# FarSync® Linux® Driver Installation Guide

FarSync PCIe

FarSync Flex

# Table of Contents

# 1 Reference Documentation

## 1.1 Quick Start Guides

[1] FarSync TxUe
[2] FarSync T4Ee
[3] FarSync K2Ee
[4] FarSync Flex

## 1.2 Application Interface

[5] FarSync® Linux® SDK User Guide

# 2 Getting Started

The FarSync Drivers for Linux software support the FarSync range of PCI Express cards and USB devices that provide connectivity with synchronous serial data communications lines.

The drivers support programmatic access to data streams to and from the communications line(s) connected to the FarSync card/device's port(s) i.e. they enable a user written application to open the ports and to receive and transmit data from/to them. The separately purchased FarSync SDK product provides the tools, documentation and sample programs required to develop such an application – please refer to **FarSync® Linux® SDK User Guide** [5], supplied with the SDK, for further details.

Installation of the FarSync products is carried out using the following 3 stages:

1. Install the hardware (one or more card/device) – see Section 3
2. Install the FarSync driver software – see Section 4
3. *Conditionally install any additional software required for the application that will be using the FarSync port. Note that this can be software that has been produced by FarSite customers using the FarSync SDK.*

This manual covers steps 1 and 2 which should enable you to install the hardware and driver for your FarSync product.

# 3 Hardware Installation

Note that specific information on installing your particular type of FarSync hardware is contained in the Quick Start Guides referenced below.

## 3.1 PCI Express (PCIe)

- FarSync TxUe [1]
- FarSync T4Ee [2]
- FarSync K2Ee [3]

### 3.1.1 General Precautions

The PCIe cards should be treated as static sensitive. Discharge any static electricity by touching the computer chassis or using an appropriate anti-static wrist strap before handling.

The PCIe cards are not hot pluggable in a PCIe slot on a standard PC or PC Server, and therefore you should turn the PC off before installing the cards or changing any jumper settings. If your system has defined hot plug slots, protected by isolation switches then they may be installed according to the methods described in your system documentation. It may still require a system restart to allow software installation to properly complete.

It is recommended you leave the system turned off whilst connecting all external cables and peripherals to a PCIe card.

The PCIe system uses a jumper-less automatic configuration system and the card can be simply installed in an empty PCIe slot. The card should be secured using the bracket retaining screw.

The FarSync PCI Express cards may be installed in any PCI Express slot.

Any issues regarding the operation of PCIe cards in a particular PC can be resolved by changing settings in the BIOS with or without the card installed. Each BIOS differs in the parameters available and their naming. Refer to your system documentation for more information.

### 3.1.2 Card Ordering

At system startup the network interfaces are assigned to the cards in PCIe slot ordering. i.e. The card in the lowest numbered slot will be assigned interfaces sync0 and sync1 (plus sync2 and sync3 for a four port card). A card in the next lowest slot will get the next available network interfaces and so on.

A problem with this scheme is that if a card is later added at a lower slot number than an existing card the interface assignment on all subsequent PCIe slots will be shifted up. This situation can also occur if a PCIe card is added to a system where the hardware assigns PCIe slots before PCI slots.

To avoid this shifting it is recommended that whenever cards are installed they are installed in the lowest numbered slot available, leaving higher numbered slots available for future use.

## 3.2 USB

- FarSync Flex [4]

There are no special handling precautions necessary when installing the FarSync Flex USB devices.

The USB devices are hot pluggable. Plug the "A" end of the supplied USB cable into a suitable USB port on the PC. Fix the "B" end to the USB device and secure the cable to the device with the fixing screws.

### 3.2.1 Device Ordering

The installation scripts will arrange that on boot up the FarSync PCIe driver is started first, followed by the Flex USB driver. This ensures that all PCIe devices are allocated device names first, so that USB devices can be connected and disconnected as required.

## 3.3 Cable installation

Various cable assemblies are available for the different cards and USB devices depending on the desired configuration. However the following rules apply to all the assemblies.

Cables should be screwed firmly to the cards/devices using the supplied connector locking screws and posts.

The cables should be secured so that the weight of any additional cables and connectors does not fall on the PCs card mounting bracket.

Don't kink or bend cables so as to stress the PC mounting bracket.

See the card/device-specific Quick Start Guide for more details.

# 4 Driver Installation

The FarSync Drivers for Linux are installed using a simple script. Since the drivers need to be actually built for the target platform, the script will carry out this building process as part of the installation sequence. In order to do this, there are some prerequisites that are detailed in Section 4.1.

## 4.1 Pre-Installation Requirements

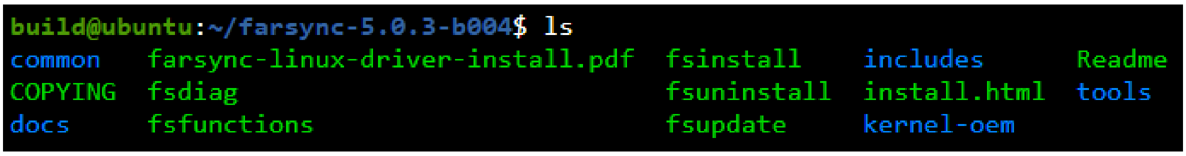
The following prerequisites are necessary for the installation of the device drivers and supporting software:

### 4.1.1 Start a terminal session on the target machine

This terminal session is then used to setup any required tools and to run the install script.

