Keonn
AdvanNet Manager 2.0.x
User Guide

Version**: 1.3**
Date: **21th March 2014**
Author: **Keonn Technologies S.L.**

Change Document record

Date	revision	Changes
20th October 2013	1.0	Initial version of the document.
11th December 2013	1.1	Added new antenna edition features.
24th January 2014	1.2	Added advance configuration chapter.
21th March 2014	1.3	Updated RF region list.

Products Covered by this Guide

This guide pertains to Keonn AdvanNet Software version 2.0 and newer releases. Keonn AdvanNet may work with: AdvanShelf, AdvanTrack, AdvanPanel, AdvanSafe, AdvanFitting and other systems.

Content table

Captures

Tables

Common terminology

Make sure all concepts and definitions in this chapter are fully understood before reading further this document. All concepts here defined are used throughout the document and are of special meaning and importance.

- **Keonn device:** in terms of AdvanNet, each Keonn product that requires a driver and therefore is controller under AdvanNet is a device.
- **Device type:** a device type defines the characteristics of a particular Keonn product. For example AdvanPanel has a device type that defines its characteristics from AdvanNet point of view.
- **Device instance**: a physical device is called an instance. Several different instances of a device type can exist in the same AdvanNet installation.
- **Device id**: each device instance has a unique id that is used to identify the device.
- **REST interface:** a software architecture for distributed applications that follows some principles like simplicity, scalability, decoupling, etc.
- **REST resource**: a resource can be any addressable concept, the representation of resources are usually a document with the current state of the resource.
- **Read cycle:** a complete iteration through all antennas a system has. The term system can have more than one sense:
  - A single RFID device.
  - Multiple RFID devices working together.

1- Introduction

1.1- About this guide

This guide covers the configuration and operation of Keonn AdvanNet Software.

1.2- Intended audience

Intended readers of this guide are systems engineers and IT staff with a minimum understanding of software configuration, communication protocols like TCP/IP and xml documents.

1.3- What is Keonn AdvanNet software

What AdvanNet is:

- A Java headless application that can run on any Java enabled platform.
- A common driver platform for Keonn products, a software layer that homogenizes all Keonn products under a common interface.
- A simple yet powerful edgeware software that can easily connect Keonn products to 3rd party systems; either RFID middleware or end-user customer applications.
- A powerful GUI (web based) test environment for any RFID deployment that uses Keonn products.

What AdvanNet does:

- Transforms low-level data into a much more usable xml messages.
- Allows filtering the low-level data.
- Offers a synchronous REST interface to access devices operations: inventory and others.
- Sends asynchronous xml messages to interested 3rd party IT systems.
- Offers a web interface to be used as a testing and deployment tool.

AdvanNet benefits:

- Takes RFID deployments to a new level of simplicity. AdvanNet is easily configured and offers a clean and simple interface the higher software levels.
- Decreases deployment time:
  - AdvanNet solves the low-level communications with RFID devices and let developers focus on business development.
  - AdvanNet is a useful and powerful deployment testing tool.
- Guarantees the best performance for Keonn products.

What AdvanNet is not:

- AdvanNet is not an RFID middleware; Keonn AdvanNet focuses solely on Keonn products and does not provide drivers for other RFID readers and devices.
- AdvanNet is not an end-user application; Keonn AdvanNet is the first software layer in any software architecture that includes Keonn products.

1.4- General information

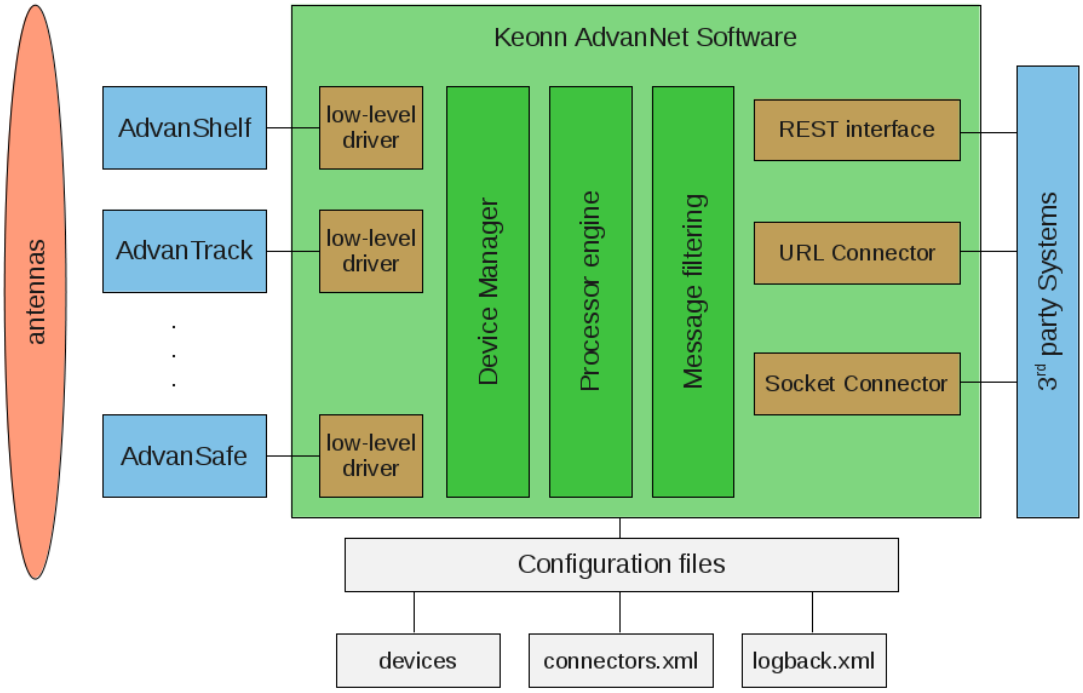
Keonn products usually need a piece of software to operate them, configure, integrate and make life easier to anyone using them. With that goal in mind Keonn provides a software to solve that: Keonn AdvanNet Software.

Keonn AdvanNet Software is a command line application written in Java that acts as the driver or firmware for any Keonn products. It can be understood as a bridge between the complexities of the low level required to talk to electronic modules and the higher level messages any IT infrastructure can manage. In other words; it's a very simple yet very flexible edge-ware.

The main characteristics of Keonn AdvanNet are:

- Knows the low-level protocol of Keonn products.
- Allows the configuration of Keonn products.
- One single instance can control several Keonn products.
- Generates inventory related xml messages.
- Offers a REST interface to access devices synchronously.
- Implements a publisher/subscriber pattern to send asynchronous data to 3rd party IT systems.
- Allows an intelligent routing of data; messages can be conditionally delivered based on data source, and other parameters.
- Allows integration with any system, integration mechanisms are universal, they are not chained to any programming language, architecture whatsoever.

The following is a block diagram centered around Keonn AdvanNet Software.



Capture 1: Keonn AdvanNet Software block diagram

1.5- Feature list

The following is a non comprehensive list of features:

Device control:

- Ability to control multiple Keonn devices.
- Ability to control each device with a different mode. For example it is possible to configure an AdvanShelf to execute an inventory twice a day while an AdvanFitting runs interrupted during work hours.

Data processing:

- Smart processing: data processing includes fork points, smart routing, processing actions, filters, etc.
- Data filtering by any of tag characteristics: EPC, rssi, location, phase, etc.
- Handle automatically GS1 Global Trade Item Number (GTIN) SGTIN-96 encoding. Can decode automatically tags using SGTIN-96.

Data export:

- Export inventory data to CSV files (Microsoft Excel compatible).
- Export inventory data to any SQL compliant database.
- Export inventory data to MS-Access by using an ODBC bridge.

Interfaces:

- Web interface for testing and deploy purposes.
- REST interface to control and access devices synchronously.
- Asynchronous data is sent out to 3rd party software connected to port 3177.
- The REST interface lets AdvanNet™ to be interfaced in virtually any programming environment.

1.6- Available examples

A list of growing examples can be downloaded from:
https://sites.google.com/a/keonn.com/support/software-downloads/samples

2- AdvanNet installation

2.1- Requirements

As a Java application, Keonn AdvanNet can run in any Java enabled platform. The following table summarizes the requirements in the most common operating systems. Please note this table has to be taken with caution, values may depend on the load due to other running software on the host machine.

Operating System	Processor	Java Virtual Machine	Minimum RAM
Windows XP or higher	Intel Pentium IV or higher. AMD Athlon 3000 or higher.	No JRE required. Windows installer is bundled with an embedded JRE.	32 Mb of dedicated RAM. 64 Mb of dedicated RAM for installations with several devices.
GNU/Linux & Unixes	Pentium IV or higher. AMD Athlon 3000 or higher.	JRE 1.6 or higher	32 Mb of dedicated RAM. 64 Mb of dedicated RAM for installations with several devices.
Mac OSX[1]	Intel Core Duo	JRE 1.6 or higher	32 Mb of dedicated RAM. 64 Mb of dedicated RAM for installations with several devices.

Table 1: System requirements

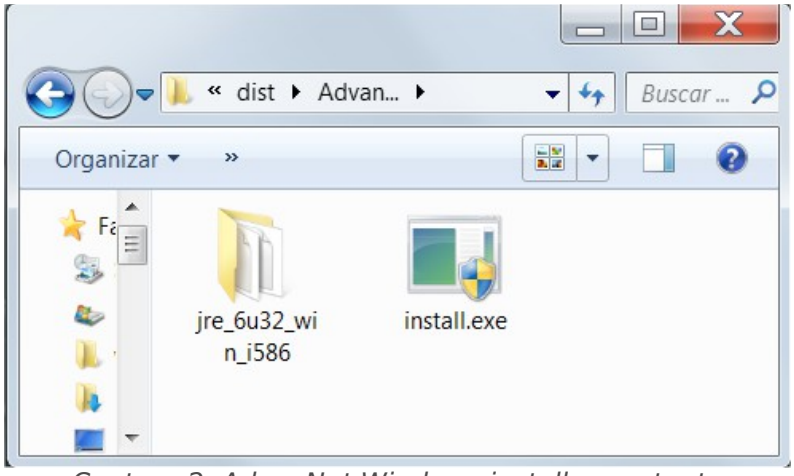
1 *Some features will not work in Mac OS. For example the keyboard emulation.*

2.2- Installation

2.2.1- Windows

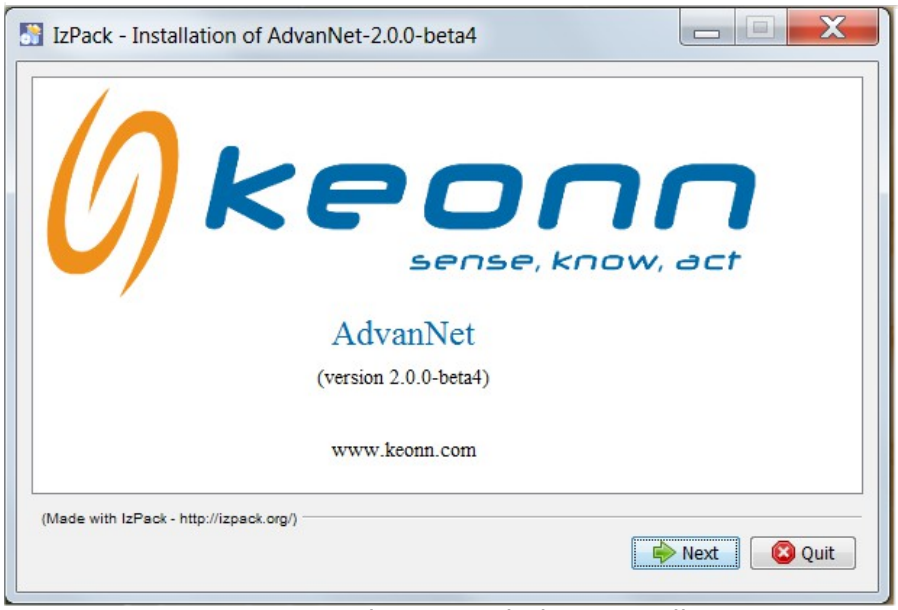
Keonn AdvanNet Software is provided as a self-contained installer that will copy all the needed files to run AdvanNet in your system. Note that administrator rights may be requested in order to install AdvanNet to privileged directories.

