

Telegesis		TG-ETRX35xDVK-PM-012-107
ETRX35xDVK Development Kit		Product Manual 1.07

# ETRX35xDVK – TELEGENESIS DEVELOPMENT KIT FOR ZIGBEE® TECHNOLOGY

## PRODUCT MANUAL



## 1 ETRX35xDVK – Development Kit Functional Summary



### Devkit Features

- ZigBee Networking “Out-of-the-Box”
- Low cost evaluation platform for ZigBee wireless mesh networking
- Designed to set up a ZigBee mesh network in only a few minutes without the need for any embedded software
- The freely downloadable Telegesis Terminal Software application offers an easy to use interface to the modules on the development boards.
- Also works with 3<sup>rd</sup> party terminal software like HyperTerminal.
- Broad selection of modules allows range testing of any possible module/antenna combination.
- Seamlessly Integrates into the Ember InSight Toolchain
- Development boards can be used as hardware platform for trials
- Battery option allows easy prototyping of end devices
- USB Drivers available for Windows from [www.telegesis.com](http://www.telegesis.com)

### Module Features

- Based on the Ember EM351 and EM357 single chip ZigBee™ / IEEE802.15.4 solutions
- 2.4GHz ISM Band
- Industry’s first ARM® Cortex-M3 based family of ZigBee modules
- 192kB (ETRX357) and 128kB (ETRX351) flash and 12kB of RAM
- 24 general-purpose I/O lines including analogue inputs (all GPIOs of the EM35x are accessible)
- Lowest Deep Sleep Current of sub 1µA and multiple sleep modes

The two Telegesis ETRX351-DVK and ETRX357-DVK development kits are an ideal starting point for development and evaluation of the ETRX351 and ETRX357 series low power 2.4GHz ZigBee modules.

The ETRX351 and ETRX357 modules are based on the third generation Ember EM351 and EM357 chipset offering the industry’s highest wireless networking performance and application code space at the lowest power consumption.

The modules’ unique AT-style command line interface allows designers to quickly integrate ZigBee technology without complex software engineering. For custom application development the ETRX35x development kits integrate with ease into Ember’s InSight development environment.

### Development Kit Contents

- 3 x USB Development Boards
- 3 x USB Cable
- 2 x ETRX35x on Carrier-Board
- 2 x ETRX35xHR on Carrier-Board
- 2 x ETRX35x-LRS on Carrier-Board
- 2 x ETRX35xHR-LRS on Carrier-Board
- 1 x ETRX3USB stick
- 2 x ½-wave Antenna
- 2 x ¼-wave Stubby Antenna

### Development Board Features

- Power can be supplied via USB, Power Jack or on board battery holder (2xAAA)
- Access to the Ember InSight Port
- Light and Temperature sensor
- 4 x Buttons, 2 x LEDs, 1 x Buzzer
- Reset and Bootload Button
- USB to Serial converter
- Breakout of all GPIO pins

### Example AT-Style Commands

AT+BCAST	Sends a Broadcast
AT+UCAST:<address>	Sends a Unicast
AT+EN	Establish PAN network
AT+JN	Join PAN

## Table of Contents

<b>1</b>	<b>ETRX35XDVK – DEVELOPMENT KIT FUNCTIONAL SUMMARY .....</b>	<b>2</b>
<b>2</b>	<b>ABSOLUTE MAXIMUM RATINGS OF THE DEVBOARD .....</b>	<b>5</b>
<b>3</b>	<b>OPERATING CONDITIONS OF THE DEVBOARD.....</b>	<b>5</b>
<b>4</b>	<b>ELECTRICAL SPECIFICATIONS.....</b>	<b>5</b>
<b>5</b>	<b>INTEROPERABILITY .....</b>	<b>5</b>
<b>6</b>	<b>OVERVIEW.....</b>	<b>6</b>
6.1	The Development Board .....	6
6.2	The carrier board.....	7
6.3	What’s in the ETRX35xDVK box? .....	7
<b>7</b>	<b>SETTING UP THE HARDWARE.....</b>	<b>8</b>
<b>8</b>	<b>ETRX35X PINOUT.....</b>	<b>8</b>
<b>9</b>	<b>THE DEVELOPMENT BOARD .....</b>	<b>10</b>
9.1	Development Board Interface Description .....	10
9.2	Development Board Sensors .....	12
<b>9</b>	<b>THE CARRIER BOARD.....</b>	<b>13</b>
<b>10</b>	<b>DRIVER INSTALLATION AND OPERATION .....</b>	<b>13</b>
10.1	Windows 9x/XP/2k Driver Installation.....	13
<b>11</b>	<b>APPLICATION SOFTWARE.....</b>	<b>18</b>
11.1	Software Set-up .....	19
11.2	Features of the Telegesis Terminal Application for ETRX35x.....	19
11.3	A Quick Start.....	20
11.3.1	Select the correct set of buttons .....	20
11.3.2	Network Setup .....	21
11.4	Configuring Buttons for your Setup .....	22
11.5	Using LEDs and ADCs on the ETRX357.....	23
11.6	Temperature display .....	24
<b>12</b>	<b>FIRMWARE UPGRADES .....</b>	<b>25</b>
12.1	Firmware Upgrades via Serial Port.....	25
12.2	Over the Air Firmware Upgrades.....	27
12.2.1	ETRX2 and ETRX357 features .....	27
12.2.2	Cloning an ETRX2 .....	27
12.2.3	Passthrough with an ETRX357 .....	27
12.2.4	Recovering on the default channel.....	28
<b>13</b>	<b>DEVBOARD SCHEMATIC.....</b>	<b>29</b>
<b>14</b>	<b>CARRIER BOARD SCHEMATIC.....</b>	<b>31</b>
<b>15</b>	<b>ETRX35X ORDERING INFORMATION .....</b>	<b>32</b>
<b>16</b>	<b>TRADEMARKS.....</b>	<b>33</b>

<b>17</b>	<b>DISCLAIMER.....</b>	<b>33</b>
<b>18</b>	<b>CONTACT INFORMATION.....</b>	<b>33</b>
<b>19</b>	<b>REFERENCES.....</b>	<b>33</b>

## 2 Absolute Maximum Ratings of the Devboard

Parameter	Min.	Max.	Units	Condition
Supply Voltage $V_{DD}$	-0.3	9	V	
Voltage on any I/O pin	-0.3	3.3	V	
Storage Temperature range	-50	150	°C	

**Table 1: Absolute Maximum Ratings**

The absolute maximum ratings given above should under no circumstances be violated. Stress exceeding one or more of the limiting values may cause permanent damage to the device.



Caution! ESD sensitive devices. Precautions should be used when handling the device in order to prevent permanent damage.

## 3 Operating Conditions of the Devboard

Typical values at 5V 25°C.

Parameter	Min.	Typ.	Max.	Units	Condition
Supply Voltage, $V_{DD}$	4	5	6	V	
Supply Current			150	mA	TX with LRS-Module
Operating ambient temperature range	-40	25	85	°C	

**Table 2. Operating Conditions**

The voltage regulators used are protected against overtemperature and overcurrent.

## 4 Electrical Specifications

See ETRX35x and ETRX35x-LRS Product Manuals

## 5 Interoperability

Unless otherwise specified the Development kits ships with Telegesis R3xx firmware based on EmberZNet4.x. Please note that the R3xx Telegesis AT-Command line interpreter is based on ZigBee PRO, but most of the functionality is implemented as a private application profile. Interoperability with wireless mesh networking solutions from other manufacturers is only possible when knowing the application profile specification of this device and using the provided transparent commands. If this is a concern please contact us for advice. Telegesis also offer separate firmware specifically developed for Smart Energy applications.

## 6 Overview

The ETRX35xDVK development kit has been designed to allow quick evaluation and prototyping using the ETRX35x ZigBee modules.

This document is intended to describe the hardware and accompanying software of the development kits. To learn more about the usage of the ETRX35x module please refer to the following documents:

- TG-ETRX35x-PM-010-xxx: ETRX35x Product manual
- TG-ETRX35x-LRS-PM-015-xxx: ETRX35x-LRS (long range) Product manual
- TG-ETRXn-R3xx-Commands: AT Style command dictionary for firmware R3xx

All our documents can be found at [http://www.telegesis.com/support/document\\_centre.htm](http://www.telegesis.com/support/document_centre.htm).

