



# USB Driver

---

Installation Guide for EV3 Devices

### USB Driver Installation Guide

For the following devices: MTSMC-EV3-U, MTSMC-EV3-MI-IP/GP, MTD-EV3, MT100UCC-EV3, MTC-EV3-B03, MTC-EV3-B04, MTPCIE-EV3

S000569, Version 1.4

### Copyright

This publication may not be reproduced, in whole or in part, without the specific and express prior written permission signed by an executive officer of Multi-Tech Systems, Inc. All rights reserved. **Copyright © 2013 by Multi-Tech Systems, Inc.**

Multi-Tech Systems, Inc. makes no representations or warranties, whether express, implied or by estoppels, with respect to the content, information, material and recommendations herein and specifically disclaims any implied warranties of merchantability, fitness for any particular purpose and non-infringement.

Multi-Tech Systems, Inc. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Multi-Tech Systems, Inc. to notify any person or organization of such revisions or changes.

### Trademarks

Multi Tech and the Multi-Tech logo are registered trademarks of Multi-Tech Systems, Inc. All other brand and product names are trademarks or registered trademarks of their respective companies.

### Contacting Multi-Tech

#### Knowledge Base

The Knowledge Base provides immediate access to support information and resolutions for all Multi-Tech products. Visit <http://www.multitech.com/kb.go>.

#### Support Portal

To create an account and submit a support case directly to our technical support team, visit: <https://support.multitech.com>.

### Support

Business Hours: M-F, 9am to 5pm CT

Country	By Email	By Phone
Europe, Middle East, Africa:	<a href="mailto:support@multitech.co.uk">support@multitech.co.uk</a>	+(44) 118 959 7774
U.S., Canada, all others:	<a href="mailto:support@multitech.com">support@multitech.com</a>	(800) 972-2439 or (763) 717-5863

### Warranty

To read the warranty statement for your product, visit [www.multitech.com/warranty.go](http://www.multitech.com/warranty.go). For other warranty options, visit [www.multitech.com/es.go](http://www.multitech.com/es.go).

### World Headquarters

Multi-Tech Systems, Inc.

2205 Woodale Drive, Mounds View, MN 55112

Phone: (800) 328-9717 or (763) 785-3500

Fax (763) 785-9874

# Contents

<b>Installing Drivers for Non-UIP EV3 Devices .....</b>	<b>4</b>
Installing on Linux .....	4
Troubleshooting Linux.....	4
Windows Release Notes .....	5
Windows Notes .....	5
Installing on USB Host Powered Devices .....	5
Installing on Non-USB Powered Devices .....	5
Downloading the Windows USB Driver .....	5
Installing on Windows 8, 7 or Vista .....	6
Installing on Windows XP .....	7
Uninstalling Windows Drivers .....	8
Windows 8 .....	8
Windows 7 or Vista .....	8
Windows XP .....	8
<b>Installing UIP Device Drivers .....</b>	<b>9</b>
Downloading and Installing the Windows Universal IP Driver .....	9
<b>Using Linux .....</b>	<b>10</b>
Shell Commands.....	10
Testing Serial Ports.....	10
Create a PPP Connection .....	10
Example.....	10
C Programming.....	11
open().....	11
read().....	12
write().....	12
close().....	13
Test Program() .....	14

# Installing Drivers for Non-UIP EV3 Devices

---

## Installing on Linux

The Linux OS includes a generic USB driver for modems supporting CDC/ACM.

Multi-Tech tested the following Linux operating systems and all used port ttyUSB0. If your system has another device using this port, your port numbers may be different.

- Ubuntu 12.10
- Debian 6.0.6
- Fedora 18

To install the device on any Linux Kernel with CDC/ACM support, connect USB cable from the device to a USB port on your computer. For most recent Linux distributions, there are no drivers to install.

If the operating system recognizes the modem, devices named `/dev/ttyUSBx` are created, for example:

- `/dev/ttyUSB0` Diagnostic port
- `/dev/ttyUSB1` NMEA port
- `/dev/ttyUSB2` Auxiliary port
- `/dev/ttyUSB3` Modem port

**Note:** AT commands are allowed on modem and auxiliary ports.

## Troubleshooting Linux

If Linux does not create devices, check for the kernel module:

```
# lsmod | grep option
```

If entries aren't found, load the kernel module with root privileges:

```
# modprobe option
```

If this returns an error response, such as `# FATAL: Module option not found`, the kernel module is not on your system. You will need to build the driver.

## Building a Linux Driver

If the Linux driver is not recognized by your system, the driver may need to be customized for your device.

**Note:** To avoid runtime loading, build the driver as part of the kernel instead of as a module. This is the recommended configuration for Android devices.

