics 51
GPIO Port Electrical Characteristics 52
Serial Port Electrical Characteristics 52
USB Port Electrical Characteristics 52
13 REGULATORY COMPLIANCE 53
FCC Compliance 53
CE Compliance 54
RoHS EU Directive 54
CE Declaration of Conformity 55
FCC Grant part B 56
FCC Grant part C 57

List of Figures

Fig. 1.1: Ion R4301P UHF Long Range Reader with GPRS/WIFI 7
Fig. 1.2: Technical drawings 8
Fig. 1.3: Ion R4301P - RFID UHF Portal Reader Development Kit 9
Fig. 3.1: Connectors View 12
Fig. 3.2: Plug Cable Connector 13
Fig. 3.3: GPIO Output Configuration Equivalent Schematic 14
Fig. 3.4: GPIO Input Configuration Equivalent Schematic 14
Fig. 3.5: DB15 Socket Connector Pins Position 15
Fig. 3.6: USB Connector Pins Position 16
Fig. 3.7: Ion R4301P DB9 Socket Connector Pins Position 17
Fig. 3.8: RFID Connector Pinout 17
Fig. 3.9: Top Cover LEDs 18
Fig. 4.1: Web Interface 19
Fig. 10.1: Bottom cover screws position 44
Fig. 10.2: SIM holder position 44

List of Tables

Tab. 3.1: Supply Voltage Connector Pinout 12
Tab. 3.2: Ethernet Connector Pinout 13
Tab. 3.3: Ethernet Connector LEDs Functionalities 13
Tab. 3.4: GPIO Connector Pinout 15
Tab. 3.5: USB Connectors Pinout 15
Tab. 3.6: GSM/GPRS Antenna Connector Pinout 16
Tab. 3.7: WIFI Antenna Connector Pinout 16
Tab. 3.8: RS232 Connector Pinout 17
Tab. 3.9: Top Cover LEDs 18
Tab. 12.1: Technical Specifications Table 50
Tab. 12.2: Reader to tag link profiles 51
Tab. 12.3: Power Supply Connector Electrical Characteristics 51
Tab. 12.4: RFID Antenna Ports Electrical Characteristics 51

Tab. 12.5: GSM/GPRS Antenna Port Electrical Characteristics (only in Mod. WR4301PXGPRS).....51
Tab. 12.6: WIFI Antenna Port Electrical Characteristics (only in Mod. WR4301PXWIFI) 51
Tab. 12.7: GPIO Port Electrical Characteristics 52
Tab. 12.8: RS232 Port Electrical Characteristics 52
Tab. 12.9: USB Port Electrical Characteristics.....52

1 INTRODUCTION

Product Description

The Ion (Model R4301P) is the top-of-the-range portal reader of the easy2read© product line.

CAEN RFID has carefully designed the device taking into account customer requests and on-field experience on RFID installations. The result is not only a RAIN RFID reader, it's a unique combination of outstanding RFID reading performances, computing power and communication capabilities.

The reader is optimized for portal installations, featuring full power to up to 4 antennas, Gen2 Dense Reader Mode management and high speed read rates.

Based on an embedded HW architecture (x86) and standard operating system (Linux), the Ion eases the development of custom software and solutions.

The on-board computing power and connectivity remove the need for an external PC and related cabling. This results in deployment and operation cost savings, thus reducing the total cost of ownership of installed devices.

The Ion is best suited for complex AutoID scenarios, where the information can be collected and fed directly to the reader from multiple sources such as Smart Card readers, Barcode readers, GPS and other in-field sensors.

All data can be handled locally through data buffering, filtering and aggregation, in order to directly provide decision-making data to higher level Business Intelligence processes. The same data can also trigger local actuators and screen displays for in-field real-time processes in a standalone mode.

The presence on board of an optional integrated GPRS modem or WIFI interface module, together with its compact and versatile form factor, allow to use it in any worldwide installation requiring RFID usage in remote areas.

As a result of all the above, the Ion allows solution providers to customize the reader to each application, thus creating their own specialized devices accordingly.

The Ion complies with and can operate in different regulatory environments (Europe, US, Australia, China, Korea, Singapore, Taiwan).

Fig. 1.1: Ion R4301P UHF Long Range Reader with GPRS/WIFI

Installation Notice

The Ion R4301P reader could be mounted either horizontally or vertically. Locate the four mounting slots on the reader, as illustrated below.

All measurements are in millimetres.

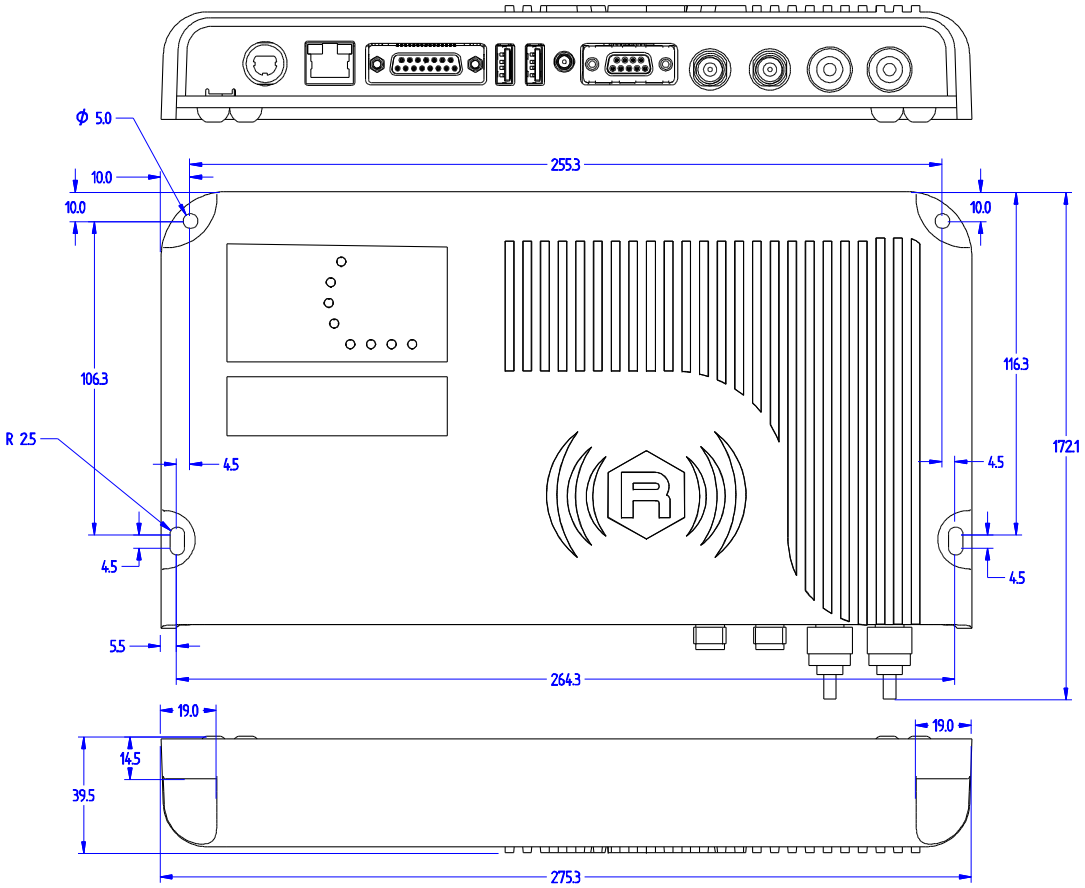


Fig. 1.2: Technical drawings

Development Kit

A development kit with power supply, antennas, cable and demo tags is available:

Fig. 1.3: Ion R4301P - RFID UHF Portal Reader Development Kit

The kit includes:

- n. 1 WALIM0000003 Power Supply
- n. 1 WANTENNAX019 Circular polarized antenna ETSI *(in Mod. WR4301PXDKEU, WR4301PDKGPE, WR4301PDKWFE)*
- n. 1 WANTENNAX020 Circular polarized antenna FCC *(in Mod. WR4301PXDKUS, WR4301PDKGPU, WR4301PDKWFU)*
- n. 1 WANTENNAX018 GPRS antenna *(in Mod. WR4301PDKGPE, WR4301PDKGPU)*
- n. 1 WANTENNAX021 WIFI antenna *(in Mod. WR4301PDKWFE, WR4301PDKWFU)*
- n. 1 Set of Labels
- n. 1 A927Z Temperature Logger Tag
- n. 1 RT0005 Temperature Logger Tag
- n. 1 Antenna cable (2.5 m)

The R4301P Ion reader and its development kit are a complete set up for a quick implementation of RFID solutions.

Ordering Options

	Code	Description
Reader	WR4301PXAAAA	R4301P - Ion - RFID UHF Portal Reader
	WR4301PXGPRS	R4301P - Ion - RFID UHF Portal reader with GPRS
	WR4301PXWIFI	R4301P - ION - RFID UHF Portal reader with WIFI Interface
Development Kit	WR4301PXDKEU	Ion - Dev Kit with ETSI RFID antenna, cable, power supply and tags (reader not included)
	WR4301PXDKUS	Ion - Dev Kit with FCC RFID antenna, cable, power supply and tags (reader not included)
	WR4301PDKGPE	Ion - Dev Kit with ETSI RFID antenna, GPRS antenna, cable, power supply and tags (reader not included)
	WR4301PDKGPU	Ion - Dev Kit with FCC RFID antenna, GPRS antenna, cable, power supply and tags (reader not included)
	WR4301PDKWFE	Ion - Dev Kit with ETSI RFID antenna, WiFi antenna, cable, power supply and tags (reader not included)
	WR4301PDKWFU	Ion - Dev Kit with FCC RFID antenna, WiFi antenna, cable, power supply and tags (reader not included)
Accessories	WANTENNAX019	Circular polarized antenna 8.5dBc (EU)
	WANTENNAX020	Circular polarized antenna 8.5dBc (US)
	WANTENNAX018	GPRS Antenna for Ion
	WANTENNAX021	WIFI Antenna for R4301P
	WALIM0000003	Power Supply for Ion R4301P
	WCAVOAAAX005	5 m RF Antenna Cable TNC/RP-N type
	WCAVOAAAX006	15 m RF Antenna Cable TNC/RP-N type
	WRA0003XAAAA	RA0003 - UHF Antenna Multiplexer
	WRA0002XAAAA	RA0002 - Digital I/O Interface Unit

2 GETTING STARTED

Introduction

This guide will help you to get started with your new Ion R4301P reader.

For more detailed information on reader configuration, connections and setup options please refer to the next chapters.

To begin, you need first to download and install the .NET framework 2.0 (only required if .NET is not already installed on your PC).

Connecting to the Ion R4301P Reader

By default, the Ion R4301P is configured with the static IP address 192.168.0.1. If your private network matches the default network configuration of the reader you can connect to it using a web browser to access the web interface or by using the Easy Controller software. Otherwise you can change the network configuration of the reader connecting to it with a serial communications program such as HyperTerminal (see § *Configure Ethernet settings* page 22).

After the Ion R4301P reader has been assigned a valid IP address you can connect it to your network and complete the configuration by using a web browser interface from a desktop PC.

Connecting to the Ion using the Easy Controller Software

1. Download the latest version of the Easy Controller software from the ion R4301P web page and install it.
2. Connect the Ion to the power supply (the power supply is an optional accessory, see § *Ordering Options* page 10).
3. Attach the Ion directly to your PC using a crossover cable or connect it to your network using a standard Ethernet cable.
4. Connect the antenna cable to port 1 of the reader.
5. Launch the *Easy Controller* by double clicking on the icon on your desktop.
6. Click on *File* → *Connect* and select the TCP/IP Connection option and insert the Ion's IP address (default address is 192.168.0.1).
7. Place a tag under the antenna field, click on *start inventory* and see the tag information displayed on the main window.

For more info on the use of the *Easy Controller*, please refer to the manual *CAEN RFID Easy Controller Software* manual available at the ion R4301P web page.

Connecting to the Ion using the serial port

You can connect to the Ion R4301P Reader using the serial port. Please refer to *Reader Configuration using the serial port* page 22.

By default, the serial port of the Ion R4301P reader is configured as a configuration console, please refer to § *Reader Configuration using the serial port* page 22 for further details on this matter. The serial port can be configured to be used as a control interface (for example to be used with the Easy Controller software or using our APIs - SDK Software Development Kit), please refer to § *Configure the serial port to communicate with the caenrfidd daemon* page *23.*

3 EXTERNAL INTERFACE DESCRIPTION

Connectors

The Ion R4301P reader is equipped with the following ports:

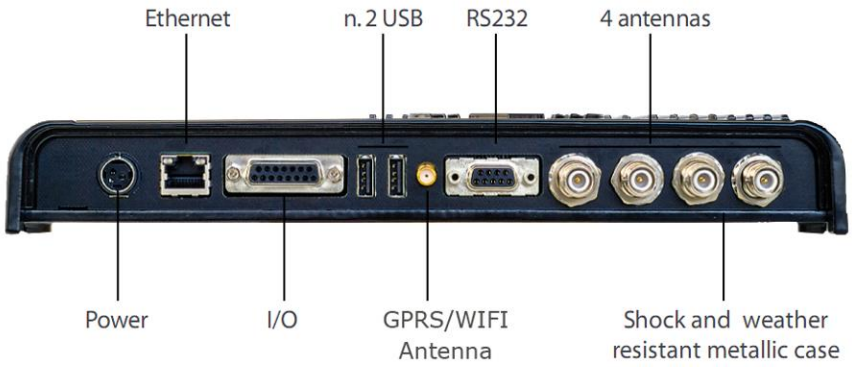


Fig. 3.1: Connectors View

Power supply

The power supply connector shall be used to provide Ion R4301P with the DC supply voltage in the range 9V ÷ 36V. The part number of the connector is KPJX-3S-S by KYCON and mates with KYCON P.N. KPPX-3P (to be used on the supply voltage cable), see § *Fig. 3.2: Plug Cable Connector* page 13.

The pinout of the connector is shown in the following table:

Pin #	Signal	Description
1	GND	Ground
2	VIN	DC input voltage
3	GND	Ground

Tab. 3.1: Supply Voltage Connector Pinout

The external shield and the pin n°3 can be left unconnected or can be connected to GND.

In the following picture the drawing of KPPX-3P is shown.

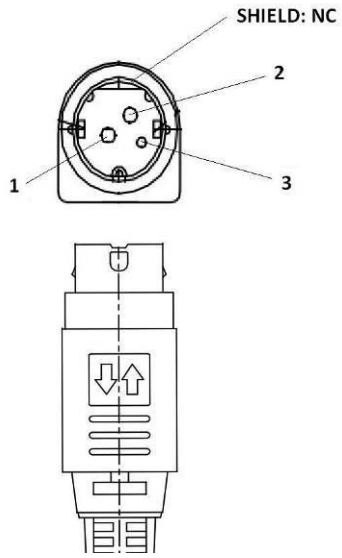


Fig. 3.2: Plug Cable Connector

A standard power supply AC/DC adapter 24Vdc output voltage (mod. WALIM0000003) for the Ion R4301P reader is available for purchase (see § *Ordering Options* page 10).