The ETRX35x module is available in eight variants, with the ETRX351 or ETRX357:

- ETRX35x – with on-board ceramic antenna and 8dBm output power
- ETRX35xHR – with Hirose coaxial connector and 8dBm output power
- ETRX35x-LRS – with on-board ceramic antenna, LNA and 19dBm output power
- ETRX35xHR-LRS – with Hirose coaxial connector, LNA and 19dBm output power

### 6.1 The Development Board

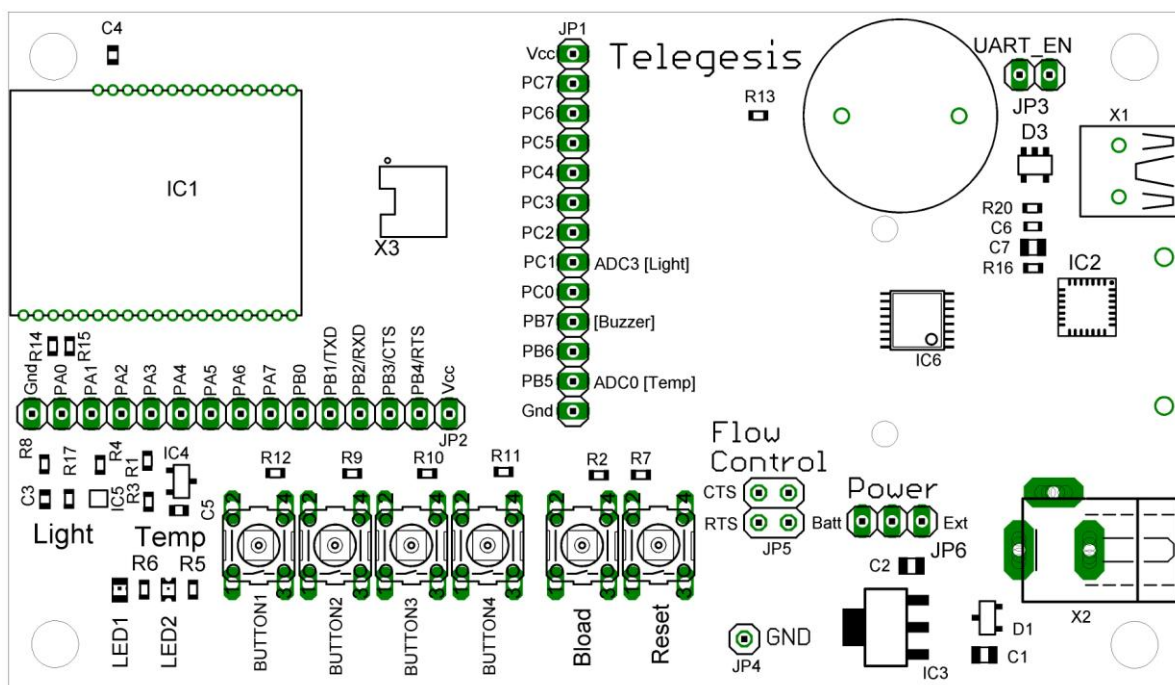


Figure 1. The development board

The development board which is part of the development kit hosts a USB to serial bridge as well as voltage regulation circuitry. Furthermore it hosts a reset switch, a bootloader switch, 4 buttons,

2 LEDs and a beeper, all of are connected to the I/Os of the module as described later in this document.

## 6.2 The carrier board

The carrier board has an ETRX35x module plus two LEDs, and a connector to attach to an Ember InSight Adaptor for reflashing the firmware. It plugs on to the development board.

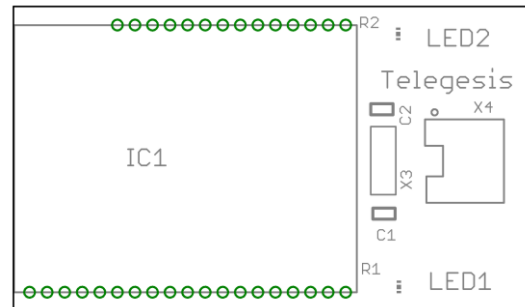


Figure 2. The carrier board

## 6.3 What's in the ETRX35xDVK box?

- 3 x ETRX35xDV Development Boards
- 3 x USB cables
- 2 x ETRX35x on carrier boards
- 2 x ETRX35xHR on carrier boards
- 2 x ETRX35x-LRS on carrier boards
- 2 x ETRX35xHR-LRS on carrier boards
- 1 x ETRX3USB USB stick
- 2 x ½-wave antennae
- 2 x ¼-wave antennae

These packages contain everything you need to immediately set up an ETRX35x development platform using the enclosed modules. The ETRX35xDVK currently includes an ETRX3USB stick built around an ETRX357 module; earlier kits included the ETRX2USB stick, but the USB drivers and the command set are the same.

## 7 Setting up the Hardware

In the development kits are all four versions of the module. They can be powered from either a USB hub, the mains (via a suitable power supply) or batteries.

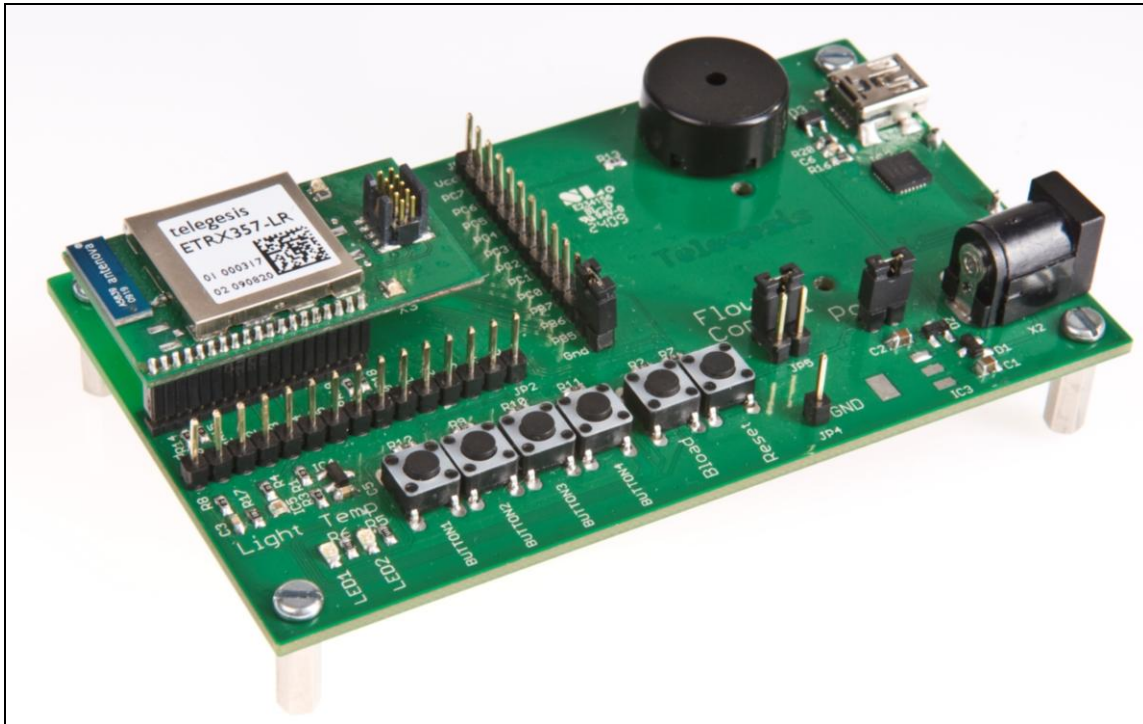


Figure 3. ETRX357-LR Module on Carrierboard plugged on to Devboard

## 8 ETRX35x pinout

The functions of each of the ETRX35x pins depend on the firmware. When using the Telegesis AT command-based firmware the S-register settings control the configuration of the I/Os. Some of them have a main function and an alternate function, as determined by the value of register S15. Holding pad PA5 low when powering up or resetting the module will cause it to enter the bootloader mode; this operation lies outside the normal firmware so it is independent of the value of S15. As the functions are firmware-dependent and may change between versions, Table 3 should be read in conjunction with the latest R3xxC AT Command Manual.

### Notes on Table 3 headings

**Name** is the designation of both the ETRX35x pad and the EM35x chip pin.

**Index** can be used to reference the individual pin in various S-registers (see the R3xx AT Command Manual)

**Pad** is the module's pad numbering

**Default direction** indicates whether the pad is normally an input or output

**Main function** and **Alternate function** indicate the purpose and connection of the pad

**Default alt fn setting** indicates whether the default firmware setting chooses the main or alternate function

Name	Index	Pad	Default direction S17= 0142CC	Main function	Alternate function	Default alt fn setting S15= 00000600
PC7	17	4	In			
PC6	16	3	In			
PC5	15	2	In		Enable TX_active on ETRX357 {4}	
PC4	14	24	In			
PC3	13	23	In			
PC2	12	22	In			
PC1	11	26	In		ADC3 (light sensor) {2}	
PC0	10	27	Out	LED		
PB7	F	28	In		ADC2, PWM {2}	
PB6	E	29	Out	LED, Button 4, IRQ3 {1}	ADC1 {2}	
PB5	D	30	In		ADC0 (temp sensor) {2}	
PB4	C	8	In			
PB3	B	6	In			
PB2	A	18	In		RXD	Enabled
PB1	9	17	Out		TXD	Enabled
PB0	8	25	In	Button 3, IRQ2 {1}		
PA7	7	5	Out	LED		
PA6	6	16	Out	LED		
PA5	5	15	In	(Bootload)		
PA4	4	14	In			
PA3	3	12	Out	Sensor supply {3}		
PA2	2	11	Out	Sensor supply {3}		
PA1	1	10	In	Button 2, IRQ1 {1}		
PA0	0	9	In	Button 1, IRQ0 {1}		

**Table 3. Module pads and functions**
**Notes**

{1} The IRQS are always enabled; it is not necessary to activate the alternate function. PB0 is not available for use on an ETRX357-LRS module as it is used internally to control the RF front-end module

{2} The ADCs are normally disabled; it is necessary to activate the alternate function

{3} In a development kit, the sensor supply outputs PA2 and PA3 must be high for the sensors to function

{4} On the ETRX357 TX\_active is an output which indicates that the RF circuit is transmitting. On the ETRX357-LRS TX\_active is always selected, and is not available as an ordinary I/O because it is connected internally to RF components.