1. Retrieve the appropriate kernel source code version for your system. This should be in your OS distribution package. Unpack/install it.
2. In the root directory open the file: `/drivers/usb/serial/option.c`
3. Check for the existence of the proper `#define` statement. `#define TELIT_PRODUCT_DE910_DUAL 0x1010`
4. If the define statement is missing, add it and then add the following in the `usb_device_id option_ids[]`:  
`USB_DEVICE(TEELIT_VENDOR_ID, TELIT_PRODUCT_DE910_DUAL) }`
5. Save changes and close the file.

6. From the unpacking root directory, type **# make menuconfig**
7. Configure the kernel according to the considered system configuration.
8. Browse to menu **Device Driver > USB Support > USB Serial Converter support** and select **USB driver for GSM and CDMA modems**.
9. To start the build once configured, type **# make**

The kernel module option.ko is in the directory drivers/usb/serial. If the kernel was built previously, compile the module by typing: **# make M=drivers/usb/serial**

To load the module use modprobe or insmod.

## Windows Release Notes

We tested driver on the following Windows operating systems.

- **Windows 8 x86 and x64, Windows 7 x86 and x64, Vista x86 and x64, XP x86 and x64, Windows Server 2012, Windows Server 2008 x86 and x64, and Windows Server 2003 x86**
  - Drivers install correctly, but may require .NET Framework version 3.5 or older.
  - After installing the driver for this device, the device may not be available when Windows comes out of a sleep/hibernate state. To correct this issue, unplug the device from the USB port and then plug it back in to the same port.
- **Windows Server 2003 x64**
  - Not supported with version 8.00.04.

## Windows Notes

### Note:

This driver is not compatible with Windows 2003 x64

## Installing on USB Host Powered Devices

(MTD, MTC, or MT100UCC)

When you connect a USB host powered device to a computer through a USB cable, the Windows **Add New Hardware Wizard** may display **Cannot Install this Hardware**. If this occurs, click **Finish**. Windows detects additional devices and prompts you to install them.

## Installing on Non-USB Powered Devices

(MTSMC, MTPCIE)

Turn on the device and wait 15 seconds before connecting the USB cable. If you connect the USB cable before supplying power to the device, the Windows **Add New Hardware Wizard** may appear and show **Cannot Install this Hardware**. If this occurs, click **Finish**. Windows detects additional devices and prompts you to install the additional devices. If Windows does not detect new device, unplug the USB cable, turn the device off and on, wait 15 seconds, insert the USB cable, and install devices when prompted.

## Downloading the Windows USB Driver

If you haven't downloaded the driver:

1. Go to the Multi-Tech Support page, [www.multitech.com/support.go](http://www.multitech.com/support.go) and select your product from the Product Families drop down list.
2. Click **Drivers**.
3. Select **ev3-u\_windriver\_8.00.04.zip** and **Save** the driver to your computer.
4. Extract the files to your computer.

## Installing on Windows 8, 7 or Vista

This process installs multiple drivers and ports.

**Note:** If you previously installed USB drivers for this device, uninstall them before installing or re-installing this driver. Uninstall all existing drivers for this device. Refer to Uninstall Windows Drivers for details.

Before you connect the device (disconnect the device if you connected it):

**CAUTION:** If you connected the device before installing the drivers, Windows may install drivers automatically. Your device may not operate correctly with these drivers. Uninstall the drivers before proceeding. See Remove Microsoft Installed Drivers for details.

1. Go to the location where you extracted the driver and open the **EV3-USB\Driver** folder.
2. Right-click on **TelitUSBInstaller\_Qc\_U8.00.04.exe** and select **Run as Administrator**.
3. Click **Yes** or **Allow** to allow the installer to make changes to your computer.
4. Click **Next** and follow the instructions in the installation wizard.
5. Click the **Install** option when prompted, for example, Install this driver software anyway.
6. Click **Finish**.
7. Connect USB cable from the device to a USB port on your computer. Windows indicates when the device is ready to use.

## Installing on Windows XP

This process installs two drivers.

**Note:** If you previously installed USB drivers for this device, uninstall them before installing or re-installing this driver. Uninstall all existing drivers for this device. Refer to Uninstall Windows Drivers for details.

Before you connect the device (disconnect the device if you connected it):

1. Go to the location where you extracted the driver and open the **EV3-USB\Driver** folder.
2. Right-click on **TelitUSBInstaller\_Qc\_U8.00.04.exe** and select **Run**.
3. Click **Next** and follow the instructions in the installation wizard.
4. Click **Continue Anyway** each time this screen appears.



5. Click **Finish**.
6. Connect USB cable from the device to a USB port on your computer. After it detects the hardware, Windows opens the New Hardware Wizard.
7. Select **No, not this time** and click **Next**.
8. If Software Installation-Windows Logo testing message appears, click **Continue Anyway**.
9. Select **Install the software automatically (Recommended)** and click **Next**.
10. Select **Finish**.
11. Repeat Steps 7-10 for each additional New Hardware Wizard. Windows indicates when the device is ready to use.