1. Unzip the installer.



Capture 2: AdvanNet Windows installer contents

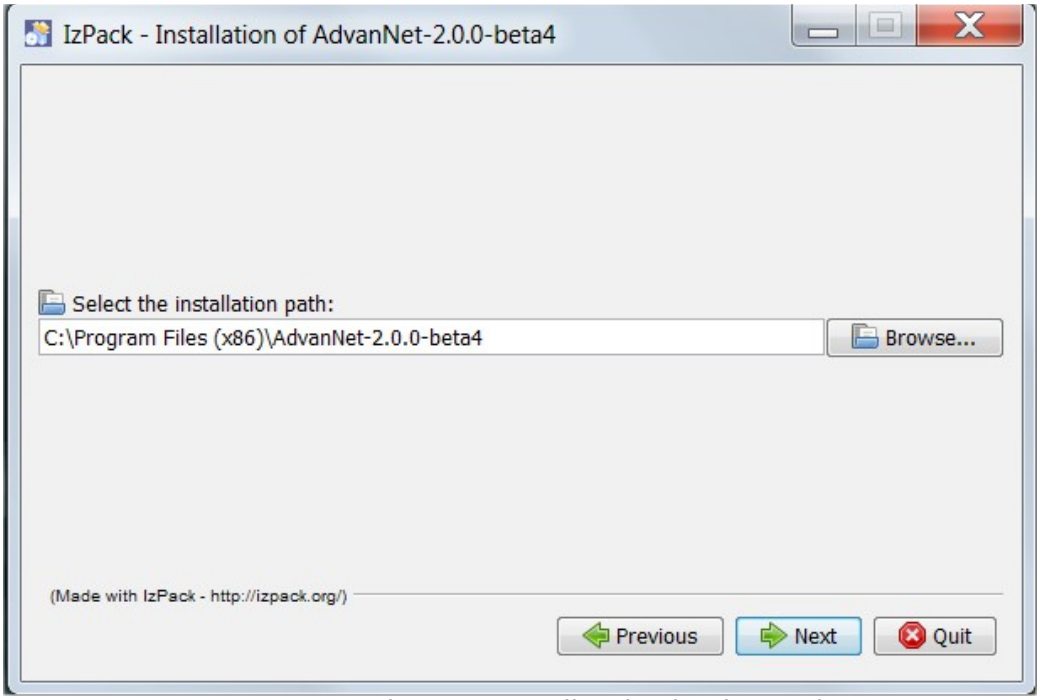
2. Run installer.bat and follow instructions



Capture 3: AdvanNet Windows Installer

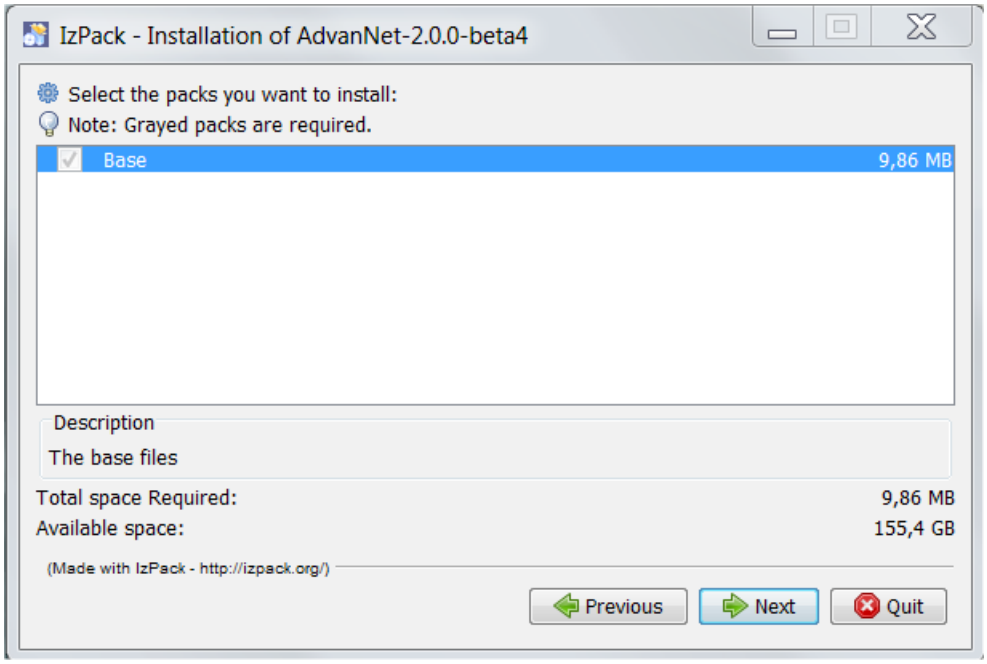
3. Select destination folder

   By default the destination folder contains the version of AdvanNet, allowing several installations in the same machine.

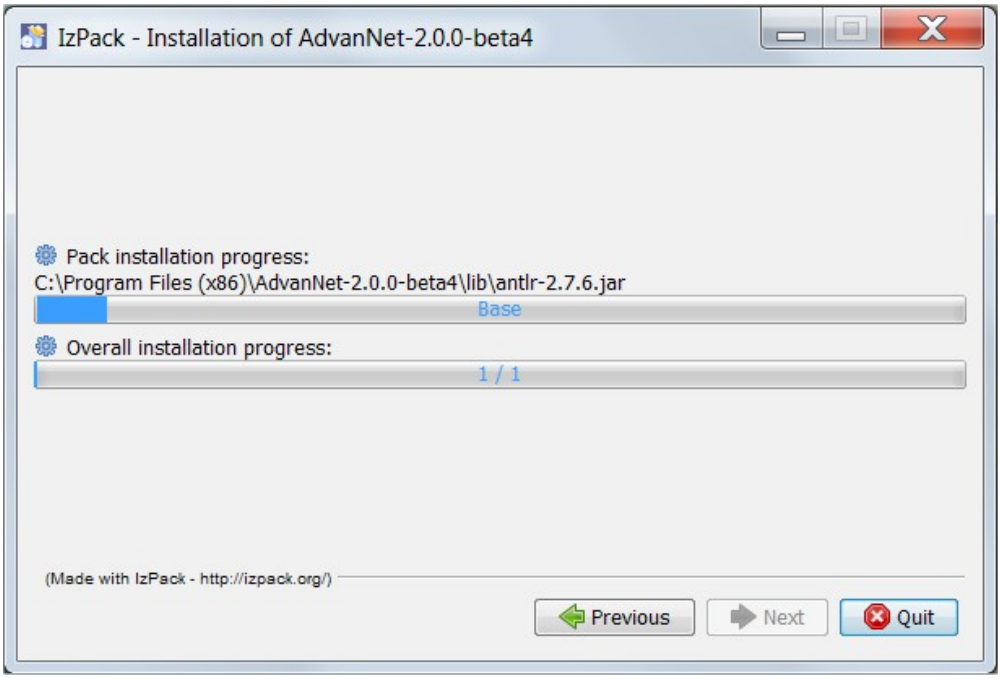


Capture 4: AdvanNet Installer destination path

4. Start installation: press *Next* when prompted.



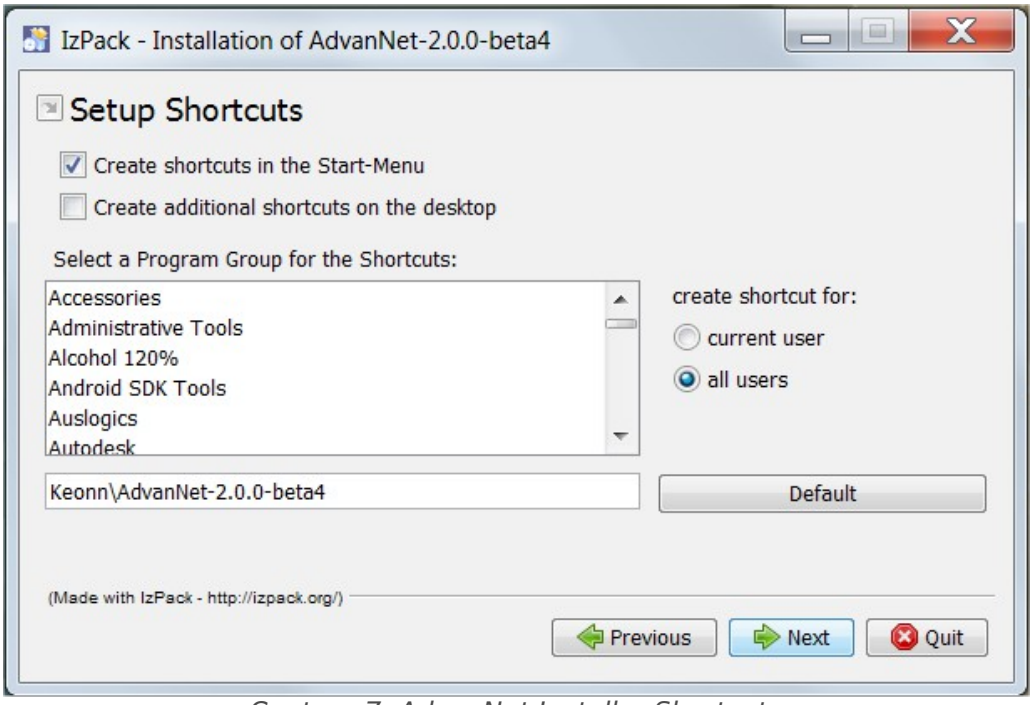
Capture 5: AdvanNet Installer files



Capture 6: AdvanNet Installer in action

5. Create shortcuts

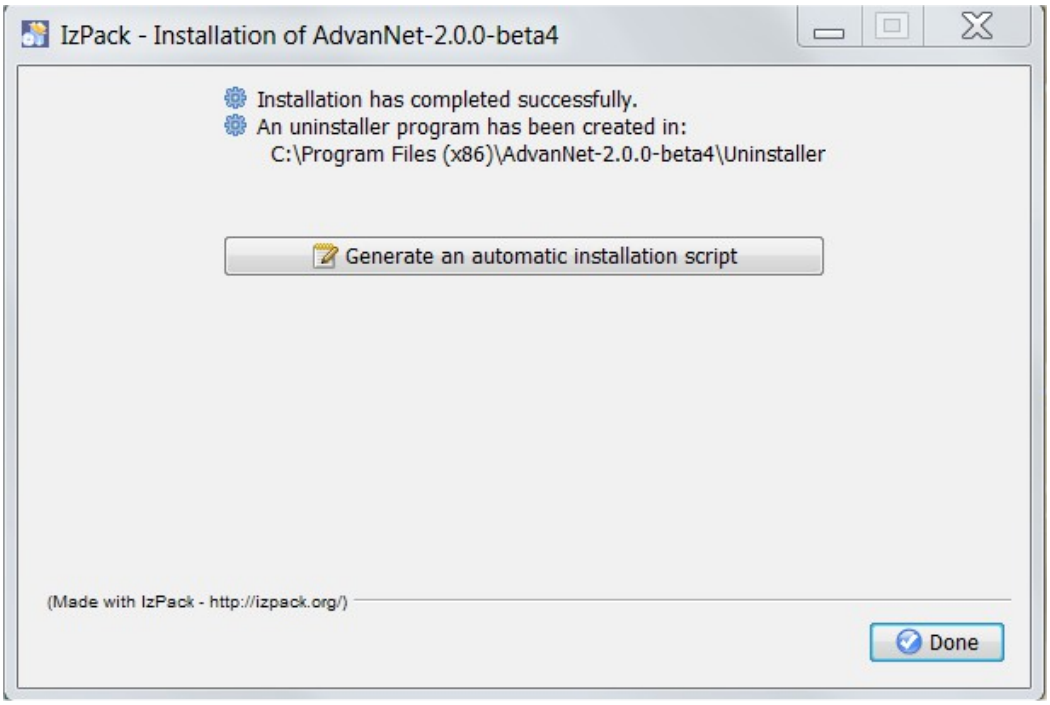
   If several versions are installed make sure no collisions are made with the shortcuts. Default options will always create shortcuts based on the version.



