



433/868 USB Dongle RFPLAYER API Specifications

V1.15 08/02/2019

Releases (Author : JP Gauthier):

V1.1 : 22/3/2016. Creation.

V1.2 : 22/6/2016. Release.

V1.3 : rename xml/json tag "value" to "v".

V1.4 : add reserved fields in public API.

V1.5 : REPEATER ON/OFF management modification. Somfy RTS, Antennas, LED signalisation explanations. Screenshots examples. Based on Firmware V1.07

V1.6 : Add ALL_ON ALL_OFF CHACON & X10. PING instruction. Based on Firmware V1.08

V1.7 : Modification of HELLO command response. Based on Firmware V1.10

V1.8 : Modification of the end of HELLO command response (free area). Based on Firmware V1.12. Instruction INITLB added.

V1.9 : adding TRACE and ALARM functions.

V1.10 : X2D thermostat frames generation precisions (X2DGAS and X2DELEC).

V1.11 : Cartelectronic TIC/Pulses and FS20 protocols added. Based on Firmware 1.22.

V1.12 : LINKY Frame added to Cartelectronic TIC. Jamming detection added. Based on Firmware 1.25.

V1.13: Correction on JAMMING infoType = 1 (no more infoType =0 as specified before, but 1 on firmware V1.25). Trace added on JAMMING pseudo-protocol. Based on Firmware 1.26.

V1.14 : Add Edisio (868.300Mhz) protocol (based on Firmware V1.29). Frequency 868.350Mhz (Deltadore-fs20-Edisio) is set as factory 'default' high frequency. Nota that old firmware (< 1.29) updated by a new firmware (>=1.29) operation will reset to factory defaults the saved parameters (the parameters saved in EEPROM ; not parrot, transcoder parameters which are saved in SFLASH). 'MAC' and 'Factory' saved parameters are exceptions and are copied to new configuration. Return to old versions (>=1.29 to <1.29) will reset all parameters without exception (including 'MAC' and 'Factory' saved parameters) : Return to old versions is NOT recommended, especially if the user had before customized its RFPLAYER device, because this customization may be lost.

V1.15 : InfoValue10, 11(X2D receiving frames) minor changes. InfoValue10, 11 Raw data[] fields are now generated with XML/JSON formats on X2D receiving frames. Add TRACE RFLINK instruction for debug (>= Firmware version V1.39). Add SIREN command.

CONFIDENTIAL

Summary

1	Introduction	3
2	USB Characteristics	3
3	Host App	3
4	Common Frame Container	4
4.1	ASCII data (used by commands Interface)	5
4.2	Binary data (used by HA protocol frames)	5
4.3	SourceDestQualifier	5
4.4	SourceDest	6
5	API 433/868	7
5.1	Host → USB Dongle (configuration and transmitting data to RF)	8
5.1.1	ASCII data (ordinary used by commands Interface)	8
5.1.2	Binary data (ordinary used to sent RF frames and by RFLINK interface)	30
5.2	USB Dongle → Host (receiving data from RF)	32
5.2.1	ASCII data (ordinary used by commands answers and asynchronous received RF Frames) 32	
5.2.2	Binary data (ordinary used to received RF Frames and by RFLINK interface)	32
5.3	Jamming detection	48
5.4	Firmware Update	51
5.5	Antennas	52
5.6	LED signalisation	53
5.7	Screenshots	54

1 Introduction

2 USB Characteristics

The USB interface emulates a serial line with a classical chip FTDI FT232R.

Parameters : Baudrate : 115,2Kb/s, 8 bits data, no parity, 1 stop bit.

The host machine must be sensitive to flow control through the CTS line (Clear to Send), especially during firmware upgrade which handles a large amount of data.

NB: CTS line sensitivity is critical only during firmware download (update operation) which handles a large amount of data. At this time, the dongle is processing this incoming flow in real time so that CTS line handling isn't mandatory. CTS line handling could be seen as 'provision'.

The USB Dongle is not sensitive to RTS line (Request to Send). The host machine has to receive the incoming flow anyway.

Incoming data are memorized in a queue (FIFO) of 2048 bytes so that many independent commands can be enqueued before processing. CTS line becomes false when this queue become almost full.

Max interpreter command line buffer is 1500 bytes length so that many instructions can be specified in one command line.

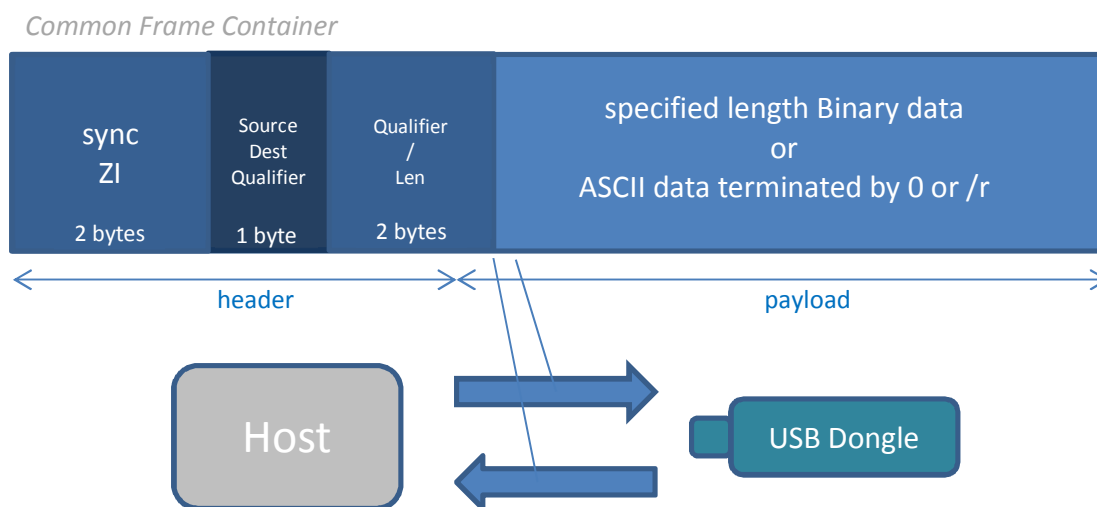
3 Host App

A Java App utility with friendly interface running on PC/Mac is available from our website to do dongle configuration. It could be useful to understand how the present API works.

4 Common Frame Container

The common Frame container is bi-directional and applicable to:

- *Host to USB Dongle* transfers
- *USB Dongle to Host* transfers



The role of the Frame Container is to help the destination to synchronize incoming flow, and multiplex or demultiplex the frame upon direction and the SourceDestQualifier field.

The frame payload is in binary or ASCII data form.

Binary form is used by the RF entity itself to handle the associated RF protocol. The Frame Container is agnostic to the payload field. The specifications of the payload field is RF entity dependent and sometimes completely specified by the provider of the chip handling this RF entity (e.g. Z-WAVE / ENOCEAN)

ASCII form is used to manage the RF entities for configuration, options setting and update processing. "AT commands style" is chosen and a commands interpreter permits to launch commands e.g. "ZIA++STATUS".

4.1 ASCII data (used by commands Interface)

Nom	Size	Contains	Remark
Sync1	1	'Z'	0x5A
Sync2	1	'I'	0x49
SourceDestQualifier	1	'A' to 'O'	0x41 to 0x4F
Qualifier	2		Printable characters Defining the frame subtype
AsciiData[0...n]	0...infinite unsigned char[]		ASCII data
terminator	1	\0 or \r	0x0 or 0x0D

4.2 Binary data (used by HA protocol frames)

Name	Size	type	Contains	Remark
Sync1	1	unsigned char	'Z'	0x5A
Sync2	1	unsigned char	'I'	0x49
SourceDestQualifier	1	unsigned char	0x00 to 0x0A	Not printable characters
Len	2	unsigned char		Len of BinaryData [] LSB first
BinaryData []		unsigned char[]		Binary data

4.3 SourceDestQualifier

bit	7	6	5	4	3	2	1	0
	0 <i>reserved</i>	<i>Value = x04 for ASCII DATA Value = 0x0 for BINARY DATA</i>			SourceDest			

When the Host emits the frame, SourceDest contains the ID of the destination of the frame (e.g RF entity = 1 for 433/868).

When the USB Dongle emits the frame to Host, SourceDest contains the ID of the source transmitting the frame (e.g RF entity = 1 for 433/868), thus, Host as destination is implicit.

4.4 SourceDest

SourceDest/ RF entity	Value	Remark
MUX Management	0	
433/868	1	ZIA
Future RF entities	Other values	ZIB, ZIC, etc...

NOTA : It is expected that future USB dongles will handle more than one RF entity.

Several flows to/from several RF entities can be multiplexed on the same physical link with ZIx header.

5 API 433/868

The 433/868Mhz interface is able to emit/receive the protocols listed below.

All received protocols are decoded simultaneously without mutual exclusion on both 433 and 868Mhz bands. One hardware receiver is used for each band and the digital processing still works simultaneously on both bands.

Low band 'L' = 433 Mhz.

High band 'H' = 868 Mhz.

The receiver of each band is tuned for a specific frequency. Consequently, receivers must be adjusted (by the management command 'FREQ') on:

- 433.420 Mhz used by Somfy RTS or 433.920 Mhz used by a lot of devices.
- 868.350 Mhz used by Deltadore X2D or 868.950 Mhz used by Visonic.

However enlarge the spectrum and thus receiving several frequencies in the same band is allowed by decreasing the selectivity of the receiver by the management command 'SELECTIVITY H/L 1'. Emissions are not affected by frequencies differences because the transmitting frequency is dynamically adjusted frame by frame upon protocols.

Brand	Protocol	Frequency Mhz	Transmitting Capability (USB RF gateway)	Receiving Capability (USB RF gateway)	Repeating Capability	Transcoding Capability: Receiving Capability → Transmitting Capability
VISONIC	PowerCode SecureCode	433.920	Yes PowerCode only	Yes PowerCode &SecureCode	Yes PowerCode &SecureCode	Yes PowerCode &SecureCode(in)
VISONIC	PowerCode SecureCode	868.950				
CHACON/DIO	CHACON V2	433.920	Yes	Yes	Yes	Yes
DOMIA	DOMIA/CHACON V1	433.920	Yes	Yes	Yes	Yes
X10	X10	433.920	Yes	Yes	Yes	Yes
DELTADORE	X2D	433.920	Yes	Yes	Yes	Yes
DELTADORE	X2D	868.350				
SOMFY	RTS	433.420	Yes	Yes	Yes	Yes
FS20	FS20	868.350	Yes	Yes	Yes	Yes
Chacon	Edisio	868.350	Yes	Yes	Yes	Yes
BLYSS (433Mhz)	BLYSS/AVIDSEN433	433.920	Yes	Yes	Yes	Yes
KD101	KD101	433.920	Yes	Yes	Yes	Yes
PARROT (Learn & Play)	-	433/868Mhz	Yes	Yes	Yes	Yes
Scientific Oregon probes	OREGON V1	433.920	No	Yes	Yes	No
Scientific Oregon probes	OREGON V2.1	433.920	No	Yes	Yes	No
Scientific Oregon probes	OREGON V3.0	433.920	No	Yes	Yes	No
OWL probes	OWL	433.920	No	Yes	Yes	No
Cartelectronic TIC	TIC	433.920	No	Yes	Yes	No

Pulses						
--------	--	--	--	--	--	--

Trademarks

X10 is a trademark of X10 Company.

VISONIC, Powercode and SecureCode are trademarks of Visonic Company.

SOMFY and RTS are trademarks of Somfy Company.

DELTADORE and X2D are trademarks of Deltadore Company.

BLYSS is a trademark of KingFisher Company.

CHACON is a trademark of Chacon Company

OREGON is a trademark of Scientific Oregon Company.

5.1 Host → USB Dongle (configuration and transmitting data to RF)

5.1.1 ASCII data (ordinary used by commands Interface)

AsciiData[0...n]	Size	Remark
Local management	0...infinite	Upon 433/868 management commands interpreter

Preamble : Due to its ASCII nature and during debug, management commands (including the container header) can be entered by a PC utility eg. *Teraterm* when the Dongle is plugged on a PC. Terminator is \r or \0. Dual or over terminators e.g . “\r\n” are permitted because the first terminator is taken into account, the following are filtered because the receiver automata is looking for “ZI” header and strips others incoming characters. Replace “\r” by “\r\n” in the incoming flow for a best viewing. The incoming flow is sensitive to backspace character (\010) but excluding the header “ZIA++” (Useful during debug if commands are sent by a keyboard).

5.1.1.1 Commands

AsciiData[0...n] are analyzed by an interpreter of commands (see the list below). This interpreter has an input buffer of more 1000 characters. A command number can precede the entire line; this number is repeated in the answer.

The commands interpreter is insensitive to upper and lowercases. "ZIA++" header is sensitive.

One time "ZIA++" is received, the RF activity is stopped. The line remaining must be entered.

The format of the answer (TEXT, XML, JSON) can be specified after the command number.

STATUS and HELLO are the single commands which returns a response. Other commands do not return answer.

Several commands can be present in a single frame with a point (',') acting as command separator.

The management commands are able to :

- Configure the device,
- Send RF orders as ON/OFF/DIM etc... (Exactly as binary requests but with a more friendly way).

Command name	parameters	Explanations
FREQ	H/L val H: val = 0 or 868950 or 868350 L: val = 0 or 433420 or 433920	<p>Modify the receiver frequency on High band (around 868Mhz) or Low band (around 433Mhz). Low and high band receivers work simultaneously.</p> <p>H band : Set the high frequency receiver to Off, 868.950 or 868.350Mhz Default is 868.950Mhz</p> <p>L band : Set the Low frequency receiver to Off or 433.420 or 433.920 Mhz Default is 433.920 Mhz</p> <p>Frequency Value of 0 leads to shutdown the selected receiver including the transmitter of the specified band.</p> <p>Examples : FREQ H 868950 FREQ L 433920 FREQ H 0</p> <p>Specified frequency is saved during shutdown.</p>

SELECTIVITY	H/L val	<p>Modify the high frequency receiver selectivity on the selected band (L:433/H:868Mhz). Selectivity is the ability to filter out of band signals.</p> <p>NOTE: Higher selectivity (low value in term of Khz) means higher RF receiver sensitivity, but out of frequency transmitting appliances could be discarded. Lower selectivity (high value in term of Khz) means lower RF receiver sensitivity (lower performance). Old cheap devices can have a large frequency offset or shift over time especially when outdoor used.</p> <p>Val : 0 : Default value - Medium selectivity (300Khz) 1 : Very low selectivity (800Khz), frequency centered between used frequencies (433420-433920 or 868350-868950) 2 : Very low selectivity (800Khz) 3 : Low selectivity (500Khz) 4 : Medium selectivity (300Khz) 5 High selectivity (200Khz)</p> <p>Default : 0</p> <p>Examples : SELECTIVITY H 3 SELECTIVITY L 1</p> <p>Specified selectivity is saved during shutdown.</p>
SENSITIVITY	H/L val	<p>Radio Frequency receiver sensitivity on the selected band (L:433/H:868Mhz) (Ultra-High Frequency analog antenna sensitivity)</p> <p>Val : 0 : Default value - High sensitivity (-0dB) 1: Very low sensitivity (-18dB) 2 : low sensitivity (-12dB) 3 : medium sensitivity (-6dB) 4 : high sensitivity (-0dB) – Default value</p> <p>Default : 0</p> <p>Example : SENSITIVITY L 1 // set very low sensitivity on 433Mhz</p> <p>Decreasing RF sensitivity could be only useful in specific cases during limited time, as pairing procedure or RF sequence learning with isolation from far RF transmitters.</p> <p>Specified sensitivity is NOT saved during shutdown.</p>
DSPTRIGGER	H/L val	<p>Digital Signal Processing trigger on the selected band (L:433/H:868Mhz) Define the smallest signal amplitude leading to start frame detection and analysis. Low trigger value means high sensitivity. Too big trigger value leads to forget useful frames. Too low trigger value leads to detect ghostly frames, generated by floor noise, and sometimes forget useful frames during this time.</p> <p>Val : 4 to 20. Unit : dBm. Val = 0 or out of bounds value lead to come back to the default value.</p>