## 9 The Development Board

### 9.1 Development Board Interface Description

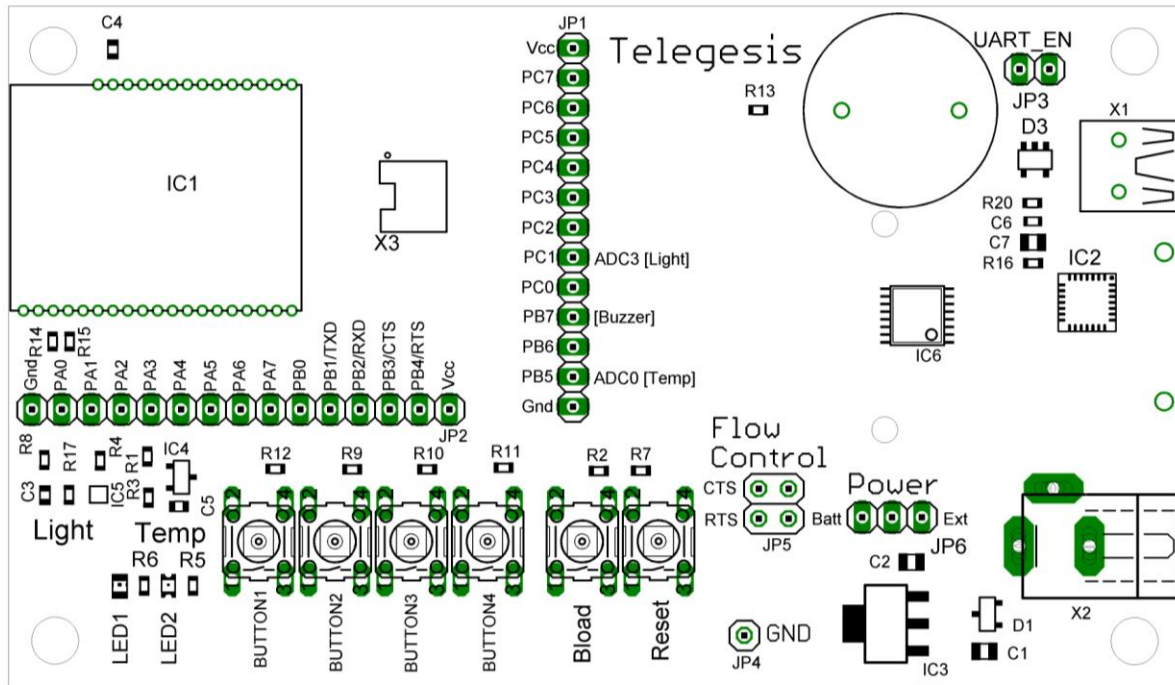


Figure 4. The development board

Figure 4 shows the location of the connectors described below.

**Programming Connector:** The 10-way programming connector X3 is used to program the ETRX35x module from an Ember InSight Adaptor. It is duplicated on the carrier board, and will not normally be fitted in the Development Board.

**USB Port:** The USB serial port allows connectivity to a PC. This provides access to the command line interface and the bootloader for firmware upgrades, and supplies DC power to the board.

**I/O connection:** JP1 and JP2 can be used to connect the I/O pins as shown in Table 4.

**Reference Ground:** JP4 is connected to the devboard's ground plane. It can be used as a reference point when making measurements on the devboard.

**I/O breakout:**

JP1 and JP2 give access to the I/O on the ETRX35x module. The individual pins are labelled on the circuit board, and the pin numbering (PA0, PB1 etc) matches that of the EM35x chip inside the module.

Pin	Devboard functionality
PA0	Button1
PA1	Button2
PA2	Temp sensor supply
PA3	Light sensor supply
PA5	Bootloader button
PB0	Button3
PB1	TXD
PB2	RXD
PB3	CTS
PB4	RTS
PB5	ADC0 (Temp sensor reading)
PB6	Button4
PB6	LED1
PB7	Buzzer
PC0	LED2
PC1	ADC3 (Light sensor reading)

Table 4. I/O Connectivity on development board

**Flow Control Selection:** JP5 is used to connect the RTS and CTS lines used for the flow control to the host. By default flow control is disabled and the corresponding lines of the module are used as standard I/Os (see the AT command dictionary on how to enable flow control), so the default setting of JP5 does not connect those lines to the host as shown in Figure 5. When flow control is enabled JP5 must be set as shown in Figure 6. Please make sure the jumpers are only set to this configuration when flow control is enabled as otherwise I/Os driving against each other (via a protective resistor) will increase the current consumption.

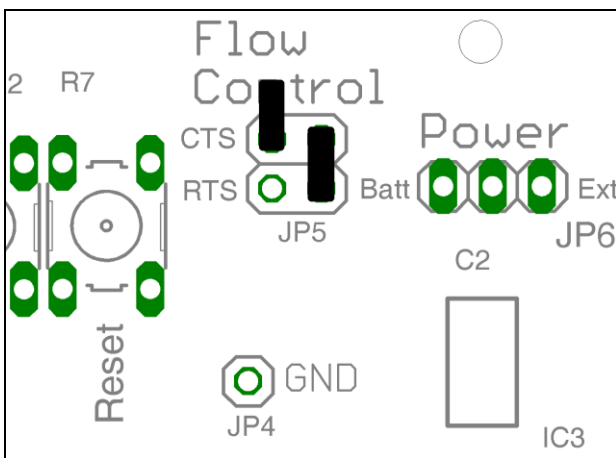


Figure 5. No Flow Control (default)

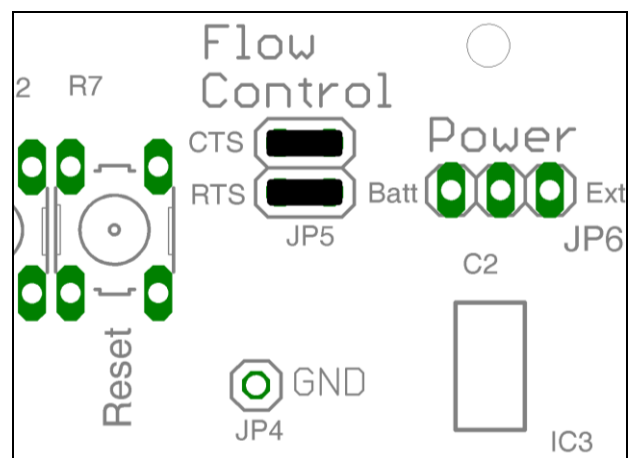
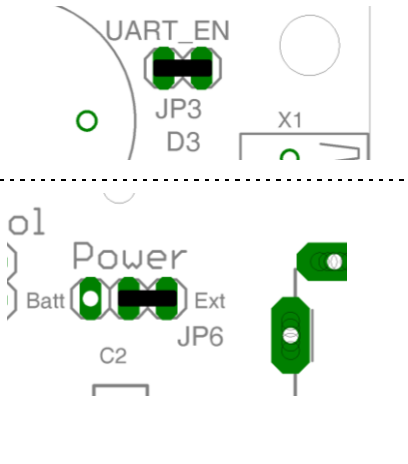
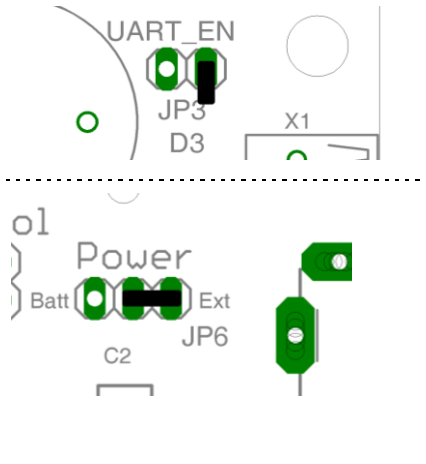
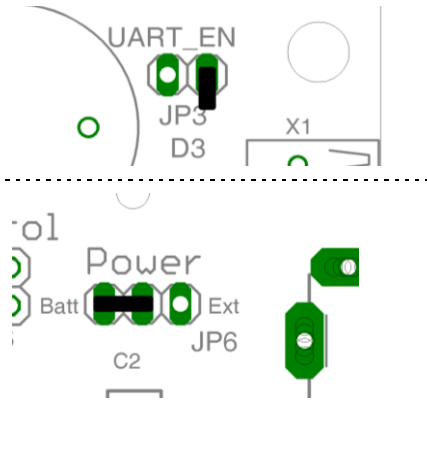


Figure 6. Flow control enabled

**Power supply jumpers:** links must be inserted at JP3 and JP6 according to the DC supply feed. JP3 can be left connected at all time, but it disconnects the serial interface lines and prevents power drain through them when the serial (USB) interface is not used. This will minimise the current drain from a battery and produce a more accurate reading of the current consumption.

It is permissible to attach a USB connection while the board is powered from a battery or external supply.

There are three options:

USB power	External power through the X2 socket	Battery power
		
<p>Insert JP3. Connect the centre pin of JP6 to 'Ext'.</p>	<p>Optionally, omit JP3. Connect the centre pin of JP6 to 'Ext'.</p>	<p>Optionally, omit JP3. Connect the centre pin of JP6 to 'Batt'.</p>

**Figure 7. Power feed options**

Instead of a shorting link, a current meter can be inserted at JP6 to monitor the power consumption.

## 9.2 Development Board Sensors

The board's temperature sensor is a National Semiconductor LM61 which has an offset of 600mV and a sensitivity of 10mV/deg. Hence for example 0°C give 600mV and 20° gives 800mV. The ETRX357's ADCs have a sensitivity of 0.1mV/LSB, so 100 LSB's  $\equiv$  1 degree.

The light sensor is an Avago APDS 9005; it indicates ambient light level but is not suitable for accurate measurements.