## Uninstalling Windows Drivers

**Note:** Disconnect the device before uninstalling drivers.

### Windows 8

To uninstall drivers from Windows 8:

1. Open Windows **Programs and Features**.
2. Uninstall all **Telit modems, ports, and USB drivers**.

### Windows 7 or Vista

To uninstall drivers from Windows 7 or Vista:

1. Open **Programs and Features** from the Windows Control Panel.
2. Uninstall the **Windows Driver Package – Telit Wireless Solutions (telitusbser) Modem**.
3. Uninstall all **Telit modems, ports, and USB**.

### Windows XP

To uninstall drivers from Windows XP:

1. Open the **Control Panel** and go to **Add or Remove Programs**.
2. Uninstall **Windows Driver Package – Telit Wireless Solutions (telitusbser) Modem**.
3. Uninstall all other **Telit modems, Ports and USB**.



# Installing UIP Device Drivers

---

## Downloading and Installing the Windows Universal IP Driver

Use this driver with MTSMC-EV3-MI-IP/GP devices.

Before you connect the device (disconnect the device if you connected it), download and install the Universal IP driver. If you already have the driver, skip to Step 5.

1. Go to the Multi-Tech Support page, [www.multitech.com/support.go](http://www.multitech.com/support.go).
2. Select your product from the Product Families drop down list.
3. Click **Drivers**.
4. Select and save the driver to your computer:
  - For x86 operating systems, select **VCP\_v1.3.1\_Setup.exe**
  - For x64 operating systems, select **VCP\_v1.3.1\_Setup\_x64.exe**
5. Double-click the **.EXE** file to start installation.
6. Click **Next** twice.
7. Click **Finish**.
8. Connect USB cable from the device to a USB port on your computer.

Windows indicates when the device is ready to use.

# Using Linux

---

## Shell Commands

### Testing Serial Ports

To test the serial ports created by the driver, type in a shell:

```
cat /dev/ttyUSBx &
# echo -en "ATE0\r" > /dev/ttyUSBx
# echo -en "AT\r" > /dev/ttyUSBx
```

**Note:** Sending ATE0 is required, to avoid issues in the terminal output. It prevents the sending/receiving spurious characters to/from the modem when used with the Linux commands “echo” and “cat”

### Create a PPP Connection

Most recent Linux distributions have GUI tools for creating PPP connections; the following instructions are for creating a PPP connection through command line interface.

PPP support must be compiled into the kernel; pppd and chat programs are also required.

pppd needs two scripts: the first script performs the environment setting and calls the second script, which is used by the chat program. For creating a PPP connection type:

```
# pppd file /etc/pppd_script &
```

### Example

```
# Debug info from pppd
debug
#kdebug 4
# Most phones don't reply to LCP echos
lcp-echo-failure 3
lcp-echo-interval 3
# Keep pppd attached to the terminal
# Comment this to get daemon mode pppd
nodetach
# The chat script (be sure to edit that file, too!)
connect "/usr/sbin/chat -v -f /etc/chatscripts/evdo_connect"
# Serial Device to which the modem is connected
/dev/ttyUSB3
# Serial port line speed
115200
dump
# The phone is not required to authenticate
#noauth
user <insert here the correct username for authentication>
name <insert here the name of the connection>
password <insert here the correct password for authentication>
# If you want to use the EV-DO link as your gateway
defaultroute
```

```
# pppd must not propose any IP address to the peer
#noipdefault
ipcp-accept-local
ipcp-accept-remote
# Keep modem up even if connection fails
#persist
# Hardware flow control
crtcts
# Ask the peer for up to 2 DNS server addresses
usepeerdns
# No ppp compression
novj
nobsdcomp
novjccomp
nopcomp
noaccomp
# For sanity, keep a lock on the serial line
lock
# Show password in debug messages
show-password
```

This script calls the option *connect* using the script *evdo\_connect*, for example: After launching a PPP connection is possible to use ftp protocol or other utilities that allow the access to the Internet.

## C Programming

The following topics show all the functions that can be used from C source code to perform read/write operations on the serial devices.

### open()

The *open()* function shall establish the connection between a file and a file descriptor. The file descriptor is used by other I/O functions to refer to that file.

#### Header File

fcntl.h

#### Prototype:

```
int open(const char *pathname, int flags)
```

#### Parameters:

pathname – file name with its own path.

flags – is an *int* specifying file opening mode: is one of O\_RDONLY, O\_WRONLY or O\_RDWR which request opening the file read-only, write-only or read/write, respectively.

#### Returns:

The new file descriptor *fildes* if successful, -1 otherwise.