		<p>Default values :</p> <p>433 Mhz = 8dBm 868Mhz = 6dBm</p> <p>Example :</p> <p>DSPTRIGGER L 15</p> <p>Specified DSPTRIGGER value is saved during shutdown.</p>
RFLINK	val	<p>RFLINK engine enabling/disabling</p> <p>RFLINK 1 : enable RFLINK RFLINK 0 : disable RFLINK</p> <p>By default RFLINK engine is enabled.</p> <p>Note : keep RFLINK enabled when Parrot is used. RFLINK state is saved during shutdown.</p>
RFLINKTRIGGER	H/L val	<p>RFLINK trigger on the selected band (L:433/H:868Mhz) Define the smallest signal amplitude leading to start frame detection and analysis. Low trigger value means high sensitivity. Too big trigger value leads to forget useful frames. Too low trigger value leads to detect ghostly frames, generated by floor noise, and sometimes forget useful frames during this time.</p> <p>Val : 4 to 20. Unit : dBm. Val = 0 or out of bounds value lead to come back to the default value.</p> <p>Default values :</p> <p>433 Mhz = 12dBm 868Mhz = 10dBm</p> <p>Example :</p> <p>RFLINKTRIGGER L 15</p> <p>Specified RFLINKTRIGGER value is saved during shutdown.</p>
LBT	val	<p><i>Listen Before Talk</i> function Val : 6 to 30. Unit : dBm</p> <p>Out of bounds value of val leads to come back to the default value. Val = 0 inhibits LBT function. Default is LBT enabled (very highly recommended).</p> <p>When enabled, the transmitter will “listen” the current activity on the same frequency and wait a silence before to “talk”. Sent frames cannot be delayed more than 3 seconds.</p> <p>Default value : 16dBm</p> <p>Example :</p> <p>LBT 10</p> <p>Specified LBT value is saved during shutdown.</p>
SETMAC	mac	<p>Set the interface MAC address to unsigned long decimal 32 bits value.</p> <p>Warning! Changing the MAC address is without effect on incoming RF Frames. But changing the MAC address will modify the ID contained in outgoing RF Frames on most protocols. Pairing between the dongle</p>

		<p>and actuators could be broken.</p> <p>Each Dongle owns a different MAC address configured at the factory.</p> <p>Examples :</p> <p>SETMAC 123456765 SETMAC 0x2AB265C3</p> <p>Specified MAC value is saved during shutdown.</p>
FACTORYRESET	[ALL]	<p>Restore factory default parameters. Include "REMAPPING PARROT" command. PARROT records and TRANSCODER configuration are not affected.</p> <p>ALL optional parameter erases all, including PARROT records and TRANSCODER configuration.</p> <p>Examples :</p> <p>FACTORYRESET FACTORYRESET ALL</p> <p>Restored values are saved during shutdown.</p>
STATUS	<p>[SYSTEM, RADIO, TRANSCODER, PARROT, ALARM]</p> <p>[TEXT, XML, JSON]</p>	<p>Gives the status of the device for a specific item : SYSTEM, RADIO, TRANSCODER, PARROT, TRANSCODER. The status can be given with several formats : TEXT, XML, JSON. (default : TEXT)</p> <p>Examples</p> <p>STATUS STATUS SYSTEM XML STATUS RADIO JSON STATUS TRANSCODER JSON // list all entries STATUS TRANSCODER ENTRY 5 JSON // list one entry STATUS PARROT XML // list all records STATUS PARROT B1 ON XML // list one record STATUS ALARM JSON // list all entries STATUS ALARM AREA 1 JSON // list all entries off area 1 STATUS ALARM ENTRY 5 JSON // list one entry regardless area</p> <p>See specific paragraph below with examples of this command.</p>
FORMAT	<p>OFF BINARY HEXA HEXA FIXED TEXT XML JSON</p>	<p>Gives the format of the received RF Frames sent to USB port. FORMAT OFF shutdowns incoming RF frames to USB port.</p> <p>FORMAT command doesn't give immediate responses, but asynchronous messages, one at each received RF frame.</p> <p>By default, format is OFF.</p> <p>After the Dongle detection, the home automation BOX has to set the chosen format with this command.</p> <p>Typically, an home automation BOX sends at startup: FORMAT BIN or FORMAT HEX or FORMAT XML or FORMAT JSON Etc...</p>

	<p>RFLINK OFF RFLINK BINARY</p>	<p>NOTE: Shutdown the repeater during GATEWAY usage is recommended because the time spent to reapeate frames is lost for receive frames during this time. Thus the instruction FORMAT automatically disables the REPEATER function with an implicit "REPEATER OFF". The REPEATER function can be re-enabled with "REPEATER ON".</p> <p>Example : FORMAT XML</p> <p>See Specific paragraph below with examples of this command.</p> <p>FORMAT RFLINK OFF disables the RFLINK flow FORMAT RFLINK BINARY enables RFLINK flow. Due to the RFLINK records length, RFLINK flow can be delivered only in binary form.</p> <p>NOTE : Frames with RFLINK format are delivered to the host machine only when the internal engine ("One Step Decoder") is unable to decode the frame.</p> <p>Example : FORMAT RFLINK BINARY</p> <p>Specified FORMAT is NOT saved during shutdown.</p>
HELLO		<p>Returns the text string "Welcome to Ziblue Dongle xxxxxxxxxxxxxxx". The beginning of the returned string is usable to recognize the Dongle at the startup of host application. xxxxxxxxxxxxxx is variable and MUST not be matched. Not sensitive to XML / JSON formats specification.</p> <p>Example : ZIA++HELLO ZIA-- Welcome to Ziblue Dongle RFPLAYER (RFP1000, Firmware V1.12 Mac 0xF6C09FA1)!</p>
PING		<p>Returns the text string "PONG". Could be useful to check periodically the alive state of the dongle. The command Hello can be used for the feature too. Not sensitive to XML / JSON formats specification.</p> <p>Example : PING</p>
<p>ON OFF DIM ASSOC DISSOC ASSOC_OFF DISSOC_OFF TOGGLE (edisiu only)</p> <p>Only X10 & CHACON: ALL_ON ALL_OFF</p>	<p>ID x or pseudo-address X10 form</p> <p>Protocol chosen in the list : VISONIC433 VISONIC868 CHACON DOMIA X10 X2D433 X2D868 X2DSHUTTER X2DELEC X2DGAS</p>	<p>Send an order over RF (ASCII form). This request exists in binary form. (see next chapters).</p> <p>ID x or pseudo-address X10 form : address of the RF appliance. ID x : with x between 0 and 255 included. pseudo-address X10 form : equivalent to IDx but with more friendly form. A—P 1—16. ID 0 = A1 ID 15 = A16 ... ID 255 = P16</p> <p>% x : parameter of DIM command. Light set at x %</p>

	<p>RTS BLYSS PARROT KD101 FS20 EDISIO</p> <p>% x BURST x QUALIFIER x</p>	<p>BURST x : define a frame repetition factor. 0 is default.</p> <p>QUALIFIER : define an optional parameter eg shutter (0) or portal (1) for Somfy RTS.</p> <p>Mandatory :</p> <ul style="list-style-type: none"> - ID x or pseudo-address X10 form - Protocol <p>NOTE : Parameters can be given in any order.</p> <p>The Host can request ASSOC (pairing dongle<->appliance) through this command. ASSOC_OFF is useful to fill one entry 'OFF' of PARROT (ASSOC fills "ON" OF PARROT). In this way, a user reminder can not be specified. The command PARROTLEARN is another way to fill ON or OFF entry of PARROT but with a user reminder.</p> <p>Examples :</p> <p>ON A3 RTS QUALIFIER 1 DIM ID 12 CHACON %40 ON KD101 ID 90500 ASSOC X2D868 F7</p> <p><u><i>X2DGAS specific case</i></u> X2DGAS protocol emulates X2D GAS/Boiler/AC thermostat. Generated frames are complex and multiples (3 types of frame: heating speed, regulation, and operating mode). Heating speed is fixed. Regulation is set by ON/OFF value and operating mode by dim value. Dim value (arg %) is used to carry the mode of the thermostat (mode used values : 0: ECO, 3: COMFORT, 4: STOP, 5: OUT OF FROST, 7: AUTO)</p> <p>Examples:</p> <p>ON X2DGAS A3 %7 generates : HEATING SPEED: state: 1 (ON) REGULATION: state: 1 (ON) OPERATING MODE: state: 7 (AUTO)</p> <p>OFF X2DGAS A3 %7 generates : HEATING SPEED: state: 1 (ON) REGULATION: state: 0 (OFF) OPERATING MODE: state: 7 (AUTO)</p> <p>ON X2DGAS A3 %5 generates : HEATING SPEED: state: 1 (ON) REGULATION: state: 1 (ON) OPERATING MODE: state: 5 (OUT OF FROST)</p> <p><u><i>X2DELEC specific case</i></u> X2DELEC protocol emulates X2D pilot wire thermostat. Generated frames are complex and multiples (2 types of frame: heating speed, and operating mode). Heating speed is set by ON/OFF value and operating mode by dim value. Dim value (arg %) is used to carry the mode of the thermostat (mode used values : 0: ECO, 3: COMFORT, 4: STOP, 5: OUT OF FROST, 7: AUTO)</p>
--	--	---

		<p>Examples: ON X2DELEC A3 %7 generates : HEATING SPEED: state: 1 (ON) OPERATING MODE: state: 7 (AUTO)</p> <p>OFF X2DELEC A3 %7 generates : HEATING SPEED: state: 0 (OFF) OPERATING MODE: state: 7 (AUTO)</p> <p>ON X2DELEC A3 %5 generates : HEATING SPEED: state: 1 (ON) OPERATING MODE: state: 5 (OUT OF FROST)</p> <p><i>FS20 specific case</i> <i>Read specs : http://fhz4linux.info/tiki-index.php?page=FS20%20Protocol</i> ON/OFF/DIM are directly supported (DIM value % mut be null) OFF : CMD code 0 (as generated by remote control) ON : CMD code 0x11 (as generated by remote control) DIM : CMD code 0...16 of the specs (entire part of percent value / 6) Example : DIM A3 FS20 %50 (FS20 CMD code = 8)</p> <p>All CMD codes (=“Befehl” field) are emulated by DIM pseudo-values 1...256 (not null) during apparent ON/OFF/ASSOC order. Thus D0...7 bit fields are supported Example : ON A3 FS20 %203 // DIM68,bidir,dest answer</p> <p><i>FS20 command field (CMD)</i> D0:4 : “befehl” tab D5 : 1: extension field exists. 0 : regular. D6 : 1 : bidirectional command. 0 : regular. D7 : 1 : Answer of a receiver..0 : regular</p>
RECEIVER	<p>Operators : + -</p> <p>Protocol chosen in the list : * X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PARROT TIC FS20 JAMMING EDISIO</p>	<p>Specify the list of enabled received protocols</p> <p>The operators specify if the subsequent protocols are to add or remove. Parameters are read from left to right.</p> <p>Examples : RECEIVER + CHACON BLYSS // add CHACON and BLYSS to the current list RECEIVER - * + CHACON // receive only chacon RECEIVER + * - CHACON // remove CHACON from the current list</p> <p>Specified list is saved during shutdown.</p>
REPEATER	<p>Operators : + -</p> <p>Protocol chosen in the list : *</p>	<p>Specify the list of enabled repeated protocols</p> <p>The operators specify if the subsequent protocols are to add (+) or remove (-).</p>

	<p>X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 TIC FS20 EDISIO</p> <hr/> <p>ON OFF</p>	<p>Parameters are read from left to right.</p> <p>Examples : REPEATER + CHACON BLYSS // add CHACON and BLYSS to the current list REPEATER - * + CHACON // remove ALL but repeate only CHACON REPEATER + * - CHACON // repeate ALL but remove CHACON from the current list</p> <p>Specified list is saved during shutdown.</p> <hr/> <p>ON : Temporary enables the repeater function. OFF : Temporary disables the repeater function. “REPEATER ON” / “REPEATER OFF” commands are NOT saved during shutdown. It is a fast way to enable/disable REPEATER without saving. After startup, “REPEATER ON” is the default state.</p> <p>NB: The instruction “FORMAT xxxx” executes automatically “REPEATER OFF” because when the dongle is connected to an HA BOX, the REPEATER is generally OFF to maximize receiving time. “REPEATER ON” permits to re-enable the REPEATER with an HA BOX.</p>
LEDACTIVITY	0 / 1	<p>Enable ou disable LED activity related to RF flow. LEDACTIVITY 0 disables LED activity LEDACTIVITY 1 enables LED activity</p> <p>Default is 1 after factory RESET.</p> <p>LED activity cannot be disabled during PARROTLEARN and TRANCODER processings or when the dongle signals an error.</p>
INITLB		<p>(re-) Initialization of the leaky buckets (LB) to full level.</p> <p>Each frequency owns a leaky bucket. At each frame sent, the leaky bucket level decreases, but idle time increases this level. Frames are not send and discarded when LB level reaches 0. The LB defines a maximum duty cycle per hour defined by European laws (e.g. 433Mhz: 10%, 868.350Mhz: 1%, 868.950Mhz: 0.1%) where the channel is usable to send frames.</p> <p>INTLB could be useful during tests to disable LB limitation of bandwidth.</p> <p>Example : INITLB. ON X10 A1</p>
PARROTLEARN	<p>ID x or pseudo-address X10 form</p> <p>ON or OFF</p> <p>[reminder]</p>	<p>Specify to PARROT learn a frame. Dongle RF sensitivity decreases during this processing. The dongle then enters into ‘capture mode’. Place the transmitter to learn at 2-3m et force it to emit frames. Decrease gradually the distance if no effect. Too near device will give bad results. The frame must be captured 2 times (1: Low frequency blue blinking then 2: High frequency blue blinking ; Good comparison : PINK Lighting, bad comparison : RED lighting).</p> <p>Without success This processing is automatically stopped :</p> <ul style="list-style-type: none"> - After around 2mn (upon your tries). - When the lateral button of the dongle is pushed

		<p>ID x or pseudo-address X10 form : address of the RF appliance. ID x : with x between 0 and 239 included. pseudo-address X10 form : equivalent to IDx but with more friendly form. A—O 1—16. P1 to P16 are reserved. A1...B16 are sensitive entries and can be recognized in real-time when the associated frames are send later by the device. C1...O16 can be played on RF but not recognized in the incoming RF flow.</p> <p>Optional parameter ON or OFF specify if the frame means one ON or one OFF. Default is ON.</p> <p>A reminder can be specified between brackets []. Max Length is 30 characters. Reminder is a text line chosen by the user to remember which device has been captured. Reminder will be displayed later with “STATUS PARROT” command</p> <p>Examples : PARROTLEARN A3 [sensor bedroom1] PARROTLEARN ID 17 ON [Gate open sensor] PARROTLEARN ID 17 OFF [Gate close sensor] PARROTLEARN B2 ON</p> <p>NOTE : Specify the chosen frequency with FREQ before to use this command. This frequency is memorized with the frame record and will be automatically used when the frame will be played.</p> <p>PARROTLEARN results are saved during shutdown.</p>
REMAPPING	<p>PARROT</p> <p>ONOFF < Send an order over RF command parameters></p> <hr/> <p>CLEAR</p>	<p>REMAPPING is a transcoder shortcut command to remap all PARROT RF sensitive entries (A1...B16) to another unique protocol. It is useful when REPEATER function is enabled. ON or OFF PARROT attributes is reported to the output protocol.</p> <p>Examples : REMAPPING PARROT ONOFF RTS D1 (remap A1...B16 PARROT ENTRY to D1...E16 on RTS protocol (shutter)) REMAPPING PARROT ONOFF RTS D1 QUALIFIER 1 (remap A1...B16 PARROT ENTRY to D1...E16 on RTS protocol (portal))</p> <p>NOTE : TRANSCODER command has a finest granularity, entry by entry. TRANSCODER has priority over REMAPPING command if transcoding conflicts appear.</p> <hr/> <p>CLEAR removes PARROT REMAPPING Exemple : REMAPPING PARROT CLEAR</p> <p>Specified REMAPPING PARROT ONOFF or CLEAR results are saved during shutdown.</p>
TRANSCODER	<p><Source> To <Destination></p> <p>[reminder]</p>	<p>TRANSCODER command permits to transcode an incoming RF frame to another protocol on one outgoing RF FRAME. TRANSCODER is a sub-function of the REPEATER. There are 32 ENTRIES. ENTRY value range : 0...31. Entries are shared between TRANSCODER</p>