## 9 The Carrier Board

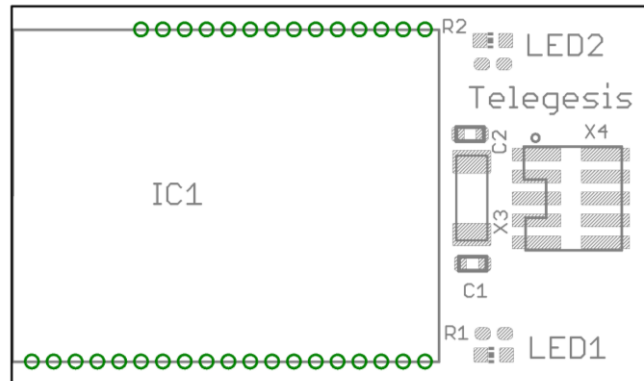


Figure 8. The carrier board

The ETRX35x carrier board has two LEDs.

Pin	Carrier board functionality
PA6	LED1
PA7	LED2

Table 5. Carrier board LEDs

**Programming Connector:** The 10-way programming connector X4 is used to program the ETRX35x module from an Ember InSight Adaptor.

**Watch crystal:** an optional 32kHz crystal and its associated capacitors can be fitted at X3, C1 and C2. In this case, R14 and R15 should be removed from the main development board. The use of this crystal is dependent on the user's firmware and the latest Ember documentation should be consulted for details of the circuit design.

## 10 Driver Installation and Operation

The ETRX35x development board uses the same USB-to-serial device as the ETRX2 products, so the same USB drivers can be used. The ETRX2USB drivers obtainable from [www.telegesis.com/telegesis\\_zigbee\\_technology\\_-\\_technical\\_support\\_/software\\_download.htm](http://www.telegesis.com/telegesis_zigbee_technology_-_technical_support_/software_download.htm) will generate a virtual COM port allowing easy access to the serial port of the embedded ETRX35x. The development kit board and the USB stick use the same drivers so you only need to install them once.

### 10.1 Windows 9x/XP/2k Driver Installation

Install the drivers before connecting the development kit using the provided USB cable. The driver package should be unzipped into a local folder. When executing the file 'TgVCPInstaller.exe' an installer will guide you through the steps required for the driver installation. If prompted that the driver hasn't passed the Windows logo test simply press 'Continue Anyway'. Note that updates to the driver package may result in the various messages differing from the screen-shots shown here.



Figure 9. Windows Logo Test

The installer will first ask you to confirm the location for the files:

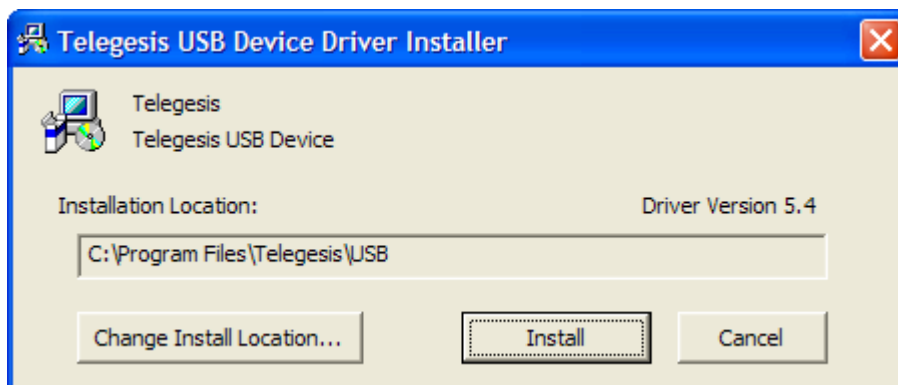


Figure 10. File location

After the files are installed you may have to restart the computer:

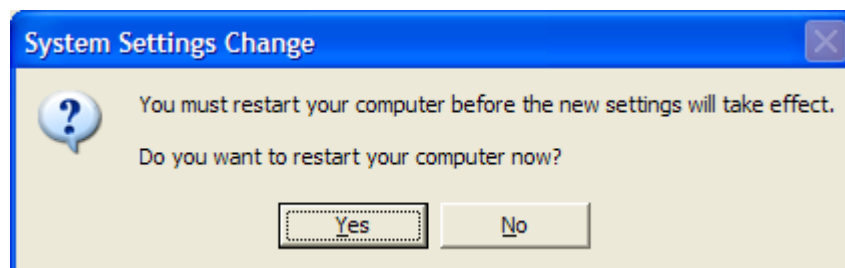


Figure 11. Request to restart

After you connect the devboard Windows® will prompt that new hardware has been found. If you have run the 'TgVCPInstaller.exe' Windows will be able to install the driver by automatically as shown in Figure 13. If you have not run 'TgVCPInstaller.exe' you will have to manually point to the directory into which you have unzipped the driver. Depending on your PC setup Windows may ask to check the internet for updated drivers, but there is no point in doing this.



Figure 12. Found New Hardware Wizard

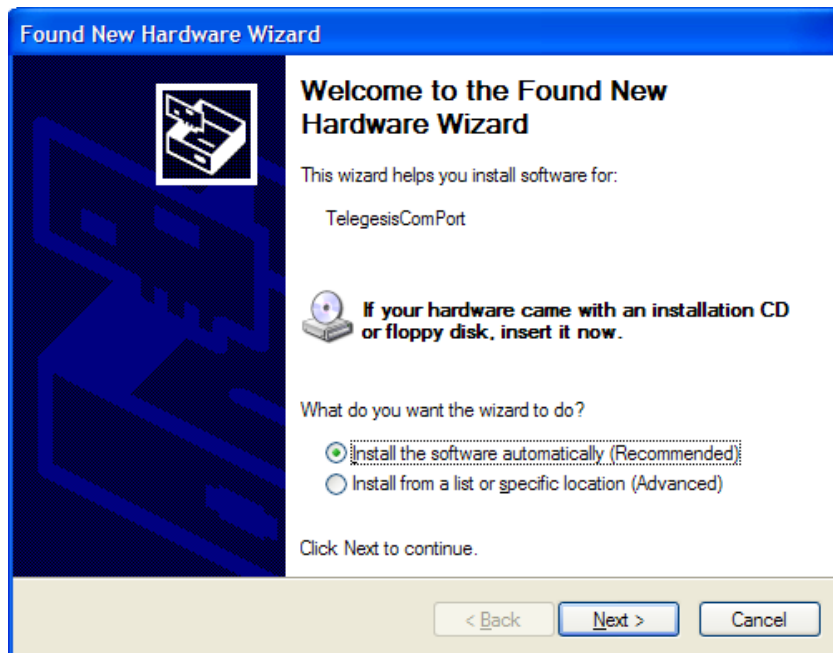
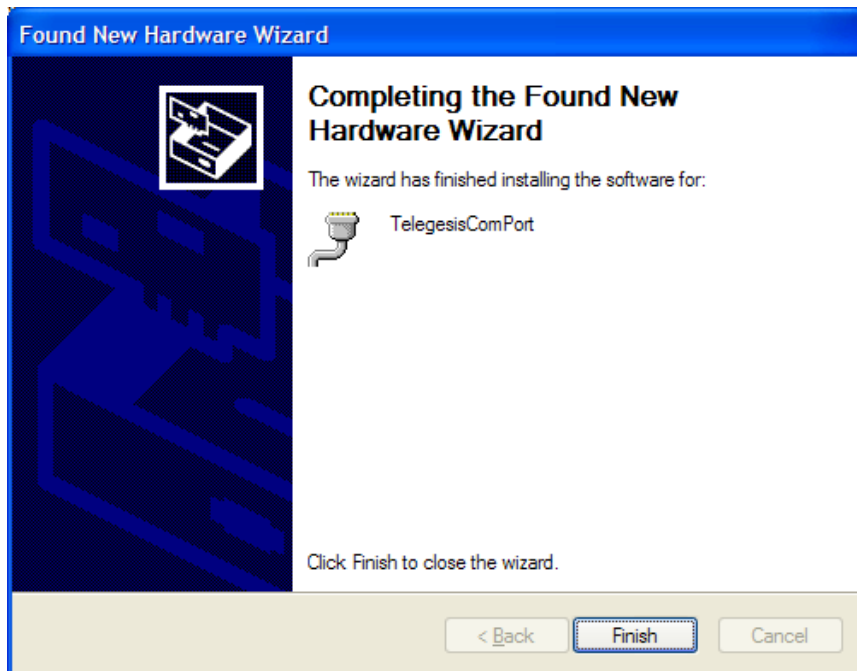


Figure 13. Install software Automatically



**Figure 14. Satisfactory completion**

Please note that each devboard and USB stick has a unique serial number which requires the installation procedure to be repeated with every new unit being attached to the computer. This allows multiple devices to be used on the same computer at any one time.

In order to find out the identity/number of the virtual COM port the devboard or USB stick has been assigned to, please open the Device Manager under the Windows Computer Management screen (see below) and click on the Ports (COM and LPT) section where you should find the new virtual COM port. By double clicking on the entry of the virtual COM port you can also change the number assigned to the virtual COM port when entering the advanced setup of the device.