Capture 7: AdvanNet Installer Shortcuts

6. Installation end

   Just click *Done*, here is no need to create the automatic script.



Capture 8: AdvanNet Installer end

For your reference only, the directory structure created at the install directory is as follows:

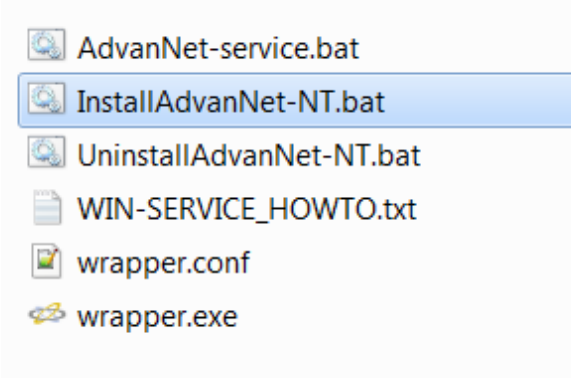
- *lib* & *native-lib*: Java libraries and native libraries.
- *bin*: executable files.
- *conf*: configuration files.
- *templates*: template files.
- *jre*: Java run time.
- *uninstaller*: uninstaller files.

2.2.1.1- Installing AdvanNet as a Windows service

After a regular installation of AdvanNet, it is possible to make it start automatically by creating a Windows service. The steps to create a new Windows service are described below:

1 – Navigate to your AdvanNet installation folder

2 – Execute /bin/win-service/InstallAdvanNet-NT.bat and wait until finish.



Capture 9: Install AdvanNet service

2.2.2- Linux, MacOS X and other Unixes

As a Java application, AdvanNet can run on any platform that supports JRE 1.6 or higher.

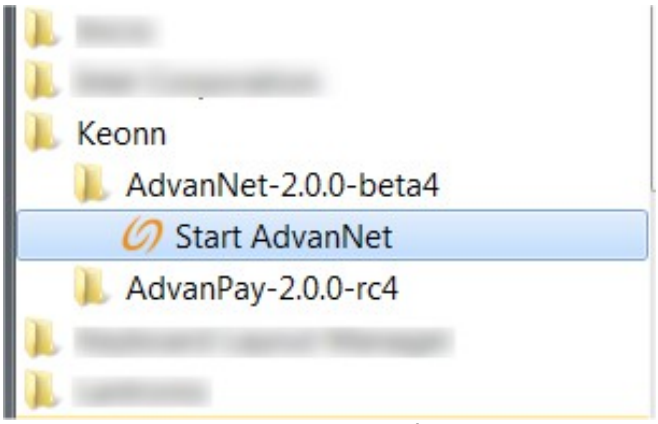
On this platforms the installer steps are:

1. Uncompress the AdvanNet library zip file
2. Install JRE1.6

2.3- Starting and stopping Keonn AdvanNet

2.3.1- Windows

To start Keonn AdvanNet execute the generated desktop shortcut or browse the application menu up to keonn/AdvanNet.



Capture 10: Start AdvanNet

A command window will open in the background.

After some seconds a new web browser tab with AdvanNet main page will load.

To stop AdvanNet just close the command window and the system will shut down gracefully. When stopped, the system waits for any operation in progress to finish and that can cause a delay, the exact delay will depend on the configuration.[2]

---

2 *For instance if the system is executing a read cycle of dozens of antennas, the shutdown will wait until the current cycle ends.*

Supported browsers are Google Chrome and Mozilla Firefox. Please make sure to use one of those browsers.

2.3.2- GNU/Linux, Mac OSX and other Unixes

To start Keonn AdvanNet execute bin/AdvanNet.sh either from the windows GUI or from a command line. A command window will open.

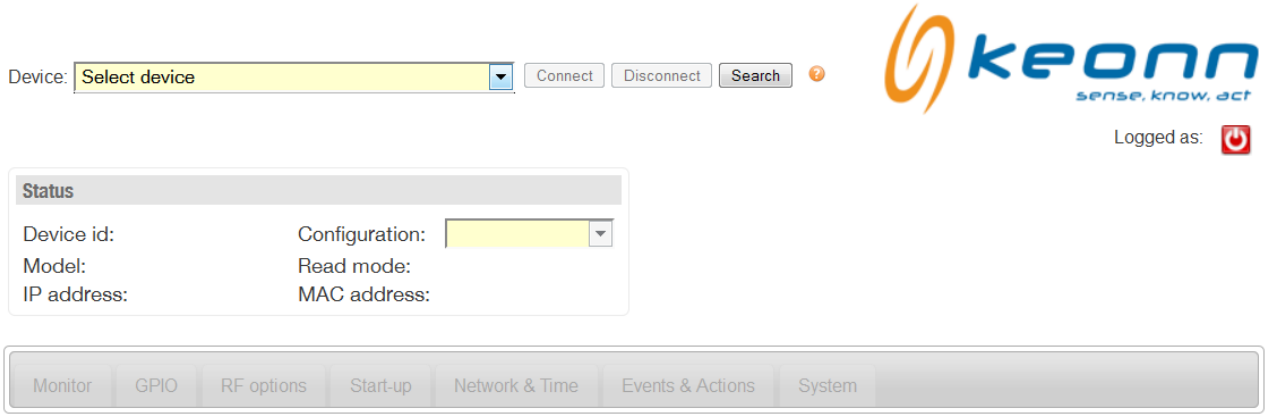
To stop it just close the command window and the system will shut down gracefully.

Make sure all .sh have executable permissions.

2.4- Verify AdvanNet starts successfully

Upon starting AdvanNet, a browser will open with the main page of AdvanNet GUI application.

You should see a page similar to this one:

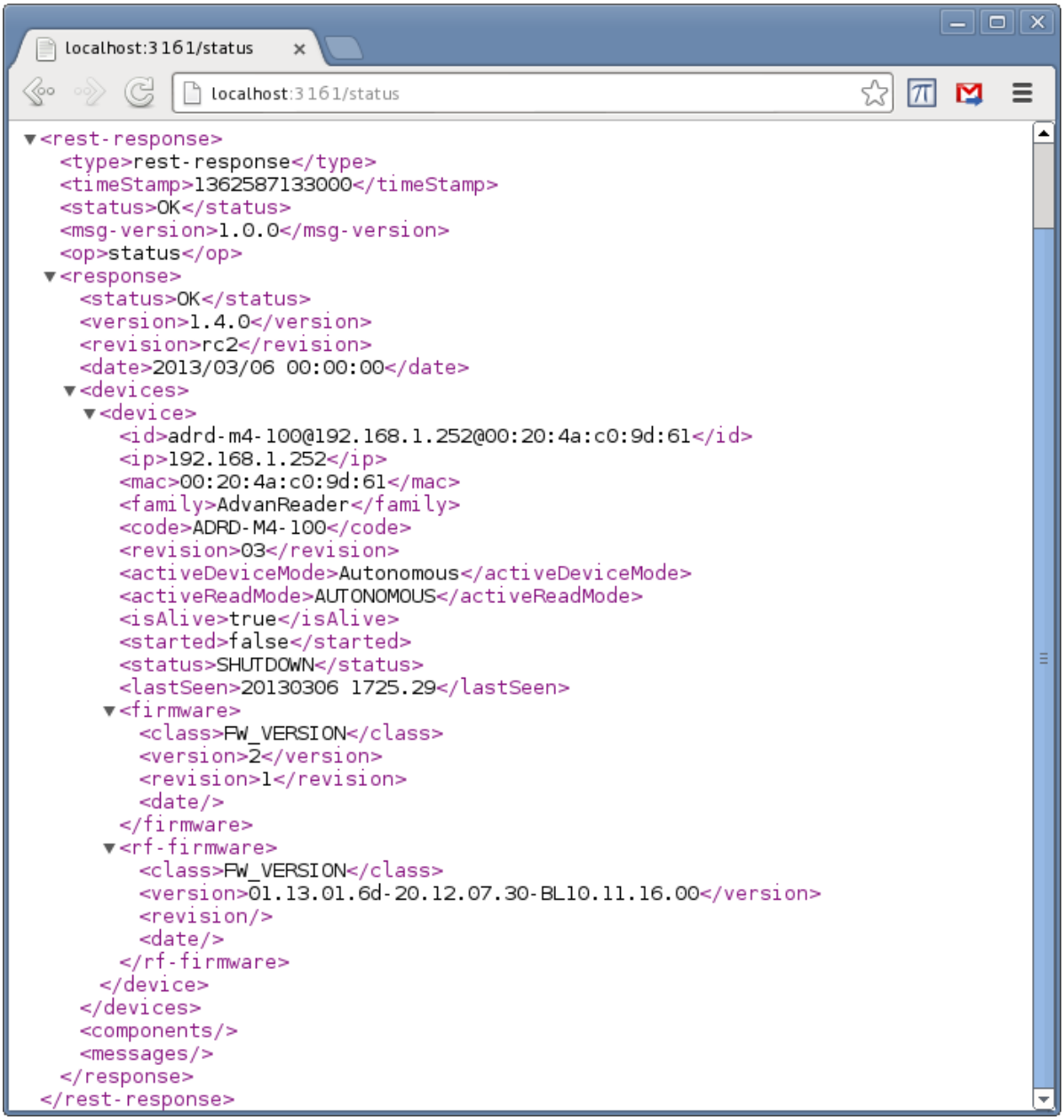


Capture 11: AdvanNet main page

A useful page to verify all systems have loaded successfully is the status page. The status page can be accessed at http://localhost:3161/status

If AdvanNet has started correctly, you should see a page with an xml file stating all the defined parameters.