		<p>and ALARM so that when one entry is used by TRANSCODER, this entry becomes unavailable for ALARM and of course when one entry is used by ALARM, this entry becomes unavailable for TRANSCODER. STATUS command response gives “maskT” label containing <i>available</i> or <i>used</i> entries by TRANSCODER, and “maskA” label for <i>available</i> or <i>used</i> entries by alarm function. It is useful for entries managing and create separate pools for TRANSCODER and ALARM.</p> <p>“maskT” and “maskA” are expressed as a 32 bits mask where D0 gives entry 0 state and D0 gives entry 31 state. “0” “ in maskT” means used by ALARM, “1” in maskT” means used by TRANSCODER or free (=available), “0” “ in maskA” means used by TRANSCODER, “1” in maskA” means used by ALARM or free (=available).</p> <p>Incoming action (ON or OFF) are reported as outgoing action on X10, DOMIA, CHACON, RTS, PARROT protocols, so that 64 different frames can be generated.</p> <p>Other protocols are strictly transcoded by destination parameters.</p> <p>In operation, frame parameters are read in real time in the incoming RF flow and compared to TRANSCODER <source> parameters. If they match, the frame is transcoded with <destination> parameters.</p> <p><source> Source can be :</p> <ul style="list-style-type: none"> - Expressed by a line of parameters - Captured in real time from the incoming RF flow (by “CAPTURE”) - Retrieved from the already memorized parameters (by “KEEP”) <p><i>Expressed by a line of parameters :</i></p> <ul style="list-style-type: none"> - ENTRY x with x between 0 and 31 included. - Protocol chosen in the list : X10, VISONIC, BLYSS, CHACON, OREGON, DOMIA, OWL, X2D, RTS, KD101, PARROT, FS20, EDISIO - ID x or X10 form : Address of the RF transmitter (most time a 32 bits ID) - Optional SUBTYPE x with x protocol dependent (default : 0) - Optional QUALIFIER x with x protocol dependent (default : 0) <p><i>Captured in real time from the incoming RF flow :</i> ‘CAPTURE’ or ‘CAPTURE Protocol’ to filter undesirable protocols When CAPTURE is specified, the dongle enters into ‘capture mode’ with blue LED blinking during 1 mn. Push the lateral button of the dongle to break the processing. The frame must be captured twice times (1: Low frequency blinking then 2: High frequency blinking ; Good matching: PINK Lighting, bad matching: RED lighting). NOTE: Transcoding starts immediately after if REPEATER is enabled (including current frame), so a short RED lighting (transmitting signal) could appear after PINK lighting.</p> <p>Retrieved from the already memorized parameters : ‘KEEP’</p> <p><destination> ON or OFF or DIM actions followed by “Send an order over RF”</p>
--	--	--

	<p>CLEAR</p>	<p>parameters.</p> <p>Examples :</p> <p>TRANSCODER ENTRY 23 X10 D8 TO ON RTS B7 TRANSCODER ENTRY 23 CHACON ID 435432766 TO ON RTS B7 TRANSCODER ENTRY 23 CHACON CAPTURE TO ON RTS B7 TRANSCODER ENTRY 23 KEEP TO ON RTS ID 12 QUALIFIER 1</p> <p><i>[reminder]</i> A reminder can be specified between brackets []. Max Length is 30 characters. Example: TRANSCODER ENTRY 23 X10 D8 TO ON RTS B7 [My Comment]</p> <p>Reminder is a text line chosen by the user to remember which device has been captured. Reminder will be displayed later with "STATUS TRANSCODER" command</p> <p>Examples :</p> <p>TRANSCODER ENTRY 23 X10 D8 TO ON RTS B7 [My reminder here]</p> <hr/> <p>CLEAR argument permits to clear all TRANSCODER entries by "TRANSCODER CLEAR" or a specific entry "TRANSCODER ENTRY <i>n</i> CLEAR"</p> <p>Specified TRANSCODER command results are saved during shutdown.</p>
TRACE	<p>Operators : + -</p> <p>Tracable functions chosen in the list :</p> <p>*</p> <p>ALARM RECEIVER TRANSMITTER TRANSCODER REPEATER JAMMING RFLINK</p>	<p>Specify traced functions. Trace function is useful for debug. It provides a flow of asynchronous log events lines given in free ASCII text form with header "ZIA55". The operators specify if the subsequent functions are to add or remove from the traced functions list. Parameters are read from left to right.</p> <p>Examples :</p> <p>TRACE + RECEIVER // add RECEIVER to the current list TRACE + RECEIVER - TRANSMITTER // add RECEIVER and remove TRANSMITTER to the current list TRACE - * + ALARM // trace only ALARM TRACE + * - ALARM // trace all excluding ALARM</p> <p>JAMMING TRACE gives a periodic info (every 3s) on Jamming detection with the current level detection (0...10). The user specified jamming threshold (0...10) is given too. NOTE: Current level >= threshold means Jamming ON.</p> <p>The Annex gives an example of TRACE RFLINK instruction response. TRACE RFLINK is reserved for debug and can not be used for operational because printing is relatively low process and new incoming frames can be lost during samples list printing. RFLINK operational mode uses binary samples printing. FORMAT RFLINK BINARY instruction isn't needed to enable RFLINK trace.</p> <p>Specified list is NOT saved during shutdown.</p>
ALARM		<p>ALARM command permits to define up to 4 areas alarm system. Areas are fully independent. Peripherals can be shared by several areas but</p>

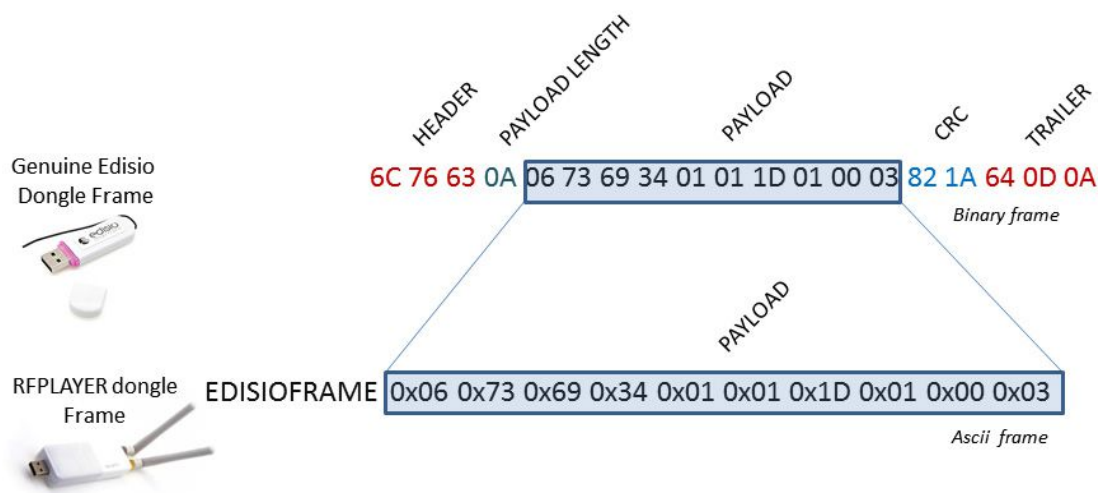
	<p>Specify global alarm parameters: PREARM x PREALARM x ALARMTIME x NTIMES x</p> <p>Save peripherals IDs: AREA x ENTRY y DETECTOR <source> AREA x ENTRY y RC_ON <source> AREA x ENTRY y RC_OFF <source> AREA x ENTRY y SIREN <dest> AREA x ENTRY y PRESIREN <dest> AREA x ENTRY y ACK <dest></p> <p>Specify autostart : AREA x AUTOSTART 0/1</p> <p>Clear entry: CLEAR AREA x CLEAR ENTRY x CLEAR</p> <p>Remote control emulation ALARM AREA x ON ALARM AREA x OFF</p>	<p>the declarations are independent. ALARM function doesn't affect other functions of RFPLAYER. Both can be running.</p> <p>AREA value range : 0...3</p> <p>ENTRY value range : 0...31. Entries are shared with TRANSCODER so that when one entry is used by TRANSCODER, this entry becomes unavailable for ALARM and of course when one entry is used by ALARM, this entry becomes unavailable for TRANSCODER. STATUS command response gives "maskT" label for available/used entries by TRANSCODER, and "maskA" label for available/used entries by alarm function. It is useful for entries managing. "maskT" and "maskA" are expressed as a 32 bits mask where D0 gives entry 0 state and D31 gives entry 31 state. "0" " in maskT" means used by ALARM, "1" in maskT" means used by TRANSCODER or free (=available), "0" " in maskA" means used by TRANSCODER, "1" in maskA" means used by ALARM or free (=available).</p> <p><u>Regular Processing</u> The alarm system, on a given area, is armed by default 30s (programmable PREARM param) after RC_ON frame receiving. When a DETECTOR frame happens, the PRESIREN is then immediately launched and the SIREN launched after 30s (programmable PREALARM param) during 2 mn (programmable ALARMTIME param). After 3 SIREN launchings (programmable NTIMES param), the alarm system is stopped up to the RC_ON frame is received again. Any number of Remote controls, detectors, sirens, presirens (buzzers), can be memorized but the sum (on all areas) must be below or equal to 32 (entries). Entries used by the TRANSCODER must be subtracted. ACK gives a feedback to the user by a short pulse ON-OFF, when the alarm is set ON or OFF by a remote control. ACK is typically linked to a buzzer so that user can verify the ON/OFF alarm state confirmation. Another ACK is send when a detector triggers the alarm system.</p> <p>NOTE : A visual ACK is done by a smooth transition on the bicolor LED : BLUE to RED : The alarm system becomes "ON". RED to BLUE : The alarm system becomes "OFF". BLUE to RED to BLUE : A detector triggers the alarm system. Presirens become ON.</p> <p>NOTE : Pushing RFPLAYER lateral button will set ALARM OFF on all areas.</p> <p><u>Global alarm parameters Setting</u> PREARM, PREALARM, ALARMTIME parameters are expressed in seconds up to 255. NTIMES is up to 255 and can be modified too. A '0' parameter means default usage. Examples: ALARM PREARM 120 // set PREARM parameter to 120 seconds. ALARM NTIMES 5 // set PREARM parameter to 5.</p> <p><u>Peripherals IDs saving</u> The saving parameters IDs are very similar to TRANSCODER function (but need AREA parameter). See <source> and <dest> description of the TRANSCODER function. <source> and <dest> are never specified on one unique line. All protocols can be mixed together by the alarm system. RC_OFF frame value must be specified only when the associated</p>
--	--	--

		<p>protocol to RF_ON doesn't specify OFF operation. Example of a complete ALARM area declaration : ALARM AREA 3 RC_ON ENTRY 1 CHACON CAPTURE [my RF RC] ALARM AREA 3 DETECTOR ENTRY 2 CHACON CAPTURE [My detector1] ALARM AREA 3 DETECTOR ENTRY 3 VISONIC CAPTURE [My detector2] ALARM AREA 3 PRESIREN ENTRY 12 TO ON CHACON A2 [My presiren] ALARM AREA 3 SIREN ENTRY 13 TO ON BLYSS A1 [My siren1] ALARM AREA 3 SIREN ENTRY 14 TO ON X10 P16 [My siren2] ALARM AREA 3 ACK ENTRY 15 TO ON CHACON A5 [My ACK]</p> <p><u>Autostart</u> An area can be set automatically at startup. It is useful when alarm ON/OFF have no meaning e.g. with smoke/water detectors etc.... By default Autostart is off. Examples: ALARM AREA 3 AUTOSTART 0 // remove autostart on area 3 ALARM AREA 3 AUTOSTART 1 // remove autostart on area 3</p> <p><u>Entry clear</u> ALARM CLEAR // Clear all entries an all areas ALARM AREA 2 CLEAR // Clear all entries of area 2 ALARM entry 12 CLEAR // Clear entry 12 regardless the area</p> <p><u>Remote control emulation</u> An alarm system area can be set ON or OFF or detector activity simulated by software through the API. Examples : ALARM AREA 2 ON // enable alarm on AREA 2 ALARM AREA 2 OFF //disable alarm on AREA 2 ALARM AREA 2 DETECTION //simulate detector on AREA 2</p> <p>NOTE : ALARM commands are saved during shutdown, excluding Remote control emulation. Current alarm state is not saved during shutdown.</p>
JAMMING	JAMMING <threshold> JAMMING SIMULATE JAMMING SIMULATE <delay>	See 'Jamming detection' chapter. Commands : JAMMING <threshold> Where <i>threshold</i> is a value between 0 and 10. Default threshold value is 7. Value 0 disables jamming detection. Value 1 is very sensitive but should lead to "false positives" events. Value 10 is less sensitive but limits "false positives" events. Minimal time to trigger "JAMMING ON" is around 15s. A simulator also allows you to send yourself alerts to virtually test the presence of a jammer by : JAMMING SIMULATE Will force receiving "JAMMING ON" frame. "JAMMING OFF" Frame is received 5 seconds later. JAMMING SIMULATE <delay> Will force receiving "JAMMING ON/OFF" frame ' <i>delay</i> ' seconds later. (eg JAMMING SIMULATE 60). Max value of delay is 255 seconds (more than 4 minutes). This

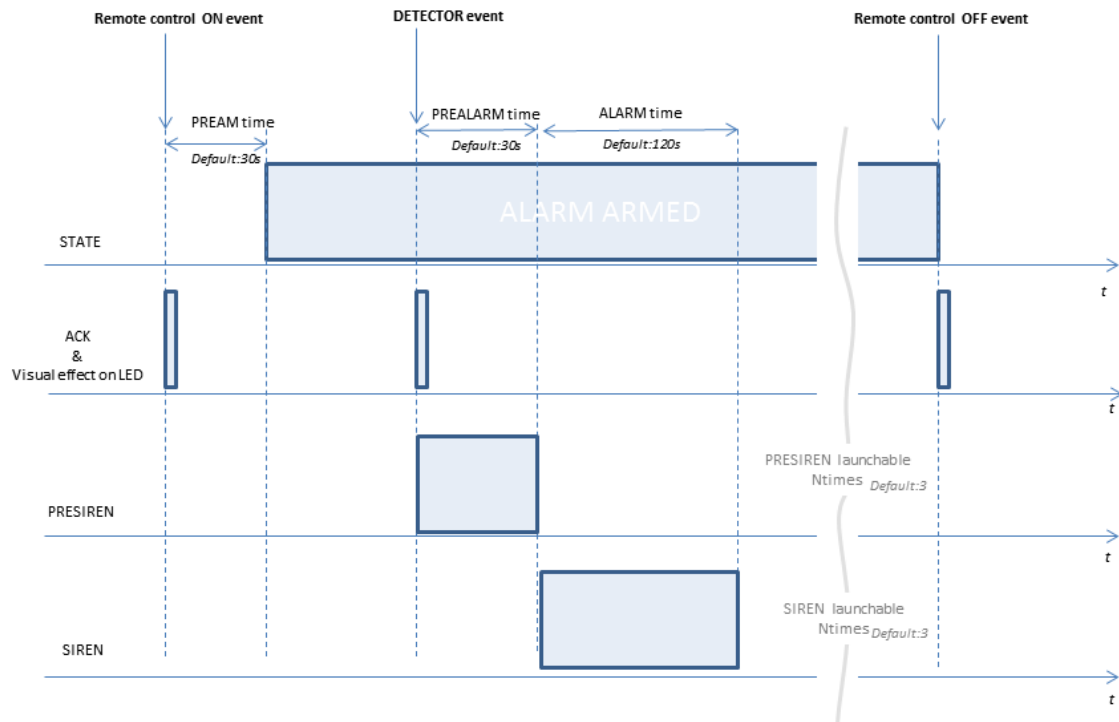
		<p>command is saved if the RFPLAYER shutdowns before JAMMING ON event. Thus, this command can be programmed on the RFPLAYER configurator, then the RFPLAYER plugged on the HA Box to test "JAMMING ON" event.</p> <p>Of course, the HA Box can send "JAMMING SIMULATE" or "JAMMING SIMULATE <delay>" command itself.</p> <p>JAMMING parameters are saved during shutdown.</p>
SIREN	SIREN <state> <time>	<p>Useful for JamTrack devices with external connector for Siren (Jack 3.5mm).</p> <p>This command permits to Home Automation Boxes to enable/disable the siren of JamTrack.</p> <p><time> is optional (default : 240s) and cannot be greater than 240s.</p> <p>Example :</p> <p>SIREN 1 10 ← Enable the siren during 10 seconds.</p> <p>SIREN 0 ← Disable Siren</p>