### Example

Open the `/dev/ttyUSBx`.

```
int fd; // file descriptor for the /dev/ttyUSBx entry
if((fd = open("/dev/ttyUSBx", O_RDONLY) < 0)
{
/* Error Management Routine */
} else {
/* ttyUSBx Device Opened */
}
{
```

### read()

The `read()` function reads *nbyte* bytes from the file associated with the open file descriptor, *filides*, and copies them in the buffer that is pointed to by *buf*.

#### Header File

unistd.h

#### Prototype:

```
ssize_t read(int fildes, void *buf, size_t nbyte)
```

#### Parameters:

fildes - file descriptor

buf - destination buffer pointer

nbyte - number of bytes that read() attempts to read

#### Returns:

The number of bytes actually read if the operation is completed successfully, otherwise it is -1.

### Example

Read `sizeof(read_buff)` bytes from the file associated with *fd* and stores them into *read\_buff*.

```
char read_buff[BUFF_LEN];
if(read(fd, read_buff, sizeof(read_buff)) < 0)
{
/* Error Management Routine */
} else {
/* Value Read */
}
```

### write()

The `write()` function writes *nbyte* bytes from the buffer that are pointed by *buf* to the file associated with the open file descriptor. *filides*.

#### Header File

unistd.h

**Prototype:**

```
ssize_t write(int fildes, const void *buf, size_t nbyte)
```

**Parameters:**

fildes – file descriptor

buf – destination buffer pointer

nbyte – number of bytes that write() attempts to write

**Returns:**

The number of bytes actually written if the operation is completed successfully, otherwise it is -1.

**Example**

Write *strlen(value\_to\_be\_written)* bytes from the buffer pointed by *value\_to\_be\_written* to the file associated with the open file descriptor, *fd*.

```
char value_to_be_written[] = "dummy_write";
if (write(fd, value_to_be_written, strlen(value_to_be_written)) < 0)
{
/* Error Management Routine */
} else {
/* Value Written */
}
```

**close()**

The *close()* function shall deallocate the file descriptor indicated by *fildes*. To deallocate means to make the file descriptor available for return by subsequent calls to *open()* or other functions that allocate file descriptors.

**Header File**

unistd.h

**Prototype:**

```
int close(int fildes);
```

**Parameters:**

fildes - file descriptor

**Returns:**

0 if successful, otherwise it is -1.

**Example**

Close the ttyUSBx file.

```
if(close(fd) < 0)
{
/* Error Management Routine */
} else {
/* File Closed */
}
```

## Test Program()

The following simple C program is useful to test the modem issuing an AT command. The program opens the /dev/ttyUSB0 interface and calls the write() and the read() function to send an AT command and receive the subsequent output.

```
#include <stdio.h> /* Standard input/output definitions */
#include <string.h> /* String function definitions */
#include <unistd.h> /* UNIX standard function definitions */
#include <fcntl.h> /* File control definitions */
#include <errno.h> /* Error number definitions */
#include <termios.h> /* POSIX terminal control definitions */
#define USB "/dev/ttyUSB0"
#define BUFSIZE 1000
#define BAUDRATE B115200
int open_port(char *port)
{
    struct termios options;
    int fd;
    fd = open(port, O_RDWR | O_NOCTTY | O_NDELAY);
    if (fd == -1)
    {
        printf("open_port: Unable to open the port - ");
    }
    else
    {
        printf ( "Port %s with file descriptor=%i",port, fd);
        fcntl(fd, F_SETFL, FNDELAY);
        tcgetattr( fd, &options );
        cfsetispeed( &options, BAUDRATE );
        cfsetospeed( &options, BAUDRATE );
        options.c_cflag |= ( CLOCAL | CREAD);
        options.c_cflag &= ~(CSIZE | PARENB | CSTOPB | CSIZE);
        options.c_cflag |= CS8;
        options.c_cflag &= ~CRTSCTS;
        options.c_lflag &= ~(ICANON | ECHO | ECHOE | ISIG);
        options.c_iflag &= ~(IXON | IXOFF | IXANY | ICRNL | INLCR | IGNCR);
        options.c_oflag &= ~OPOST;
        if ( tcsetattr( fd, TCSANOW, &options ) == -1 )
            printf ("Error with tcsetattr = %s\n", strerror ( errno ) );
        else
            printf ( "%s\n", "succeed" );
    }
    return (fd);
}
int main()
{
    int serialFD = open_port(USB);
    char buf[BUFSIZE];
```

```
memset(buf,0,BUFSIZE);
write(serialFD, "AT\r" , strlen("AT\r"));
sleep(1);
read( serialFD, buf, BUFSIZE );
printf("The string is: %s\n", buf);
close(serialFD);
return 0;
}
```

The sleep instruction is required because the modem response after issuing the AT command is not immediate, so you need to wait a bit before reading. There are more efficient ways to do this, for example, you can put the read call in a while loop and exit when the read buffer contains a certain string.