Ethernet port

The Ethernet interface of Ion R4301P can be used to connect the reader to a 10/100/1000BaseT network using a RJ45 cable.

The pinout of the connector in shown in the following table:

Pin #	Signal	Description
1	DA+	Bi-directional pair A +
2	DA-	Bi-directional pair A -
3	DB+	Bi-directional pair B +
4	DC+	Bi-directional pair C +
5	DC-	Bi-directional pair C -
6	DB-	Bi-directional pair B -
7	DD+	Bi-directional pair D +
8	DD-	Bi-directional pair D -

Tab. 3.2: Ethernet Connector Pinout

Note: the Ion R4301P does not support Power Over Ethernet (POE).

The Ethernet connector has two LEDs with the following functionalities:

LED	FUNCTION	TYPE
Right	Network activity	Blinking Yellow
Left	10Mbps connection	OFF
Left	100Mbps connection	Yellow
Left	1000Mbps connection	Green

Tab. 3.3: Ethernet Connector LEDs Functionalities

GPIO ports

The Ion R4301P reader has 13 GPIO bi-directional pins to be used to control external device or to send trigger signals to the reader. The default configuration of all the GPIO lines after a power on reset or a general reset is set to Input.

The configuration of each GPIO line can be changed individually using the *API - CAEN RFID SDK* (Software Development Kit available at the ion R4301P web page), the web configuration interface, the Easy Controller software or the configuration console (see § *GPIO* chapter page 30 for further details).

In the case a GPIO line is configured as an output line, the pin is an open drain port with 12mA current sinking capability and an internal 10kΩ pull-up resistor to 5Vdc.

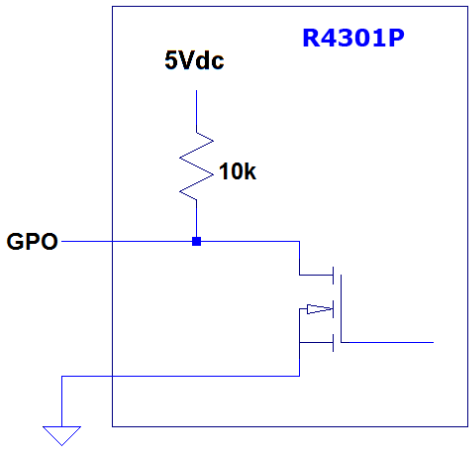


Fig. 3.3: GPIO Output Configuration Equivalent Schematic

In case a GPIO line is configured as an input line, the pin is a logic port with Schmitt trigger input with a voltage range from 0V to 5V.

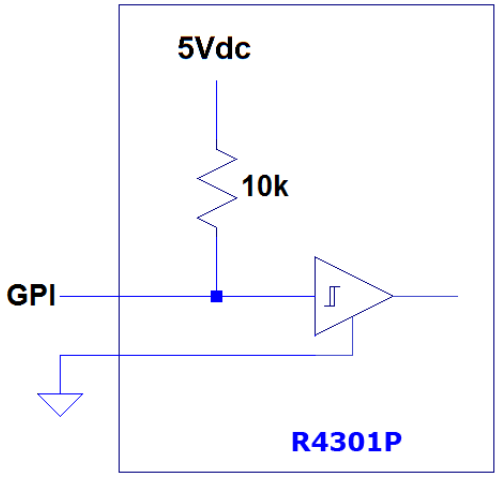


Fig. 3.4: GPIO Input Configuration Equivalent Schematic

The reader supply voltage is available on the pin #15 of the GPIO connector (DC_BYPASS, connected to pin #2 of the power supply connector) in order to supply external equipment with an overall maximum current need of 500mA.

In the following table the pinout of the GPIO DB15 socket connector is shown. Connector external shell is connected to GND.

Pin #	Signal	Description
1	GPIO0	General-purpose IO #0
2	GPIO2	General-purpose IO #2
3	GPIO4	General-purpose IO #4
4	GPIO6	General-purpose IO #6
5	GPIO8	General-purpose IO #8
6	GPIO10	General-purpose IO #10
7	GPIO12	General-purpose IO #12
8	GND	Ground
9	GPIO1	General-purpose IO #1
10	GPIO3	General-purpose IO #3
11	GPIO5	General-purpose IO #5
12	GPIO7	General-purpose IO #7
13	GPIO9	General-purpose IO #9
14	GPIO11	General-purpose IO #11
15	DC_BYPASS	Input supply voltage bypass (500mA max)
-	Shell	External shell (connected to Ground)

Tab. 3.4: GPIO Connector Pinout

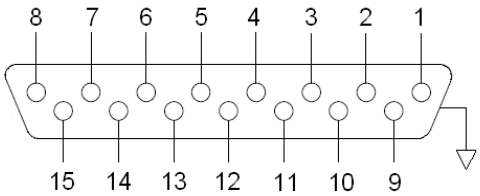


Fig. 3.5: DB15 Socket Connector Pins Position

Warning: avoid connecting the DC_BYPASS signal to any of GPIO pins, otherwise the reader can be permanently damaged.

USB host ports

Ion R4301P reader has 2 USB 2.0 High Speed host ports that can be used to connect external devices like WI-FI or Bluetooth adapters, GPS receivers or flash pen drives.

Each port is able to provide up to 500mA supply current at 5Vdc voltage.

The connector is a 4 pins TYPE A receptacle. The pinout is described in the following table:

Pin #	Signal	Description
1	Vbus	+5V USB bus voltage output (500mA max)
2	D-	USB D- signal
3	D+	USB D+ signal
4	GND	Ground
-	Shell	External shell (connected to Ground)

Tab. 3.5: USB Connectors Pinout

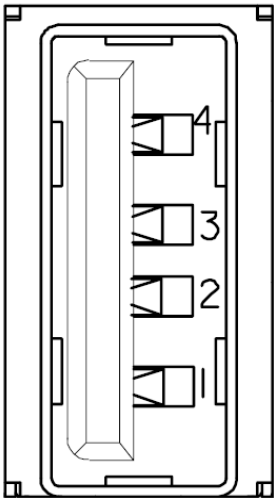


Fig. 3.6: USB Connector Pins Position

GSM/GPRS antenna port (only in Mod. WR4301PXGPRS)

RF port for the external GSM/GPRS antenna is a 50Ω SMA jack connector.

The suggested antenna to be used with Ion R4301P/GPRS reader is Amphenol P.N. TAN015-1100104BS.

Ion R4301P/GPRS is approved to operate with 1dBi gain max. antennas connected to the GSM/GPRS port.

Antenna VSWR shall be lower than 2:1 in order to achieve optimum performances: the use of antenna with VSWR > 10: 1 can damage the module.

Pin	Signal	Description
Inner	RF OUT	RF output (2W peak max.)
Outer	GND	Ground

Tab. 3.6: GSM/GPRS Antenna Connector Pinout

Note: the standard Ion R4301P model has the SMA connector but the internal GSM/GPRS is not present.

Note: for more information on the insertion of SIM CARD please refer to *SIM card placement* paragraph page 44.

WIFI antenna port (only in Mod. WR4301PXWIFI)

RF port for the external WIFI antenna is a 50Ω SMA jack connector.

The suggested antenna to be used with Ion R4301P/WIFI reader is Pulse W1010.

Ion R4301P/WIFI is approved to operate with 2dBi gain max. antennas connected to the WIFI port.

Antenna VSWR shall be lower than 2:1 in order to achieve optimum performances: the use of antenna with VSWR > 10: 1 can damage the module.

Pin	Signal	Description
Inner	RF OUT	RF output (100mW peak max.)
Outer	GND	Ground

Tab. 3.7: WIFI Antenna Connector Pinout

Note: the standard Ion R4301P model has the SMA connector but the internal WIFI interface is not present.

RS232

The Ion R4301P RS232 port is available on a DB9 socket connector.

The pinout is the following:

Pin #	Signal	Description
1	-	Not connected
2	TX	TX output
3	RX	RX input
4	-	Not connected
5	GND	Ground
6	-	Not connected
7	CTS	CTS input
8	RTS	RTS output
9	-	Not connected
-	Shell	External shell (connected to Ground)

Tab. 3.8: RS232 Connector Pinout

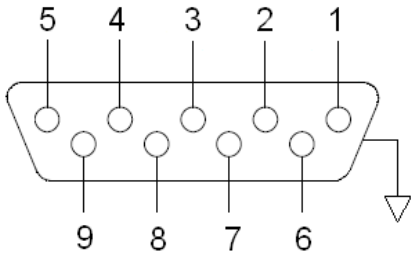


Fig. 3.7: Ion R4301P DB9 Socket Connector Pins Position

RFID antenna ports

The Ion R4301P has 4 antenna ports available on 50Ω RP-TNC jack connectors.

The pinout is the following:

Pin	Signal	Description
Inner	RF OUT	RF output (32dBm peak max.)
Outer	GND	Ground

Fig. 3.8: RFID Connector Pinout

In order to achieve the best reading performances the VSWR of the antenna shall be lower than 1.5 : 1.

LEDs

The Ion R4301P top cover panel houses the following LEDs (see § *Fig. 3.9: Top Cover LEDs* page 18):

LED	FUNCTION	TYPE
POWER	Power ON	Green LED
RF ACTIVE	RFID Transmitting activity	Blinking Yellow LED
GSM/GPRS	GSM/GPRS modem or WIFI interface power ON	Blinking Blue LED
	Communication activity	Blinking Blue LED
FAULT[1]	Boot phase	Red LED
	Ready to work	OFF
ANTENNA[2]	Selected TX antenna	Yellow LED

Tab. 3.9: Top Cover LEDs

Fig. 3.9: Top Cover LEDs

[1] In Mod. WR4301PXAAAA the GSM/GPRS LED is always OFF.

[2] The LED indicates the antenna configured via software to transmit. By default the selected antenna is Ant1. Please note that the LED does not indicate if an antenna is physically attached to the reader connector or not.

4 READER CONFIGURATION

Introduction

The Ion R4301P Reader can be configured using either the configuration web interface or the standard linux bash shell available via the serial port. In the following paragraphs we will guide you through the reader configuration using both the methods.

Reader Configuration using the Web Interface

The Ion Web Interface is built around Webmin, a well-known web-based configuration tool for Linux systems. Being derived from Webmin, the Ion web interface takes all the advantages of expandability and modularity of the Webmin tool itself.

By factory default, the Ion web interface is reachable at the following URL: https://192.168.0.1:10000 (see the figure below).

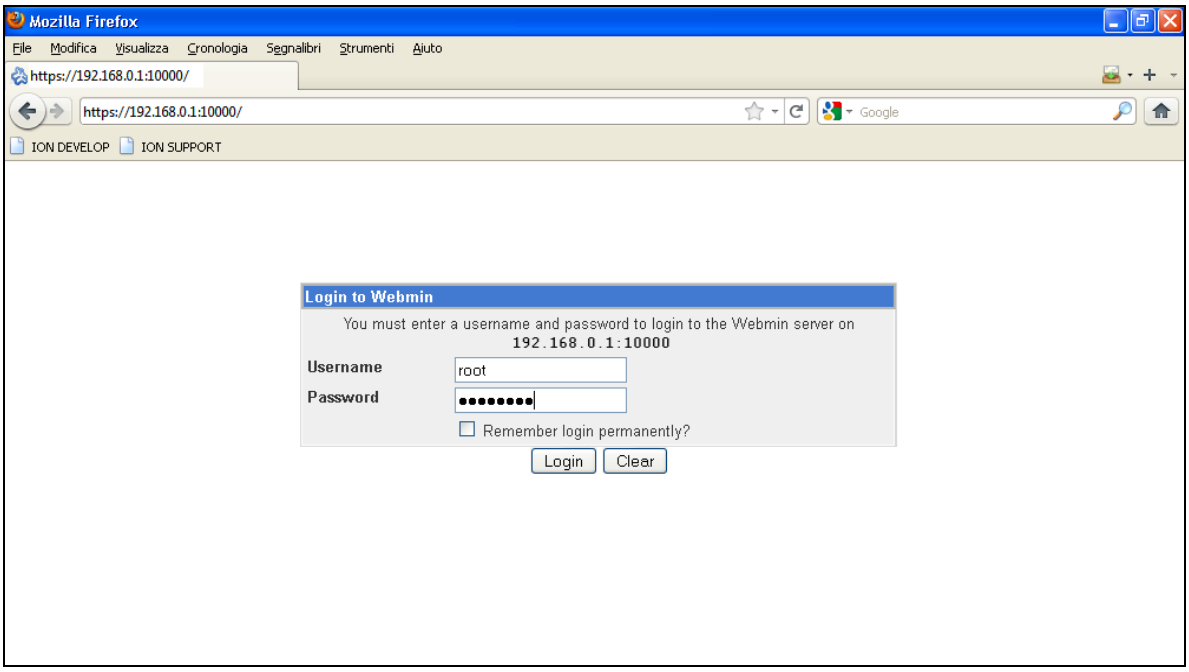


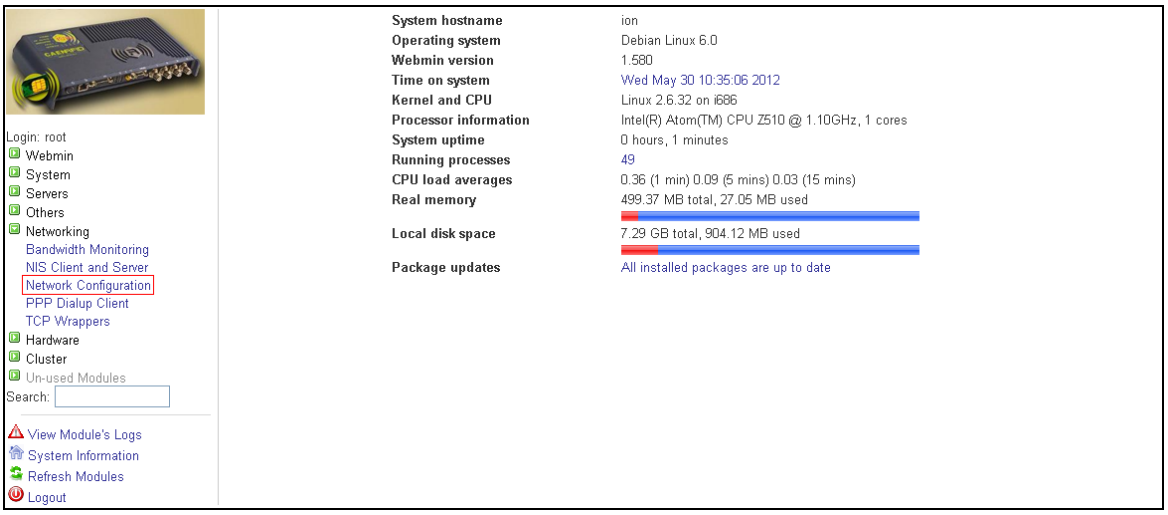
Fig. 4.1: Web Interface

To login, type ***root*** in the Username text box and ***rootroot*** in the Password textbox.