EDISIOFRAME	EDISIOFRAME <bytes of frame>	<p>EDISIOFRAME gives the possibility to send and receive edisio frames in a relatively <u>raw format</u>.</p> <p>This RFPLAYER format is a simplified and friendly version of the binary API of the genuine Edisio dongle. It is useful in the case where a software driver has already been developed for this dongle and have to be adapted to RFPLAYER.</p> <p>The RFPLAYER format offers the advantage of avoiding binary data manipulation, frame delineation and CRC calculation (CRC calculations is done by RFPLAYER).</p> <p>EDISIOFRAME action with or without parameters is used to enable asynchronous receiving of Edisio Frames with ZIA66 header. This enabling isn't saved during shutdown.</p> <p><u>Examples :</u> Sending an Edisio Frame (Edisio regular length is 10 bytes but could be longer): ZIA++ EDISIOFRAME 0x06 0x73 0x69 0x34 0x05 0x01 0x1D 0x01 0x00 0x03</p> <p>Receiving an asynchronous Edisio Frame: ZIA66 EDISIOFRAME 0x06 0x73 0x69 0x34 0x05 0x01 0x1D 0x01 0x00 0x03</p> <p>Nota that syntax is the same for transmitting and receiving. Receiving is always in Hexadecimal form (with 0x as prefix at each byte). Transmitting uses decimal or Hexadecimal form (with 0x as prefix at each byte for hexadecimal).</p> <p>See below paragraph upon RFPLAYER Edisio Frame.</p>
-------------	------------------------------	---

EDISIOFRAME versus Genuine Edisio Dongle Frame



5.1.1.1.1 ALARM chronogram



5.1.1.1.2 STATUS answer examples

The line “ZIA++1234 STATUS SYSTEM” will return :

ZIA--1234 STATUS SYSTEM

systemStatus request number=1234

Version: 1.17, Time: 29s, Mac: 0xFD0CBA71, LBT: 16dBm, Factory: 1488874419, ClusterID: 0, RTdenials: 0, MaskT: 0xFFBF8FF9, MaskA: 0x3FFFFFFE,

transmitter available: VISONIC433 VISONIC868 CHACON DOMIA X10 X2D433 X2D868 X2DSHUTTER X2DELEC X2DGAS RTS BLYSS PARROT KD101

receiver available: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PARROT

receiver enabled: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PARROT

repeater available: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PARROT

repeater enabled: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PARROT

The line “ZIA++1234 STATUS SYSTEM XML” will return :

```
ZIA--<?xml version="1.0" encoding="ISO8859-1" ?><systemStatus><reqNum>1234</reqNum><i> <n>Version</n> <v>1.17</v> <unit></unit> <c></c></i><i>
<n>Time</n> <v>138</v> <unit>s</unit> <c></c></i><i> <n>Mac</n> <v>0xFD0CBA71</v> <unit></unit> <c></c></i><i> <n>LBT</n> <v>16</v>
<unit>dBm</unit> <c></c></i><i> <n>Factory</n> <v>1488874419</v> <unit></unit> <c></c></i><i> <n>ClusterID</n> <v>0</v> <unit></unit> <c></c></i><i>
<n>RTdenials</n> <v>0</v> <unit></unit> <c></c></i><i> <n>MaskT</n> <v>0xFFBF8FF9</v> <unit></unit> <c></c></i><i> <n>MaskA</n>
<v>0x3FFFFFFE</v> <unit></unit>
<c></c></i></transmitter><available><p>VISONIC433</p><p>VISONIC868</p><p>CHACON</p><p>DOMIA</p><p>X10</p><p>X2D433</p><p>X2D868</p><p>
X2DSHUTTER</p><p>X2DELEC</p><p>X2DGAS</p><p>RTS</p><p>BLYSS</p><p>PARROT</p><p>KD101</p></available></transmitter><receiver><available>
<p>X10</p><p>RTS</p><p>VISONIC</p><p>BLYSS</p><p>CHACON</p><p>OREGONV1</p><p>OREGONV2</p><p>OREGONV3/OWL</p><p>DOMIA</p><p>X
2D</p><p>KD101</p><p>PARROT</p></available></receiver><receiver><enabled><p>X10</p><p>RTS</p><p>VISONIC</p><p>BLYSS</p><p>CHACON</p><p>
OREGONV1</p><p>OREGONV2</p><p>OREGONV3/OWL</p><p>DOMIA</p><p>X2D</p><p>KD101</p><p>PARROT</p></enabled></receiver><repeater><
available><p>X10</p><p>RTS</p><p>VISONIC</p><p>BLYSS</p><p>CHACON</p><p>OREGONV1</p><p>OREGONV2</p><p>OREGONV3/OWL</p><p>DOMI
A</p><p>X2D</p><p>KD101</p><p>PARROT</p></available></repeater><repeater><enabled><p>X10</p><p>RTS</p><p>VISONIC</p><p>BLYSS</p><p>CH
ACON</p><p>OREGONV1</p><p>OREGONV2</p><p>OREGONV3/OWL</p><p>DOMIA</p><p>X2D</p><p>KD101</p><p>PARROT</p></enabled></repeater
></systemStatus>
```

The line “ZIA++1234 STATUS SYSTEM JSON” will return :

```
ZIA--{"systemStatus": {"reqNum": "0", "info": [{"n": "Version", "v": "1.17", "unit": "", "c": ""}, {"n": "Time", "v": "313", "unit": "s", "c": ""}, {"n": "Mac", "v": "0xFD0CBA71", "unit": "", "c": ""}, {"n": "LBT", "v": "16", "unit": "dBm", "c": ""}, {"n": "Factory", "v": "1488874419", "unit": "", "c": ""}, {"n": "ClusterID", "v": "0", "unit": "", "c": ""}, {"n": "RTdenials", "v": "0", "unit": "", "c": ""}, {"n": "MaskT", "v": "0xFFBF8FF9", "unit": "", "c": ""}, {"n": "MaskA", "v": "0x3FFFFFFE", "unit": "", "c": ""}], "transmitter": {"available": {"p": ["VISONIC433", "VISONIC868", "CHACON", "DOMIA", "X10", "X2D433", "X2D868", "X2DSHUTTER", "X2DELEC", "X2DGAS", "RTS", "BLYSS", "PARROT", "KD101"]}}, {"receiver": {"available": {"p": ["X10", "RTS", "VISONIC", "BLYSS", "CHACON", "OREGONV1", "OREGONV2", "OREGONV3/OWL", "DOMIA", "X2D", "KD101", "PARROT"]}}, {"receiver": {"enabled": {"p": ["X10", "RTS", "VISONIC", "BLYSS", "CHACON", "OREGONV1", "OREGONV2", "OREGONV3/OWL", "DOMIA", "X2D", "KD101", "PARROT"]}}, {"repeater": {"available": {"p": ["X10", "RTS", "VISONIC", "BLYSS", "CHACON", "OREGONV1", "OREGONV2", "OREGONV3/OWL", "DOMIA", "X2D", "KD101", "PARROT"]}}, {"repeater": {"enabled": {"p": ["X10", "RTS", "VISONIC", "BLYSS", "CHACON", "OREGONV1", "OREGONV2", "OREGONV3/OWL", "DOMIA", "X2D", "KD101", "PARROT"]}}}}
```

The line “ZIA++1234 STATUS RADIO” will return :

ZIA--

radioStatus request number=1234

Frequency: 433920Khz Adapted to most 433Mhz devices,

Selectivity: 0 Default value, FloorNoise: -97dBm A bit noisy, DspTrigger: 8dBm, RFlink: 1 Enabled, RFlinkTrigger: 12dBm, sentFrames: 272, discardedFrames: 0, authorizedUsage: 3600000ms/h ETSI EN 300 220-1, remainingUsage: 3599906ms,

Frequency: 868950Khz Adapted to Visonic/Meian devices,

Selectivity: 0 Default value, FloorNoise: -108dBm Very small noise, DspTrigger: 6dBm, RFlink: 1 Enabled, RFlinkTrigger: 10dBm, sentFrames: 0, discardedFrames: 0, authorizedUsage: 3600ms/h ETSI EN 300 220-1, remainingUsage: 3600ms,

The line “ZIA++1234 STATUS RADIO XML” will return :

```
ZIA--<?xml version="1.0" encoding="ISO8859-1" ?><radioStatus><reqNum>1234</reqNum><band><i> <n>Frequency</n> <v>433920</v> <unit>Khz</unit>
<c> Adapted to most 433Mhz devices</c></i><i> <n>Selectivity</n> <v>0</v> <unit></unit> <c> Default value</c></i><i> <n>FloorNoise</n> <v>-97</v>
<unit>dBm</unit> <c> A bit noisy</c></i><i> <n>DspTrigger</n> <v>8</v> <unit>dBm</unit> <c></c></i><i> <n>RFlink</n> <v>1</v> <unit></unit> <c>
Enabled</c></i><i> <n>RFlinkTrigger</n> <v>12</v> <unit>dBm</unit> <c></c></i><i> <n>sentFrames</n> <v>272</v> <unit></unit> <c></c></i><i>
<n>discardedFrames</n> <v>0</v> <unit></unit> <c></c></i><i> <n>authorizedUsage</n> <v>3600000</v> <unit>ms/h</unit> <c> ETSI EN 300 220-
1</c></i><i> <n>remainingUsage</n> <v>3599906</v> <unit>ms</unit> <c></c></i></band><band><i> <n>Frequency</n> <v>868950</v> <unit>Khz</unit>
<c> Adapted to Visonic/Meian devices</c></i><i> <n>Selectivity</n> <v>0</v> <unit></unit> <c> Default value</c></i><i> <n>FloorNoise</n> <v>-108</v>
<unit>dBm</unit> <c> Very small noise</c></i><i> <n>DspTrigger</n> <v>6</v> <unit>dBm</unit> <c></c></i><i> <n>RFlink</n> <v>1</v> <unit></unit> <c>
Enabled</c></i><i> <n>RFlinkTrigger</n> <v>10</v> <unit>dBm</unit> <c></c></i><i> <n>sentFrames</n> <v>0</v> <unit></unit> <c></c></i><i>
<n>discardedFrames</n> <v>0</v> <unit></unit> <c></c></i><i> <n>authorizedUsage</n> <v>3600</v> <unit>ms/h</unit> <c> ETSI EN 300 220-1</c></i><i>
<n>remainingUsage</n> <v>3600</v> <unit>ms</unit> <c></c></i></band></radioStatus>
```

The line “ZIA++1234 STATUS RADIO JSON” will return :

```
ZIA--{"radioStatus": {"reqNum": "1234", "band": [{"i": [{"n": "Frequency", "v": "433920", "unit": "Khz", "c": " Adapted to most 433Mhz devices"}, {"n":
"Selectivity", "v": "0", "unit": "", "c": " Default value"}, {"n": "FloorNoise", "v": "-97", "unit": "dBm", "c": " A bit noisy"}, {"n": "DspTrigger", "v": "8", "unit":
"dBm", "c": ""}, {"n": "RFlink", "v": "1", "unit": "", "c": " Enabled"}, {"n": "RFlinkTrigger", "v": "12", "unit": "dBm", "c": ""}, {"n": "sentFrames", "v": "323",
"unit": "", "c": ""}, {"n": "discardedFrames", "v": "0", "unit": "", "c": ""}, {"n": "authorizedUsage", "v": "3600000", "unit": "ms/h", "c": " ETSI EN 300 220-
1"}, {"n": "remainingUsage", "v": "3599737", "unit": "ms", "c": ""}], [{"i": [{"n": "Frequency", "v": "868950", "unit": "Khz", "c": " Adapted to Visonic/Meian
devices"}, {"n": "Selectivity", "v": "0", "unit": "", "c": " Default value"}, {"n": "FloorNoise", "v": "-108", "unit": "dBm", "c": " Very small noise"}, {"n":
"DspTrigger", "v": "6", "unit": "dBm", "c": ""}, {"n": "RFlink", "v": "1", "unit": "", "c": " Enabled"}, {"n": "RFlinkTrigger", "v": "10", "unit": "dBm", "c": ""},
{"n": "sentFrames", "v": "1", "unit": "", "c": ""}, {"n": "discardedFrames", "v": "0", "unit": "", "c": ""}, {"n": "authorizedUsage", "v": "3600", "unit": "ms/h",
"c": " ETSI EN 300 220-1"}, {"n": "remainingUsage", "v": "3600", "unit": "ms", "c": ""}]}]}
```

5.1.1.1.3 FORMAT answer examples

FORMAT TEXT

ZIA44 FRAME: frameType: 0, dataFlag: 1,

rfLevel: -49dBm, floorNoise: -108dBm, rfQuality: 10

protocol: 2 (VISONIC), infoType: 2, frequency: 868950Khz

subType: 0 (Detector/Sensor), id: 1166992416, qualifier: 1 (Tamper)

ZIA44 FRAME: frameType: 0, dataFlag: 0,

rfLevel: -34dBm, floorNoise: -97dBm, rfQuality: 10

protocol: 4 (CHACON), infoType: 1, frequency: 433920Khz

subType: 1, id: 146139014 (ON)

ZIA44 FRAME: frameType: 0, dataFlag: 0,

rfLevel: -58dBm, floorNoise: -97dBm, rfQuality: 7

protocol: 1 (X10), infoType: 0, frequency: 433920Khz

subType: 1, id: 33 (ON, C2)