If you see error messages on the status page, please go to the troubleshooting chapter to fix the error.




```
▼<rest-response>
  <type>rest-response</type>
  <timeStamp>1362587133000</timeStamp>
  <status>OK</status>
  <msg-version>1.0.0</msg-version>
  <op>status</op>
 ▼<response>
   <status>OK</status>
   <version>1.4.0</version>
   <revision>rc2</revision>
   <date>2013/03/06 00:00:00</date>
  ▼<devices>
   ▼<device>
     <id>adrd-m4-100@192.168.1.252@00:20:4a:c0:9d:61</id>
     <ip>192.168.1.252</ip>
     <mac>00:20:4a:c0:9d:61</mac>
     <family>AdvanReader</family>
     <code>ADRD-M4-100</code>
     <revision>03</revision>
     <activeDeviceMode>Autonomous</activeDeviceMode>
     <activeReadMode>AUTONOMOUS</activeReadMode>
     <isAlive>true</isAlive>
     <started>false</started>
     <status>SHUTDOWN</status>
     <lastSeen>20130306 1725.29</lastSeen>
    ▼<firmware>
      <class>FW_VERSION</class>
      <version>2</version>
      <revision>1</revision>
      <date/>
     </firmware>
    ▼<rf-firmware>
      <class>FW_VERSION</class>
      <version>01.13.01.6d-20.12.07.30-BL10.11.16.00</version>
      <revision/>
      <date/>
     </rf-firmware>
    </device>
   </devices>
   <components/>
   <messages/>
  </response>
 </rest-response>
```


Capture 12: AdvanNet status page

Under some circumstances the loading of the GUI application is faster than the device discovery process. In that scenario:

- Reload the web page (F5) until the devices appear in the device drop-down list.
- Devices are updated automatically, but that may require some seconds.

3- AdvanNet GUI application

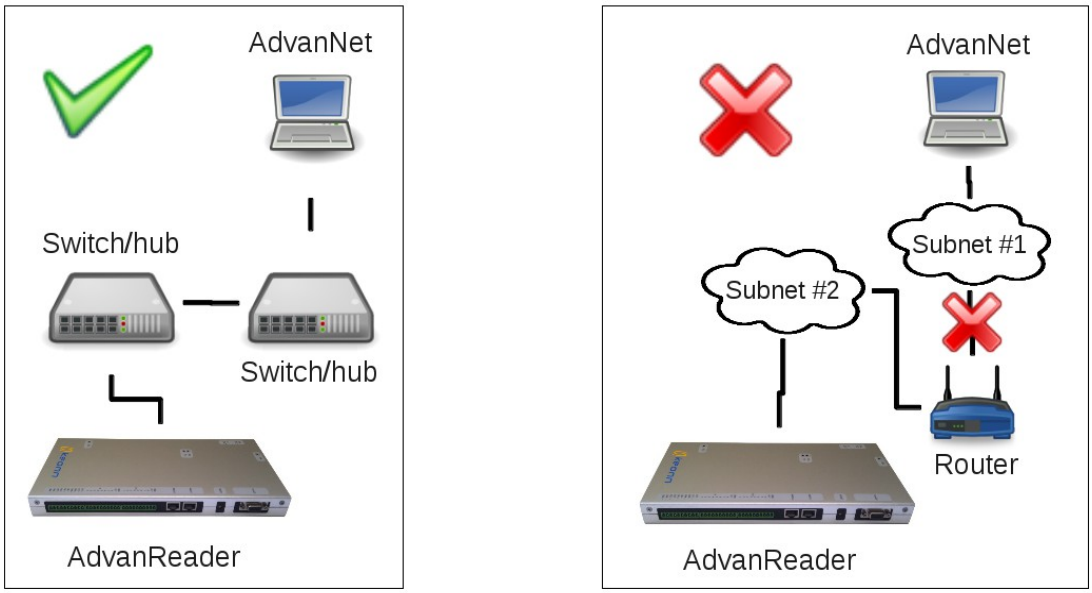
In this chapter AdvanNet is presented as a 'zero configuration' system. That means AdvanNet can be used to control Keonn devices almost effortlessly from the user point of view.

3.1- Device discovery

Once AdvanNet is started, an automatic self-discovery process is performed, the process identifies any Keonn product connected in the same sub-network AdvanNet is running[3].

There is only one requirement for the self-discovery process to work: Keonn devices and AdvanNet installation must be in the same sub-net, in other words, no routers must exist between AdvanNet and Keonn devices.

The following diagram show two possible connection scenarios: a right approach versus a wrong approach.

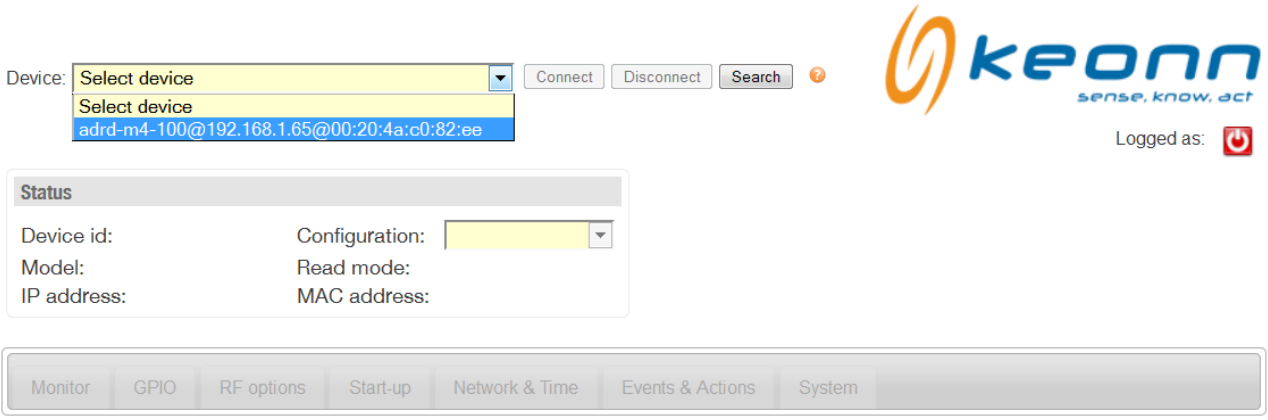


Capture 13: Device Discovery Connections

For statically configured devices, the existence of routers is not an issue. Please read on to learn about statically configured devices.

Devices that are configured manually but can't be reached are showed in gray in device candidate list.

3 *Keonn products based on AdvanReader-m4-100 with hardware revision .02 and higher*



Capture 14: Device selection

3.1.1- Device naming

Devices are loaded from two different sources:

- Automatically discovered devices.
- Configured devices.

According to its source, the device ID is different.

Device source	ID
Discovered	product_code@ip_address@mac_address. The ID helps us identifying the type of device and their network location. Example: adrd-m4-100@192.168.1.5@00:20:4a:de:e7:98
Configured	ID is defined in the configuration file

Table 2: Device IDs

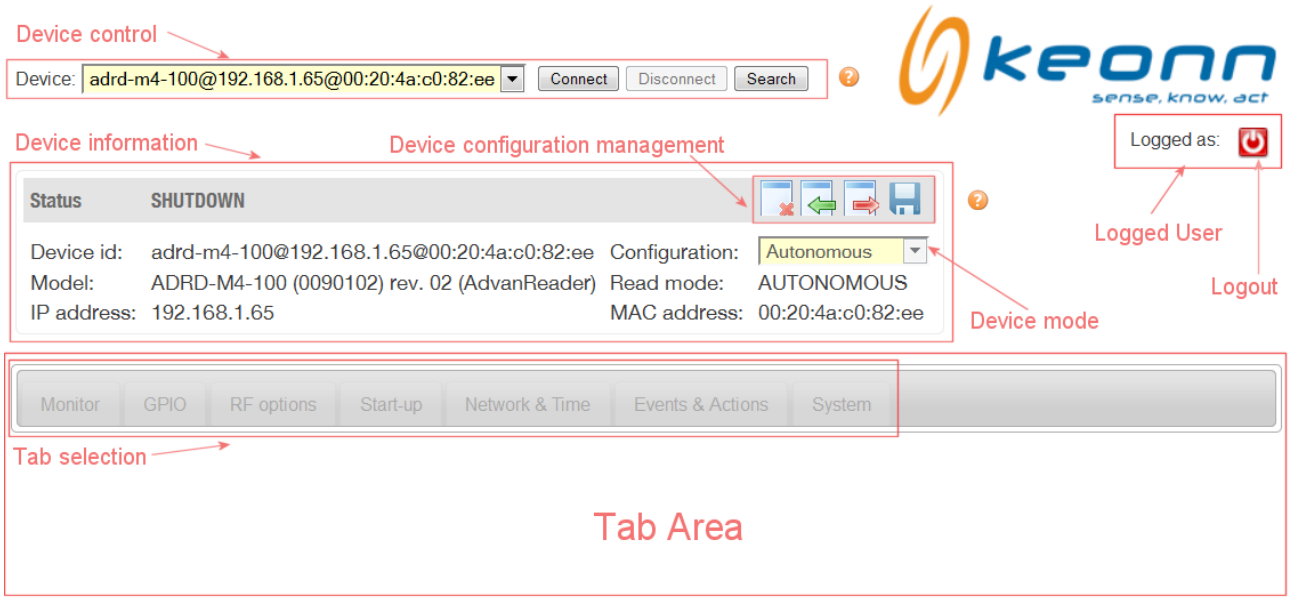
3.2- AdvanNet GUI organization

After starting AdvanNet, a new web browser will appear with AdvanNet GUI web application. If the page is not opened, you can easily access to it by typing http://localhost:3161/index.html on a web browser.

Only Google Chrome and Firefox are supported by AdvanNet.

The AdvanNet GUI application is the Keonn testing and deployment tool. From this page Keonn devices can be managed, configured and monitored.

The main page contains everything is required to start operating Keonn products. The page is organized as follows:



Capture 15: GUI application organization

3.2.1- Device control

Allows selecting devices and connecting/disconnecting them from the current AdvanNet instance.

- **Connect:** Establishes a connection between the device and the GUI application. After the connection is successfully established, the control tabs are activated, and the device info box is populated automatically.
- **Disconnect**: finally, to disconnect and release resources we must use the *disconnect* button. A disconnected device can be used by another AdvanNet instance.
- **Search**: Use this button to add manually the IP of a device you want to control through AdvanNet(useful when the auto-descovery system can't find it).

One device can only be connected by one instance of AdvanNet.
Although several AdvanNet instances can run together without causing any problem.

3.2.2- Logged user

AdvanNet supports the use of username/password for authentication.

This feature is disabled by default but can be enabled. See Advanced configuration chapter.

3.2.3- Log-out

Icon to log out from the current session. The log-out action makes sense only if user authentication has been enabled.

3.2.4- Device info

Info box with the following device characteristics:

- Device id: identifier for the device.
- Model: device model.
- Read mode: currently active RFID read mode (more on this later on)
- IP address: current IP address of the device.
- MAC address: hardware MAC address of the device.

3.2.5- Device mode

Active wide device configuration.

On connected devices it is possible to change the active device configuration.

A device mode defines how the hardware will act. This is best understood with a few examples:

- An anti-theft system may work in different device modes:
  - EPC pattern recognition.
  - NXP EAS bit detection
  - Tag memory access
  - Etc

  Each example could be configured as a device mode and will control the internal operation of the RF hardware.
- A general purpose RF reader may have several working modes:
  - Autonomous
  - Sequential: to be used with multiplexers.
  - Etc

3.2.6- Device configuration management

Allow to handle the system configuration:

- Import: imports a configuration file for the selected device.
- Export: exports device's current configuration to configuration file (xml).
- Save: persists the current active configuration of the device. Once the configuration is persisted it will be applied to the same device even after a complete system reboot.

- Reset: Returns all the device's parameters to the default value.

3.2.7- Tab selection

By clicking on a tag we will change the active tab.