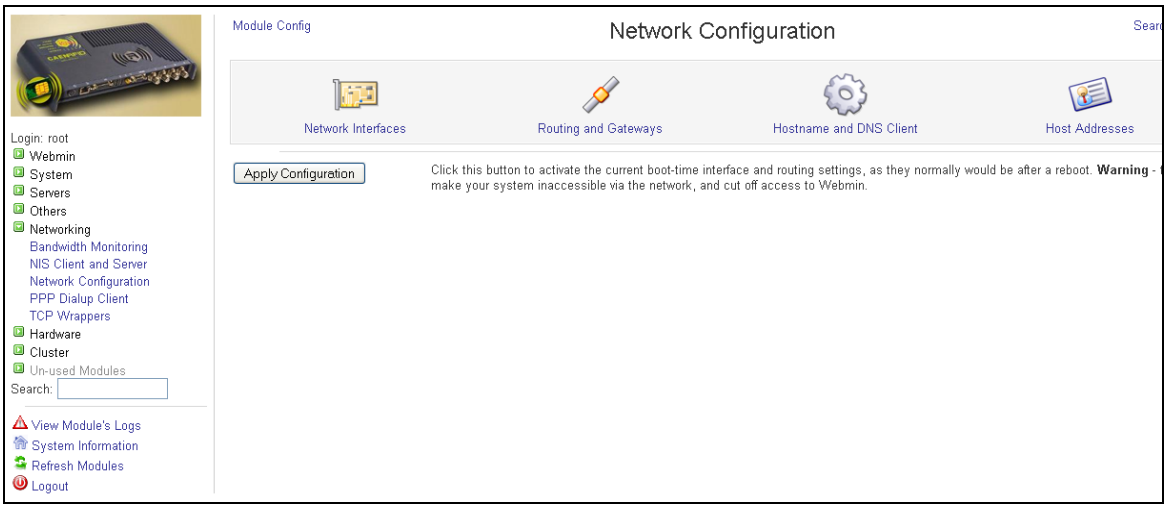
Configure Ethernet settings

To change the network configuration of the Ion R4301P reader using the web interface:

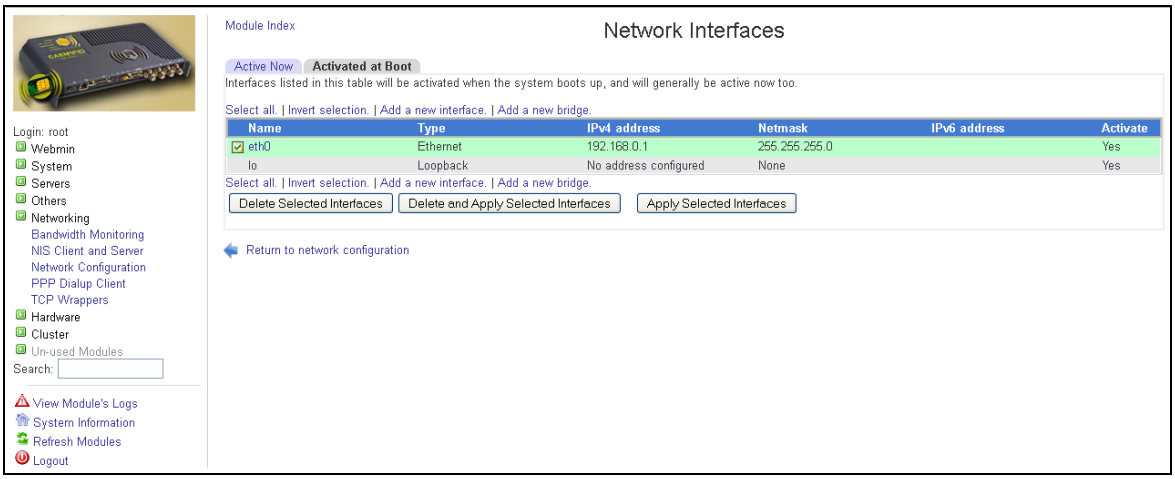
1. Select the *Network Configuration* option in the Networking menu:



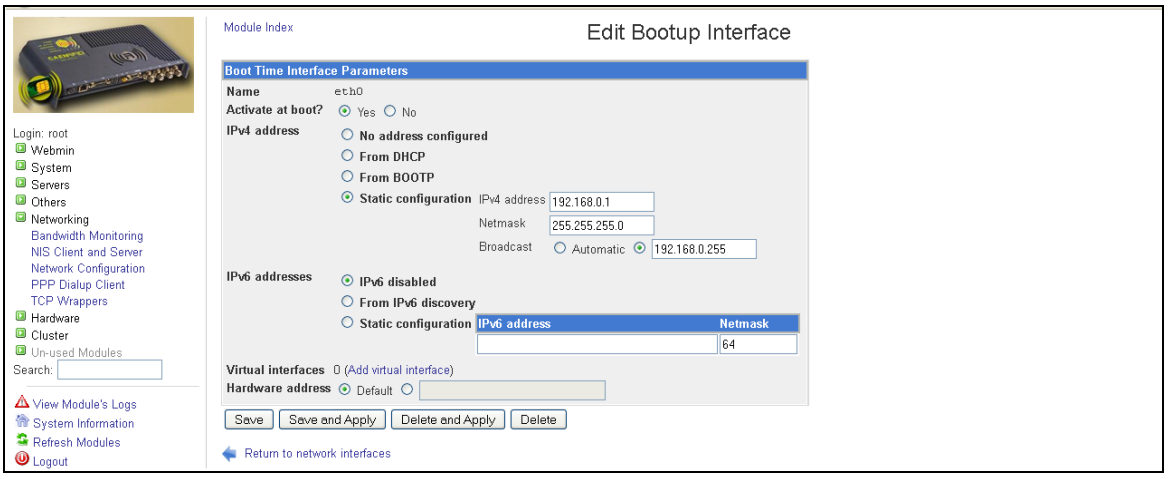
2. Click on the *Network Interfaces* icon:



3. On the Activated at Boot tab, enable the eth0 switch and click on the *eth0* label:



4. Fill the *Boot Time Interface Parameters* form and press the save button:



The new settings will become effective at next reader reboot.

Set Date and Time

To change the reader date and time using the web interface:

1. Select the *System Time* option in the Hardware Menu.
2. On the *Set Time* tab you can set date and time both for system time and hardware time or decide to set one time based on the other using the *Set System Time To Hardware Time* or the *Set Hardware Time To System Time* buttons.
3. On the *Change timezone* tab you can select your timezone.
4. On the *Time server sync* tab you can enable time server synchronization. In the first textbox you can provide the IP address of the server, and check if the hardware clock must sync with it. The synchronization can be scheduled by clicking on the radiobutton "Yes, at time below…" and can be done either choosing a simple scheduling by a list box or specifying a datetime by the six lists provided in the tab. When all settings are done, click on *Sync and apply* button in the bottom of the tab to make the changes effective.

Reader Configuration using the serial port

By default, a standard bash shell is available on the Ion serial port; to access the reader using this method, you have first to configure your preferred serial terminal monitor (Hyperterminal, minicom, etc...) with the following settings:

Baud rate:	115200 b/s
Data bit:	8
Stop Bit:	1
Parity:	None
Flow Control:	None

To login, type ***root*** at the Username prompt and ***rootroot*** at the Password prompt.

The default text editor on Ion R4301P reader is the vi (a screen-oriented text editor originally created for the Unix operating system). However it is possible to install any other Debian Linux compatible text editor and to use it as replacement of the *vi*.

Configure Ethernet settings

To modify the reader network parameters you have to edit the network interface configuration file in the `/etc/network` directory[3]:

```
root@ion:~# vi /etc/network/interface
# The loopback network interface
auto lo eth0
iface lo inet loopback

# The primary network interface uses static address
iface eth0 inet static
        address 192.168.0.1
        netmask 255.255.255.0
        broadcast 192.168.0.255
        network 192.168.0.0
        #gateway 192.168.0.254
```

Change the default values according to your needs.

If needed, you can also add your preferred DNS server to the `/etc/resolv.conf` file:

```
nameserver 192.168.0.254
```

To reboot the network stack and make the changes active, use the following commands (`stop` and `start`):

```
root@ion:~# /etc/init.d/networking stop
root@ion:~# /etc/init.d/networking start
```

[3] Commands on the Development Virtual Machine are marked by the suffix `develop:~#`, while commands on the Ion R4301P Reader shell are marked by the suffix `root@ion:~#`.

Set Date and Time

To set date and time on the Ion reader, you can use the linux date command; as an example, to set the current time and date to 29 May 2012, 16:30:00 type:

```
root@ion:~# date -s "29 May 2012 16:30:00"
```

The above command sets the system time only. To set the hardware clock, based on the system clock, type the command:

```
root@ion:~# /sbin/hwclock --systohc
```

Configure the serial port to communicate with the *caenrfidd* daemon

As said in the previous paragraphs, by default the RS232 port is used only as a console for the bash shell.

If needed, you can use the RS232 port, instead of the ethernet port, to communicate directly with the *caenrfidd* daemon (see § *DEVELOPING APPLICATION* page 35). The steps required for the serial port reconfiguration are listed below.

1. Disable writing bootloader output messages to the serial console. By editing the `/etc/default/grub` file and comment out the following lines by adding a # as follow:

   ```
   # Enable the serial console
   # GRUB SERIAL COMMAND="serial --speed=115200 --unit=0 --word=8 --parity=no --stop =1"
   # GRUB TERMINAL INPUT=serial
   # GRUB_TERMINAL_OUTPUT=serial
   ```

2. Disable writing kernel output messages to the serial console. By editing the `/etc/default/grub` file and modify the kernel command line by removing the console option line:

   ```
   GRUB_CMDLINE_LINUX="loglevel=6 reboot=bios"
   ```

3. Disable writing init output messages to the serial console. By editing the `/etc/inittab` file and comment out the line beginning with *T0* by adding a # as follow:

   ```
   #T0:23:respawn:/sbin/getty -L ttyS0 115200 vt100
   ```

4. Upgrade the grub settings by using the following command:

   ```
   root@ion:~# update-grub
   ```

5. If not already available, install the *socat* program using the following command:

   ```
   root@ion:~# aptitude install socat
   ```

6. Redirect all *caenrfidd* network messages to the serial port by using the following command:

   ```
   root@ion:~# socat TCP:localhost:1000 /dev/ttyS0,raw,echo=0
   ```

   Redirection is immediately active after the above command execution and will be performed until reader power off. If you want this setting to be persistent, add the above command to the *rc.local* file in the `/etc` directory:

   ```
   #!/bin/sh -e
   #
   # rc.local

   # Decomment below to run the install script automatically...
   # /sbin/getty -L ttyS0 -n -l /opt/install 115200 vt100
   socat TCP:localhost:1000 /dev/ttyS0,raw,echo=0
   ```

Please note that, once all the above steps will be completed, the output of the bash shell will be no more directed to the RS232 serial port. However, the bash shell will be still available via a ssh connection.

For example, to enter again the bash shell from the development virtual machine[4] you can type:

```
develop@ion:~# ssh <ion ip address>
```

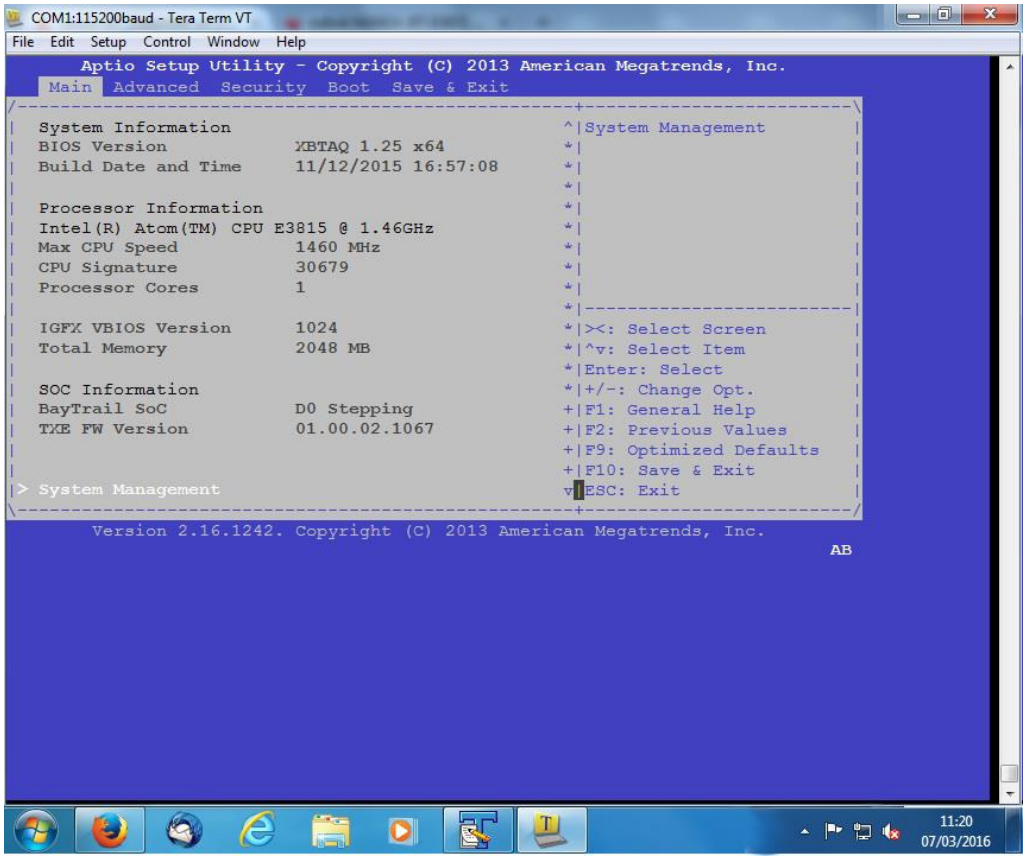
where *<ion ip address>* is the IP address of the Ion reader.

[4] See § *Getting Started with C* page 32. The chapter explains how to use our developing virtual machines and how you can use it in order to develop your custom applications for the Ion R4301P reader.