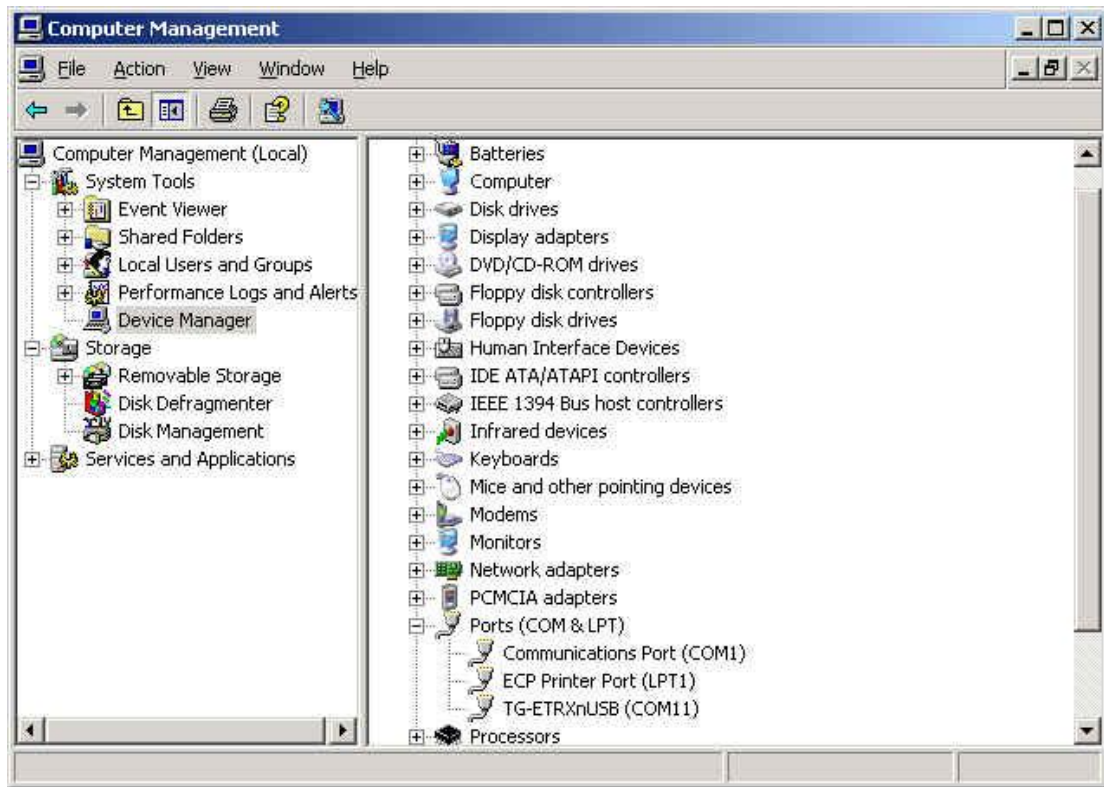


Figure 15. Device Manager

Once the correct COM port has been selected, the Telegesis Terminal software can be used to control the devboard as described in chapter 10.

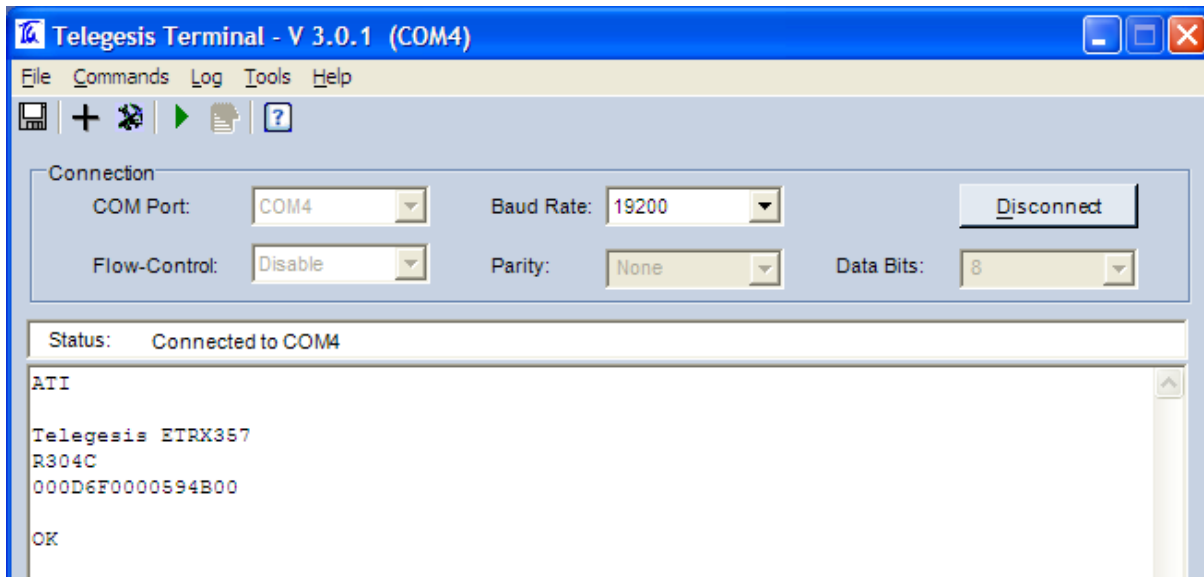


Figure 16. Telegesis Terminal



**11.1 Software Set-up**

Before installing Telegesis Terminal, you must install .NET Framework Version 1.1 Redistributable Package from Microsoft. This is currently available at

[www.microsoft.com/downloads/en/details.aspx?familyid=262D25E3-F589-4842-8157-034D1E7CF3A3&displaylang=en](http://www.microsoft.com/downloads/en/details.aspx?familyid=262D25E3-F589-4842-8157-034D1E7CF3A3&displaylang=en)

There is also a Service Pack for version 1.1. Note that Version 2 or later is not sufficient as it supplements Version 1.1 but does not replace it.

Then download Telegesis Terminal from our website at [www.telegesis.com/telegesis\\_zigbee\\_technology\\_-\\_technical\\_support\\_/telegesis\\_terminal.htm](http://www.telegesis.com/telegesis_zigbee_technology_-_technical_support_/telegesis_terminal.htm). After installing the Telegesis Terminal Application program the command buttons for the firmware will be shown at the bottom of the window. In order to use the Telegesis Terminal software, select the correct COM port and the connection parameters (ETRX35x default 19200, 8 bits, no parity, no flow control) and press the connect button. These settings are automatically retained each time the software is re-started.

**11.2 Features of the Telegesis Terminal Application for ETRX35x**

To make life easier all of the commands have been pre-defined and conveniently grouped at the bottom of the terminal window. Pressing a button (where no parameters are required) causes the corresponding command to be issued instantly. Where a parameter is required the command is shown in the Command bar and the required parameter can be entered manually. In order to issue the command from the command bar simply press the <enter> or the Send button. To see the parameters for a specific command, move and hold the mouse pointer over the chosen command button - Figure 18.

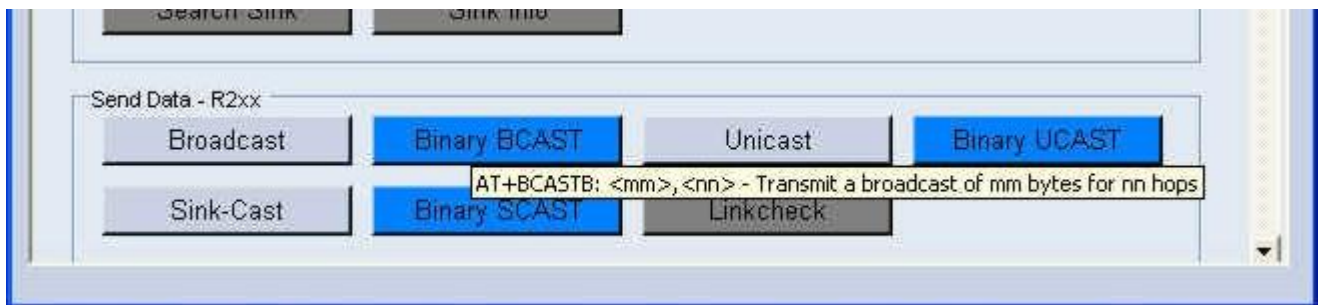


Figure 18. Screen tips for parameters

Every EUI64 number reporting in is listed in a separate window which can be opened or closed by clicking on the button labelled '**Device List**'. If a device ID is required as a parameter in the command bar, simply double click on an entry in the device list and its EUI64 will be automatically transferred to the current cursor position of the command bar.

To allow for easier identification, the EUI64 IDs in the device list can be named. When right clicking on any EUI64 ID a name can be associated with the respective ID - Figure 19.

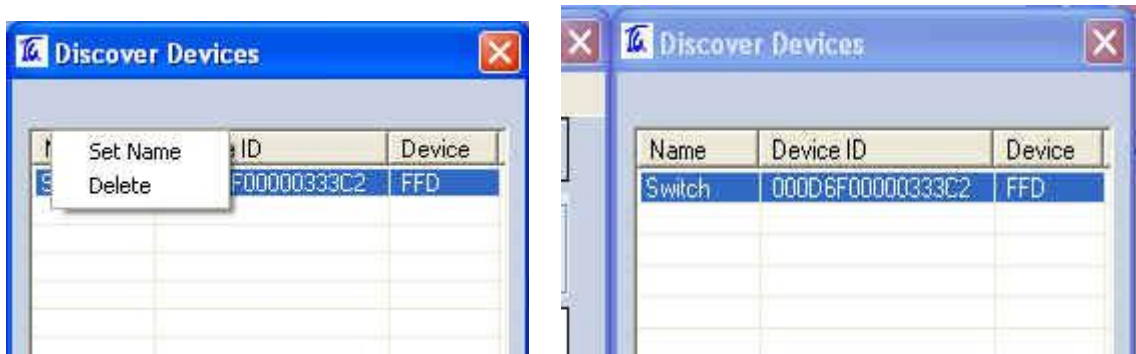


Figure 19. Device Naming

The application software also allows you to add custom command buttons and edit existing command buttons. New groups of commands can be created and command buttons can be moved between groups. To learn more about how to do this please refer to the help files of the Telegesis Terminal Software by selecting **Help / Contents**.