3.2.8- Tab area

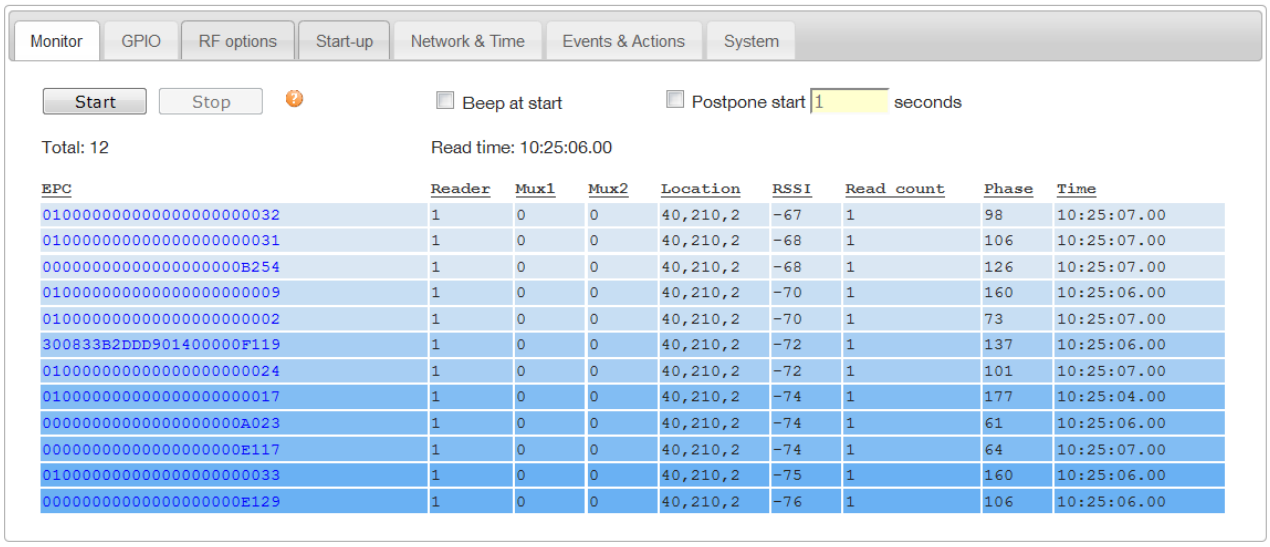
Area with tab contents. The specific contents will depend on the active tab and on the selected device mode.

3.3- Common tabs

Regardless the active device mode, some of the tabs are always the same.

3.3.1- Monitor

From the monitor page we can start/stop the RF operation of the active device mode.



EPC	Reader	Mux1	Mux2	Location	RSSI	Read count	Phase	Time
010000000000000000000032	1	0	0	40,210,2	-67	1	98	10:25:07.00
010000000000000000000031	1	0	0	40,210,2	-68	1	106	10:25:07.00
000000000000000000000B254	1	0	0	40,210,2	-68	1	126	10:25:07.00
010000000000000000000009	1	0	0	40,210,2	-70	1	160	10:25:06.00
010000000000000000000002	1	0	0	40,210,2	-70	1	73	10:25:07.00
300833B2DDD90140000F119	1	0	0	40,210,2	-72	1	137	10:25:06.00
010000000000000000000024	1	0	0	40,210,2	-72	1	101	10:25:07.00
010000000000000000000017	1	0	0	40,210,2	-74	1	177	10:25:04.00
000000000000000000000A023	1	0	0	40,210,2	-74	1	61	10:25:06.00
000000000000000000000E117	1	0	0	40,210,2	-74	1	64	10:25:07.00
010000000000000000000033	1	0	0	40,210,2	-75	1	160	10:25:06.00
000000000000000000000E129	1	0	0	40,210,2	-76	1	106	10:25:06.00

Capture 16: Monitor tab

The tag result table will display all the inventoried tags with the tag properties.

Tag properties	
EPC	Tag EPC in hexadecimal
Reader	Reader port where the tag has been read
Mux1[4]	Output port of the first level multiplexer where the tag has been read.
Mux2	Output port of the second level multiplexer where the tag has been read.
Location	Estimated location of the tag. Location information has to be filled in during antenna setup.
RSSI	RSSI if the received tag.
Read count	Read count[5]
Phase	Phase value of the detected tag
Time	Time stamp of the reading

Table 3 Tag properties

Be aware that depending on the active mode the tag table display may var. For example:

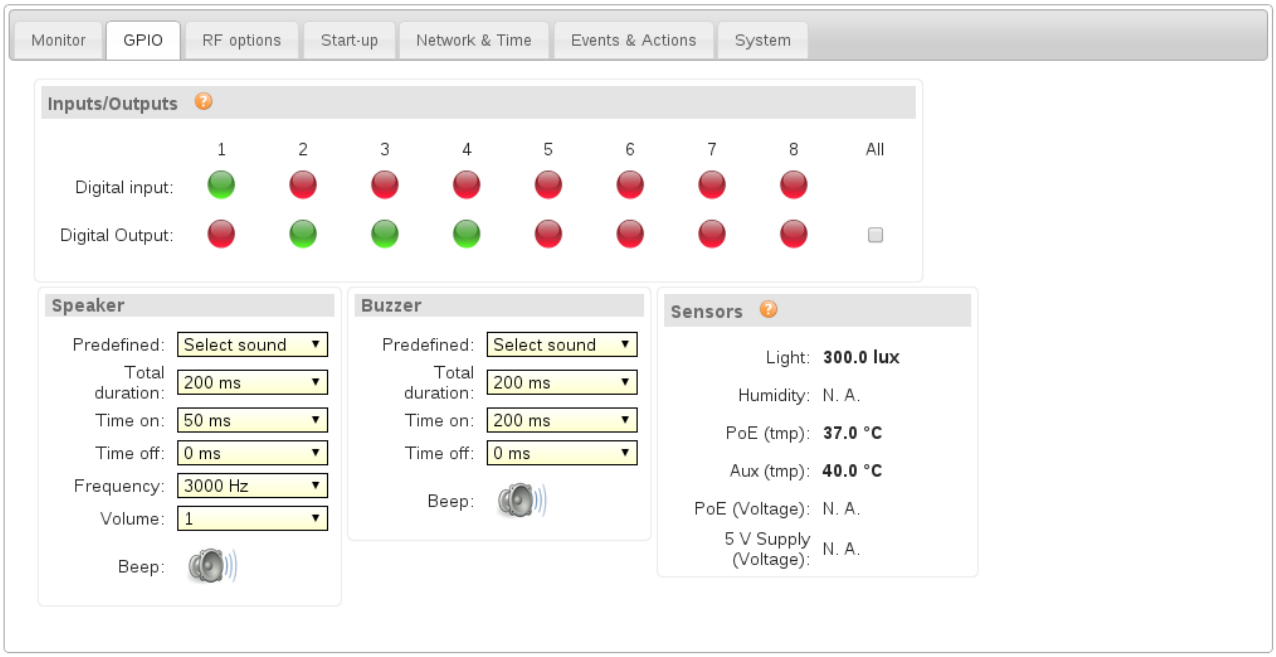
- NXP modes do not use the standard SELECT EPCGen2 command and therefore no results will be displayed. Even though tags are detected with the NXP bit set.
- EPC alarm modes will obviously filter out non-matching tags.

4 *Mux1 and Mux2 values only make sense in sequential mode where multiplexers can be used.*

5 *In Autonomous mode the read count will always have a value of 1. In sequential mode the read count may be greater than 1.*

3.3.2- GPIO[6]

The GPIO tab allow access to all I/O interfaces.



Capture 17: GPIO tab

The available interfaces are summarized in the tables that follows:

Digital inputs (read only)

Input lines	
GPI #1	General Purpose Input #1
GPI #2	General Purpose Input #2
GPI #3	General Purpose Input #3
GPI #4	General Purpose Input #4
GPI #5	General Purpose Input #5
GPI #6	General Purpose Input #6
GPI #7	General Purpose Input #7
GPI #8	General Purpose Input #8

Table 4 GPI table

6 *Tables in this chapter represents AdvanReader-m4-100. Other devices may have a different number of I/O devices.*

Digital outoputs: click on any GPO line to activate/deactivate it.

Output lines	
GPO #1	General Purpose Output #1
GPO #2	General Purpose Output #2
GPO #3	General Purpose Output #3
GPO #4	General Purpose Output #4
GPO #5	General Purpose Output #5
GPO #6	General Purpose Output #6
GPO #7	General Purpose Output #7
GPO #8	General Purpose Output #8

Table 5 GPO table

Sound devices: adjust parameters and play sounds

Sound devices	
Speaker	External speaker
Buzzer	Internal buzzer

Table 6 IO devices table

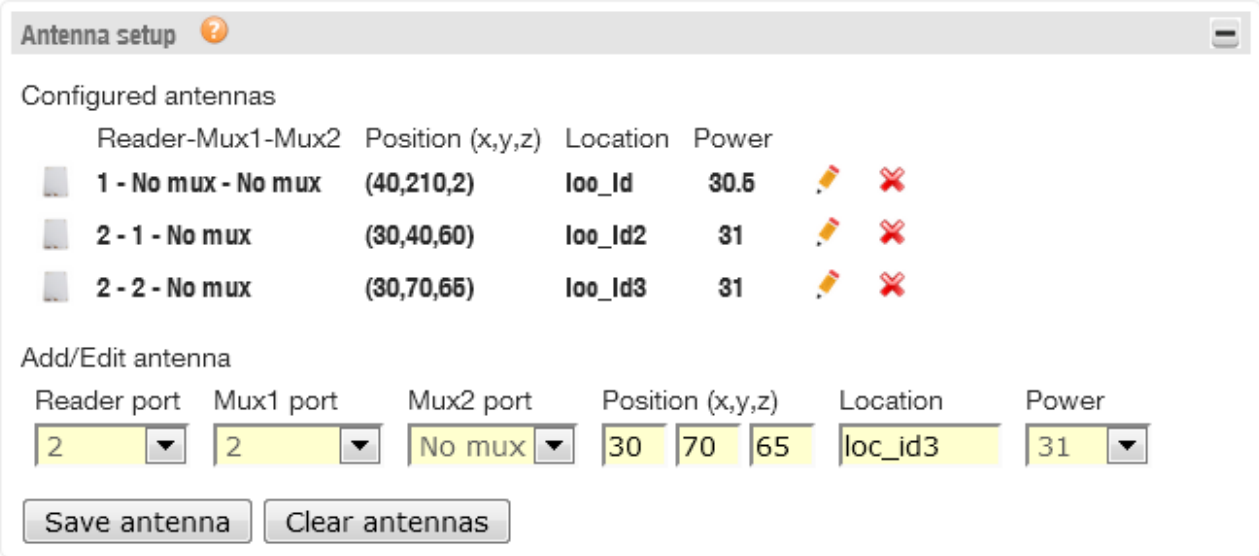
On-board sensors (read only)

On-board sensors[7]	
Light	On-board light sensor
Humidity	On-board humidity sensor
PoE temperature	On-board temperature sensor of the PoE power supply. For monitoring purposes. This is not ambient temperature.
+24 Vcc power supply temperature	On-board temperature sensor of the +24 Vcc power supply. For monitoring purposes. This is not ambient temperature.
PoE voltage	On-board voltage sensor of the PoE power supply. For monitoring purposes.
+5 Vcc voltage	On-board voltage sensor of the +5 Vcc power supply. For monitoring purposes.

Table 7 Sensor table

3.3.3- RF Options

The RF options tab is specific to the device mode selected. However, the antenna menu that is showed in the RF Options tab is common for all device:



Capture 18: Antenna setup

In AdvanNet terms, any antenna is then composed by those 3 parts, for example antenna [1,1,4]:

- antenna connected at port #1 of the reader.