ZIA44 FRAME: frameType: 0, dataFlag: 0,

rfLevel: -78dBm, floorNoise: -97dBm, rfQuality: 3

protocol: 5 (OREGON), infoType: 4, frequency: 433920Khz

subType: 0, id_PHY: 0x1A2D (THGR122/228/238/268,THGN122/123/132)

adr_channel: 54273, adr: 212, channel: 1

qualifier: 32, lowBatt: 0, measures:

temperature: +23.4 Celsius

hygrometry: 75 %

ZIA44 FRAME: frameType: 0, dataFlag: 0,

rfLevel: -45dBm, floorNoise: -98dBm, rfQuality: 10

protocol: 7 (OWL), infoType: 8, frequency: 433920Khz

subType: 0, id_PHY: 0x0000 (CM119/160)

adr_channel: 62851, adr: 3928, channel: 3

qualifier: 4, lowBatt: 0, measures:

energy: 0 Wh

power: 0 W

ZIA44 FRAME: frameType: 0, dataFlag: 0,

rfLevel: -64dBm, floorNoise: -98dBm, rfQuality: 6

protocol: 5 (OREGON), infoType: 6, frequency: 433920Khz

subType: 0, id_PHY: 0x1A89 (WGR800)

adr_channel: 40192, adr: 157, channel: 0

qualifier: 48, lowBatt: 0, measures:

wind speed: 0.5 m/s

direction: 225 degree

FORMAT XML

```
ZIA22<?xml version="1.0" encoding="ISO8859-1" ?><frame><header><frameType>0</frameType> <dataFlag>0</dataFlag> <rfLevel>-73</rfLevel><dBm
<floorNoise>-98</floorNoise><dBm <rfQuality>5</rfQuality>/10 <protocol>5</protocol> <protocolMeaning>OREGON</protocolMeaning>
<infoType>4</infoType> <frequency>433920</frequency>Khz</header><infos><subType>0</subType> <id_PHY>0xFA28</id_PHY>
<id_PHYMeaning>THGR810</id_PHYMeaning> <adr_channel>59650</adr_channel> <adr>233</adr> <channel>2</channel> <qualifier>48</qualifier>
```

```
<lowBatt>0</lowBatt></infos> <measures> <type>temperature</type> <value>+23.1</value> <unit>Celsius</unit> </measures> <measures>
<type>hygrometry</type> <value>81</value> <unit>%</unit> </measures></frame>
```

```
ZIA22<?xml version="1.0" encoding="ISO8859-1" ?><frame><header><frameType>0</frameType> <dataFlag>0</dataFlag> <rfLevel>-41</rfLevel>dBm
<floorNoise>97</floorNoise>dBm <rfQuality>10</rfQuality>/10 <protocol>3</protocol> <protocolMeaning>BLYSS</protocolMeaning>
<infoType>1</infoType> <frequency>433920</frequency>Khz</header><infos><subType>1</subType> <id>4261483730</id>
<subTypeMeaning>ON</subTypeMeaning> </infos></frame>
```

```
ZIA22<?xml version="1.0" encoding="ISO8859-1" ?><frame><header><frameType>0</frameType> <dataFlag>0</dataFlag> <rfLevel>-64</rfLevel>dBm
<floorNoise>98</floorNoise>dBm <rfQuality>6</rfQuality>/10 <protocol>5</protocol> <protocolMeaning>OREGON</protocolMeaning>
<infoType>6</infoType> <frequency>433920</frequency>Khz</header><infos><subType>0</subType> <id_PHY>0x1A89</id_PHY>
<id_PHYMeaning>WGR800</id_PHYMeaning> <adr_channel>40192</adr_channel> <adr>157</adr> <channel>0</channel> <qualifier>48</qualifier>
<lowBatt>0</lowBatt></infos> <measures> <type>wind speed</type> <value>0.0</value> <unit>m/s</unit> </measures> <measures>
<type>direction</type> <value>225</value> <unit>degree</unit> </measures></frame>
```

```
ZIA22<?xml version="1.0" encoding="ISO8859-1" ?><frame><header><frameType>0</frameType> <dataFlag>0</dataFlag> <rfLevel>-56</rfLevel>dBm
<floorNoise>93</floorNoise>dBm <rfQuality>7</rfQuality>/10 <protocol>9</protocol> <protocolMeaning>RTS</protocolMeaning> <infoType>3</infoType>
<frequency>433920</frequency>Khz</header><infos><subType>0</subType> <subTypeMeaning>Shutter</subTypeMeaning> <id>6793524</id>
<qualifier>7</qualifier><qualifierMeaning>Up/On</qualifierMeaning></infos></frame>
```

```
ZIA22<?xml version="1.0" encoding="ISO8859-1" ?><frame><header><frameType>0</frameType> <dataFlag>0</dataFlag> <rfLevel>-43</rfLevel>dBm
<floorNoise>98</floorNoise>dBm <rfQuality>10</rfQuality>/10 <protocol>2</protocol> <protocolMeaning>VISONIC</protocolMeaning>
<infoType>2</infoType> <frequency>433920</frequency>Khz</header><infos><subType>0</subType>
<subTypeMeaning>Detector/Sensor</subTypeMeaning> <id>614725408</id>
<qualifier>0</qualifier><qualifierMeaning></qualifierMeaning></infos></frame>
```

```
ZIA22<?xml version="1.0" encoding="ISO8859-1" ?><frame><header><frameType>0</frameType> <dataFlag>0</dataFlag> <rfLevel>-76</rfLevel>dBm
<floorNoise>96</floorNoise>dBm <rfQuality>4</rfQuality>/10 <protocol>5</protocol> <protocolMeaning>OREGON</protocolMeaning>
<infoType>4</infoType> <frequency>433920</frequency>Khz</header><infos><subType>0</subType> <id_PHY>0x1A2D</id_PHY>
<id_PHYMeaning>THGR122/228/238/268,THGN122/123/132</id_PHYMeaning> <adr_channel>54273</adr_channel> <adr>212</adr> <channel>1</channel>
<qualifier>32</qualifier><lowBatt>0</lowBatt></infos> <measures> <type>temperature</type> <value>+23.4</value> <unit>Celsius</unit> </measures>
<measures> <type>hygrometry</type> <value>75</value> <unit>%</unit> </measures></frame>
```

FORMAT JSON

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "0", "rfLevel": "-71", "floorNoise": "-98", "rfQuality": "5", "protocol": "5", "protocolMeaning":
"OREGON", "infoType": "9", "frequency": "433920"}, "infos": {"subType": "0", "id_PHY": "0x2A19", "id_PHYMeaning": "PCR800", "adr_channel": "39168",
"adr": "153", "channel": "0", "qualifier": "48", "lowBatt": "0", "measures": [{"type": "total rain", "value": "1040.1", "unit": "mm"}, {"type": "current rain",
"value": "0.00", "unit": "mm/h"}]}}
```

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "0", "rfLevel": "-41", "floorNoise": "-97", "rfQuality": "10", "protocol": "3", "protocolMeaning":
"BLYSS", "infoType": "1", "frequency": "433920"}, "infos": {"subType": "0", "id": "4261483730", "subTypeMeaning": "OFF"}}}
```

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "0", "rfLevel": "-64", "floorNoise": "-97", "rfQuality": "6", "protocol": "5", "protocolMeaning":
"OREGON", "infoType": "6", "frequency": "433920"}, "infos": {"subType": "0", "id_PHY": "0x1A89", "id_PHYMeaning": "WGR800", "adr_channel": "40192",
"adr": "157", "channel": "0", "qualifier": "48", "lowBatt": "0", "measures": [{"type": "wind speed", "value": "0.4", "unit": "m/s"}, {"type": "direction",
"value": "225", "unit": "degree"}]}}
```

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "1", "rfLevel": "-50", "floorNoise": "-107", "rfQuality": "10", "protocol": "2", "protocolMeaning":
"VISONIC", "infoType": "2", "frequency": "868950"}, "infos": {"subType": "0", "subTypeMeaning": "Detector/Sensor", "id": "1166992416", "qualifier": "8",
"qualifierMeaning": { "flags": ["Supervisor/Alive"]}}}}
```

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "1", "rfLevel": "-52", "floorNoise": "-107", "rfQuality": "10", "protocol": "2", "protocolMeaning":
"VISONIC", "infoType": "2", "frequency": "868950"}, "infos": {"subType": "0", "subTypeMeaning": "Detector/Sensor", "id": "1166992416", "qualifier": "1",
"qualifierMeaning": { "flags": ["Tamper"]}}}}
```

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "0", "rfLevel": "-41", "floorNoise": "-97", "rfQuality": "10", "protocol": "4", "protocolMeaning":
"CHACON", "infoType": "1", "frequency": "433920"}, "infos": {"subType": "1", "id": "146139014", "subTypeMeaning": "ON"}}}
```

```
ZIA33{ "frame" :{"header": {"frameType": "0", "dataFlag": "0", "rfLevel": "-41", "floorNoise": "-97", "rfQuality": "10", "protocol": "7", "protocolMeaning":
"OWL", "infoType": "8", "frequency": "433920"}, "infos": {"subType": "0", "id_PHY": "0x0003", "id_PHYMeaning": "CM180", "adr_channel": "784", "adr": "49",
"channel": "0", "qualifier": "6", "lowBatt": "0", "measures": [{"type": "energy", "value": "45150", "unit": "Wh"}, {"type": "power", "value": "345", "unit":
"W"}, {"type": "P1", "value": "345", "unit": "W"}, {"type": "P2", "value": "0", "unit": "W"}, {"type": "P3", "value": "0", "unit": "W"}]}}
```

5.1.2 Binary data (ordinary used to sent RF frames and by RFLINK interface)

5.1.2.1 Received from USB port and send to the RF transmitter

<i>BinaryData []</i>	<i>Size</i>	<i>type</i>	<i>Remark</i>
FrameType	1	unsigned char	Type of packet. Value = 0
Cluster	1	unsigned char	Set always 0. Future use.
Protocol	1	unsigned char	Define the protocol. See table
action	1	unsigned char	Define operation. See table
DeviceID	4	unsigned long	LSB first DeviceID : 0...255. House Code (A-P) must be on deviceID[7...4] for pseudo X10 address format 32 bits form is only used for KD101 protocol.
DimValue	1	unsigned char	0% ... 100%
Burst	1	unsigned char	Set 0 by default.
Qualifier	1	unsigned char	Set 0 by default.
Reserved2	1	unsigned char	Set 0 by default

Home Automation Protocols, excepted X10 and DOMIA LITE, use large MacDeviceIDs to distinguish separate installations (and avoid to command easily appliances of your neighbors!). Thus, an internal USB Dongle MAC 32 bits address is used to generate the MacDeviceIDs sent over the RF. The API requests only a small 8 bits DeviceID with the internal calculation $MacDeviceID = f(DeviceID)$.

The internal USB Dongle MAC 32 bits address is preset at factory with a random value. It can be read or set to another value with a management command.

5.1.2.2 Protocol

Protocol	Value	Remark
VISONIC_433	1	PowerCode only
VISONIC_868	2	PowerCode only
CHACON_433	3	
DOMIA_433	4	
X10_433	5	
X2D_433	6	Alarm
X2D_868	7	Alarm
X2D_SHUTTER_868	8	Qualifier7:6 =1 to emit X2D variant, else Qualifier=0
X2D_HA_ELEC_868	9	Qualifier7:6 =1 to emit X2D variant, else 0
X2D_HA_GAS_868	10	Qualifier7:6 =1 to emit X2D variant, else 0
SOMFY_RTS_433	11	Qualifier D0=1 to emit portal frame, else Qualifier D0=0 (or nothing) DIM values between 0—15 emulate specific RTS functions. Specify DIM %4 to emulate RTS "My" function. Qualifier D1=1 can emulate the same specific RTS functions

		but specified on D7:4 of the qualifier. Specify D7:4 to 4 to emulate RTS "My" function.
BLYSS_433	12	
PARROT_433_OR_868	13	ID0 to ID254 (A1...P15). ID255 (P16) is reserved and must not be used.
FS20	14	ID=D0...D7 defines accesses by pseudo-address X10 A1-A16...P1-P16. D8...D31 must 0. ID=D8...D31 defines accesses by 24 bits ID. The form is the same as ID at the receiving side. Qualifier defines extension byte.
KD101_433	16	Smoke detector. 32 bits ID
FS20	22	
EDISIO	23	

5.1.2.3 Action

action	Value	Remark
OFF	0	Used by most protocols
ON	1	Used by most protocols
DIM	2	Used by some protocols
BRIGHT	3	<i>not used</i>
ALL_OFF	4	<i>Used by BLYSS</i>
ALL_ON	5	<i>Used by BLYSS</i>
ASSOC	6	Used by most protocols
DISSOC	7	<i>provision</i>
ASSOC_OFF	8	Used by PARROT for its OFF entries
DISSOC_OFF	9	<i>provision</i>
TOGGLE	10	<i>Used by Edisio</i>

5.1.2.4 RFLINK interface (USB to RF)

See Chapter *RFLINK interface (RF to USB)* in the opposite direction.

Packet structure is the same.

5.2 USB Dongle → Host (receiving data from RF)

5.2.1 ASCII data (ordinary used by commands answers and asynchronous received RF Frames)

Header	Remark
ZIA--	Synchronous answers of the commands interpreter
ZIA00	Asynchronous received RF Frames. Enabled by "FORMAT HEXA"
ZIA11	Asynchronous received RF Frames. Enabled by "FORMAT HEXA FIXED"
ZIA22	Asynchronous received RF Frames. Enabled by "FORMAT XML"
ZIA33	Asynchronous received RF Frames. Enabled by "FORMAT JSON"
ZIA44	Asynchronous received RF Frames. Set by "FORMAT TEXT"
ZIA55	Asynchronous events log on specified functions. Set by "TRACE" debug command.
ZIA66	Asynchronous received RF Edisio Frames. Set by "EDISIOFRAME" even without argument

5.2.2 Binary data (ordinary used to received RF Frames and by RFLINK interface)

5.2.2.1 Received RF Frames from regular decoder

Binary Data[]	Size	Type	Remark
FrameType	1	unsigned char	0: received RF Frames from regular decoder
Cluster	1	unsigned char	Reserved.
DataFlag	1	unsigned char	0: 433Mhz, 1: 868Mhz
RFLevel	1	signed char	Unit : dB (high signal :-40dB to low : -110dB)
FloorNoise	1	signed char	Unit : dB (high signal :-40dB to low : -110dB)
RFQuality	1	unsigned char	RF signal quality : 1/10 (poor) to 10/10 (best)
Protocol	1	unsigned char	See below. Not significant with RFLINK frame
InfosType	1	unsigned char	See below. Not significant with RFLINK frame
Infos[0...9]	20	Signed or unsigned short upon context	LSB first. Define provided data by the device Not significant with RFLINK frame.

NOTE : Binary received RF Frames flow is enabled by "FORMAT BINARY" command.

NOTE : Binary format is an alternate way to receive RF Frames. Consider FORMAT HEXA, HEXA FIXED, XML and JSON with ASCII form to develop/debug more easily a Dongle driver.

5.2.2.2 RFLINK interface (RF to USB)

The Dongle is able to provide not decoded frames records to the Host system. The Dongle assumes entirely the real time frame recording and the host (e.g Linux system) can analyze these records without real time constraints .

Visit <http://www.nemcon.nl/blog2/> for further information about RFLINK. Visit <https://github.com/> to get sources.

Binary Data[]	Size	Type	Remark
FrameType	1	unsigned char	1: RFLINK Frame
frequency	4	unsigned long	Frequency expressed in Khz Available : 433420, 433920, 868350, 868950.
RFLevel	1	signed char	Unit : dB (high signal :-40dB to low : -110dB)
FloorNoise	1	signed char	Unit : dB (high signal :-40dB to low : -110dB)
PulseElementSize	1	unsigned char	Value : 1
number	2	unsigned short	Number of Pulses upon RFLINK definition
Repeats	1	unsigned char	Number of re-transmits upon RFLINK definition
Delay	1	unsigned char	Delay in ms. after trans. upon RFLINK definition
Multiply	1	unsigned char	Real pulse unit in microseconds upon RFLINK definition Value = 40
Time	4	unsigned long	Timestamp indicating when the signal was received upon RFLINK definition
Pulses[]	number+2	unsigned char[]	Pulses[0] and Pulses[number+1] are set to 0 upon historical RFLINK definition

NOTE : Binary RFLINK Frames flow is enabled by “FORMAT RFLINK BINARY” command.

5.2.2.3 DataFlag

bit	7	6	5	4	3	2	1	0
	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	<i>reserved</i>	868Mhz Flag 0: 433 1: 868

5.2.2.4 RFLevel / FloorNoise

They are expressed in dB. One RFLevel above -70dB (-70...-40) has to be considered high.