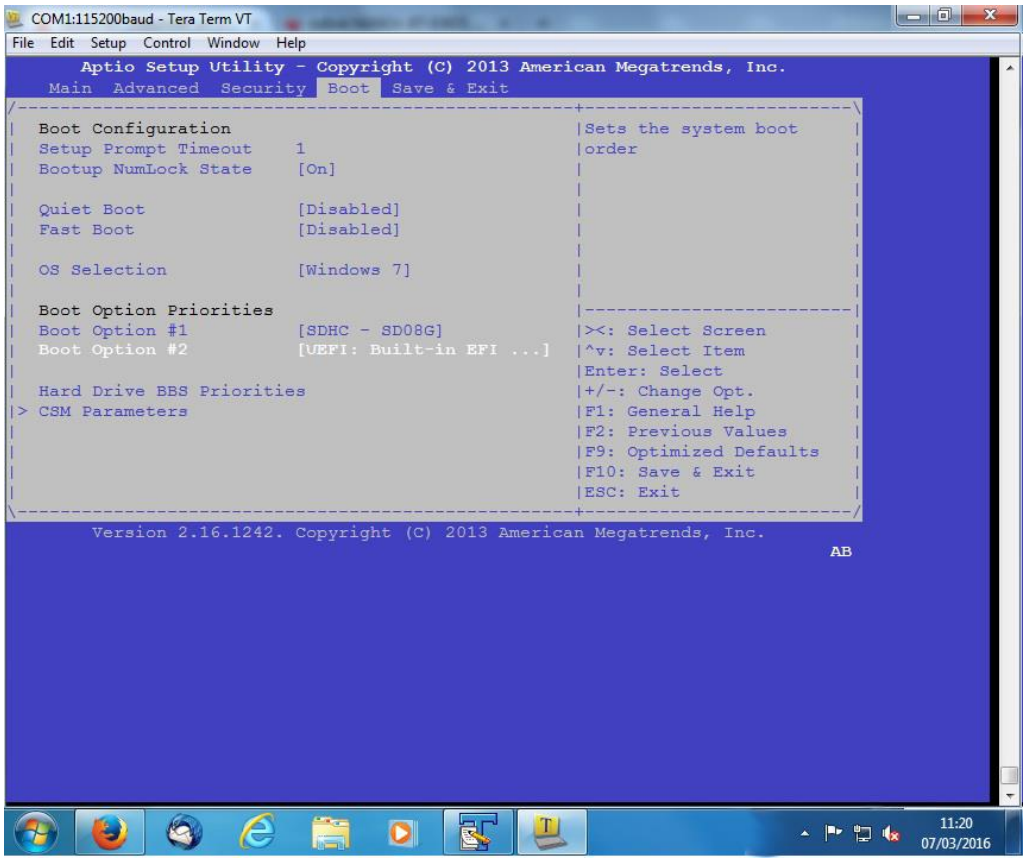
How to restore the R4301P default factory settings

Follow the instructions below for resetting the Ion firmware to its factory default.

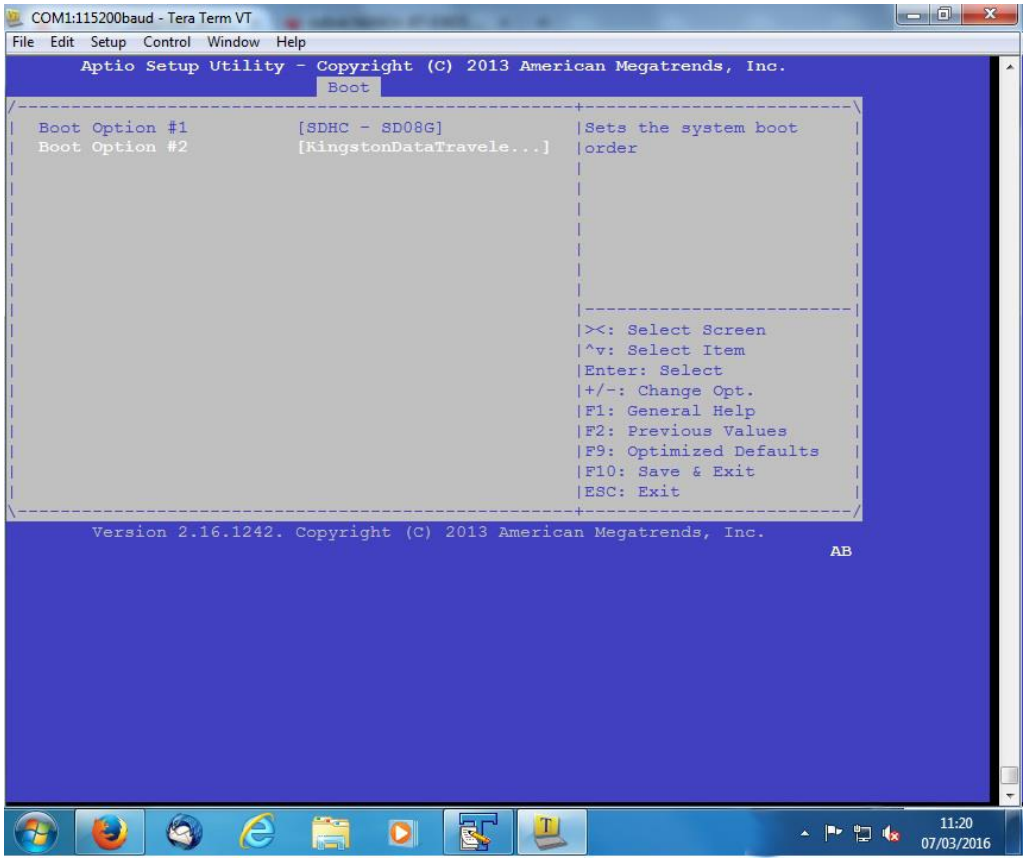
1. Download the latest zipped Ion R4301P image from the ion R4301P web page and unzip it.
2. Flash the Ion R4301P image into an USB flash drive (at least 4 GB of memory).
   For that purpose you can use the HDDRaw tool (http://hddguru.com/software/HDD-Raw-Copy-Tool/) or any other equivalent software.
3. Power ON your PC, insert the USB flash drive into one of the Ion USB ports and attach a serial cable to the Ion RS232 port.
4. Open a serial port monitor window with the following settings:
   baud rate: 115200
   Data bit: 8
   Stop bit: 1
   Parity: none
   Flow control: none
5. Power ON the reader and during the boot, press the ESC key:



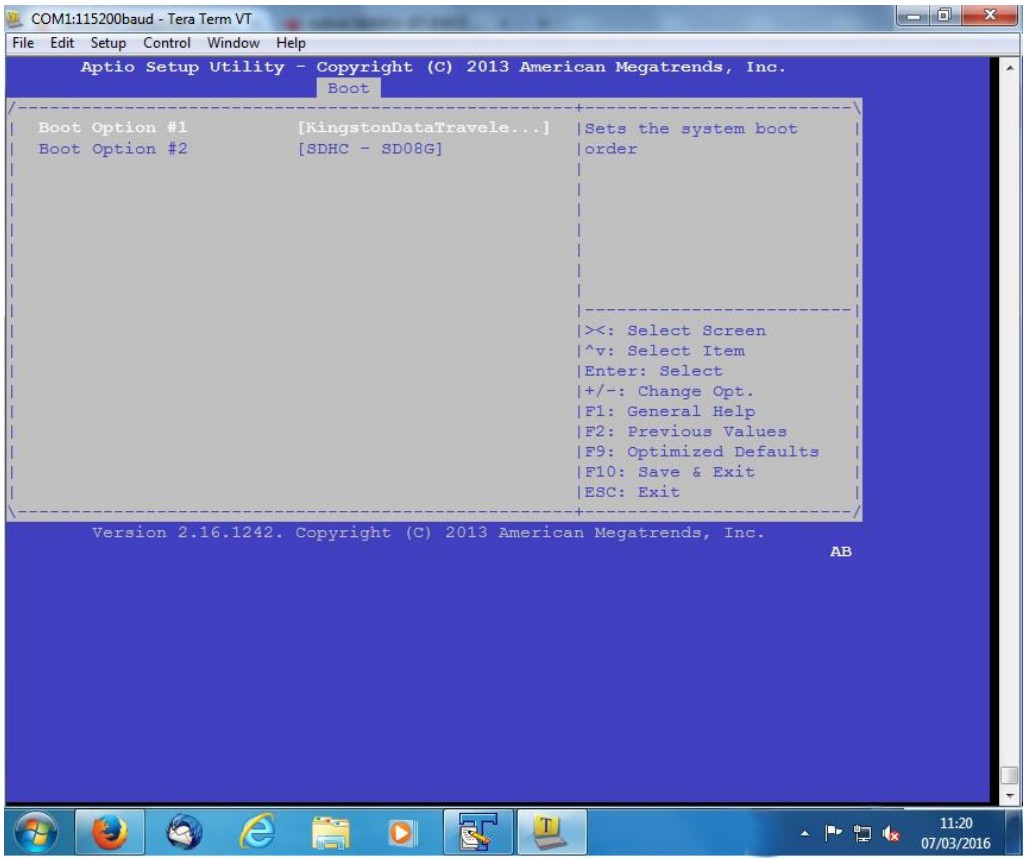
6. Press the right arrow to move until “Boot” and enter this option by pressing “Enter” key:



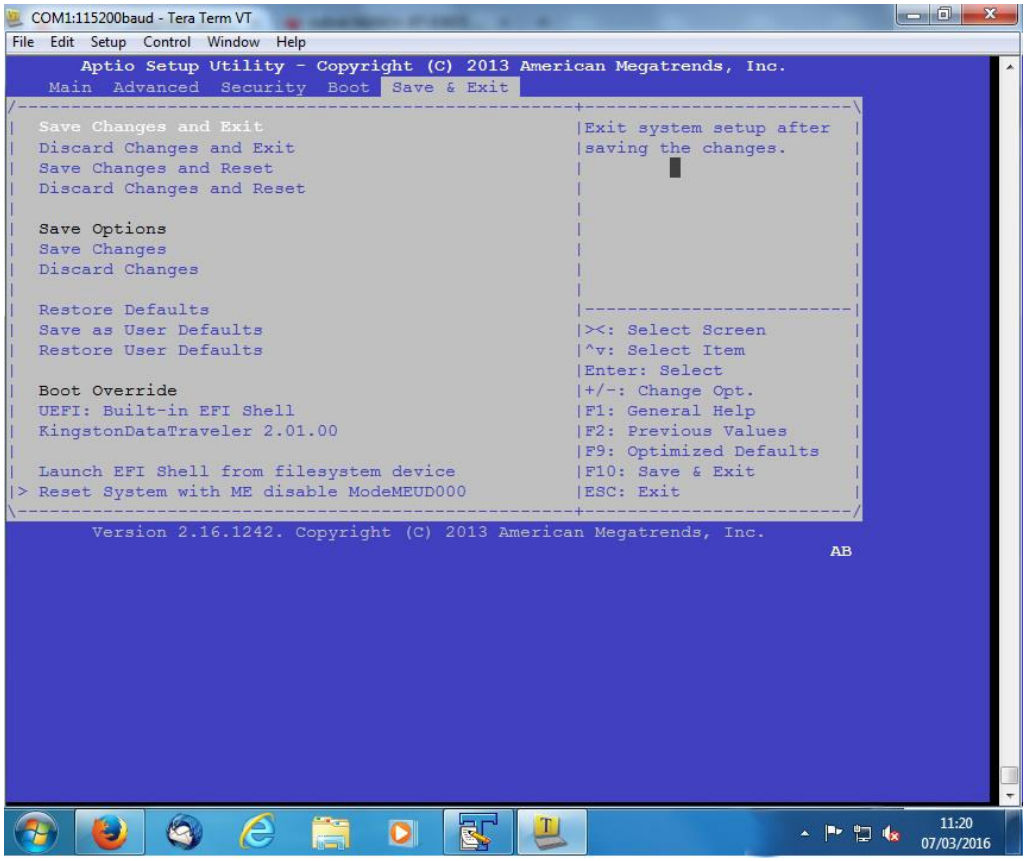
7. Using the down arrow, move until “Hard drive BBS Priorities” and enter this option by pressing “Enter” key:



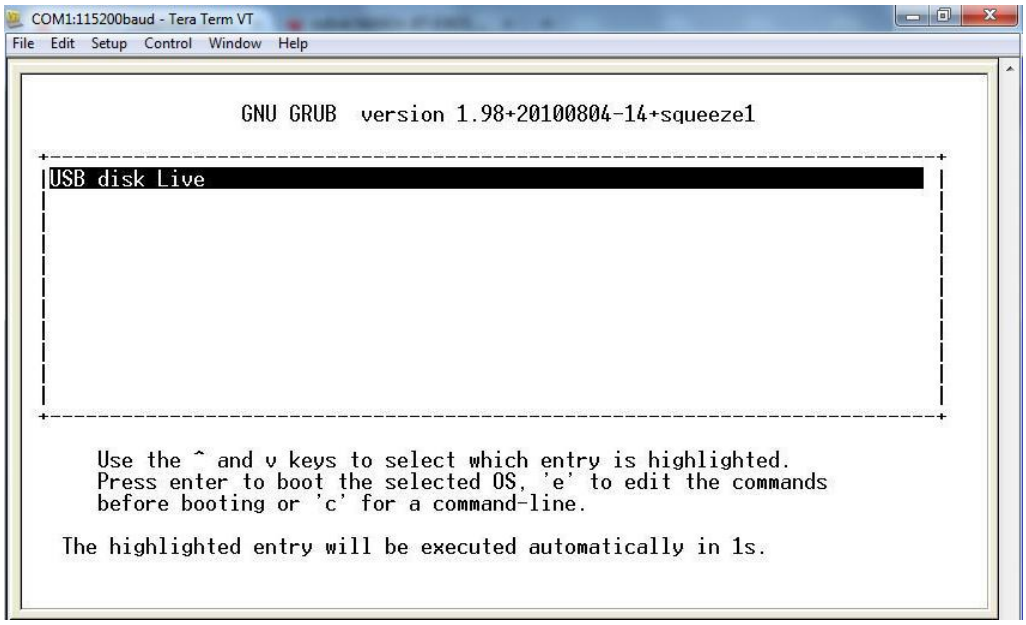
8. Select the USB pen drive using the down arrow and change it to *Boot Option #1* using the "+/-" keys:



9. Press the ESC key, go to "Save&Exit" using the right arrow and select "Save Changes and Exit". Press Enter.



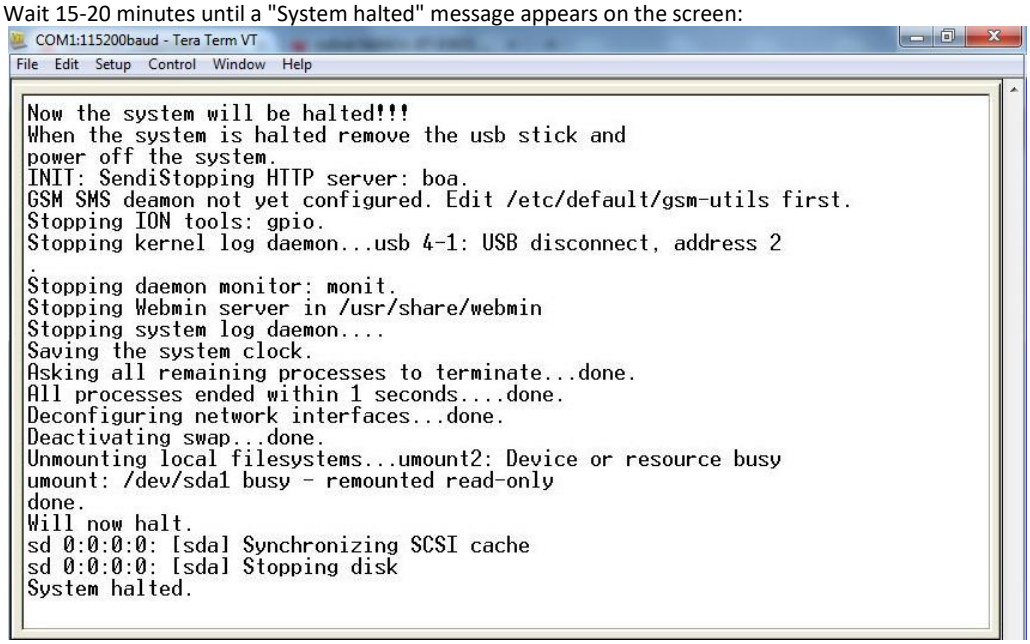
10. If everything is ok the bootloader should boot from the USB disk image as shown in the picture:



11. Type root at the Ion login prompt and rootroot at the password request.

12. When entered in the bash shell type the following command:
    /opt/install
    and then press enter.

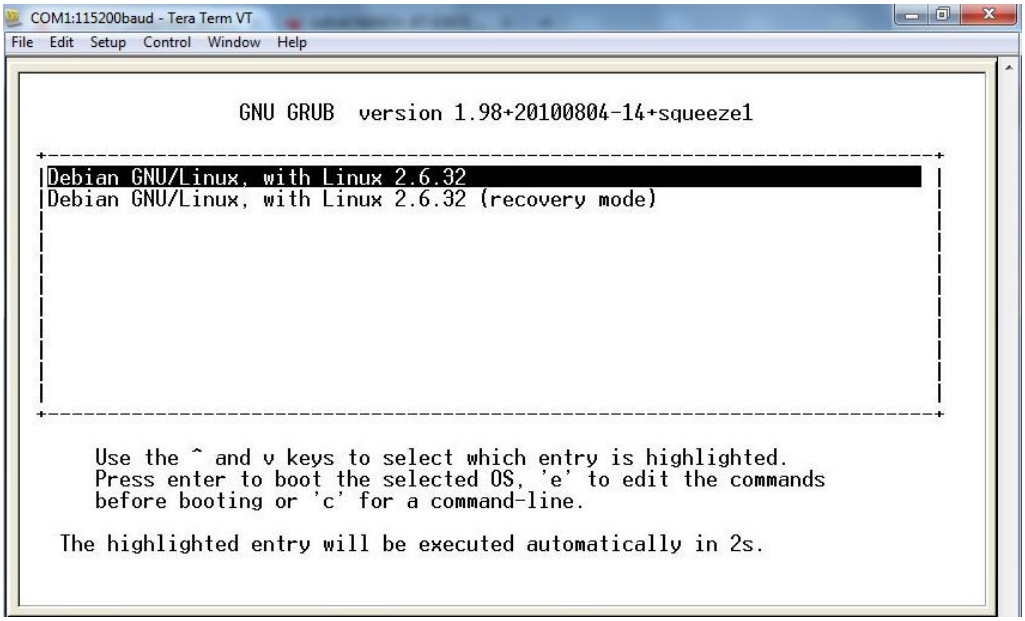
13. Wait 15-20 minutes until a "System halted" message appears on the screen:

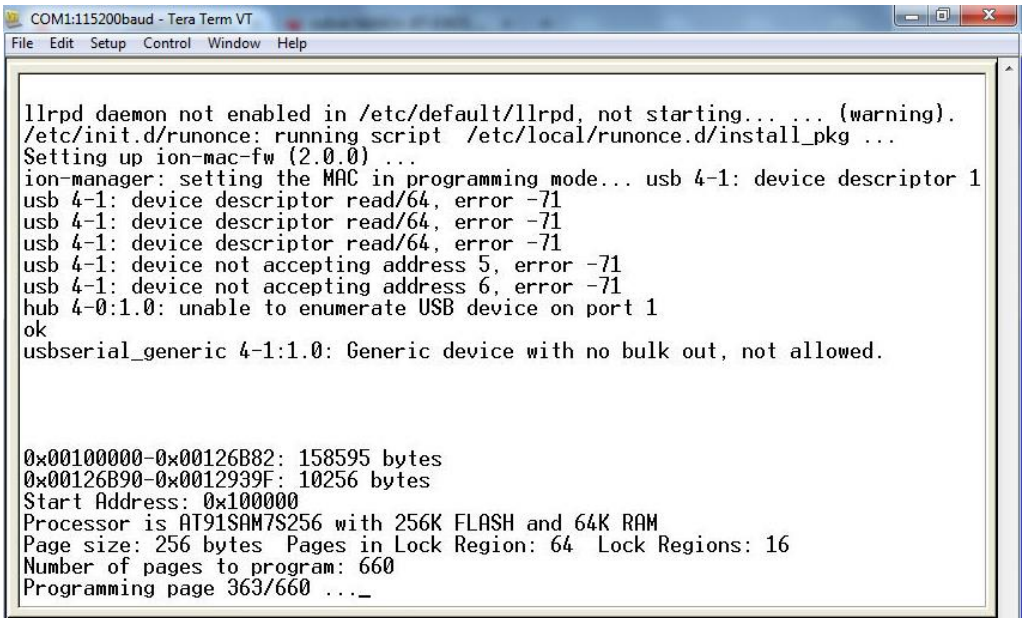


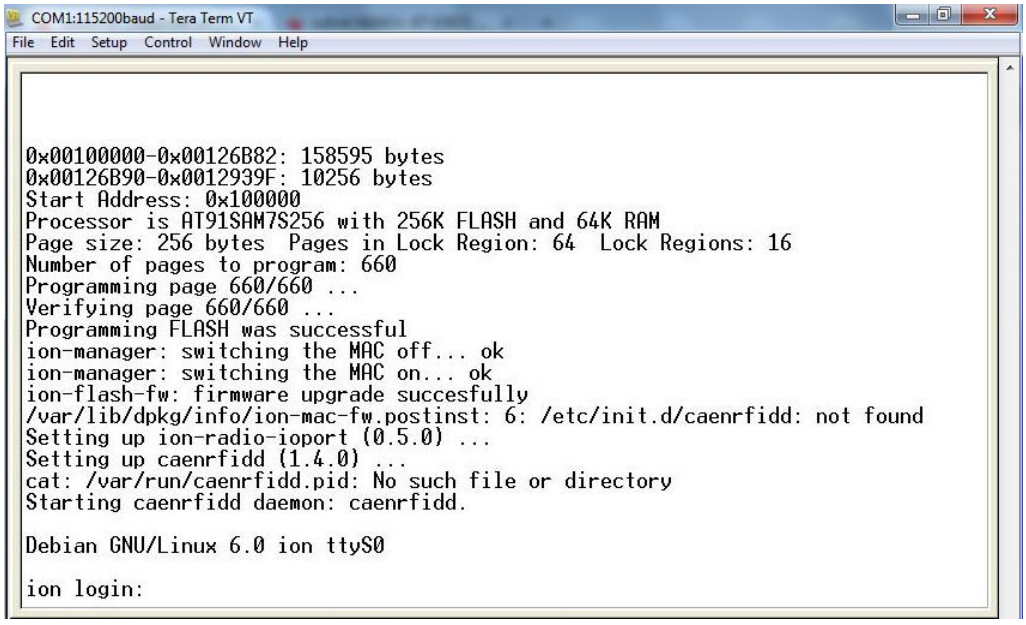
14. Power off the reader and remove the USB flash drive.

15. Power on the reader.

16. Wait until all the new packages have been installed (see the following three pictures):


COM1:115200baud - Tera Term VT
File Edit Setup Control Window Help
GNU GRUB version 1.98+20100804-14+squeeze1
Debian GNU/Linux, with Linux 2.6.32
Debian GNU/Linux, with Linux 2.6.32 (recovery mode)
Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the commands
before booting or 'c' for a command-line.
The highlighted entry will be executed automatically in 2s.


COM1:115200baud - Tera Term VT
File Edit Setup Control Window Help
llrpd daemon not enabled in /etc/default/llrpd, not starting... ... (warning).
/etc/init.d/runonce: running script /etc/local/runonce.d/install_pkg ...
Setting up ion-mac-fw (2.0.0) ...
ion-manager: setting the MAC in programming mode... usb 4-1: device descriptor 1
usb 4-1: device descriptor read/64, error -71
usb 4-1: device descriptor read/64, error -71
usb 4-1: device descriptor read/64, error -71
usb 4-1: device not accepting address 5, error -71
usb 4-1: device not accepting address 6, error -71
hub 4-0:1.0: unable to enumerate USB device on port 1
ok
usbserial_generic 4-1:1.0: Generic device with no bulk out, not allowed.
0x00100000-0x00126B82: 158595 bytes
0x00126B90-0x0012939F: 10256 bytes
Start Address: 0x100000
Processor is AT91SAM7S256 with 256K FLASH and 64K RAM
Page size: 256 bytes Pages in Lock Region: 64 Lock Regions: 16
Number of pages to program: 660
Programming page 363/660 ...



17. When the Ion login prompt appears the restore procedure is completed and the Ion is ready to be used again.

5 GPIO

Using Ion GPIO Interface

The Ion has 13 general purpose input and output (GPIO) interfaces. You can use them to control external devices and/or get specific signals from the environment.

The GPIO lines can be managed directly from the console Linux shell of the reader. Looking into the `/dev` directory, you can find an inner dedicated directory for each GPIO line:

```
root@ion:~# ls -d /dev/gpio*
/dev/gpio0   /dev/gpio11  /dev/gpio3  /dev/gpio6  /dev/gpio9
/dev/gpio1   /dev/gpio12  /dev/gpio4  /dev/gpio7
/dev/gpio10  /dev/gpio2   /dev/gpio5  /dev/gpio8
```

For instance, the first GPIO line is mapped to the gpio0 directory and, looking into it, you'll find the following pseudo-files:

```
root@ion:~# ls /dev/gpio0/
device       direction  power  subsystem  uevent  value
```

using the pseudo-file *direction* you can configure the GPIO line as an input or as an output while using the pseudo-file *value* you can get/set the GPIO line level.

For instance, to know the configuration of the first GPIO line you can perform the following commands:

```
root@ion:~# cat /dev/gpio0/direction
in
```

To get the level of the first GPIO line you can perform the following commands:

```
root@ion:~# cat /dev/gpio0/value
1
```

Since all the GPIO lines are configured as input lines by default at the startup, in order to set a GPIO line to a specified level you have to configure it as an output line before. Let consider the following example where we set the GPIO line 0 to the low level:

```
root@ion:~# echo out > /dev/gpio0/direction
root@ion:~# cat /dev/gpio0/direction
out
root@ion:~# echo 0 > /dev/gpio0/value
```

For technical information on GPIO ports please refer to *GPIO ports* page 14 and *GPIO Port Electrical Characteristics* page 52.

6 *CAENRFIDD* DAEMON

Introduction

The *caenrfidd* daemon represents the server side implementation of the CAEN RFID protocol, it handles the request coming from the host computer and sends back the replies after having executed the requested actions. The client side of the CAEN RFID protocol is implemented by the CAEN RFID SDK (Software Development Kit) available at the ion R4301P web page.

Configuring the daemon

The *caenrfidd* daemon can be configured by modifying its configurations files. After editing the configurations files you should restart the daemon in order to make effective the new settings:

```
# /etc/init.d/caenrfidd restart
```

The system wide configuration file

The *caenrfidd* daemon uses file `/etc/defaults/caenrfidd` as system wide configuration file. Here the default content of file such on the Ion:

```
# Default settings for caenrfidd server. This file is sourced by /bin/sh from
# /etc/init.d/caenrfidd.

# Enable/disable the deamon.
# Use "yes" to enable.
CAENRFIDD ENABLE=yes

# Set the default reader name
READER_NAME=ion

# Enable debugging mode
#DEBUG=yes
```

The variable `CAENRFIDD_ENABLE` can be used to enable/disable the daemon at startup while `READER_NAME` defines the default reader name (Ion, in this example).

The `DEBUG` variable enable/disable debugging messages into system log facility. You can see debugging messages into file `/var/log/syslog` (use command `tail -f /var/log/syslog` to log all data on your terminal).

The main configuration file

The main configuration file is `/etc/caenrfidd/caenrfidd.conf` and it defines basic daemon configurations. Here the default content of this file on the Ion:

```
# Package generated configuration file

# System configuration directory
#
# Set the default system configuration directory.
#
# Default:
#       config dir = "/etc/caenrfidd"

# Loggin level
#
# Set the default log level.
```

```
#
# Valid values are: debug, error and info.
#
# Default:
#       log_level = error

# Log to stderr
#
# Enable logging to standard error too.
#
# Default:
#       log_to_stderr = false

# Lock file
#
# Set the lock file name.
#
# Default:
#       lock file = "/var/lock/caenrfidd"

# PID file
#
# Set the PID file name. You can kill the program by using the command:
#       $ kill
#
# Default:
#       pid file = "/var/run/caenrfidd.pid"

# Run as user
#
# If executed as root the server will drop privileges to the specified
# user.
#
# Note: the selected user shold be able to read/write USB devices!
#
# Default:
#       user = "daemon"
user = "root"

# Listening port
#
# Change the default listening port.
#
# CAENRFID default is 1000 so you need setting:
#
#    user = "root"
#
# in order to bind to the default CAENRFID port or the server will refuse
# to start.
#
# Default:
#       port = 1000

# Radios directory
#
# Set the default radios' searching directory.
#
# Default:
#       radios dir = "/usr/lib/caenrfidd/radios"

# IO ports directory
#
# Set the default IO ports' searching directory.
#
# Default:
#       ioports_dir = "/usr/lib/caenrfidd/ioports"
```

All settings are self-explanatory and they define general daemons settings.