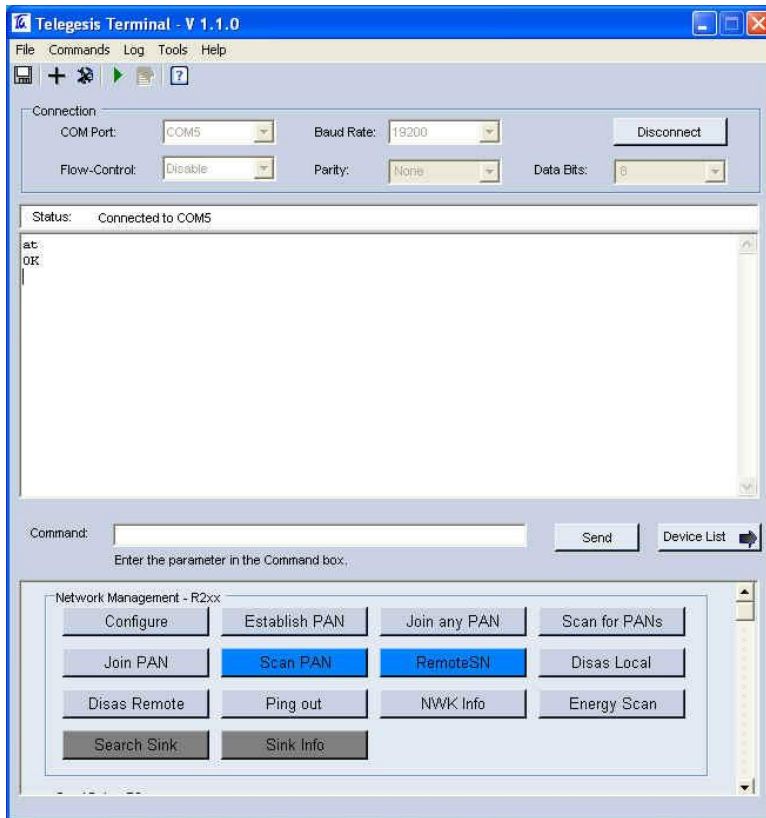
### 11.3 A Quick Start

This section gives you a quick introduction on how to get started. Power up the node connected to the PC and type **AT** followed by **<enter>**. If the communication to the module is working the module will prompt **OK**, if not check the USB connectivity and driver status and make sure you have connected to the correct COM port with the correct speed. Please note that even when connected to the wrong COM port there may be a reply to AT, for example when you are communicating with your PC's modem or Bluetooth device. To double check you are actually communicating with a Telegesis ZigBee device use the ATI command described below.

#### 11.3.1 Select the correct set of buttons

Type **ATI<enter>** to verify your firmware version (eg R303C). Then select File → Open Layout on the menu bar and open the R3xx.xml file as appropriate. This attaches the correct set of AT commands to the buttons. You can create your own button configurations for different situations and save them as separate files. Telegesis Terminal starts up using the configuration of the previous session.

### 11.3.2 Network Setup



To establish a PAN network issue the **AT+EN** command, or alternatively press the **'Establish PAN'** button.

The local unit will now scan all available 16 channels and establish a PAN with a random PAN ID, on the quietest available one. This may take up to 16 seconds and leads to the node becoming the networks coordinator. When successful the module will prompt **'JPAN'** followed by the details of the newly created PAN. If you get an error message instead, it is likely that the module was already part of a PAN, so you will need to issue the **AT+DASSL** command or press the **'Disas Local'** button to leave the PAN before going back to starting a new one. In order to find out about the network status simply issue the **AT+N** command or press the **'NWK Info'** button.

Figure 20. Command Line Interactions

Once the network is established, remote nodes can be powered up ready to join in. If you have serial access to remote nodes simply issue the **AT+JN** command or press the **'Join any PAN'** button, to join the newly established PAN. If you do not have serial access to the remote nodes (as is the case of the two MCBs provided with the previous ETRX2 based DVKA) you will just need to wait for them to join the network automatically. By default, once every minute all nodes (except coordinators) are set up to check whether there are any neighbours on the same PAN nearby, or if they have been orphaned. If no neighbours are found after 5 consecutive tries, the unit will leave the (deserted) PAN and try to join into a new one, once every minute.

This initial network set-up can take a few minutes, especially with no serial access to remote nodes, but once the network is set up it will remain set even after power cycles. New nodes joining will cause a prompt **'NEWNODE:'** on the remote side and display the **JPAN** message locally as described above.

The ETRX357 stores its network parameters in non-volatile memory, so if you reset or power-cycle a module there is no need to re-establish or join the PAN again.

If devices do not join the desired PAN automatically, check that they are not already in a network of their own. The command **AT+PANSCAN** will reveal the presence of other networks in the vicinity.

To learn more about setting up and maintaining a PAN please refer to the user guide and the AT Command Dictionary.



## 11.5 Using LEDs and ADCs on the ETRX357

The LEDs are connected to the ETRX357 in a pull-down arrangement so each output must be set to '0' to turn on its LED. Refer to Table 3 for details of the various connections; since they are distributed across the 16 I/O pins some examples of typical functions are given here for convenience. The four LED pins are configured as outputs by default so they can be used immediately. Before each ADC can be used, its pin must be set to use the alternate function in register S15, but it does not matter whether it is defined as a digital input or output. The temperature and light sensors on the circuit board are powered from two of the ETRX357's outputs, so these pins must be set high in order for the sensors to function.

Turn on local LEDs	ATS18=00000000
Turn off local LEDs	ATS18=000140C0
Turn off all remote LEDs in the network	ATSALL,FFFF,18=000140C0
Turn on local LEDs and supply power to temperature and light sensors	ATS18=0000000C
Enable temperature and light ADCs	ATS15=0002E600
Read local light sensor	ATS22?

Note that these immediately change the actual values of the registers described but not their default values.

The ADC registers can be read on a local or remote device by using the ATS or ATREMS commands, but they can also be sent automatically to the network sink by using function 0110 or 0130; consult the AT Command Manual for the details, and see section 11.6.

## 11.6 Temperature display

Telegesis Terminal includes a demonstration of how data can be collected from remote nodes and displayed on a central “sink” node. To activate the data-gathering function:

1. Define the module on the data-gathering development board as the network sink with the command

```
ATS104=1
```

2. Instruct one or more Carrier Boards to send their ADC readings at regular intervals by setting a Timer/Counter on the CB, eg to send every second using Timer/Counter 5 set:

```
S33=0004
```

```
S34=8130
```

3. Activate all four ADCs on each Carrier Board with the command

```
ATS15=0002E600
```

4. Turn on the sensor power on each Carrier Board with the commands

```
ATS182=1
```

```
ATS183=1
```

5. The ADC readings will appear at the sink in the form of a “FN0130” prompt, which also contains the digital I/O values and the contents of the S46 counter register. See the AT Command Manual for full details of the prompt and the registers used here.

It is usually easiest to set the sensor device’s registers by accessing them via USB, but you can also use the ATREMS command from the sink node.

You can activate a graph by selecting Tools -> R2xx Temperature Awareness but it is not suitable for the ETRX357. Telegesis Terminal was written for the ETRX2; the ETRX357’s ADCs are ten times more sensitive so its temperature readings will appear strange. The graph displays ADC1 which was correct for an ETRX2 carrier board, but the ETRX357 carrier board uses ADC0 instead. Built-in function 8130 can send all four ADC readings, though not all will be shown on the graph.

## 12 Firmware upgrades

If required, the firmware of the ETRX35x modules can be upgraded serially as well as over the air. Over-the-air upgrading is primarily a function of the Ember bootloader and it is not available with the early bootloader versions. See section 12.2 for more details.

### 12.1 Firmware Upgrades via Serial Port

In order to upgrade the firmware of the ETRX35x module using the serial bootloader, issue the “**AT+BLOAD**” command either by typing it in, or by pressing the respective button in the “**Module Control**” group of the Telegesis Terminal Application.

Alternatively the button labelled “Blood” can be pressed on the development board whilst the reset button is pressed and released.

After entering the bootloader, the connection parameters need to be changed to 115200bps, 8 data bits, 1 stop bit, no parity, no flow control (providing that it is not already set to these values). This is achieved by pressing the ‘**Disconnect**’ button, changing the settings and then pressing the ‘**Connect**’ button (if only the connection speed needs to be changed disconnecting and reconnecting may not be required, depending on your PC’s operating system).

After pressing ‘**Enter**’, the bootloader menu will be shown in the terminal window as shown in Figure 22.

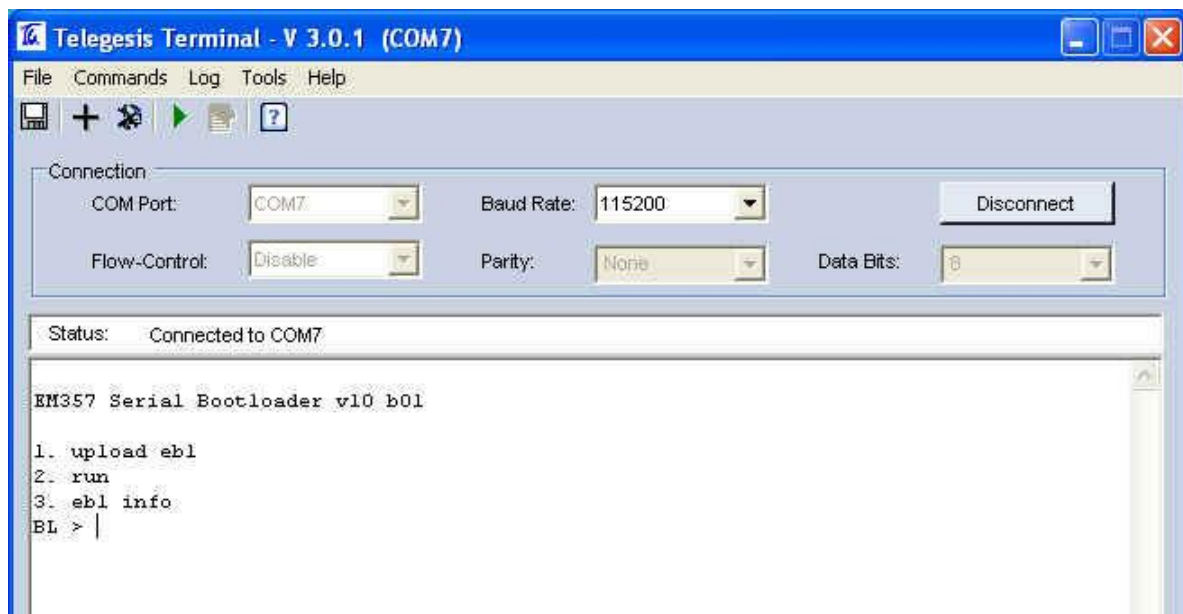


Figure 22. Bootloader Menu

Pressing ‘**1**’ initiates the upload of the new firmware and a number of ‘**C**’ characters will indicate that the ETRX35x is ready to receive data. Within 60 seconds, select **Tools / Transfer File...** and browse for the new firmware file.