One RFLevel under -70dB (-110dB... -70) has to be considered low. Difference between RFLevel and FloorNoise is representative of the quality of the signal (1/10 to 10/10).

The FloorNoise parameter is classically under -90dB. 868Mhz is often less noisy than 433hz because it is less used. The Floor Noise must be keep as low as possible. Avoid to set the dongle near electrical devices.

5.2.2.5 Protocol

Protocol	Value	Remark
X10	1	Use InfoType 0, InfoType 1
VISONIC	2	Use InfoType 2
BLYSS	3	Use InfoType 1
CHACON	4	Use InfoType 1
OREGON	5	Use InfoType 4, 5, 6, 7, 9
DOMIA	6	Use InfoType 0
OWL	7	Use InfoType 8
X2D	8	Use InfoType 10, 11
RTS	9	Use InfoType 3
KD101	10	Use InfoType 1
PARROT	11	Use InfoType 0
TIC	13	Use InfoType 13
FS20	14	Use InfoType 1, 14
JAMMING	15	Use InfoType 1
EDISIO	16	Use InfoType 15
<i>reserved</i>	16-255	-

5.2.2.6 InfoType

InfoType value	Infos[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
0	subType	id	-	-	-	-	-	-	-	-
1	subType	id_lsb	id_msb							
2, 14	subType	id_lsb	id_msb	qualifier	-	-	-	-	-	-
3	subType	id_lsb	id_msb	qualifier	-	-	-	-	-	-
4	subType	id_PHY	adr_channel	qualifier	temp	hygro	-	-	-	-
5	subType	id_PHY	adr_channel	qualifier	temp	hygro	pressure	-	-	-
6	subType	id_PHY	adr_channel	qualifier	speed	direction	-	-	-	-
7	subType	id_PHY	adr_channel	qualifier	UV	-	-	-	-	-
8	subType	id_PHY	adr_channel	qualifier	Energy_lsb	Energy_msb	power	P1	P2	P3
9	subType	id_lsb	id_lsb	qualifier	TotalRain_lsb	TotalRain_msb	rain			
10	subType	id_lsb	id_lsb	qualifier	Mode	0	Data 0-1	Data 2-3	Data 4-5	Data 6-7
11	subType	id_lsb	id_lsb	qualifier	0	0	Data 0-1	Data 2-3	Data 4-5	Data 6-7
12	-	-	-	-	-	-	-	-	-	-
13	subType	id_lsb	id_lsb	qualifier	infos	Cnt1Lsb	Cnt1Msb	Cnt2Lsb	Cnt2Msb	power
15	subType	id_lsb	id_lsb	qualifier	infos	[add1]	[add2]			

5.2.2.7 *InfoType 0*

InfoType 0	Used by X10 / DOMIA LITE protocol / PARROT
subType	0: OFF, 1: ON, 2: BRIGHT, 3: DIM, 4: ALL_OFF, 5 : ALL_ON
id	Id of the device in X10 format, housecode (A...P) =D[7:4], dev (1...16) =D[3:0]

5.2.2.8 *InfoType 1*

InfoType 1	Used by X10 (24/32 bits ID), CHACON , KD101, BLYSS, FS20, JAMMING
subType	0: OFF, 1: ON/Alert ; X10 & CHACON add 4: ALL_OFF, 5 : ALL_ON
id_lsb	Unsigned short. Lsb of the device ID FS20 : address (8bits) on D8...15 JAMMING : id_lsb = 0
id_msb	Unsigned short. Msb of the device ID FS20 : Housecode (16bits) JAMMING : id_msb = 0

5.2.2.9 *InfoType 2*

InfoType 2	Used by VISONIC
subType	VISONIC 0: detector/sensor(PowerCode device), 1: remote control (CodeSecure device)
id_lsb	Unsigned short. Lsb of the device ID
id_msb	Unsigned short. Msb of the device ID
qualifier	Visonic : detector/sensor/ PowerCode device : D0 : Tamper Flag, D1: Alarm Flag, D2: Low Batt Flag, D3: Supervisor Frame Flag (Nota : D0:3 of the id always set to 0 so that id and qualifier can be ORed) remote control device (MCT-234 style) : Key Id (4 buttons) values : 0x08, 0x10, 0x20, 0x40. (Nota : D0:7 of the id always set to 0) so that id and qualifier can be ORed.

5.2.2.10 *InfoType 3*

InfoType 3	Used by RTS protocol
subType	0: shutter device, 1: Portal device
id_lsb	Unsigned short. Lsb of the device ID
id_msb	Unsigned short. Msb of the device ID.
qualifier	Shutters Remote control) : D0:4 : code function 1 : Down /OFF 4 : My 7 : Up / ON 13 : ASSOC portals Remote control : D0:4 : code function 5 : Left button 6 : Right button

5.2.2.11 InfoType 4

InfoType 4	Used by Scientific Oregon protocol (thermo/hygro sensors)
subType	0: regular sensor frame
id_PHY	Unsigned short. Define the recognized physical device. See Table.
adr_channel	adr_channel[15:8] defines a random device address chosen after reset or battery change. Application can take or not into account this field. adr_channel[7:0] defines the channel programmed on device. : 1..N
qualifier	D7:4 = 1 : Oregon protocol V1, 2 : Oregon V2, 3 : Oregon V3 D0 = 0: Batt Ok, D0=1 : Low Batt (less than 20% battery remaining for some devices)
temp	Signed short. Temperature (Unit : 1/10 of degrees Celsius, e.g. 213 means 21.3°C)
hygro	Hygrometry (0...100%). 0 means hygrometry not available on device

5.2.2.12 InfoType 5

InfoType 5	Used by Scientific Oregon protocol (Atmospheric pressure sensors)
subType	0: regular sensor frame
id_PHY	Unsigned short. Define the recognized physical device. See Table.
adr_channel	iadr_channel[15:8] defines a random device address chosen after reset or battery change. Application can take or not into account this field. adr_channel[7:0] defines the channel programmed on device.
qualifier	D7:4 = 1 : Oregon protocol V1, 2 : Oregon V2, 3 : Oregon V3 D0 = 0: Batt Ok, D0=1 : Low Batt (less than 20% battery remaining for some devices)
temp	Signed short. Temperature (Unit : 1/10 of degrees Celsius, e.g. 213 means 21.3°C)
hygro	Hygrometry (0...100%).
pressure	Atmospheric/barometric pressure. (Unit : hPa)

5.2.2.13 InfoType 6

InfoType 6	Used by Scientific Oregon protocol (Wind sensors)
subType	0: regular sensor frame
id_PHY	Unsigned short. Define the recognized physical device. See Table.
adr_channel	adr_channel[15:8] defines a random device address chosen after reset or battery change. Application can take or not into account this field. adr_channel[7:0] defines the channel programmed on device.
qualifier	D7:4 = 1 : Oregon protocol V1, 2 : Oregon V2, 3 : Oregon V3 D0 = 0: Batt Ok, D0=1 : Low Batt (less than 20% battery remaining for some devices)
speed	Averaged Wind speed (Unit : 1/10 m/s, e.g. 213 means 21.3m/s)
direction	Wind direction 0...359° (Unit : degrees)

5.2.2.14 InfoType 7

InfoType 7	Used by Scientific Oregon protocol (UV sensors)
subType	0: regular sensor frame
id_PHY	Unsigned short. Define the recognized physical device. See Table.
adr_channel	adr_channel[15:8] defines a random device address chosen after reset or battery change. Application can take or not into account this field. adr_channel[7:0] defines the channel programmed on device.
qualifier	D7:4 = 1 : Oregon protocol V1, 2 : Oregon V2, 3 : Oregon V3 D0 = 0: Batt Ok, D0=1 : Low Batt (less than 20% battery remaining for some devices)
UV	UV index (Unit : 1/10 index)

5.2.2.15 InfoType 8

InfoType 8	Used by OWL (Energy/power sensors)
subType	0: regular sensor frame
id_PHY	Define the recognized physical device. CM119/CM160 = 0, CM130 = 1, CM180=2, CM180i=3
adr_channel	adr_channel[15:4] defines a random device address chosen after reset or battery change for discriminate devices on a same channel. Application can take into account or ignore this field. adr_channel[3:0] defines the channel programmed by mini-switches on the device.
qualifier	D0 = 0: Batt Ok, D0=1 : Low Batt D1=0 : Only the total instantaneous Power is given. D1=1 : Power on each input 1, 2, 3 are added (CM180i only).
Energy_lsb	Define the measured energy since the RESET of the device (32 bits value). Unit : Wh
Energy_msb	
Power	Define the total instantaneous measured power. Unit : W. P=UI. U=230V
P1	Define the instantaneous measured power on I1 input. Unit : W P=UI. U=230V
P2	Define the instantaneous measured power on I2 input. Unit : W. P=UI. U=230V
P3	Define the instantaneous measured power on I3 input. Unit : W. P=UI. U=230V

5.2.2.16 InfoType 9

InfoType 9	Used by Scientific Oregon protocol (Rain sensors)
subType	0: regular sensor frame
id_PHY	Unsigned short. Define the recognized physical device. See Table.
id_channel	id_channel[15:8] defines a random device address chosen after reset or battery change. Application can take or not into account this field. id_channel[7:0] defines the channel programmed on device.
qualifier	D7:4 = 1 : Oregon protocol V1, 2 : Oregon V2, 3 : Oregon V3 D0 = 0: Batt Ok, D0=1 : Low Batt (less than 20% battery remaining for some devices)
TotalRain_lsb	Define the rain measured since the RESET of the device (lsb value). Unit : 0.1mm
TotalRain_msb	Define the rain measured since the RESET of the device (msb value). Unit : 0.1 mm
Rain	Define the instantaneous measured rain. Unit : 0.01 mm/h

5.2.2.17 InfoType 10

InfoType 10	Used by Thermostats X2D protocol
subType	0: GENERIC 1: RADIO TYBOX 2: TYBOX BUS 3: PACK LABEL 4: DELTA 200 5: DRIVER RF 6: STARBOX F03 7: OTHER 8: REC BIDIR
id_lsb	Unsigned short. Lsb of the device ID D3:0 : Area
id_msb	Unsigned short. Msb of the device ID.
qualifier	D0 : Tamper Flag, D1: anomaly device, D2: Low Batt Flag, D3: -, D4: Test/Assoc D5 : Domestic frame D7:6 : X2D variant
Function	D8:0 Values : 0 : SPECIAL 1 : HEATING_SPEED 2 : OPERATING_MODE 12 : REGULATION 26 : THERMIC_AREA_STATE
Mode/State	Case with with Function = HEATING_SPEED or Function = REGULATION or Function = THERMIC_AREA_STATE :

	<p>D7:0 State :</p> <p>0: OFF</p> <p>1: ON</p> <p>Case with Function = OPERATING_MODE</p> <p>D7:0 State :</p> <p>0: ECO</p> <p>1: MODERAT</p> <p>2: MEDIO</p> <p>3: COMFORT</p> <p>4: STOP</p> <p>5: OUT OF FROST</p> <p>6: SPECIAL</p> <p>7: AUTO</p> <p>8: CENTRALISED</p>
--	--

5.2.2.18 InfoType 11

InfoType 11	Used by Alarm X2D protocol / Shutter
subType	0: detector/sensor device, 1: remote control device / shutter
id_lsb	Unsigned short. Lsb of the device ID D12:8: sensor type, D7:0 buttons expected for remote control devices
id_msb	Unsigned short. Msb of the device ID.
qualifier	<p>SubType=0</p> <p>D0 : Tamper Flag, D1: Alarm Flag, D2: Low Batt Flag, D3: - , D4: Test/Assoc</p> <p>D5 : Domestic frame</p> <p>D7:6 : X2D variant</p> <p>SubType=1</p> <p>On : 1</p> <p>Off : 2</p> <p>Stop : 3</p>

5.2.2.19 InfoType 12

Deprecated

5.2.2.20 InfosType 13

InfosType 13	Used by Cartelectronic TIC / Pulses (Linky/Teleinfo historic data or pulses counting) https://www.carlectronic.fr/home/124-tic-pulses-868mhz.html Contact carlectronic.fr for more precise specifications.
subType	D0: 0 : Teleinfo mode, 1 : Encoder mode, 2 : Linky mode.
id_lsb	Unsigned short. Lsb of the device ID ,D0:15 <i>Linky mode :</i> Device ID (id_lsb+ id_msb) = 'Counter Id' field of Linky mode : Bit 0-19 Sub-id of the counter Bit 20-22 Type of the Linky counter Bit 23-27 year of the counter Bit 28-31 manufacturer code
id_msb	Unsigned short. Msb of the device ID, D15:31
qualifier	<i>Teleinfo mode:</i> D0:7 : (states field) D0 : battery (1 : low) D1 : PAPP (Apparent power) (1: Valid) D2 : TeleInfo (1: Not present) D3:4 PEJP or DEMAINE 0 : no change price time warning 1 : white 2 : blue 3 : Red / PEJP D8:15 : Msb (D32-39) of device ID (D0:39 defines "ADCO" parameter) <i>Encoder mode :</i> D0: Low Batt Flag (1 : low) <i>Linky modes:</i> D0:7 : (states field) D0 : battery (1 : low) D1 : PAPP (Apparent power) (1: Valid) D2 : TeleInfo (1: Not present) D3:4 : PEJP or color price forecast for tomorrow (DEMAIN) (See 'Tab') D:5:6 : color price for today (See 'Tab') 'Tab' : 0 : no change price time warning 1 : white 2 : blue 3 : Red / PEJP
infos	<i>Teleinfo mode:</i> D0:7 Contract_type / Current Price Time

	<i>Linky mode</i> D0:15 Current Index/Spare D0:3 : spare D4:7 : current index D8:15 : Average voltage
Cnt1_lsb	<i>Teleinfo</i> : 32 bits counter 1 (lsb) <i>Telecounters</i> : 32 bits pulse counter 1 (lsb) Linky : 32 bits Running Index (lsb)
Cnt1_msb	<i>Teleinfo</i> : 32 bits counter 1 (msb) <i>Telecounters</i> : 32 bits pulse counter 1 (msb) Linky : 32 bits Running Index (msb)
Cnt2_lsb	<i>Teleinfo</i> : 32 bits counter 2 (lsb) <i>Telecounters</i> : 32 bits pulse counter 2 (lsb) Linky : 32 bits production Index (lsb)
Cnt2_msb	<i>Teleinfo</i> : 32 bits counter 2 (msb) <i>Telecounters</i> : 32 bits pulse counter 2 (msb) Linky : 32 bits production Index (msb)
Apparent_power	<i>Teleinfo</i> : D0:15 (unit :Watt)

Some infos:

TeleInfos :

counter 1 : (Wh) : Base / Hc: "Heures creuses" / EJPPM

counter 2 : (Wh) : Hp : "Heures pleines" / EJPHn

Linky :

Running Index (Unit: Wh)

The Running Index is the consumption measure in use. For information there can be up to 10 counters on a Linky device. Each meter corresponds to a tariff period. The tariff period corresponds to the Current Index information (0 to 9 counter index). The Linky "rate period" field does exist, but, it is a text field, and it may be different from one vendor / constructor to another.

Production Index (Unit: Wh)

A Linky device can also handle the solar production aspect. In this case, the consumption counter still exists, and the second 32-bit counter is used for the power generation aspect.

The customer can thus know the consumption and production indexes (if existing) with his home automation system, and be able to use the meters to know the consumption / production hourly, daily, etc., if the home automation system proposes. With this knowledge, he will be able to master these consumptions, his production of energy in the day, the days of sun, cloudy...

Telecounters/Encoder :

32 bits pulse counters

Regarding the counters of the Encoder message, they are in pulses, the number of pulses since the start of the counter. It is thus necessary to take this value, and to multiply by the correspondence of an impulse, to obtain the index of consumption / use / production:

Example:

1 pulse / 0.25l for water meters: counter = 24532 => 6133 liters => 6,133 m3 (absolute meter)

The same applies to electrical energy.