The reader configuration file

By using the `READER_NAME` above variable the daemon will load the reader configuration file named `/etc/caenrfidd/READER_NAME.conf`; so, in our example, the daemon will read file `/etc/caenrfidd/ion.conf` to set specific reader settings.

The CAENRFID Ion

The Ion reader configuration file is by default:

```
# Package generated configuration file
# See the caenrfidd(8) manpage for details

#
# CAENRFID (read only) configuration settings
#

# Reader Manufacturer
#
# Define custom reader manufacturer ID.
#
# Default:
#       vendor = 21336  # CAEN

# Reader Model
#
# Define custom reader model ID.
#
# Default:
#       model = "(null)"
model = "R4301P"    # Ion

# Reader radios list
#
# Define which radios are connected to the reader and each per radio
# configuration data.
#
# Note:
#     * antennae ID numbering starts from 0.
#     * invalid antenna IDs are simply discarted.
#     * no radios and/or no antennae is an error!
#
# radio dummy {
#       antennae_enabled = { 0 }
# }
# radio impinj {
#       antennae enabled = { 0, 1 }
# }
radio impinj {
     antennae_enabled = { 0, 1, 2, 3 }
}

# Reader IO ports list
#
# Define which IO ports are connected to the reader.
#
# Default:
#       ioports = { }
ioports = { ion }

#
# CAENRFID (client writable) configuration settings
#

# Read cycles during continuos inventory (Ver. 3.1)
#
# Define how many rounds the reader should execute during continuos
# inventories.
#
# Default:
#       read cycles = 1

# RF Power
#
# Define the default RF power in mW.
#
# Default:
#       rf_power = 680

# Q value
#
# Define the default Q value for the inventory algorithm.
#
# Allowed values are in 0..15 range.
```

```
#
# Default
#       q = 3

# Session
#
# Define the default session for the query algorithm.
#
# Allowed values are: S0, S1, S2 and S3
#
# Default
#       session = "S0"
```

The settings are self-explanatory.

7 DEVELOPING APPLICATION

Introduction

The Ion R4301P reader supports C, Java and C# applications. Your application communicates with the Ion through one of two RFID services:

- The *caenrfidd* server, which controls the reader by the CAEN RFID communication protocol. For more information on the *caenrfidd* server, see § *Configuring the daemon* page 31. For more information on the CAEN RFID easy2read communication protocol refer to the SDK (Software Development Kits) library (download it from the ion R4301P web page).

There are two ways to use the Ion R4301P from your application:

- Execute the application on a remote host computer. In this case, all the processing is performed by the host.
- Execute the application locally on the Ion R4301P reader. In this case, the application resides on the Ion reader and much of the processing occurs on the reader itself and not remotely on the host computer. In this case the programming languages that are currently supported are C and Java; .NET will be supported in future revisions of the firmware.

Running your application on the Ion reader improves system scalability by minimizing network traffic, since the Ion reader can handle many processing tasks such as data filtering. Furthermore, you can access directly to all the HW interfaces present on the reader and to the peripheral connected to the USB host ports.

In order to simplify the development of application running directly on the Ion R4301P reader, CAEN RFID provides a virtual machine image compatible with VirtualBox. In the following we explain how to use the development virtual machine for that purpose.

Development virtual machine image

Ask for our development virtual machine at CAEN RFID Technical Support.

Note: The virtual machine image is inside a zip archive, each archive should contain at least 3 files named `name.vmdk`, `name.ovf` and `name.mf`. You need all these files to import the virtual machine into your VirtualBox installation.

After having downloaded the zip file you can check its consistency by using GPG, the signature file named `file.sig` and our public key as follow:

```
$ gpg --verify ion-develop-1.0.0.zip.sig ion-develop-1.0.0.zip
gpg: Signature made Tue 08 Jun 2010 01:53:33 PM CEST using DSA key ID 859BDF89
gpg: Good signature from "CAEN RFID packages maintainer <maintainer@support.caenrfid.it>"
```

Installing the development virtual machine

To install the development virtual machine you need the VirtualBox suite version 3.2.12 or above. Please refer to the VirtualBox home site in order to know how to install it on your PC.

After having downloaded the desired release you can import the machine by selecting the *Import Appliance* into the *File* menu of the main VirtualBox application window.

```
$ VBoxManage import <machine>.ovf
```

where `<machine>.ovf` is the filename of the virtual machine.

Note: once the import procedure is finished you may need blessing the new machine by edit the machine settings and then just pressing the *Ok* button.

Using the development virtual machine

Once you started the development virtual machine, you can login with username *user* and password *useruser*. The superuser *root* password is *rootroot* instead.

You can use the development machine as the preferred development environment for the Ion R4301P reader since the installed system is very similar to the system installed on the reader itself. This permits to develop and test your application on the virtual machine before to install the software on the reader avoiding possible mistakes during the job and reducing the development time. The development machine can be also used to test the installation of additional packages so that you will add the packages into the reader only after having verified that the new software do not interfere with the system.

When you develop an application for the Ion R4301P you have to use the C or the Java language and the API – SDK (Software Development Kit) we provide and the TCP/IP protocol as a connection interface, as a result you can test your application running on the development machine and communicating to a remote reader connected to the same TCP/IP network. When you install your application on the reader itself you can just change the remote IP address of the reader with the special address for the local host (127.0.0.1) and the application will communicate with the local reader. As you can note there is no difference between an application running on a remote host or directly inside the reader. Due to this, installing your application into the reader it's just a matter of copying the binary files on the reader using the *scp* command, for example. In the following we show an example of a C application and one of a Java application including the testing phase and the installation on the reader.

Note: If you plan to auto-start your application when the Ion R4301P reader boots, we recommend that you install your application on the reader and start it manually to verify that the executable runs properly. Then you can use the configuration web interface to configure the application to auto-start at boot time.

8 GETTING STARTED WITH C

Development code installation

If you want to develop C applications using our CAENRFID C library you must first install the *libcaenrfid* development package into your development virtual machine[5]:

```
develop@ion:~# sudo aptitude install libcaenrfid-dev
```

New and standard API

Package *libcaenrfid-dev* containes the header files required for developing C applications using the CAENRFID C API (SDK Software Development Kit). Along with the C standard interface, *libcaenrfid-dev* contains also a new CAENRFID API which is designed to be platform independent; you can decide to use the standard CAENRFID SDK if you want to be backward compatible or the new library if you want your code to be platform independent.

To use the standard API you must add the include file `caenrfid_old.h` into your code (see § *Standard API example* below).

Ask for the source files used described in this chapter at CAEN RFID Technical Support.

Standard API example

We show a simple example of an application performing an inventory cycle and presenting the results on the console.

Only four functions are required to perform a basic inventory cycle:

`CAENRFID_Init()`,

`CAENRFID_InventoryTag()`,

`CAENRFID_FreeTagsMemory()`

`CAENRFID_End()`.

The code below shows how to use them.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>

#define  YES I KNOW USING OLD CAENRFID API IS EVIL
#include <caenrfid.h>
#include <caenrfid_old.h>

.........................

int main(int argc, char *argv[])
{
     int i;
     caenrfid_handle_t handle;
        char string[] = "Source 0";
        CAENRFIDTag *tag;
        int size;
        char *str;
        int ret;
```

[5] Commands on the Development Virtual Machine are marked by the suffix `develop:~#`, while commands on the Ion R4301P Reader shell are marked by the suffix `root@ion:~#`.

```
     if (argc < 2) {
             fprintf(stderr, "usage: %s: <server addr>\n", argv[0]);
             exit(EXIT FAILURE);
     }

        /* Start a new connection with the CAENRFIDD server */
        ret = CAENRFID Init(TCP, argv[1], &handle);
        if (ret != CAENRFID StatusOK) {
                fprintf(stderr, "cannot init lib (err=%d)\n", ret);
                exit(EXIT_FAILURE);
        }

     /* Do the inventory */
        ret = CAENRFID InventoryTag(&handle, string, &tag, &size);
        if (ret != CAENRFID StatusOK) {
                fprintf(stderr, "cannot get data (err=%d)\n", ret);
                exit(EXIT_FAILURE);
        }

     /* Report results */
        for (i = 0; i < size; i++) {
                str = bin2hex(tag[i].ID, tag[i].Length);
                if (!str) {
                        fprintf(stderr, "cannot allocate memory!\n");
                        exit(EXIT FAILURE);
                }

                printf("%.*s %.*s %.*s %d\n",
                        tag[i].Length * 2, str,
                        MAX LOGICAL SOURCE NAME, tag[i].LogicalSource,
                        MAX READPOINT NAME, tag[i].ReadPoint, tag[i].Type);

                free(str);
        }

     /* Free inventory data */
        CAENRFID FreeTagsMemory(&tag);

        /* Close the connection */
        ret = CAENRFID_End(&handle);
        if (ret != CAENRFID StatusOK)
                fprintf(stderr, "improper caenrfidlib closure!\n");

     return 0;
}
```

The define `_YES_I_KNOW_USING_OLD_CAENRFID_API_IS_EVIL` is used to suppress compiler warning generated by the standard API.

To compile the code, you can use the `make` command:

```
develop@ion:~#  make inv test std
cc  inv_test_std.c  -lcaenrfid -o inv_test_std
```

When the compile operation is finished you can test the program from your virtual machine:

```
develop@ion:~# ./inv_test_std <ion reader_address>
cae00000000000000000002 Source_0 Ant0 3
cae00000000000000000003 Source 0 Ant0 3
cae00000000000000000007 Source 0 Ant0 3
cae0000000000000000000a Source_0 Ant0 3
```

Where *<ion reader address>* is the address of your ion reader (by default is 192.168.0.1).

To run the same program directly on the RFID reader, first transfer it to the Ion:

```
develop@ion:~# scp ./inv_test_std  <ion reader_address>:
```

and then execute it with the following command:

```
root@ion:~# ./inv_test_std 127.0.0.1
cae00000000000000000002 Source 0 Ant0 3
cae00000000000000000003 Source 0 Ant0 3
cae00000000000000000007 Source 0 Ant0 3
cae0000000000000000000a Source_0 Ant0 3
```

New API usage example

On the contrary to the standard API, only three functions are required to perform a basic inventory cycle using the new API:

`caenrfid_open()`,

`caenrfid_inventory()`

`caenrfid_close()`

The code below shows how to use them.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <string.h>

#include <caenrfid.h>


.......................


      int main(int argc, char *argv[])
{
     int i;
     struct caenrfid_handle handle;
        char string[] = "Source_0";
     struct caenrfid tag *tag;
        size t size;
        char *str;
        int ret;

     if (argc < 2) {
             fprintf(stderr, "usage: %s: <server addr>\n", argv[0]);
             exit(EXIT FAILURE);
     }

        /* Start a new connection with the CAENRFIDD server */
        ret = caenrfid open(CAENRFID PORT TCP, argv[1], &handle);
        if (ret < 0) {
                fprintf(stderr, "cannot init lib (err=%d)\n", ret);
                exit(EXIT_FAILURE);
        }

     /* Do the inventory */
        ret = caenrfid inventory(&handle, string, &tag, &size);
        if (ret < 0) {
                fprintf(stderr, "cannot get data (err=%d)\n", ret);
                exit(EXIT_FAILURE);
        }

     /* Report results */
        for (i = 0; i < size; i++) {
                str = bin2hex(tag[i].id, tag[i].len);
                if (!str) {
                        fprintf(stderr, "cannot allocate memory!\n");
                        exit(EXIT FAILURE);
                }

                printf("%.*s %.*s %.*s %d\n",
                        tag[i].len * 2, str,
                        CAENRFID SOURCE NAME LEN, tag[i].source,
                        CAENRFID READPOINT NAME LEN, tag[i].readpoint,
                    tag[i].type);

                free(str);
        }

     /* Free inventory data */
     free(tag);

        /* Close the connection */
        ret = caenrfid_close(&handle);
        if (ret < 0)
                fprintf(stderr, "improper caenrfidlib closure!\n");
```

```
    return 0;
}
```

Note that this time the `caenrfid_old.h` file is no longer needed.

To compile the code, you can use the `make` command:

```
develop@ion:~# make inv_test
cc     inv_test.c  -lcaenrfid -o inv_test
```

When the compile operation is finished you can test the program from your virtual machine:

```
develop@ion:~# ./inv_test <ion reader address>
cae00000000000000000002 Source_0 Ant0 3
cae00000000000000000003 Source 0 Ant0 3
cae00000000000000000007 Source 0 Ant0 3
cae0000000000000000000a Source_0 Ant0 3
```

Where *<ion reader address>* is the address of your Ion reader (by default equal to 192.168.0.1).

To run the same program directly on the RFID reader, first transfer it to the Ion:

```
develop@ion:~# scp ./inv_test  <ion reader_address>:
```

and then execute it with the following command:

```
root@ion:~# ./inv_test 127.0.0.1
cae00000000000000000002 Source_0 Ant0 3
cae00000000000000000003 Source 0 Ant0 3
cae00000000000000000007 Source 0 Ant0 3
cae0000000000000000000a Source_0 Ant0 3
```

9 GETTING STARTED WITH JAVA

Introduction

This quick guide will explain how to start developing in Java with your CAEN RFID R4301P reader. It starts from the assumption that you have already installed the Development Virtual Machine as described by the relevant chapter and that you will use it as the development platform[6].

Setup the development environment

The first step in setting up the development environment is to install the JDK on the virtual machine, for this purpose type the following commands on a shell window:

```
develop:~# aptitude update
develop:~# aptitude install default-jdk
```

Then, in order to use the SDK (Software Development Kits), you need to download the java libraries from the ion R4301P web page. The libraries (CAENRFIDLibrary.jar and RXTXcomm.jar) are included into the SDK package, extract them from the SDK_4.0.0.zip file using the unzip command from a shell window or using the archive manager (squeeze) from the GUI.

Testing the development environment

At this point you are able to compile and test the first, very simple, Java program:

```
public static void main(String[] args) {
import java.math.BigInteger;
import com.caen.RFIDLibrary.*;

class CAENTest {

        CAENRFIDReader myReader = new CAENRFIDReader();
        CAENRFIDReaderInfo myInfo;
        CAENRFIDLogicalSource myLS;
        CAENRFIDTag myTags[];

        System.out.println("CAENTest starting...");

        try {
            myReader.Connect(CAENRFIDPort.CAENRFID_TCP, args[0]);

            myInfo = myReader.GetReaderInfo();
            String s = myInfo.GetModel();

            System.out.println(s);

            myLS = myReader.GetSources()[0];

            myTags = myLS.InventoryTag();
            if( myTags != null ) {
                 BigInteger bi = new BigInteger(myTags[0].GetId());
                 System.out.println(bi.toString(16));
            } else {
                 System.out.println("No tags found!");
            }
```

[6] Commands on the Development Virtual Machine are marked by the suffix `develop:~#`, while commands on the Ion R4301P Reader shell are marked by the suffix `root@ion:~#`.

```
            myReader.Disconnect();
        } catch (Exception err) {
            System.out.println(err.toString());
        }
    }
}
```

This simple java code instructs the reader, indicated by the IP address passed as the only argument, to execute an inventory cycle and print on the console window the EPC code of the first detected tag, if at least one is detected.