Firmware files for the ETRX35x will be in the format ETRX35x\_R3xxC.ebl. After checking that the protocol is set to XMODEM (128 Bytes), press the **Send** button and the new firmware will be downloaded as shown in Figure 23.

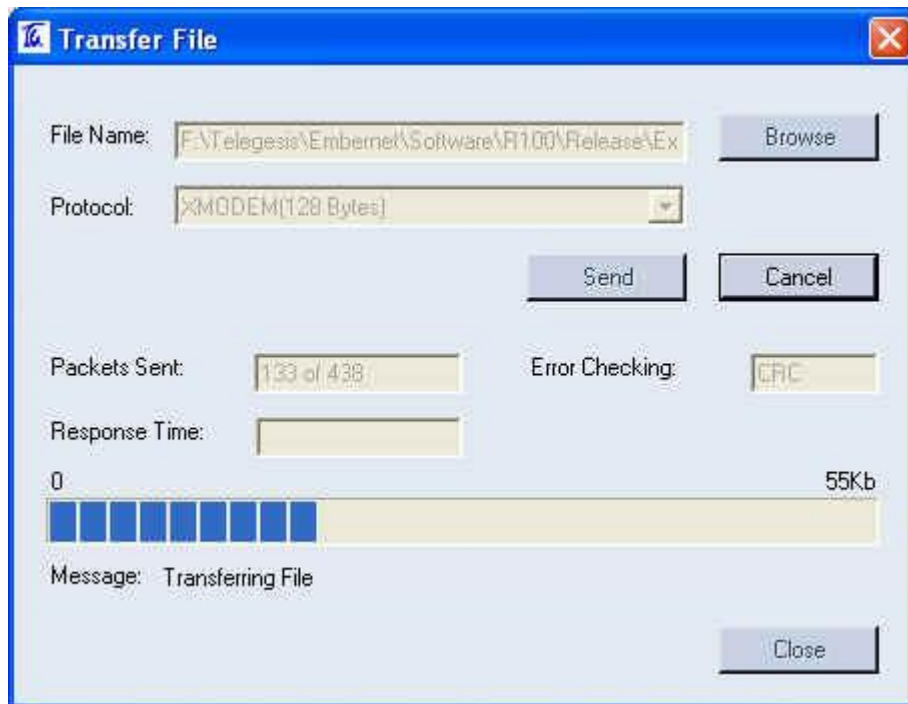


Figure 23. File Transfer Window

When the transfer has been completed successfully, press **Enter** again in order to return to the bootloader menu (shown in figure 10) and option '2' to run the downloaded application software. If the application software has a baudrate other than 115200bps, this will need to be changed to the application baudrate as described above – 19200 baud in the case of R3xx firmware.

## 12.2 Over the Air Firmware Upgrades

### 12.2.1 ETRX2 and ETRX357 features

With the ETRX2, upgrading over the air is possible by cloning a local node's firmware to a remote node, so if new firmware has to be introduced to the network it can be downloaded serially to a master node, which then can clone itself to one node after the other in turn given the target node is only a single hop away. Cloning between some firmware releases may not always be possible, so check the Firmware Revision History for relevant comments.

The ETRX357 uses passthrough OTA bootloading, so the new file is stored in the host processor and loaded on to the target device by transmitting it through the local device. The local device is therefore not required to have the same firmware as the updated target.

The ETRX2 and ETRX357 have different bootloaders and firmware files as well as different modes of transferring the new file. Consequently you cannot upgrade an ETRX357 via an ETRX2 or vice-versa.

### 12.2.2 Cloning an ETRX2

In order to clone the local node's firmware use the command

**AT+CLONE:<EUI64 of the target device>:<Target device's password>**

This will initiate the cloning process given the target node is only a single hop from the local node. After successful cloning the remote node will perform a reset and the local node will continue normal operation without reset.

### 12.2.3 Passthrough with an ETRX357

You should first verify that the module's bootloaders are sufficiently recent, as this feature was not included in the earliest versions. Start the bootloader with the AT+BLOAD command, enter a <cr> and check the response, then exit the bootloader without altering the firmware by selecting option 2 ('run'). The bootloader needs to be version v42 or higher for passthrough to be possible. A new bootloader can only be installed by reflashing the module's memory with an Ember Insight Adaptor.

The passthrough process is similar to bootloading a file on to the local device, except that there is no need to change the baud rate. Use the command

**AT+PASSTHROUGH:<EUI64 of the target device>:<Target device's password>**

The device will respond with a prompt and a sequence of 'C' characters. Select **Tools / Transfer File...** from the drop-down menu of Telegesis Terminal and browse for the new firmware file.

Firmware files for the ETRX35x will be in the format ETRX35x\_R3xxC.ebl. After checking that the protocol is set to XMODEM (128 Bytes), press the **Send** button and the new firmware will be downloaded as shown in Figure 23.

Passthrough bootloading on to an End Device or across multiple hops is not guaranteed to be successful as the network performance cannot be predicted.

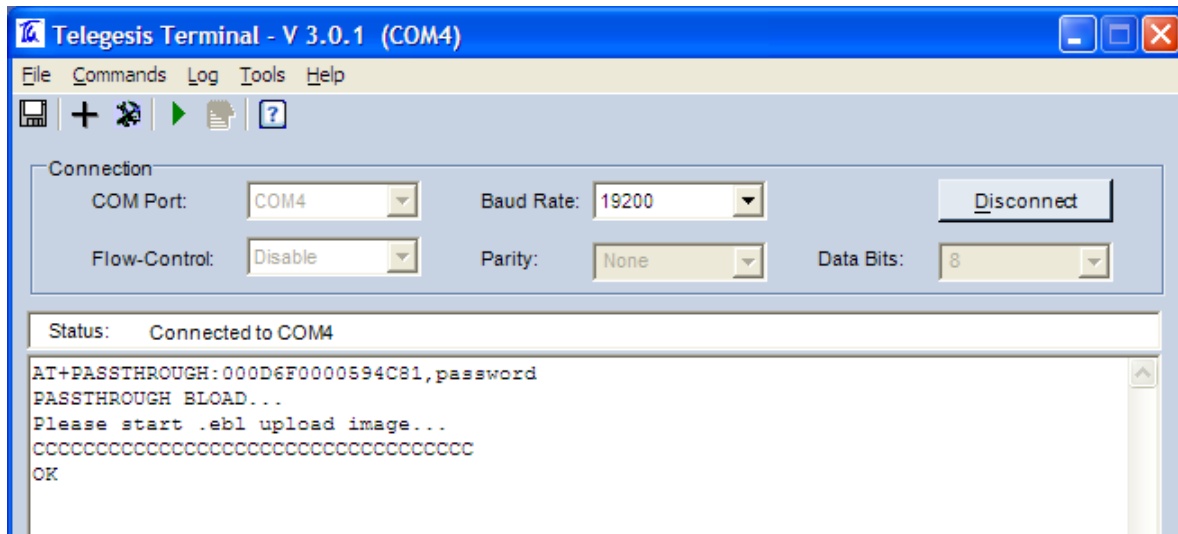


Figure 24. Passthrough bootloading

**12.2.4 Recovering on the default channel**

If the target device is reset or power cycled whilst in the bootloader, the unit will listen for new firmware files on channel 13. It is therefore required to set up a node on channel 13 and repeat the recovery action described in the previous section.