5.2.2.21 *InfoType 14*

InfoType 14	Used by FS20
subType	<p>Read : http://fhz4linux.info/tiki-index.php?page=FS20%20Protocol</p> <p>Simple Commands ON/OFF [D0...7: 0: OFF ; D0...7: 0x11 :ON] are mapped to InfoType1 to easily link Home automation boxes.</p> <p><i>command field</i></p> <p>D0:4 : "befehl" tab</p> <p>D5 : 1: extension field exists. 0 : regular.</p> <p>D6 : 1 : bidirectional command. 0 : regular.</p> <p>D7 : 1 : Answer of a receiver..0 : regular.</p>
id_lsb	D8...15 : address (8bits)
id_msb	D0...15 : Housecode (16bits)
qualifier	Optional « Erweiterung » Field (= Extension field)

5.2.2.22 *InfoType 15*

Note : Consider EDISIOFRAME action to handle EDISIO protocol (case of genuine Edisio Dongle emulation)

InfoType 15	Used by Edisio
subType	D0...D7 : Command field (CMD) See table Below
id_lsb	Address LSB
id_msb	Address MSB
qualifier	CID : D0...D7 : Channel Identifier (button)
infos	MID/BL MID : D0...D7 : Model Field See Table below BL : D8...D15 : Battery Level (unit : 1/10 V)
[ADD0]	Additional data byte 0, byte 1
[ADD1]	Additional data byte 2, byte 3

Edisio Command Field

CMD - Command	Value hex	Additional data length	Description
NULL	0x00	0	Null instruction
ON	0x01	0	Channel Switched ON
OFF	0x02	0	Channel Switched OFF
TOGGLE	0x03	0	Channel Toggled
DIM	0x04	1 Byte	Channel Dimmed to x% (x=hex 0x03 to 0x64)
DIM-UP	0x05	0	Dim Channel Up 3%
DIM-DOWN	0x06	0	Dim Channel Down 3%
DIM-A	0x07	0	Dim Channel Toggle
DIM-STOP	0x08	0	Dim Stop
SHUTTER_OPEN	0x09	0	Channel Switched to OPEN
SHUTTER_CLOSE	0x0A	0	Channel Switched to CLOSE
SHUTTER_CLOSE	0x1B	0	Channel Switched to CLOSE (USB dongle V1.0)
SHUTTER_STOP	0x0B	0	Channel Switched to STOP
RGB	0x0C	3 Bytes	R, G and B bytes
RGB_C	0x0D	-	Reserved
RGB_PLUS	0x0E	-	Reserved
OPEN_SLOW	0x0F	-	Reserved
SET_SHORT	0x10	0	Pairing Request (only with CID value hex 0x09)
SET_5S	0x11	0	Unpairing 1 Channel (only with CID value hex 0x09)
SET_10S	0x12	0	Unpairing All Channels (only with CID value hex 0x09)
STUDY	0x16	0	Pairing Request by Gateway
DEL_BUTTON	0x17	0	Reserved for gateway
DEL_ALL	0x18	0	Reserved for gateway
SET_TEMPERATURE	0x19	2 Bytes	Temperature sent by the temperature sensor
DOOR_OPEN	0x1A	0	Status sent by the sensor
BROADCAST_QUERY	0x1F	0	Reserved

QUERY_STATUS	0x20	0	Request sent by the gateway to know receiver status
REPORT_STATUS	0x21	N Byte(s)	Response from the receiver after 'QUERY_STATUS'
READ_CUSTOM	0x23	0	Read Customizable Data
SAVE_CUSTOM	0x24	N Byte(s)	Save Customizable Receiver Data
REPORT_CUSTOM	0x29	N Byte(s)	Response from the receiver after READ_CUSTOM and SAVE_CUSTOM
SET_SHORT_DIMMER	0x2C	N Byte(s)	Dimmer Pairing Request
SET_SHORT_SENSOR	0x2F	N Byte(s)	Sensor Pairing Request
In progress	0x		Automatic Acknowledgment
In progress	0x		Disable Automatic Acknowledgment

Edisio Model Field

Model	Value hex	Device	Function	Products Ref.
EMITRBTN	0x01	Emitter 8 Channels	8x On/Off/Pulse/Open/Stop/Close	ETC1/ETC4/EBP8
Reserved	0x02	Reserved	Reserved	
Reserved	0x03	Reserved	Reserved	
Reserved	0x04	Reserved	Reserved	
Reserved	0x05	Reserved	Reserved	
Reserved	0x06	Reserved	Reserved	
EMS-100	0x07	Motion sensor	On/Off/Pulse/Pilot wire//Open/Close	EMS-100
ETS-100	0x08	Temperature sensor	Temperature	ETS-100
EDS-100	0x09	Door sensor	On/Off/Pulse/Open/Close	EDS-100
EMR-2000	0x10	Receiver 1 Output	1x On/Off or Pulse	EMR-2000
EMV-400	0x11	Receiver 2 Outputs	2x On/Off or 1x Open/Stop/Close	EMV-400
EDR-D4	0x12	Receiver 4 Outputs	4x On/Off or Dimmer	EDR-D4
EDR-B4	0x13	Receiver 4 Outputs	4x On/Off/Pulse - 2x Open/Stop/Close	EDR-B4
EMSD-300A	0x14	Receiver 1 Output	1x On/Off or Dimmer	EMSD-300A
EMSD-300	0x15	Receiver 1 Output	1x On/Off or Dimmer	EMSD-300
EGW	0x16	Gateway	Send and Receive all commands	EGW
EMM-230	0x17	Emitter 2 Channels	2x On/Off/Pulse/Open/Stop/Close	EMM-230
EMM-100	0x18	Emitter 2 Channels	2x On/Off/Pulse/Open/Stop/Close	EMM-100
Reserved	0x19	Reserved	Reserved	Reserved
ED-TH-01	0x0E	Thermostat	Thermostat	ED-TH-04
ED-LI-01	0x0B	Receiver 1 Output	1x On/Off	ED-LI-01
ED-TH-02	0x0F	Receiver 1 Output	1x On/Off Heater/Cooler	ED-TH-02
ED-TH-03	0x0C	Receiver 1 Output FP	1x Off/Eco/Comfort/Auto (Fil Pilote)	ED-TH-03
ED-SH-01	0x0D	Receiver 2 Outputs	1x Open/Stop/Close	ED-SH-01
IR-TRANS	0x1B	IR Transmitter	Trigger	EGW-PRO/EN
STM-250	0x1E	Enocean sensor	Door	STM-250
STM-330	0x1F	Enocean sensor	Temperature	STM-330
PTM-210	0x20	Enocean switch	Switch	PTM-210
IR-DEVICE	0x21	Virtual device	IR Remote Control	EGW-PRO/EN
EN-DET	0x22	Enocean motion	Motion sensor	EGW-PRO/EN

EN-SOC	0x23	Enocean socket	Socket	EGW-PRO/EN
--------	------	----------------	--------	------------

5.2.2.23 id_PHY Oregon

id_PHY	Commercial Scientific Oregon device references	Fonction	Prot.
0x0	Probes Oregon protocol V1	thermometer	V1
0x1A2D	THGR122/ 228 /238/268, THGN122/123/132	Thermo+hygro	V2
0xCA2C	THGR328	Thermo+hygro	V2
0x0ACC	RTGR328	Thermo+hygro	V2
0xEA4C	THC238/268, THWR288 , THRN122, THN122/132, AW129/131	thermometer	V2
0x1A3D	THGR918 /928, THGRN228, THGN50	Thermo+hygro	V2
0x5A6D	THGR918N	Temp+Pressure	V2
0x1A89	WGR800	Wind sensor	V3
0xCA48	THWR800	S. pool thermo	V3
0xFA28	THGR810 , THGN800	Thermo+hygro	V3
0x2A19	PCR800	Rain sensor	V3
0xDA78	UVN800	UV sensor	V3

5.3 Jamming detection

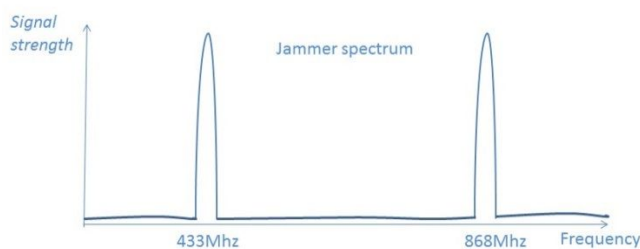
The RFPLAYER equipment has a jamming detection function, which has been developed with real jammers that can be found on the market. The jammers are classically used to fool the alarm systems by making wireless transmission impossible. They unfortunately have a formidable efficiency, even at a relatively large distance. For example, transmissions at home may be blocked by a jammer on the street. The emission power of a home automation device is a few mW and is limited by law. The transmit power of a jammer is a few W (watts) and is not limited by law because the jammers are naturally prohibited. The power ratio between home automation device and jammer is 1000...

One of the difficulties encountered in managing interference detection is the sensitivity to be given in order to be effective without having “false positives”. It is common to have short periods when communications are disrupted without being a jammer’s victim. A jamming event/flag may therefore appear during every transmission disturbances (that are not due to a malicious intent): This event/flag is called a “false positive”. But the goal is not to detect poor transmission quality but detect a jammer and the bad intention behind, then alert the user !

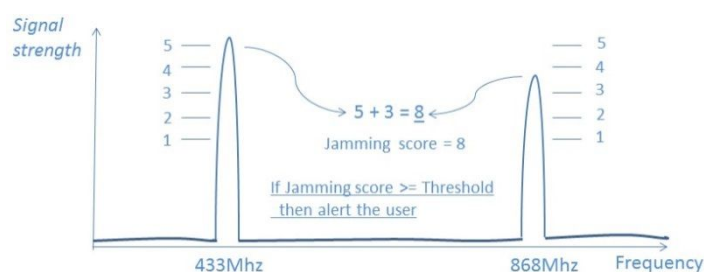
A robber does not know the characteristics of your alarm system and therefore has an interest in using a “large spectrum” jammer, which in practice covers simultaneously the 433Mhz and 868Mhz bands (and even 315MHZ US band in the same device !), which is most likely. The number of bands covered can be seen by classically counting the number of antennas...



The jamming detector can therefore be sensitive to several frequency bands in order to reduce the false positives. If the transmissions are heavily disturbed simultaneously on several bands then it is very likely that a jammer is active. A jammer with a very strong signal with random pseudo-modulation and is agnostic to a specific radiofrequency protocol. Any protocol is disrupted and can be even completely frozen.



The RFPLAYER controls both frequencies 433Mhz and 868Mhz. Each frequency receives a score between 0 and 5. Then, the 2 notes are added and the result is compared to a user-programmable threshold. If the result is greater than or equal to the threshold, the user is alerted.



Note that with this principle, a threshold between 1 and 5 detects a disturbance on a single frequency, while a threshold between 6 and 10 requires disturbances on both frequencies, which threshold will dramatically reduce false positives. Of course, do not disable RECEIVER 433Mhz and/or 868Mhz when these receivers check jamming. Reception Frequency on each band is not important because jammers are dirty generators and do not transmit on precise frequencies.

To be easily managed by the home automation BOX, the jamming detector is considered a pseudo "JAMMING" protocol with a single virtual device that sends "ON" frames when a jamming is detected ("ON JAMMING ID 0") and OFF when this jamming stops ("OFF JAMMING ID 0").

When jamming occurs, the RFPLAYER's LED flashes blue at high frequency.

You can try to trigger the jamming detector with a simple remote control 433Mhz or 868Mhz (preferably with a RF protocol that is not known to RFPLAYER). For this, set the threshold to 1. Then continually push the buttons of the remote control at 20 cm of the RFPLAYER during 20 to 30 seconds. Keep in mind that a threshold of 10 isn't 10 times harder to reach than a threshold of 1, but more 100 times, and needs dual band triggering... Despite this, our tests with a real jammer triggered the detector at 40m RFPLAYER through a house (not with a direct sight) with a threshold of 10.

When Jamming happens:

*FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rfLevel: -63dBm, floorNoise: -73dBm, rfQuality: 2
protocol: 15 (JAMMING), infoType: 1, frequency: 433920Khz
subType: 1, id: 0 (ON)*

When Jamming stops :

*FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rfLevel: -87dBm, floorNoise: -96dBm, rfQuality: 2
protocol: 15 (JAMMING), infoType: 1, frequency: 433920Khz
subType: 0, id: 0 (OFF)*

The most given parameters frequency eg. rfLevel, FloorNoise, rfQuality are irrelevant (eg. the “frequency: 433920Khz” above). Only, the frametype, protocol, infoType, subtype, id are significant.

The jammer threshold is set to 7 by default. It can be redefined by the command :

JAMMING <threshold>

Where *threshold* is a value between 0 and 10.

Value 0 disables jamming detection.

Value 1 is very sensitive but should lead to “false positives” events.

Value 10 is less sensitive but limits “false positives” events.

Minimal time to generate “JAMMING ON” is around 15s of “jamming detected situation”.

A simulator also allows you to send yourself alerts to virtually test the presence of a jammer by :

JAMMING SIMULATE

Will force receiving “JAMMING ON” frame. “JAMMING OFF” Frame is received 5 seconds later.

JAMMING SIMULATE <delay>

Will force receiving “JAMMING ON/OFF” frame ‘*delay*’ seconds later. (eg *JAMMING SIMULATE 60*).

Max value of delay is 255 seconds (more than 4 minutes). This command is saved if the RFPLAYER shutdowns before JAMMING ON event. Thus, this command can be programmed on the RFPLAYER configurator, then the RFPLAYER plugged on the HA Box to test “JAMMING ON” event.

Of course, the HA Box can send “*JAMMING SIMULATE*” command itself.

5.4 Firmware Update

Firmware can be updated by a simple encrypted text file. Firmware files are provided by Ziblue.

A given version of firmware cannot be flashed twice times and is rejected the subsequent times. Return to older versions is allowed.

Current firmware version is visible by STATUS SYSTEM instruction.

The update time is a relatively long (2mn):

- One minute is spent to download the firmware through the serial line.
- One minute is spent to verify file integrity at each step and then program the embedded controller.

The LED is RED and blinking during this time. The LED remains inactive if the downloaded firmware version is already running. Wait the end of RED blinking.

The new firmware automatically starts after programming. Restart the dongle (unplug then plug again the dongle) if it doesn't restart.

A new firmware can be programmed by different ways :

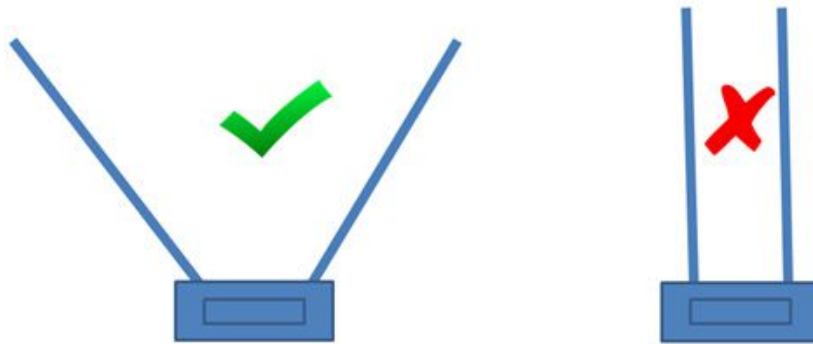
- By the Ziblue RFPLAYER configurator,
- By serial emulators available on the market (eg. TeraTerm, do “push a file”),
- By Home Automation BOX. Of course, do not send home automation commands during this time.

Previous user's configuration is preserved.

NB: The principle is “entirely store, verify, then flash, then verify again”, so that power can be down at each step of the programming without damage and the programming/flashing phase isn't critical (no “brick” effect).