Compile the proposed code into the java bytecode:

```
develop:~# javac -cp "./:./CAENRFIDLibrary.jar:./RXTXcomm.jar" CAENTest.java
```

In the above command line the CAENRFIDLibrary.jar and the RXTXcomm.jar files are supposed to be in the same directory of the source code. As a result you'll obtain the CAENTest.class bytecode file that can be executed with the following command line:

```
develop:~# java -cp "./:./CAENRFIDLibrary.jar:./RXTXcomm.jar" CAENTest 10.0.32.201
```

where 10.0.32.201 is the IP address of your Ion reader.

Running Java applications on the Ion reader

In order to run Java applications on the Ion reader you need first to install the Java runtime. You don't need to install all the Java Development Kit since you can compile your code on the development machine and then transfer the executable bytecode on the Ion reader.

Obtain a shell window on the Ion reader with a SSH client and install the JRE using the following command line:

```
root@ion:~# aptitude update
root@ion:~# aptitude install default-jre
```

Transfer the CAENRFIDLibrary.jar, the RXTXcomm.jar and the CAENTest.class files on the Ion reader using, for example, a secure copy command (scp):

```
root@ion:~# scp CAENRFIDLibrary.jar 10.0.32.201:
root@ion:~# scp CAENTest.class 10.0.32.201:
```

where 10.0.32.201 is the IP address of your Ion reader.

Now you can run the application on your Ion reader using the following command:

```
root@ion:~# java -cp "./:./CAENRFIDLibrary.jar:./RXTXcomm.jar" CAENTest 127.0.0.1
```

assuming that CAENTest.class, CAENRFIDLibrary.jar and RXTXcomm.jar files are in the same directory. As you can notice, the only difference with the command line used to execute the application on the development machine is the IP address of the reader that, in this latter case, is the special localhost address identifying the same machine where the application is running.

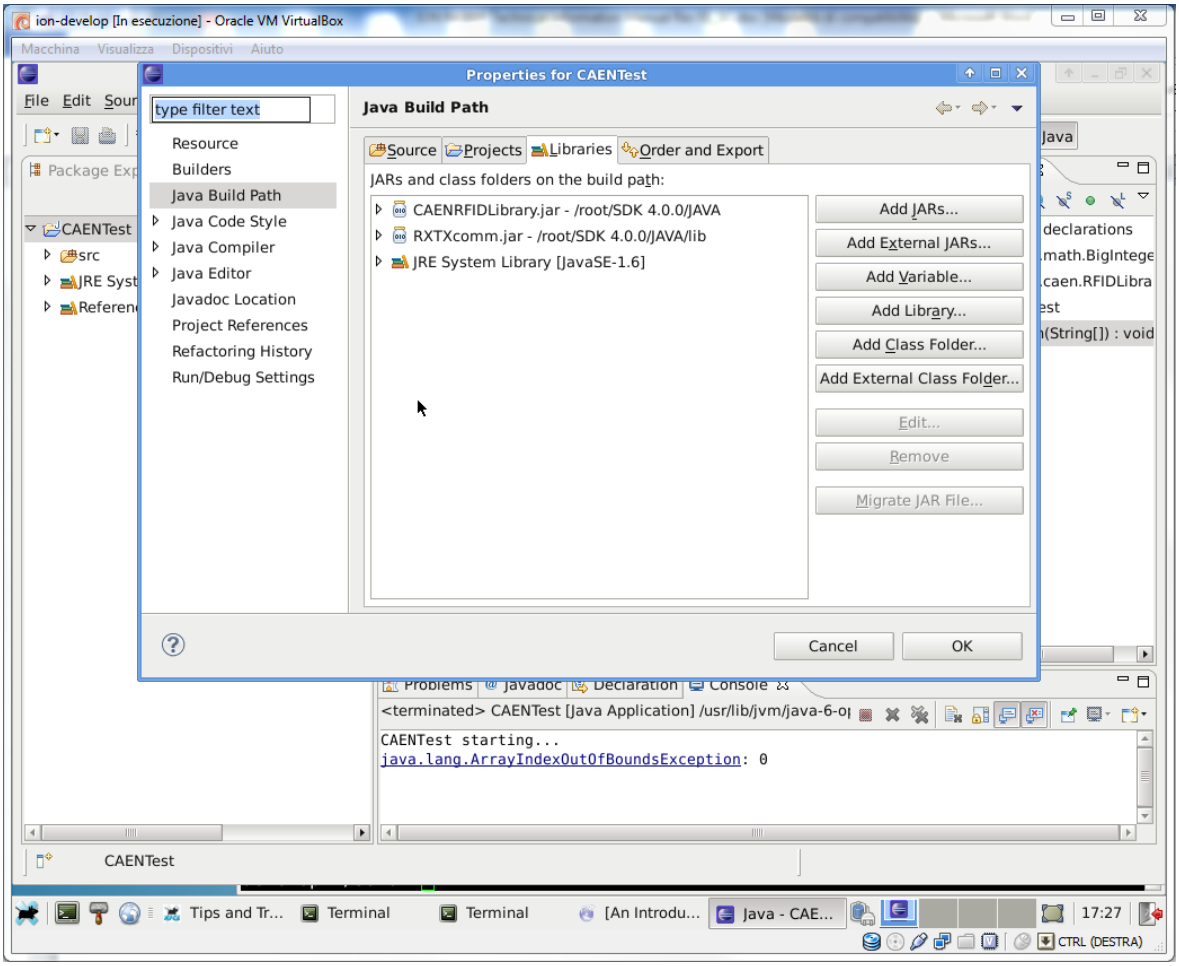
Optional: using the Eclipse IDE

A simple project like the one we described in the previous paragraphs can be managed using a simple text editor and the command line but managing more complex projects may require the usage of an Integrated Development Environment. Here is explained how to install and configure the Eclipse IDE in order to create application for the Ion reader.

The first step is to install the Eclipse IDE itself on the development virtual machine by the following command line:

```
develop:~# aptitude update
develop:~# aptitude install eclipse
```

Once the IDE is installed you can run it by clicking on the Eclipse icon in the Development menu item of the XFCE main menu. After having created a new project you need to add the CAENRFIDLibrary.jar and the RXTXcomm.jar to the project going into the Project Properties, Java Build Paths, Libraries, Add External JARs…:



You are now ready to develop complex projects using the Eclipse IDE on the Development Machine, Enjoy!

10 USING THE GSM

SIM card placement

In order to insert the SIM card inside the reader the cover shall be opened by removing the 6 screws (please see the red circle in the *Fig. 10.2: SIM holder position* page 44 for the correct position).

Fig. 10.1: Bottom cover screws position

The SIM card shall be inserted in the GPRS modem SIM holder.

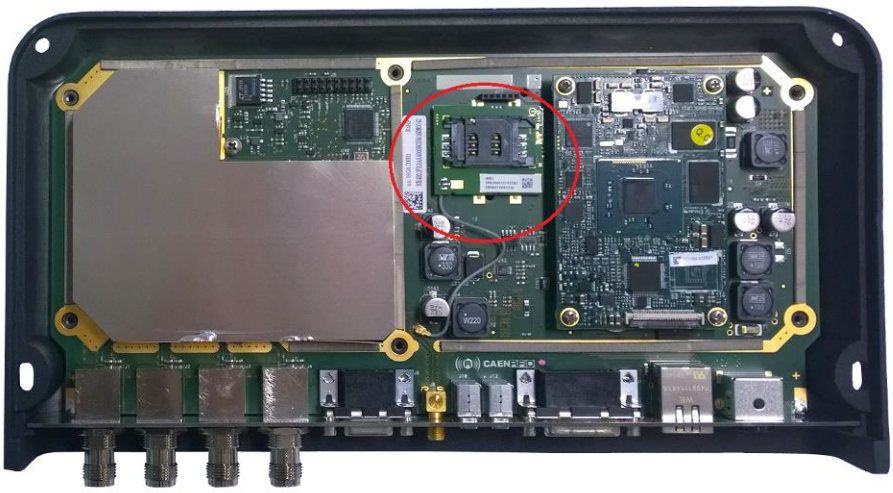


Fig. 10.2: SIM holder position

Warning: Use a standard format SIM card (25×15 mm).

Setting up the GSM module

The Ion R4301P reader has an (optional) internal GSM/GPRS module that can be used to send/receive SMS or as Internet gateway. Here some basic instructions about how to send SMS and how to connect to the Internet using the GSM/GPRS module.

The GSM/GPRS module is connected to the system through the second serial port (the first is used for the console) mapped to the pseudo-file /dev/ttyS2.

We suggest to create a symbolic link for the pseudo-file /dev/ttyS2 in order to have a mnemonic shortcut for the future use of the module:

```
root@ion:~# ln -s /dev/ttyS2 /dev/mobilephone
```

Then you need to switch on the GSM/GPRS module that is switched off by default; this can be obtained invoking the *ion-manager* script in the following way:

```
root@ion:~# ion-manager gsm switch
ion-manager; switching the GSM on/off... ok
```

You can verify if the GSM/GPRS module is correctly activated by issuing the *gsmctl* command:

```
root@ion:~# gsmctl all
gsmctl[ERROR]: ME/TA error 'SIM PIN required' (code 311)
```

The message above is what you should obtain and indicate that the module is active and working, it is just waiting for the PIN of the SIM card.

In the following snippet you can see how to provide the PIN of the SIM card to the module and the reply to the *gsmctl* call when the module is ready to be used:

```
root@ion:~# gsmctl -I "+cpin=NNNN"
root@ion:~# gsmctl all
<ME0>  Manufacturer: Telit
<ME1>  Model: GC864-QUAD
<ME2>  Revision: 10.00.063
<ME3>  Serial Number: 359294031006030
<FUN>  Functionality Level: 1
...
```

Sending a SMS message

Once the GSM/GPRS module is correctly configured, you can send SMS messages in a very simple way form the command shell, for example the following command line will send a SMS message with the body “Hello world!” to the specified number:

```
root@ion:~# gsmsendsms <phone_number> 'Hello world!'
```

Obviously you can create shell scripts that include the gsmsendsms calls in order to automate SMS message delivering.

C language

If you need sending a SMS from a C program the easiest way is by using the `system(3)` function with the command above. Please refer to system Linux man pages for further info.

Surfing the Internet

In order to connect to the Internet using the internal GSM/GPRS moduel, you can use the *wvdial* program. Just edit its configuration file named `/etc/wvdial.conf` as follow:

```
[Dialer Defaults]
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
Init3 = AT+CGDCONT=1,"IP","<YOUR APN>"
Modem Type = Analog Modem
Baud = 115200
New PPPD = yes
Modem = /dev/ttyS2
ISDN = 0
Phone = "<YOUR PHONE NUMBER>"
Password = <YOUR PASSWORD>
Username = <YOUR USERNAME>
```

and replace parameters <YOUR APN>, <YOUR PHONE NUMBER>, <YOUR PASSWORD> and <YOUR USERNAME> according to your network provider.

After having completed the modification of the configuration file, just type the *wvdial* command to start the connection:

```
root@ion:~# wvdial
'--> WvDial: Internet dialer version 1.60
--> Initializing modem.
--> Sending: ATZ
ATZ
OK
--> Sending: ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
OK
--> Sending: AT+CGDCONT=1,"IP","web.omnitel.it"
AT+CGDCONT=1,"IP","web.omnitel.it"
OK
--> Modem initialized.
--> Sending: ATDT*99***1#
--> Waiting for carrier.
ATDT*99***1#
CONNECT
--> Carrier detected.  Waiting for prompt.
```

Here the program may hang for a while but then the output continues:

```
--> Don't know what to do!  Starting pppd and hoping for the best.
--> Starting pppd at Fri Oct  7 10:11:43 2011
--> Pid of pppd: 1949
--> Using interface ppp0
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> local  IP address 109.114.117.107
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> remote IP address 109.114.126.226
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> primary   DNS address 83.224.70.54
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
--> secondary DNS address 83.224.70.77
--> pppd: (~U[07][08]@~S[07][08]~H~[[07][08]
```

If you want, you can make the *wvdial* process run in background using the Ctrl+Z key combination and the *bg* command (as usual for Linux processes). That sequence will give you the possibility to use the same shell for further commands instead of having the needs to open another one.

After that you can check the newly created *pppX* network adapter that correspond to the GSM/GPRS module:

```
root@ion:~# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:109.114.117.107  P-t-P:109.114.126.226  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:64 (64.0 B)  TX bytes:97 (97.0 B)
```

The routing table and the file `/etc/resolv.conf` should be modified accordingly in order to establish the internet connection.

For example, you may need to assign the *ppp0* virtual network card as you default gateway, this can be obtained by deleting the previous default gateway setting and assigning the new one:

```
root@ion:~# route del default
root@ion:~# route add default ppp0
```

11 UPGRADE PROCEDURE

Upgrade the Ion packages

The Ion R4301P reader operating system is a standard Linux Debian installation so the upgrade procedure is based, as any Debian system, on the *aptitude* tool. Here we show the basic usage of this method in order to upgrade or install new packages on the system, for further details on the aptitude tool please refer to the specific documentation (see § Debian Documentation on line resource).