To set up a node on channel 13 use the following commands:

**AT+DASSL** – leave the current network (will show an error if not currently part of a network)

**ATS00=0004** – Only allow channel 13

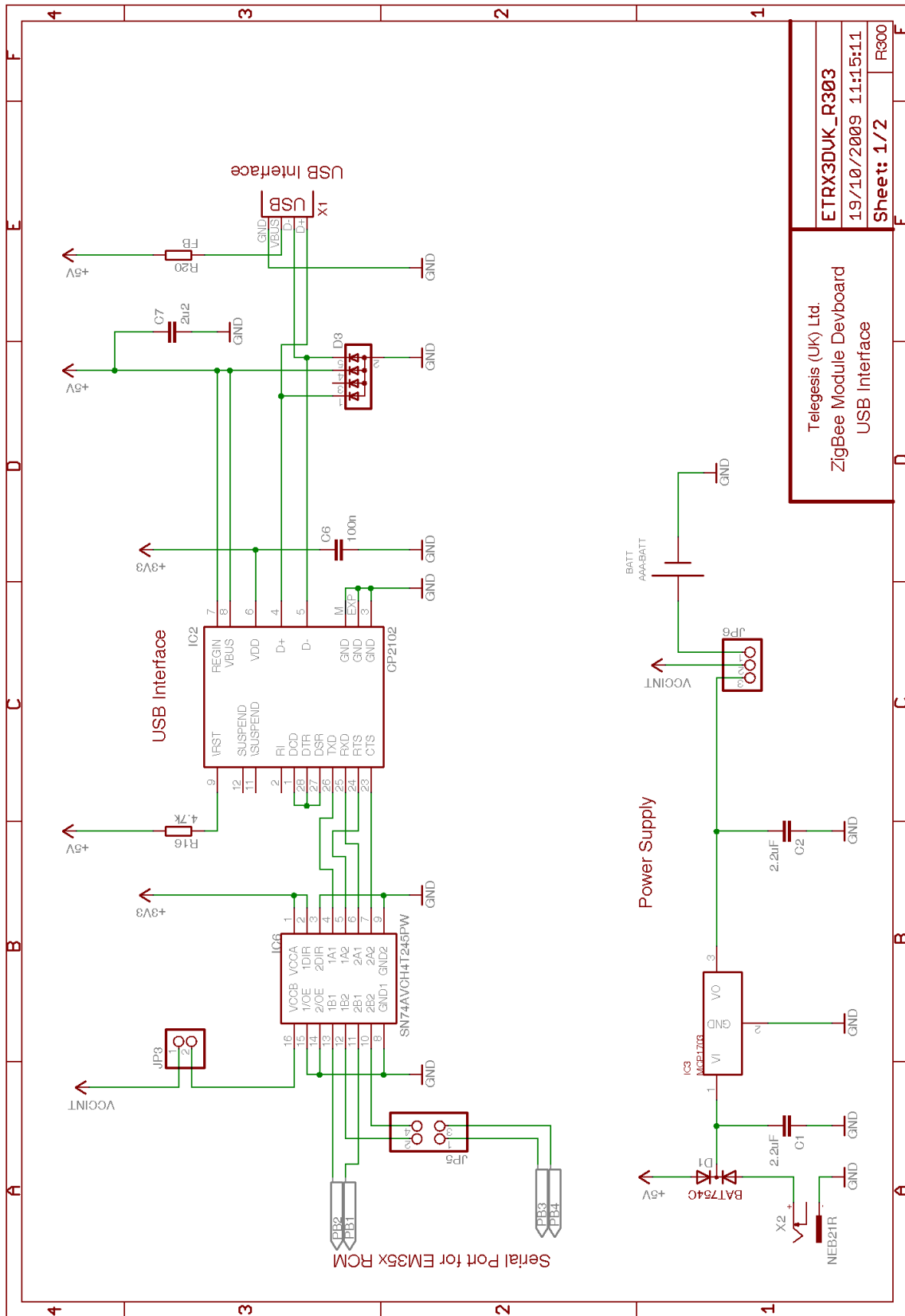
**AT+EN** – establish new network on channel 13

This will cause the local node to become a coordinator on channel 13.

**AT+RECOVER**

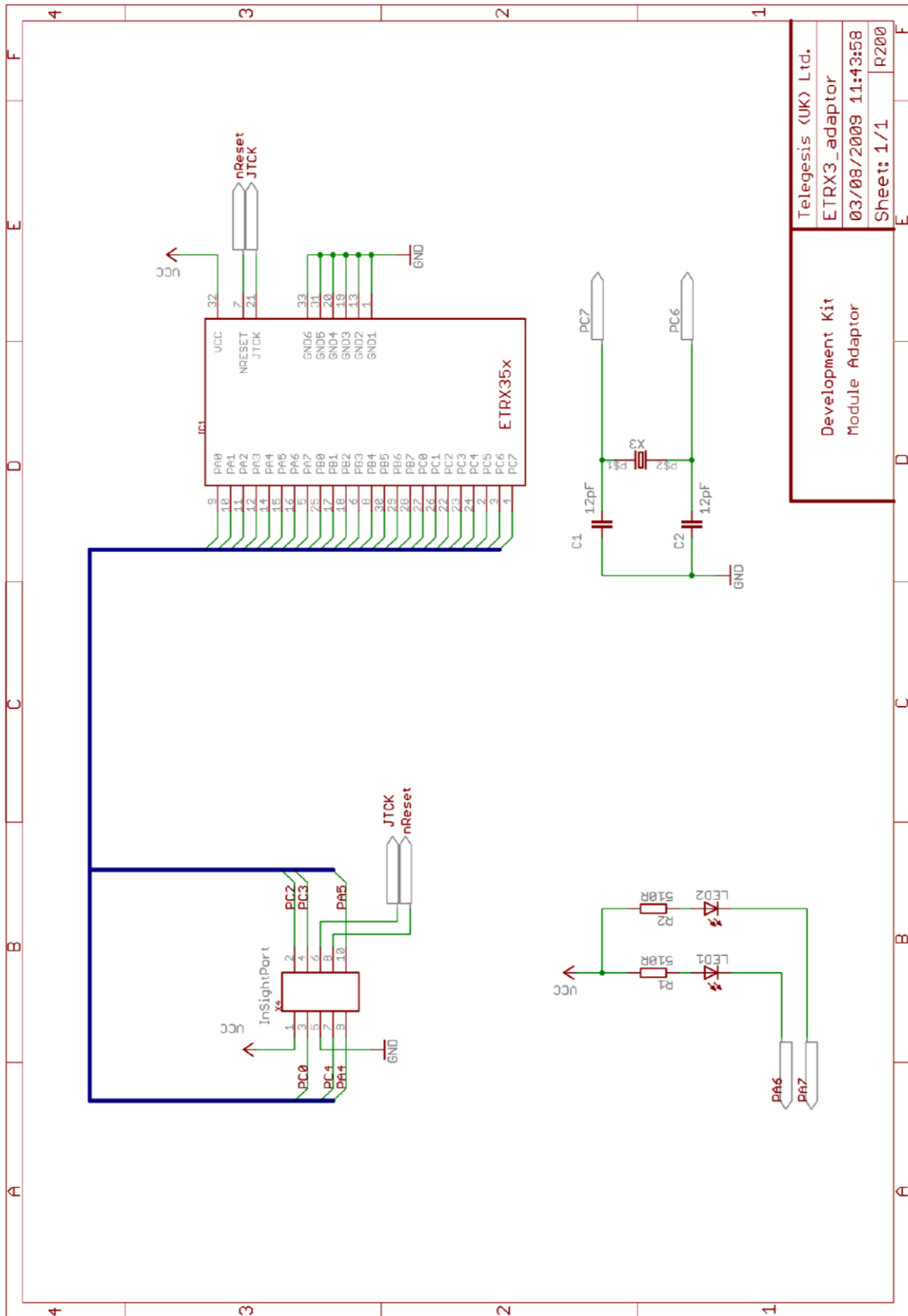
The local node will search for a remote node in bootload mode and clone the local firmware to that remote node.

13 Devboard Schematic





## 14 Carrier Board Schematic



## 15 ETRX35x Ordering Information

Ordering/Product Code	Description
<b>ETRX35x</b>	Telegesis Wireless Mesh Networking Module with Ember ZigBee® Technology: <ul style="list-style-type: none"> <li>• Telegesis AT Style Command Interpreter based on Ember's EmberZNet stack</li> <li>• Integrated 2.4GHz Antenna</li> </ul>
<b>ETRX35xHR</b>	Telegesis Wireless Mesh Networking Module with Ember ZigBee® Technology: <ul style="list-style-type: none"> <li>• Telegesis AT Style Command Interpreter based on Ember's EmberZNet stack</li> <li>• Hirose U.FL Antenna Connector</li> </ul>
<b>ETRX35xDVK</b>	Telegesis Development Kit with: <ul style="list-style-type: none"> <li>• 3 x ETRX35xDV Development Boards</li> <li>• 2 x ETRX35x Modules on Carrier Boards</li> <li>• 2 x ETRX35xHR Modules on Carrier Boards</li> <li>• 2 x ETRX35x-LRS Modules on Carrier Boards</li> <li>• 2 x ETRX35xHR-LRS Modules on Carrier Boards</li> <li>• 1 x ETRX3USB stick</li> <li>• 2 x ½-wave antennae</li> <li>• 2 x ¼-wave antennae</li> <li>• 3 x USB Cables</li> </ul>

**Notes:**

- Customers' PO's must state the Ordering/Product Code.
- There is no "blank" version of the ETRX35x Module available. All Modules carry the Telegesis AT style Command Layer based on the EmberZNet Stack. (Where customers wish to add their own firmware they can re-program the flash memory of the embedded EM35x).
- Please contact Telegesis if you require additional AT style commands or specific integration assistance.

## 16 Trademarks

All trademarks, registered trademarks and products names are the sole property of their respective owners.

## 17 Disclaimer

Product and Company names and logos referenced may either be trademarks or registered trademarks of their respective companies. All information is correct at time of issue. We reserve the right to make modifications and/or improvements to documentation or products without prior notification. Telegesis (UK) Ltd does not convey any license under its patent rights or assume any responsibility for the use of the described product.

## 18 Contact Information

Website: [www.telegesis.com](http://www.telegesis.com)

E-mail [sales@telegesis.com](mailto:sales@telegesis.com)

Telegesis (UK) Limited  
Abbey Barn Business Centre  
Abbey Barn Lane  
High Wycombe  
Bucks  
HP10 9QQ  
UK

Tel: +44 (0)1494 510199

Fax: +44 (0)5603 436999

## 19 References

Telegesis - [www.telegesis.com](http://www.telegesis.com)

Ember - [www.ember.com](http://www.ember.com)