5.5 Antennas

Best sensitivity



5.6 LED signalisation




At startup

Pink flash during 1 second	REPEATER function is enabled
-------------------------------	------------------------------

One very short blue flash then...	Hardware error :
1x RED flash	Low Frequency transceiver failure (433Mhz)
2x RED flashes	High Frequency transceiver failure (868Mhz)
3x RED flashes	EEPROM failure
4x RED flashes	SFLASH failure

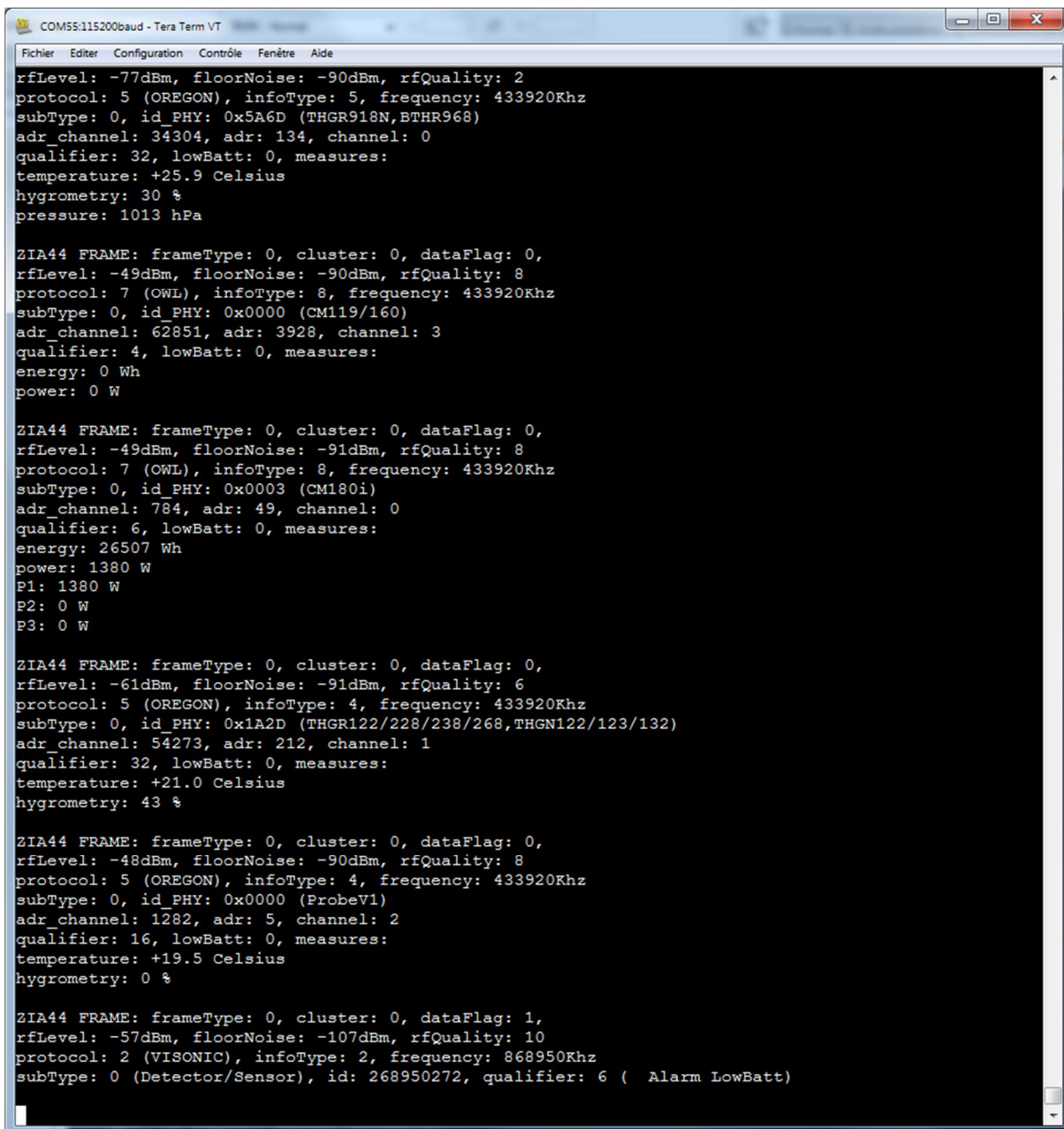
Blue/Red smooth variation during 2 seconds	Frequency calibration error
---	-----------------------------

During operation

	RF signal <u>is received</u> <i>blue</i>
	RF Frame <u>is received and recognized</u> <i>pink</i>
	RF Frame <u>is transmitted</u> <i>red</i>

5.7 Screenshots

After FORMAT TEXT command :



```

COM55:115200baud - Tera Term VT
Fichier  Editer  Configuration  Contrôle  Fenêtre  Aide
rfLevel: -77dBm, floorNoise: -90dBm, rfQuality: 2
protocol: 5 (OREGON), infoType: 5, frequency: 433920Khz
subType: 0, id_PHY: 0x5A6D (THGR918N,BTHR968)
adr_channel: 34304, adr: 134, channel: 0
qualifier: 32, lowBatt: 0, measures:
temperature: +25.9 Celsius
hygrometry: 30 %
pressure: 1013 hPa

ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rfLevel: -49dBm, floorNoise: -90dBm, rfQuality: 8
protocol: 7 (OWL), infoType: 8, frequency: 433920Khz
subType: 0, id_PHY: 0x0000 (CM119/160)
adr_channel: 62851, adr: 3928, channel: 3
qualifier: 4, lowBatt: 0, measures:
energy: 0 Wh
power: 0 W

ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rfLevel: -49dBm, floorNoise: -91dBm, rfQuality: 8
protocol: 7 (OWL), infoType: 8, frequency: 433920Khz
subType: 0, id_PHY: 0x0003 (CM180i)
adr_channel: 784, adr: 49, channel: 0
qualifier: 6, lowBatt: 0, measures:
energy: 26507 Wh
power: 1380 W
P1: 1380 W
P2: 0 W
P3: 0 W

ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rfLevel: -61dBm, floorNoise: -91dBm, rfQuality: 6
protocol: 5 (OREGON), infoType: 4, frequency: 433920Khz
subType: 0, id_PHY: 0x1A2D (THGR122/228/238/268,THGN122/123/132)
adr_channel: 54273, adr: 212, channel: 1
qualifier: 32, lowBatt: 0, measures:
temperature: +21.0 Celsius
hygrometry: 43 %

ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rfLevel: -48dBm, floorNoise: -90dBm, rfQuality: 8
protocol: 5 (OREGON), infoType: 4, frequency: 433920Khz
subType: 0, id_PHY: 0x0000 (ProbeV1)
adr_channel: 1282, adr: 5, channel: 2
qualifier: 16, lowBatt: 0, measures:
temperature: +19.5 Celsius
hygrometry: 0 %

ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 1,
rfLevel: -57dBm, floorNoise: -107dBm, rfQuality: 10
protocol: 2 (VISONIC), infoType: 2, frequency: 868950Khz
subType: 0 (Detector/Sensor), id: 268950272, qualifier: 6 ( Alarm LowBatt)

```

After STATUS command :

```
COM122:115200baud - Tera Term VT
Fichier  Editer  Configuration  Contrôle  Fenêtre  Aide
ZIA--STATUS
systemStatus request number=0
Version: 1.17, Time: 53309s, Mac: 0xFD0CBA71, LBT: 16dBm, Factory: 1488874419, ClusterID: 0, RTde
nials: 0, MaskT: 0xFFBF8FF9, MaskA: 0x3FFFFFFE,
transmitter available: VISONIC433 VISONIC868 CHACON DOMIA X10 X2D433 X2D868 X2DSHUTTER X2DELEC X2
DGAS RTS BLYSS PARROT KD101
receiver available: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 P
ARROT
receiver enabled: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PAR
ROT
repeater available: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 P
ARROT
repeater enabled: X10 RTS VISONIC BLYSS CHACON OREGONV1 OREGONV2 OREGONV3/OWL DOMIA X2D KD101 PAR
ROT

ZIA--
radioStatus request number=0
Frequency: 433920Khz Most 433Mhz devices,
Selectivity: 1 Very Low (+ Somfy RTS), FloorNoise: -88dBm High noise, DspTrigger: 8dBm, RFlink: 1
Enabled, RFlinkTrigger: 12dBm, sentFrames: 17019, discFrames: 0, dutyCycle: 360000ms/h by ETSI,
remainDC: 359784ms,
Frequency: 868950Khz Visonic,
Selectivity: 0 Default value, FloorNoise: -108dBm Very small noise, DspTrigger: 6dBm, RFlink: 1 E
nabled, RFlinkTrigger: 10dBm, sentFrames: 16, discFrames: 0, dutyCycle: 3600ms/h by ETSI, remainD
C: 3600ms,
```

Tera Term configuration :



Tera Term: Config. port série

Port: COM55

Vitesse: 115200

Données: 8 bit

Parité: none

Stop: 1 bit

Ctrl. de flux: hardware

Délai de transmission

0 msec/car 0 msec/ligne

OK

Effacer

Aide



Tera Term: Configuration du Terminal

Taille du Terminal

150 x 54

Terminal = taille fenêtre

Redim Auto fenêtre

Type Terminal VT100

Réponse:

Aller à la ligne

Réception LF

Emission: CR

Echo local

Commut. auto (VT<->TEK)

OK

Effacer

Aide

TRACE of TRACE RFLINK command (only for debug) with ZIA55 response.

```

Fichier Editer Configuration Contrôle Fenêtre Aide
rflevel: -58dBm, floorNoise: -92dBm, rfQuality: 8
protocol: 5 (OREGON), infoType: 4, frequency: 433920Khz
subType: 0, id_PHY: 0x0000 (ProbeV1)
adr_channel: 2817, adr: 11, channel: 1
qualifier: 16, lowBatt: 0, measures:
temperature: +20.2 Celsius
hygrometry: 0 %

ZIA55 RFLINK REC frame F=433920Khz level=-69dBm noise=-92dBm Samples number=193 list below= level:time(us)
ZIA55 RFLINK line 0= 0:0, 1:160, 0:720, 1:280, 0:680, 1:320, 0:640, 1:320, 0:640, 1:360, 0:640, 1:280, 0:720, 1:280, 0:680, 1:320,
ZIA55 RFLINK line 1= 0:680, 1:280, 0:640, 1:360, 0:640, 1:360, 0:640, 1:280, 0:680, 1:320, 0:640, 1:320, 0:640, 1:320, 0:640, 1:320,
ZIA55 RFLINK line 2= 0:640, 1:320, 0:640, 1:360, 0:600, 1:320, 0:680, 1:320, 0:640, 1:320, 0:640, 1:360, 0:640, 1:360, 0:640, 1:320,
ZIA55 RFLINK line 3= 0:1160, 1:760, 0:1160, 1:840, 0:1120, 1:800, 0:1160, 1:320, 0:680, 1:800, 0:1120, 1:360, 0:640, 1:760, 0:760, 1:240,
ZIA55 RFLINK line 4= 0:1160, 1:320, 0:640, 1:360, 0:640, 1:320, 0:680, 1:280, 0:680, 1:800, 0:1120, 1:320, 0:720, 1:280, 0:680, 1:320,
ZIA55 RFLINK line 5= 0:640, 1:320, 0:640, 1:800, 0:1120, 1:360, 0:680, 1:800, 0:720, 1:280, 0:1120, 1:280, 0:720, 1:760, 0:1200, 1:280,
ZIA55 RFLINK line 6= 0:640, 1:840, 0:1160, 1:320, 0:640, 1:320, 0:640, 1:320, 0:680, 1:320, 0:640, 1:320, 0:640, 1:360, 0:640, 1:360,
ZIA55 RFLINK line 7= 0:640, 1:280, 0:680, 1:320, 0:680, 1:280, 0:720, 1:280, 0:680, 1:320, 0:640, 1:320, 0:640, 1:360, 0:640, 1:280,
ZIA55 RFLINK line 8= 0:720, 1:280, 0:640, 1:840, 0:640, 1:360, 0:640, 1:320, 0:1080, 1:840, 0:1200, 1:280, 0:640, 1:840, 0:1120, 1:360,
ZIA55 RFLINK line 9= 0:640, 1:280, 0:720, 1:760, 0:1160, 1:320, 0:640, 1:360, 0:640, 1:360, 0:640, 1:320, 0:640, 1:800, 0:680, 1:320,
ZIA55 RFLINK line 10= 0:1120, 1:360, 0:640, 1:320, 0:640, 1:320, 0:680, 1:280, 0:720, 1:320, 0:640, 1:320, 0:640, 1:760, 0:1200, 1:320,
ZIA55 RFLINK line 11= 0:640, 1:320, 0:640, 1:840, 0:1120, 1:840, 0:1120, 1:320, 0:640, 1:280, 0:720, 1:280, 0:640, 1:840, 0:640, 1:360,
ZIA55 RFLINK line 12= 0:640, 1:320, 0:0.
ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rflevel: -61dBm, floorNoise: -91dBm, rfQuality: 7
protocol: 5 (OREGON), infoType: 4, frequency: 433920Khz
subType: 0, id_PHY: 0x1A2D (THGR122/228/238/268,THGN122/123/132)
adr_channel: 7425, adr: 29, channel: 1
qualifier: 32, lowBatt: 0, measures:
temperature: +21.9 Celsius
hygrometry: 33 %

ZIA44 FRAME: frameType: 0, cluster: 0, dataFlag: 0,
rflevel: -62dBm, floorNoise: -91dBm, rfQuality: 7
protocol: 5 (OREGON), infoType: 4, frequency: 433920Khz
subType: 0, id_PHY: 0x1A2D (THGR122/228/238/268,THGN122/123/132)
adr_channel: 7425, adr: 29, channel: 1
qualifier: 33, lowBatt: 1, measures:
temperature: +21.9 Celsius

```

one RF incoming RFLINK Frame (to USB) dump (colored) example

```

struct RFLINK_HEADER {
    unsigned char FrameType; // always the first element of the structure
    unsigned long KhzFrequency;
    signed char RFLevel;
    signed char FloorNoise;
    unsigned char PulseElementSize;
    unsigned short Number; // Number of pulses, times two as every pulse has a mark and a space.
    unsigned char Repeats; // Number of re-transmits on transmit actions.
    unsigned char Delay; // Delay in ms. after transmit of a single RF pulse packet
    unsigned char Multiply; // Pulses[] * Multiply is the real pulse time in microseconds
    unsigned long Time; // Timestamp indicating when the signal was received (millis())
} IS_PACKED;

struct REC_RFLINK_PACKET // Raw signal variable places in a struct
{ // keep homogeneous with original_RFLINK_RawSignalStruct
    struct RFLINK_HEADER header;

    union {
        unsigned char cPulses[RFLINK_RAW_BUFFER_SIZE_WITH_MARGIN];
        unsigned short sPulses[RFLINK_RAW_BUFFER_SIZE_WITH_MARGIN_SHORT_SAMPLE];
    } data;
} IS_PACKED;

5A 49 01 96 00 01 00 9F 06 00 B0 95 01 83 00 00      ZI.-...ÿ..°.f..
00 28 66 25 02 00 00 07 44 07 07 06 21 06 08 06      .(f%....D...!...
22 06 08 06 21 06 08 06 21 06 21 06 09 06 08 06      "...!...!...!...
21 06 07 07 22 06 07 07 21 06 21 06 09 06 07 06      !..."...!...!...
22 06 21 06 09 06 21 06 08 06 07 06 21 07 22 05      ".!...!...!...!".
08 07 07 06 21 06 22 06 08 06 22 06 07 06 21 07      ...!"..."...!...
08 06 22 06 07 06 09 05 22 06 08 06 21 06 21 06      .."....."....!...
09 05 22 06 08 06 21 06 09 05 22 06 08 06 07 06      .."....!".....
22 06 08 06 22 06 21 06 07 07 07 06 22 07 07 06      "...!".....
21 06 22 06 08 05 09 06 21 06 00                      !"...!...

```

END of document.