By default, the CAENRFID APT repository list and the Debian APT repository list are already set up into your system so, if you want to upgrade the Ion reader with the latest available packages, you have first to setup your Ion network with the proper settings (see § *READER CONFIGURATION* page 19) and then type at the command shell the following two commands[7]:

```
root@ion:~# aptitude update
root@ion:~# aptitude upgrade
```

Install new Debian software

As a first step, you may need to search for a specific package by using the *apt-cache* command as follow:

```
root@ion:~#  apt-cache search <string>
```

The command searches each string occurrences into all Debian packages and prints the results on standard output. Once you have found the name of the package on the list resulting from the above commands, you can install it by using the apt-get command as follow:

```
root@ion:~# apt-get install <package>
```

For example, if you wish to install the Java Run Time environment (JRE) into your Ion reader, you can proceed as shown in the following:

```
root@ion:~# apt-cache search jre

default-jre-headless - Standard Java or Java compatible Runtime (headless)
default-jre - Standard Java or Java compatible Runtime
```

Then use the command:

```
# apt-get install default-jre

apt-get install defaul-jre
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package defaul-jre
root@ion:~# apt-get install default-jre
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
 ca-certificates ca-certificates-java default-jre-headless defoma file
 fontconfig fontconfig-config hicolor-icon-theme java-common
 libaccess-bridge-java libaccess-bridge-java-jni libasound2 libasyncns0
 libatk1.0-0 libatk1.0-data libavahi-client3 libavahi-common-data
 libavahi-common3 libcairo2 libcups2 libdatrie1 libflac8
 libfont-freetype-perl libfontconfig1 libfontenc1 libgcrypt11 libgif4
 libglib2.0-0 libglib2.0-data libgnutls26 libgpg-error0 libgtk2.0-0
 libgtk2.0-bin libgtk2.0-common libice6 libjasper1 libjpeg62 liblcms1
```

[7] Commands on the Development Virtual Machine are marked by the suffix `develop:~#`, while commands on the ION R4301P Reader shell are marked by the suffix `root@ion:~#`.

```
libmagic1 libnspr4-0d libnss3-1d libogg0 libpango1.0-0 libpango1.0-common
libpcre3 libpixman-1-0 libpng12-0 libpulse0 libsm6 libsndfile1 libtasn1-3
libthai-data libthai0 libtiff4 libvorbis0a libvorbisenc2 libx11-6
libx11-data libxau6 libxcb-render-util0 libxcb-render0 libxcb1
libxcomposite1 libxcursor1 libxdamage1 libxdmcp6 libxext6 libxfixes3
libxfont1 libxft2 libxi6 libxinerama1 libxml2 libxrandr2 libxrender1
libxtst6 openjdk-6-jre openjdk-6-jre-headless openjdk-6-jre-lib sgml-base
shared-mime-info ttf-dejavu-core ttf-dejavu-extra tzdata-java
x-ttcidfont-conf x11-common xfonts-encodings xfonts-utils xml-core
........
```

12 TECHNICAL SPECIFICATIONS

Technical Specifications Table

Frequency Band	902÷928 MHz (FCC part 15) 865.600÷867.600 (ETSI EN 302 208)
Output Power	Up to 32 dBm (1.6W) conducted (ETSI) Up to 30 dBm (1W) conducted (FCC)
CPU	Intel Atom E3815 CPU @ 1.46Ghz
Memory	2Gbytes RAM, 8 Gbytes MicroSD
Operating System	Linux (Debian)
Scripting	Java Virtual Machine
Host Interface Protocols	- CAEN RFID host-to-reader protocol - EPCGlobal LLRP RFID host-to-reader protocol
Antenna Connector	4 TNC Reverse Polarity
Receiving Capability	Gen 2 Dense Reader Mode Management Data rate up to 400 Kbits/s
Standard Compliance	EPC C1 G2/ISO 18000-6C
Digital I/O	13 GPIO pins, TTL level
Connectivity	RS232 Serial Communication (DB9); USB 2.0 High Speed Host Port; Ethernet 10/100/1000BASE-T (RJ45)
Wireless Communication	GSM/GPRS (SMA) (optional) WiFi (SMA) (optional)
Internal Interfaces	MicroSD slot SIM card housing (optional)
IP Rating	IP42
MTBF	135.000 hours
Dimensions	(W)275 x (L)155 x (H)39 mm^3 (10.8 x 6.1 x 1.5 $inch^3$)
DC Power	9÷36 Vdc (30W)
Operating Temperature	-20 °C to 55 °C
Weight	1.3 kg

Tab. 12.1: Technical Specifications Table

Warning: The RF settings must match the country/region of operating to comply with local laws and regulations. The usage of the reader in different countries/regions from the one in which the device has been sold is not allowed.

Reader – Tag Link Profiles

Ion R4301P reader supports different modulations and return link profiles according to EPC Class1 Gen2 protocol.

In the following table are reported all available profiles and the compatibility with the different regulations:

Link profile #	Regulation	Modulation	Return Link
1	ETSI	PR–ASK; f=40kHz	Miller (M=4); f = 300kHz
2	FCC	PR–ASK; f=40kHz	Miller (M=4); f = 250kHz
3	FCC	DSB–ASK; f=160kHz	FM0; f = 400kHz

Tab. 12.2: Reader to tag link profiles

Power Supply Connector Electrical Characteristics

Pin name	Pin #	Parameter	Min	Typ.	Max	Unit
GND	1	Input Current			5	A
VIN	2	Input DC voltage level	9		36	V
		Input Current			5	A

Tab. 12.3: Power Supply Connector Electrical Characteristics

RFID Antenna Ports Electrical Characteristics

Parameter	Min	Typ.	Max	Unit
RF output power – ETSI	0.05		1.6	W
	17		32	dBm
RF output power – FCC	0.05		1	W
	17		30	dBm
Output power vs. power setting accuracy			± 1	dB
Output power setting step		0.1		dB
RF port impedance		50		Ω
Recommended antenna VSWR			1.5:1	-

Tab. 12.4: RFID Antenna Ports Electrical Characteristics

GSM/GPRS Antenna Port Electrical Characteristics

Parameter	Min	Typ.	Max	Unit
RF output power			2	W
			33	dBm
RF port impedance		50		Ω
Recommended antenna VSWR			2:1	-
Absolute maximum antenna VSWR			10:1	-
Allowed antenna gain			1	dBi

Tab. 12.5: GSM/GPRS Antenna Port Electrical Characteristics (only in Mod. WR4301PXGPRS)

WIFI Antenna Port Electrical Characteristics

Parameter	Min	Typ.	Max	Unit
RF output power			100	mW
			20	dBm
RF port impedance		50		Ω
Recommended antenna VSWR			2:1	-
Absolute maximum antenna VSWR			10:1	-
Allowed antenna gain			2	dBi

Tab. 12.6: WIFI Antenna Port Electrical Characteristics (only in Mod. WR4301PXWIFI)

GPIO Port Electrical Characteristics

Parameter	Min	Typ.	Max	Unit
VOL			0.4	V
VOH	2.0		5.0	V
Output current (low level)			12	mA
Output current (high level)			0.3	mA
VIL	-0.5		0.8	V
VIH	2		5.5	V
Input current			0.5	mA
Internal pull-up resistor		10		kΩ

Tab. 12.7: GPIO Port Electrical Characteristics

Serial Port Electrical Characteristics

Pin name	Pin #	Parameter	Min	Typ.	Max	Unit
TX	2	RS232 Voltage Levels	±5	±8		V
		Output Impedance	300			Ω
RX	3	RS232 Voltage Levels	-30		+30	V
		Input Impedance	3	5	7	kΩ
CTS	7	RS232 Voltage Levels	-30		+30	V
		Input Impedance	3	5	7	kΩ
RTS	8	RS232 Voltage Levels	±5	±8		V
		Output Impedance	300			Ω

Tab. 12.8: RS232 Port Electrical Characteristics

USB Port Electrical Characteristics

Pin name	Pin #	Parameter	Min	Typ.	Max	Unit
Vbus	1	Voltage level	4.75V	5.00V	5.25V	V
		Output Current			500	mA
D-	2	VOL	0		0.3	V
		VOH	2.8		3.6	V
		Input Impedance		27		Ω
D+	3	VOL	0		0.3V	V
		VOH	2.8		3.6	V
		Input Impedance		27		Ω

Tab. 12.9: USB Port Electrical Characteristics

13 REGULATORY COMPLIANCE

FCC Compliance

This equipment has been tested and found to comply with Part 15 of the FCC Rules.

NOTE:

a. Any changes or modification not approved by CAEN RFID could void the user's authority to operate the equipment.

b. Ion R4301P, R4301PXGPRS and R4301PXWIFI readers are approved for operation with the following antennas:

- RFID ports: power rating 1W max.; CAENRFID antenna Mod. WANTENNAX020 Circular polarized antenna 8.5dBc (US)
- GSM/GPRS port: Amphenol Mod. TAN015-1100104BS (Linear polarized antenna 1dBi gain 870-960/1710-1990MHz); ordering option WANTENNAX018-
- WIFI port: Pulse mod. W1010 (Dipole antenna 2dBi gain 2.4–2.5GHz); ordering option WANTENNAX021.

c. Use of other than the approved antenna with this unit may result in harmful interference with other users, and cause the unit to fail to meet regulatory requirements.

d. When installing the RFID antennas ensure a minimum separation distance of 25 cm between each antenna and all persons.

e. When installing the GSM/GPRS antenna ensure a minimum separation distance of 20 cm between the antenna and all persons and 20 cm distance between GSM/GPRS antenna and each RFID antennas.

f. When installing the WIFI antenna ensure a minimum separation distance of 20 cm between the antenna and all persons and 20 cm distance between WIFI antenna and each RFID antennas.

g. Ion R4301P, R4301PXGPRS and R4301PXWIFI readers must be professionally installed to correctly set the TX power for the RF cable selected.

h. Maximum RF conducted power at RFID port is 1W (30dBm). The installer shall set the maximum allowed RF power in order to comply with 36dBm E.I.R.P FCC radiated limit taking into account the cable loss used for the connection to the approved antenna according to the following formula:

Max. allowed RF output power (dBm) = 36 – Antenna Gain (dBi or dBic) + Cable Loss (dB)

Example:

Approved antenna WANTENNAX020 and WCAVOAAAX005 (5 m RF Antenna Cable TNC/RP-N type; 0.32dB/m)

Max. allowed RF output power (dBm) = 36 – 8.5dBic + 1.6dB = 29.1dBm

Reference documents: Test report n. *FCC-16574B* [RD2] and *FCC-16547* [RD3].

See § ***FCC Grant part B*** and ***FCC Grant part C*** page 56 and 57 for the ion R4301P FCC Compliance Certificate.

CE Compliance

Reference standard:

ETSI EN 301 489-1 V2.2.0:2017

ETSI EN 301 489-3 V2.1.1:2017

ETSI EN 301 489-17 V2.2.1:2012

ETSI EN 302 208 V3.1.1:2017

ETSI EN 300 328 V2.1.1:2017

ETSI EN 301 511 V12.5.1 2018

ETSI EN 301 489-7 V1.3.1:2006

EN 55032:2012

CEI EN 55024:2013

CEI EN 60950-1:2007 +/A11:2010 +/A1:2012 +/A12:2012

CEI EN 50364:2011

See § ***CE Declaration of Conformity*** page 55 for the ion R4301P CE Compliance Certificate.

RoHS EU Directive

The Ion R4301P Reader is compliant with the EU Directive 2011/65/EU on the Restriction of the Use of certain Hazardous Substances in Electrical and Electronic Equipment (RoHS2).

ion R4301P
CE DECLARATION OF CONFORMITY

We

CAEN RFID Srl
Via Vetraia, 11
55049 Viareggio (LU) Italy
Tel.: +39.0584.388.398 Fax: +39.0584.388.959
Mail: info@caenrfid.com
Web site: www.caenrfid.com

herewith declare under our own responsibility that the products:

WR4301PXAAAA R4301P - Ion - RFID UHF Portal Reader

WR4301PXGPRS R4301P - Ion - RFID UHF Portal reader with GPRS

WR4301PXWIFI R4301P - Ion - RFID UHF Portal reader with WIFI Interface

correspond in the submitted version to the following standards:

ETSI EN 301 489-1 V2.2.0:2017

ETSI EN 301 489-3 V2.1.1:2017

ETSI EN 301 489-17 V2.2.1:2012

ETSI EN 302 208 V3.1.1:2017

ETSI EN 300 328 V2.1.1:2017

ETSI EN 301 511 V12.5.1 2018

ETSI EN 301 489-7 V1.3.1:2006

EN 55032:2012

CEI EN 55024:2013

CEI EN 60950-1:2007 +/A11:2010 +/A1:2012 +/A12:2012

CEI EN 50364:2011

and declare under our sole responsibility that the specified products meet the principle requirements and other applicable regulations of directives 2014/53/EU (RED) and 2011/65/EU (RoHS2)

Date: 22/03/2019

CAEN RFID srl
Via Vetraia, 11
55049 VIAREGGIO - ITALY
VAT IT 02032050466

Adriano Bigongiari (Chief Executive Officer)

On the basis of this declaration, these products will bear the following mark: CE

ion R4301P
FCC GRANT PART B

TCB

GRANT OF EQUIPMENT AUTHORIZATION

TCB

Certification
Issued Under the Authority of the
Federal Communications Commission
By:

EMCCert Dr. Rasek GmbH
Stoernhofer Berg 15
91364 Unterleinleiter,
Germany

Date of Grant: 05/30/2016

Application Dated: 05/30/2016

CAEN RFID srl
via Vetraia, 11 - 55049 Viareggio (LU) - ITALY
Viareggio, 55049
Italy

Attention: Adriano Bigongiari , CEO

NOT TRANSFERABLE

EQUIPMENT AUTHORIZATION is hereby issued to the named GRANTEE, and is VALID ONLY for the equipment identified hereon for use under the Commission's Rules and Regulations listed below.

FCC IDENTIFIER: UVECAENRFID022
Name of Grantee: CAEN RFID srl
Equipment Class: **Part 15 Class B Computing Device Peripheral**
Notes: **R4301P/R4301PXGPRS/R4301PXWIFI-ION-RFID UHF Reader**

Grant Notes	FCC Rule Parts	Frequency Range (MHZ)	Output Watts	Frequency Tolerance	Emission Designator
CC	15B	-			

CC: This device is certified pursuant to two different Part 15 rules sections.

ion R4301P
FCC GRANT PART C

TCB **TCB**

GRANT OF EQUIPMENT AUTHORIZATION

Certification
Issued Under the Authority of the
Federal Communications Commission
By:

EMCCert Dr. Rasek GmbH
Stoernhofer Berg 15
91364 Unterleinleiter,
Germany

Date of Grant: 05/30/2016

Application Dated: 05/30/2016

CAEN RFID srl
via Vetraia, 11 - 55049 Viareggio (LU) - ITALY
Viareggio, 55049
Italy

Attention: Adriano Bigongiari , CEO

NOT TRANSFERABLE

EQUIPMENT AUTHORIZATION is hereby issued to the named GRANTEE, and is VALID ONLY for the equipment identified hereon for use under the Commission's Rules and Regulations listed below.

FCC IDENTIFIER: UVECAENRFID022
Name of Grantee: CAEN RFID srl
Equipment Class: **Part 15 Spread Spectrum Transmitter**
Notes: **R4301P/R4301PXGPRS/R4301PXWIFI-ION-RFID UHF Reader**

Grant Notes	FCC Rule Parts	Frequency Range (MHZ)	Output Watts	Frequency Tolerance	Emission Designator
CC	15C	902.75 - 927.25	1.0		

Power output listed is peak conducted. The antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter except in accordance with FCC accepted multi-transmitter procedures. End-users and installers must be provided with antenna installation instructions and transmitter operating conditions for satisfying RF exposure compliance.

This device is required to be professionally installed by a properly trained technician.

CC: This device is certified pursuant to two different Part 15 rules sections.