7 Ob-board sensors are available based on device. Please use the User Guide of each device.

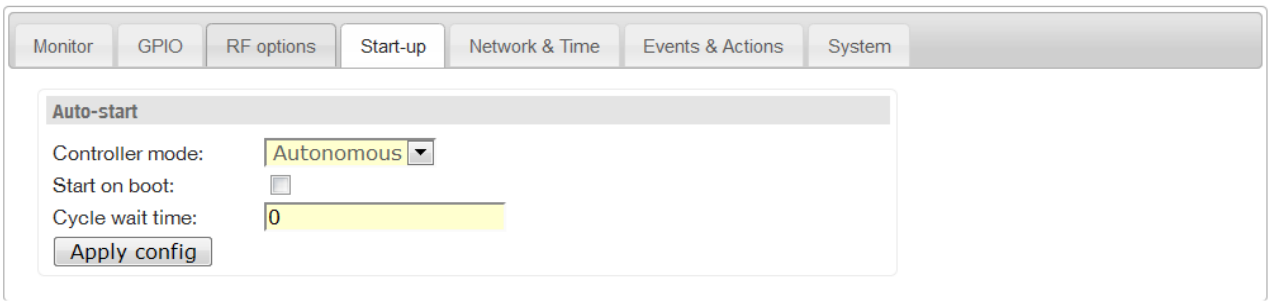
- connected at output #1 of the first multiplexing level.
- connected at output 4 at the second multiplexing level.

Antenna setup		Possible values
Reader port	Reader RF port the antenna is connected to	[1,2,3,4]
Mux-1 port	Level one multiplexer RF output port the antenna is connected to	[1..16][8]
Mux-2 port	Level two multiplexer RF output port the antenna is connected to	[1..16]
Location	Identifier of the location of the antenna	Any string
Pos X	X position of the antenna given certain location system	Any integer
Pos Y	Y position of the antenna given certain location system	Any integer
Pos Z	Z position of the antenna given certain location system	Any integer
Power	Specific power to be applied to that antenna	[5 .. 31.5] dBm (0.5 dBm steps)

Table 8 RF options Antenna setup

3.3.4- Start-up

System start up options. From this tab we can control the start up properties of the device.



Capture 19: Start up tab

The following combination of values are very useful:

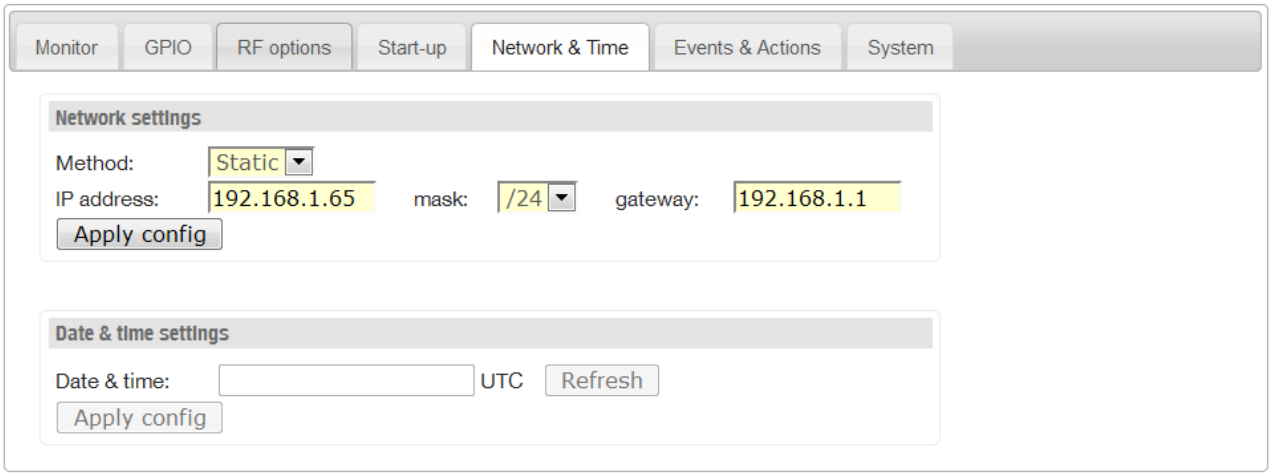
- Autonomous + start on boot enabled: to make sure the system start the RF operation as soon as the device is powered up.
- Autonomous + start on boot disabled: in case we prefer to start the RF operation manually, from the monitor tab.
-

8 *AdvanMux-8 only support up to 8 outputs. AdvanMux-4 only supports 4 output ports*

3.3.5- Network & Time

From Network & Time tab it is possible to:

- Adjust network settings[9].
- Adjust time settings[10]: time is not relevant for RF operation but it's used in the logs and other sub-systems.



Capture 20: Network 6 Time tab

To change the IP of AdvanReader 50 and AdvanReader 100, an additional program called Lantronix DeviceInstaller should be installed.

9 *Only for certain models. See the specific device User Guide.*

10 *Only for certain models. See User Guide to understand each device capabilities.*

3.3.6- System

From the system tab we access advanced features of the system

Capture 21: System tab

Features:

- Save system configuration: Persists current AdvanNet software configuration.
- Export system configuration: Exports AdvanNet software configuration values to a file.
- Import system configuration: Imports AdvanNet software configuration from an external file.
- Reset system configuration: Sets all the values of the software to the default values

Reset to defaults won't reset the device settings or other Operating System settings.

3.4- Specific tabs

Some tabs change their contents according to the device model and device mode selected. Tabs that change their contents are:

- **RF tab:** the device mode controls the RF operations and therefore the RF tab contents will change according to the selected mode. It's not the same adjusting options for an RF portal than adjusting options for an anti-theft system.
- **Events & Actions**: according to the device mode selected the system will trigger certain events. For example in an anti-theft system, one possible events is TAG_ALARM. However that event makes no sense in other modes.

3.4.1- RF options

The examples shown in this chapter represent the configuration for a general purpose AdvanReader-m4-100. However, the ideas and concepts will be the same across any system.

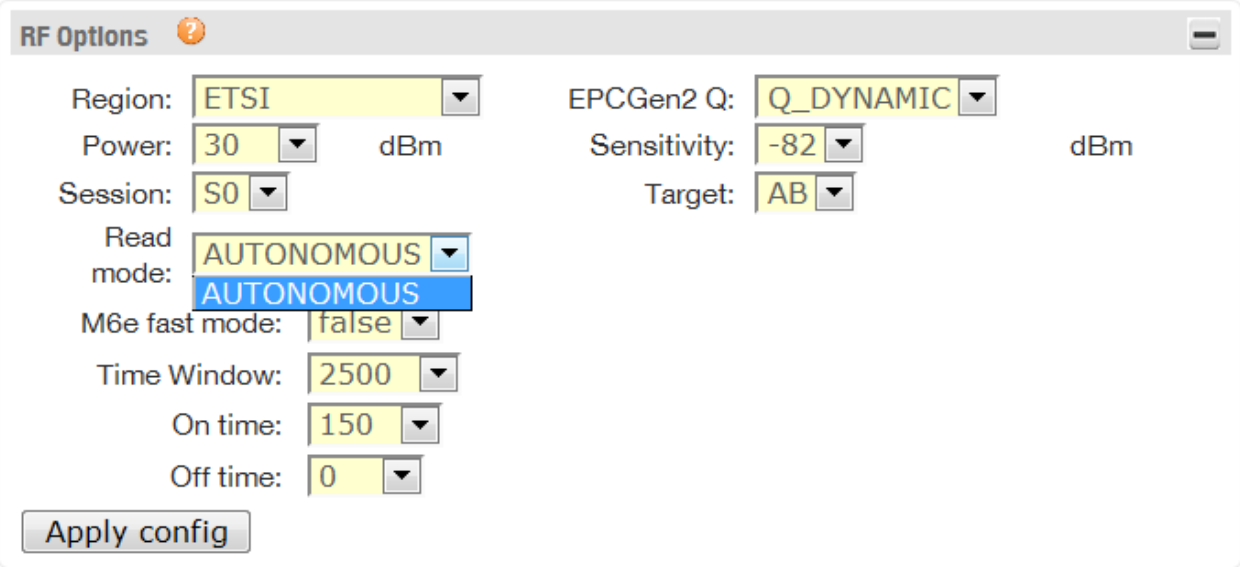
RF Options		Possible values
Region	RF operation region[11]	ETSI FCC AUSTRALIA CHINA INDIA JAPAN KOREA MALAYSIA PERU SINGAPORE TAIWAN
Power	RF conducted power (0 dBm to 30 dBm)	5-31.5 dBm (0.5 steps)
Sensitivity	Mnimum received sensitivity (software filter)	[-60, -80] dBm[12]
Session	EPCGen2 session	[0,1,2,3]
Target	EPCGen2 target	[A,B,AB,BA]
Q	EPCGen2 Q value	Dynamic, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048

Table 9 RF options common parameters

11*Supported RF regions may change from product to product*

12 *AdvanReader-m4-100 does not offer -80 dBm sensitivity. The -80 value expresses the maximum reader sensitivity will be used.*

Autonomous mode

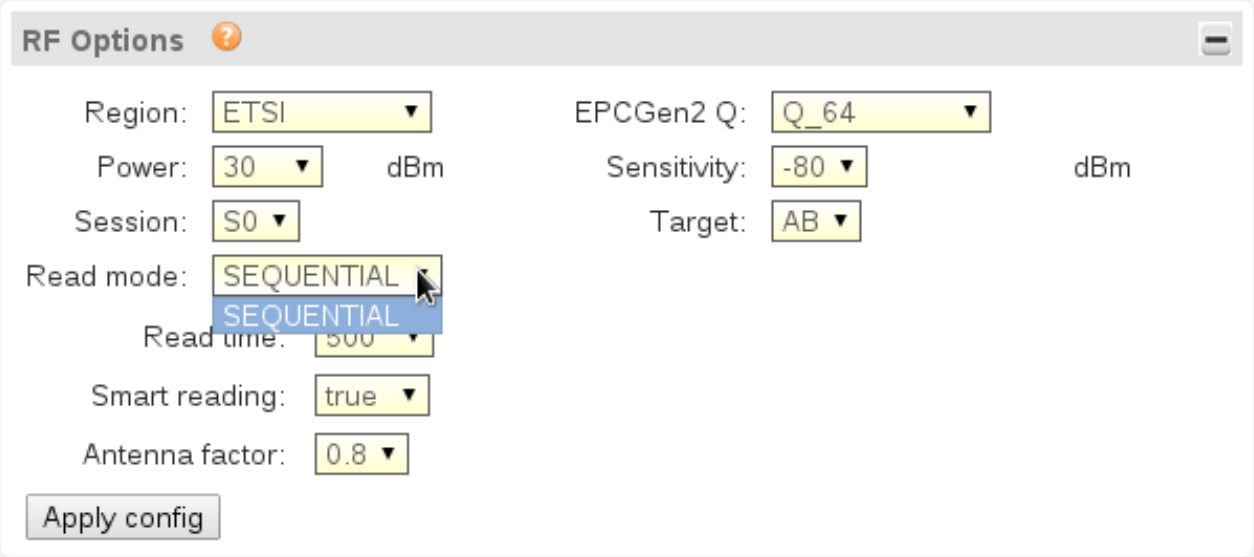


Capture 22: RF options in Autonomous device mode

RF Options		Possible values
Read mode (fixed)	AUTONOMOUS	
Time window	The time a tag read will be held into the monitor tab since the last read. A tag read at time T for the last time will disappear from the monitor tab at time (T + time window)	[2500 .. 10000] ms
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms

Table 10 RF options table for AUTONOMOUS device mode

Sequential mode

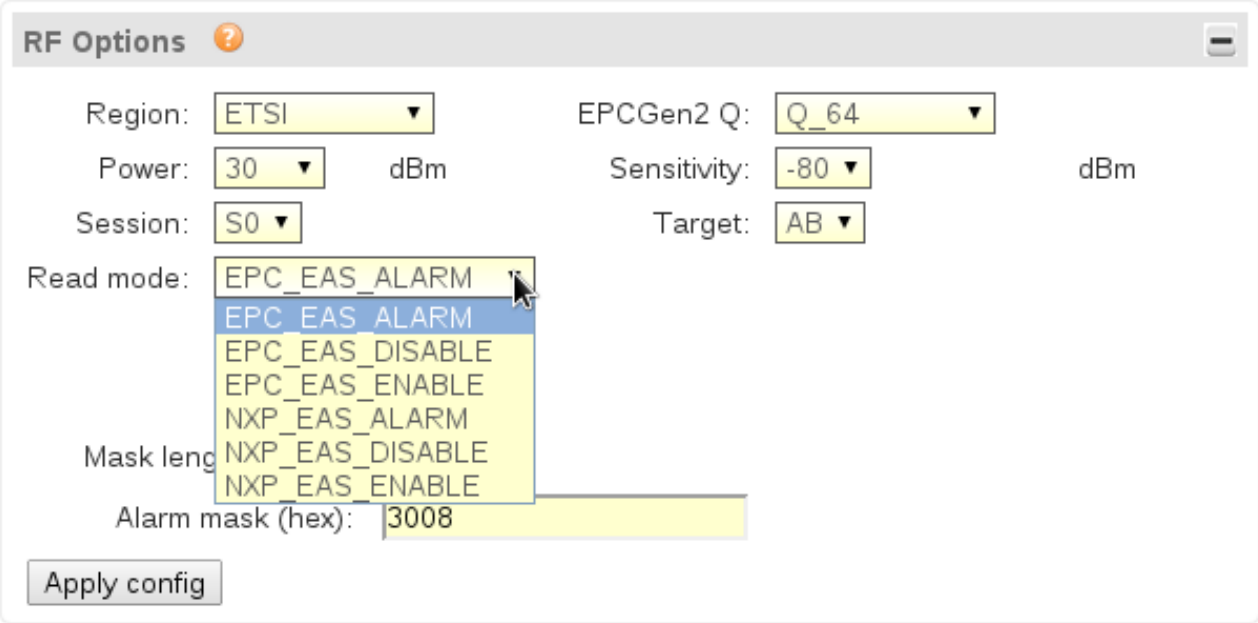


Capture 23: RF options in Sequential device mode

RF Options		Possible values
Read mode (fixed)	SEQUENTIAL	
Time window	The time a tag read will be held into the monitor tab since the last read. A tag read at time T for the last time will disappear from the monitor tab at time (T + time window)	[2500 .. 10000] ms
Smart reading	When enabled, the system uses the symmetry of antennas and multiplexers to reduce the total time of a read cycle. The time reduction is never done by reducing location accuracy.	true/false
Antenna factor	Internal parameter used in the smart reading mode. Small values result in a reduction of the overall read time caused by reducing the time spent in each antenna.	[0.1 … 1]

Table 11 RF options table for Sequential device mode

Alarm mode



Capture 24: RF options in Alarm device mode

This device mode has several read modes. Each read mode represents a different alarm operation:

- EPC_EAS_ALARM: the alarm recognizes EPC patterns to trigger appropriate events.
- EPC_EAS_DISABLE: disables the alarm pattern on each found tag.
- EPC_EAS_ENABLE: enables the alarm pattern on each found tag.
- NXP_EAS_ALARM: the alarm recognizes the NXP EAS bit to trigger appropriate events.
- NXP_EAS_DISABLE: disables the NXP EAS bit on any found tag.
- NXP_EAS_ENABLE: enables the NXP EAS bit on any found tag.

EPC_EAS_ALARM

RF Options		Possible values
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms
SW filter	Under some circumstances, basically when tag population with different memory organization exist, tags can be detected as alarmed when they are not. The SW filter is applied on those *stray* tags.	true/false
Mask length (byte)	Mask length of the EPC pattern used. Lengtn is word base, so only multiples of 2 are allowed.	[2,4,6,8,10,12,14,16,18,20,22,24] bytes
Alarm mask	The EPC pattern in hexadecimal.	Any hexadecimal pattern of the length defined in mask length

Table 12 RF options table for EPC_EAS_ALARM

EPC_EAS_DISABLE

EPC_EAS_DISABLE RF Options		Possible values
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms
Mask length (byte)	Mask length of the EPC pattern used. Lengtn is word base, so only multiples of 2 are allowed.	[2,4,6,8,10,12,14,16,18,20,22,24] bytes
EPC mask	The EPC pattern in hexadecimal that will be replaced in the tag EPC.	Any hexadecimal pattern of the length defined in mask length
Access password (hex)	Write access password for the EPC bank.	Any hexadecimal value of length 8.
EPC bank locked	Disable only when the tags being used are no protected against EPC write operations.	true/false

Table 13 RF options table for EPC_EAS_DISABLE

EPC_EAS_ENABLE

RF Options		Possible values
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms
Mask length (byte)	Mask length of the EPC pattern used. Lengtn is word base, so only multiples of 2 are allowed.	[2,4,6,8,10,12,14,16,18,20,22,24] bytes
EPC mask	The EPC pattern in hexadecimal that will be replaced in the tag EPC.	Any hexadecimal pattern of the length defined in mask length
Access password (hex)	Write access password for the EPC bank.	Any hexadecimal value of length 8.
EPC bank locked	Disable only when the tags being used are no protected against EPC write operations.	true/false

Table 14 RF options table for EPC_EAS_ENABLE

NXP_EAS_ALARM

RF Options		Possible values
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms
NXP Chip type	Specific NXP chip. Leave the default value	G2IL/G2XL

Table 15 RF options table for EPC_EAS_ENABLE

NXP_EAS_DISABLE

RF Options		Possible values
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms
NXP Chip type	G2IL/G2XL Leave the default value set.	G2IL/G2XL
Expected tag population	Advanced configuration. Leave to 1 for best performance	[1,5,10,20]
Operation timeout	NXP comamnd timeout. Leave default value.	[50 … 1000] ms

Table 16 RF options table for NXP_EAS_DISABLE

NXP_EAS_ENABLE

RF Options		Possible values
On time	RF on time. Together with the off time it can control the duty cycle. The duty cycle is: onTime / (onTime+offTime)	[50...1000] ms
Off time	RF off time.	[0...200] ms
NXP Chip type	G2IL/G2XL Leave the default value set.	G2IL/G2XL
Expected tag population	Advanced configuration. Leave to 1 for best performance	[1,5,10,20]
Operation timeout	NXP comamnd timeout. Leave default value.	[50 … 1000] ms

Table 17 RF options table for NXP_EAS_ENABLE

3.4.2- Events & Actions tab

The Events & Actions tab is where events are configured to trigger actions.

An event is generated as a result of a Rf operation, a GPIO operation, etc. Once an event is generated it is processed to see if it has any action attached. When an action is found, it is executed.

Possible examples are:

- An alarmed tag is detected at antenna 1, then trigger the GPO #1
- An alarmed tag is detected at any antenna. Then trigger a speaker action.

Every device mode has a set of allowed events and actions. For example the EPC EAS mode define the TAG_ALARM event but that particular event does not make sense if we are in Autonomous mode.

3.4.2.1- Events

List of all events defined. Not all device modes will implement all the events.

Device events	
TAG_READ	Event that is generated when a tag is read in any of the connected antennas.
TAG_READ_ANTENNA_1	Event that is generated when a tag is read in the antenna connected at port #1
TAG_READ_ANTENNA_2	Event that is generated when a tag is read in the antenna connected at port #2
TAG_READ_ANTENNA_3	Event that is generated when a tag is read in the antenna connected at port #3
TAG_READ_ANTENNA_4	Event that is generated when a tag is read in the antenna connected at port #4
TAG_ALARM	Event that is generated when an alarmed tag is detected at any of the connected antennas. The meaning of *alarmed tag* will change according to the device mode.
TAG_ALARM_ANTENNA_1	Event that is generated when an alarmed tag is detected at thye antenna connected at port #1
TAG_ALARM_ANTENNA_2	Event that is generated when an alarmed tag is detected at thye antenna connected at port #2
TAG_ALARM_ANTENNA_3	Event that is generated when an alarmed tag is detected at thye antenna connected at port #3
TAG_ALARM_ANTENNA_4	Event that is generated when an alarmed tag is detected at thye antenna connected at port #4
TAG_ALARM_ENABLED	Event that is generated when a tag is alarmed at any of the connected antennas.
TAG_ALARM_DISABLED	Event that is generated when a tag is disalarmed at any of the connected antennas.

Device events	
DEVICE_ERROR	Event that is generated when an internal error ocurrs in the RF module.
SYSTEM_ERROR	Event that is generated when an internal error ocurrs in the Configuration Utility software.

Table 18 Available events

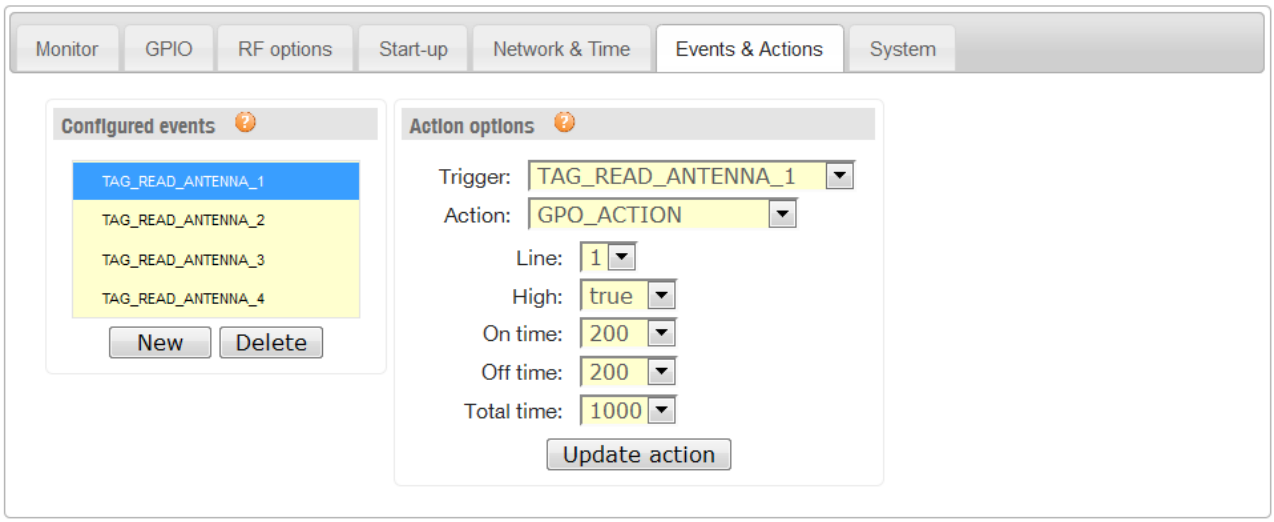
3.4.2.2- Actions

List of all available actions

Device actions	
GPO_ACTION	Action that uses the GPO lines
BUZZER_ACTION	Action that uses the on-board buzzer
SPEAKER_ACTION	Action that uses the external speaker

Table 19: Available actions

The following is the pre-defined configuration for the Autonomous mode.



Capture 25: Events & Actions tab for Autonomous mode

Update existing defined events:

1. Select the event by clicking on it. You should see it highlighted in blue.

2. Adjust trigger, action and parameters
3. Click on the *Update action* button.

Add new event:

1. Click on the *New* button
2. Adjust trigger, action and parameters.
3. Click on the *Save new action* button.

Delete event:

1. Select the event by clicking on it. You should see it highlighted in blue.
2. Click on the *Delete action* button.

3.4.3- Actions parameters

GPO action		
Line	1..5	GPO line
High	true/false	Value that will be set
On time	0..65535 ms	Time the GPO line will be set to *high* value before changing the sate to !*high*
Off time	0..65535 ms	Time the GPO line will remain in !*high* value before changing the sate to *high*
Total time	0..65535 ms	Total time of the action. The final state of the GPO line is alwas set to! *high*

Table 20: Available actions

Buzzer action		
On time	0..65535 ms	Time the buzzer will be driven on before changing the state
Off time	0..65535 ms	Time the buzzer will be driven off before changing the state
Total time	0..65535 ms	Total time of the action. The final state is always off

Table 21: Available actions

Speaker action		
Frequency	500..5000 HZ	Frequency in Hz of the speaker tone.
Volume	1..10	Volume of the speaker. The theoritical maximum output power is 2 Watt. The volume scale is not linear and therofore a volume is 5 is not exactly in the middleof the maximum power.
On time	0..65535 ms	Time the speker will be driven on before changing the state
Off time	0..65535 ms	Time the speaker will be driven off before changing the state
Total time	0..65535 ms	Total time of the action. The final state is always off

Table 22: Available actions

4- Advanced configuration

4.1- Enable logback

Logback is a powerful logging system. AdvanNet supports it but it's disabled by default.

To enable it:

1. Open the file $ADVANNET/conf/META-INF/connectors.xml
2. Add one line at the beginning of the file

```
<?xml version='1.0' encoding='utf-8'?>
<aliases>
        <load-order-scope>default</load-order-scope>
        <load-order>20000</load-order>

        <CONFIGURATION>
                <gui.useLogin>false</gui.useLogin>
                <logger>net.ihg.util.log.SLF4JLogger</logger>
        </CONFIGURATION>

        …
</aliases>
```

3. Make sure the $ADVANNET/lib directory contains the files:
   - logback-classic-0.9.24.jar[13]
   - logback-core-0.9.24.jar
   - janino.jar[14]

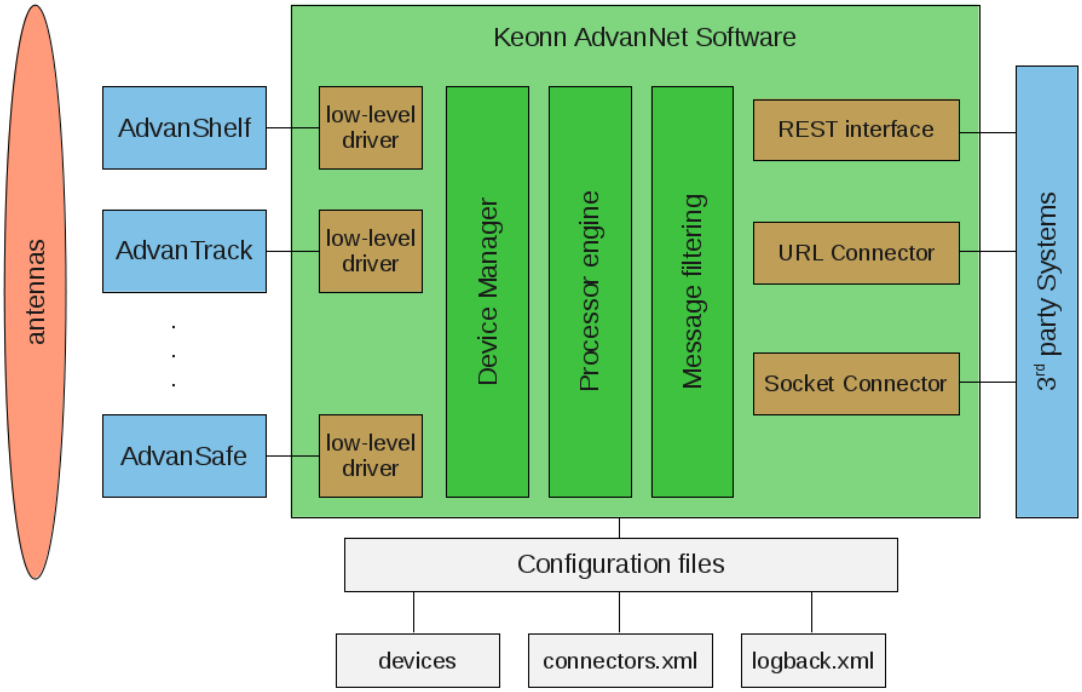
13 *http://logback.qos.ch/*

14 *http://docs.codehaus.org/display/JANINO/Home*

5- AdvanNet internals

This chapter is an introductory explanation on AdvanNet device operation, device handling and data processing. This is a chapter intended to help users understand how AdvanNet works internally.

Let's start refreshing the block diagram of AdvanNet



Capture 26: AdvanNet block diagram

5.1- Configuration files

At the current release Keonn AdvanNet is configured by using xml files. All configuration files reside in the `$ADVANNET/conf` directory, where $ ADVANNET is the directory where AdvanNet has been installed.

The *conf/* directory follows the tree below:

META-INF/devices/*.xml

The devices directory contains one file per device on the system. That is, a system with 2 AdvanShelf will contain two files inside the devices directory.

- The files under the devices/ directory are called device definition files.
- Device definition files templates can be found under

$ADVANNET/templates/devices.

- Regardless of the contents in devices/, the self-discovery operation will be performed.

META-INF/connectors.xml

Configuration file where most of the logic of AdvanNet is defined.

Properties defined there can be changed at runtime by using the REST interface. Please see AdvanNet Reference Guide.

META-INF/logback.xml

Configuration of the logging facility. It does not usually require any change.

5.2- Device operation

The first step to understand how AdvanNet handle devices is to get to know the operation of devices themselves. Devices can be categorized in two groups as shown in the table below:

Category	Description	External operation
Synchronous or on demand device	Devices that are waiting for commands and react only when commands arrive.	A command Is issued and a response data is received synchronously. The basic operation is the *inventory*.
Asynchronous or autonomous device	Devices that are running continuously once they are started.	Data is generated internally and device listeners are notified as data is being created. The basic operation is the *notify* method applied to the device listeners.

Table 23: device classification

While some device have a nature that define how it will operate, some others can be forced to operate in any of the two possible modes.

For example, AdvanMat will most sure run as an asynchronous device, thus notifying listeners any time a tag passes through its coverage area. It wouldn't make much sense to control AdvanMat synchronously.

On the other hand some devices can be controlled both modes. Imagine the case of a smart shelf, synchronous mode would make sense if we were interested in performing a manual inventory once or twice a day. Also we could be interested in real time inventory, then using perhaps the asynchronous mode.

That said, almost any devices in AdvanNet can be configured to work either synchronous or asynchronously regardless of the physical sense. Default configuration always chooses the method that is best suited with device nature and expected operation. AdvanMat is by

default configured to be used asynchronously while AdvanShelf is configured as a synchronous device.

The following table summarizes how a device reacts to a command according to its operation mode.

Category	Inventory operation	Notify
Synchronous	Returns the current inventory seen by the device.	A synchronous device **never** notifies registered listeners.
Asynchronous	Returns the tags seen in the last *TimeWindow* seconds. This operation does not cause any interference with the normal asynchronous operation. (*TimeWindow* can be configured)	An asynchronous device will **always** notify all registered listeners.

Table 24: device operations

5.3- Device controlling

As we have seen, synchronous and asynchronous devices are handled differently, while the first follow a request-response cycle the latter notifies registered listeners about events as soon as they are generated. How do we issue a request to a synchronous device? How do we get notified when a device generates data?

The answer to those questions is what we are about to see.

5.3.1- REST interface

AdvanNet uses a common interface to handle devices and to access most of the services devices offer. AdvanNet uses an HTTP based REST[15] interface to handle devices.

During AdvanNet start up an embedded HTTP server is launched and starts serving requests at the local URL:

http://localhost:3161

Details on how to use the REST interface are provided in AdvanNet operation chapter.

5.4- Data processing

AdvanNet has a fairly simple process engine that allows some basic operations on data captured by RF devices.

Data processing can be performed at two different stages:

1. **Device level:** a device can perform an initial filtering of the data. A typical example

15 *http://en.wikipedia.org/wiki/Representational_state_transfer*

is to filter out tags given a defined pattern or by received RSSI.

2. **Global level:** data generated by asynchronous devices can be processed as a previous step before being sent to device listeners.

Processors can be chained in a hierarchy of master/slave like configuration.

The most common processors available are summarized in the table below.

Name	Description
EPC filter	High level EPC processor that filters out any EPC not matching a given regular expression.
Echo processor	Prints on the standard output and in the logs the data being captured.
Statictics processor	Analyses data being captured and prints it on the standard output and in the logs.
Inventory	Processes the data captured to resolve inventory data and to generate xml messages that will be sent to device listeners.

Table 25: Processing methods

The default setup includes three processors: EPC filter processor (disabled), echo processor (disabled) and inventory processor (required to format xml messages).

5.5- Connectors

After the device controller and the configured processors have done its job, it's time the resulting inventory messages are sent to registered listeners.

Sending messages has to be understood as a two step process:

1. Message filtering: a final filtering of the messages could be performed as a way to organize data in a more usable way. No need to say this step can be skipped and left for higher levels of the software architecture.
2. Message transport: the last process is the sending of the messages, AdvanNet supports currently two methods: by TCP socket or by an HTTP URL.

5.5.1- Message filtering

Why messages are filtered? Keonn products are very different in nature, AdvanSafe is an anti-theft system while AdvanTrack is a real-time inventory system, those differences makes very natural that a 3rd party system might be interested in being notified for events generated at AdvanSafe and didn't care about AdvanTrack events. That being one of the reasons why filtering messages is important.

5.5.2- Message Transport

The filtering has been already covered in previous chapters, so we will focus in transport mechanisms. At the time of this writing two mechanisms are possible:

- Socket transport: a socket is opened and the message/s sent to the socket. No response is never expected. This is best suit for quick integration.
- HTTP transport: a connection is opened to a given URL and the message/s sent to it by executing a HTTP POST operation to the given URL. This mechanism expects the 3rd party to send back a response as HTTP defines.

AdvanNet does not support integration by using SOAP.

5.6- Tag life cycle

A very easy and simple way to understand the internals of AdvanNet is to have a look at a tag life cycle; from the point where it is first singulated up to the point where t becomes a useful business piece of data.

5.6.1- Synchronous devices

The following is a quick glimpse of a tag life cycle:

1. Upon a user requesting an inventory, a given device performs a singulation process.
2. The tag is singulated and it's basic data is captured and temporarily stored at a Keonn product instance. Basic tag data are:
    - EPC
    - RFID characteristics: received powered, read count, etc.
    - Context data: location data of the receiving antenna.
3. The tag data along with any other tag data singulated is passed to the processor engine.
4. Once the processor engine has ended its job and given the tag has not been remove due to the processing itself, the tag is returned to the original requester.

5.6.2- Asynchronous devices

The following is a quick glimpse of a tag life cycle:

1. An instance of a Keonn product configured as an asynchronous device is scanning continuously the RF field searching for tags.
2. The tag is singulated and it's basic data is captured and temporarily stored at a Keonn product instance. Basic tag data are:
    - EPC
    - RFID characteristics: received powered, read count, etc.

- Context data: location data of the receiving antenna.

3. The tag data along with any other tag data singulated is passed to the processor engine.

   One of the processors might be a transport service that will eventually send tag data to registered listeners.