### 4.1.2 Copy the installation files

Extract the contents of the release tar.gz file into a local directory of the target machine, retaining the supplied directory structure
e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ ls
common   farsync-linux-driver-install.pdf  fsinstall    includes      Readme
COPYING  fsdiag                            fsuninstall  install.html  tools
docs     fsfunctions                       fsupdate     kernel-oem
```

### 4.1.3 Build Tools

The install script requires access to Kernel development tools including:

- C compiler
- Make system

If these are not already present, they must be installed prior to running the FarSync driver install script.

How the tools are obtained/installed will depend on the particular Linux distribution that you are using. Some current examples [Mar22] are given below:

#### *4.1.3.1 Ubuntu [20.04.4 LTS - 5.13.0-35]:*

- sudo apt update
- sudo apt install build-essential

#### *4.1.3.2 RedHat [8.4 - 4.18.0-305]*

- sudo yum install kernel-devel
- sudo yum install patch
- sudo yum install gcc
- sudo yum install make
- sudo yum install elfutils-libelf-devel
- sudo yum install libncurses*  [i.e. all variants of the lib will be installed]

#### *4.1.3.3 openSUSE [Leap 15.3 - 5.3.18-150300]*

- sudo zypper install -t pattern devel_basis
- sudo zypper se -t pattern
- sudo zypper install -t pattern devel_kernel
- sudo zypper install insserv-compat
- sudo zypper in libncurses5

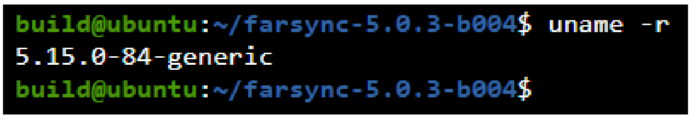
### 4.1.4 Kernel Header Reference

Since the building of the drivers requires access to the kernel headers specific to the target platform, you need to ensure you have a symbolic link called **/usr/src/linux** which points to where your Kernel Source tree or Kernel Header files are located

e.g.

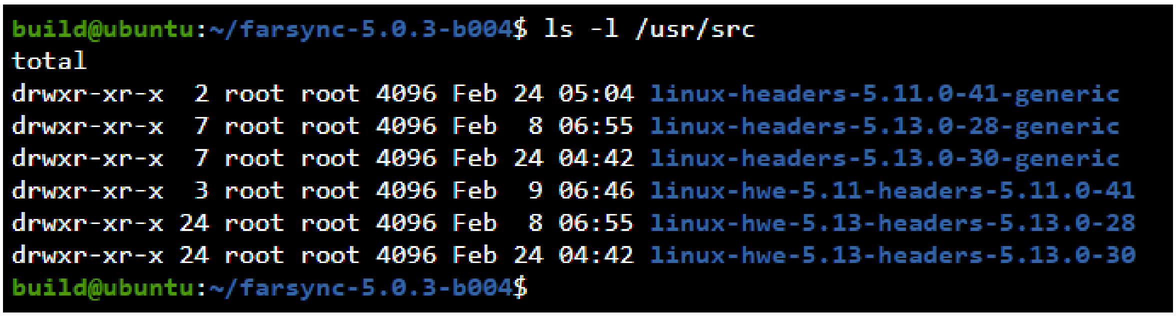
Determine the kernel release that you are using by running the **uname** command:

**uname -r**



```
build@ubuntu:~/farsync-5.0.3-b004$ uname -r
5.15.0-84-generic
build@ubuntu:~/farsync-5.0.3-b004$
```

Locate the available kernel headers:

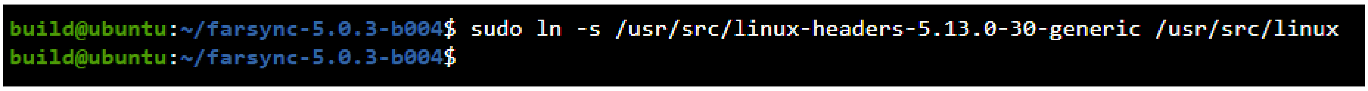


```
build@ubuntu:~/farsync-5.0.3-b004$ ls -l /usr/src
total
drwxr-xr-x  2 root root 4096 Feb 24 05:04 linux-headers-5.11.0-41-generic
drwxr-xr-x  7 root root 4096 Feb  8 06:55 linux-headers-5.13.0-28-generic
drwxr-xr-x  7 root root 4096 Feb 24 04:42 linux-headers-5.13.0-30-generic
drwxr-xr-x  3 root root 4096 Feb  9 06:46 linux-hwe-5.11-headers-5.11.0-41
drwxr-xr-x 24 root root 4096 Feb  8 06:55 linux-hwe-5.13-headers-5.13.0-28
drwxr-xr-x 24 root root 4096 Feb 24 04:42 linux-hwe-5.13-headers-5.13.0-30
build@ubuntu:~/farsync-5.0.3-b004$
```

If there is not already a soft link, /usr/src/linux[1] pointing to the corresponding headers for use by the script, you should create one now using the **ln** command:

**sudo ln -s /usr/src/[target-folder-name] /usr/src/linux**

e.g. **sudo ln -s /usr/src/linux-headers-`uname -r` /usr/src/linux**



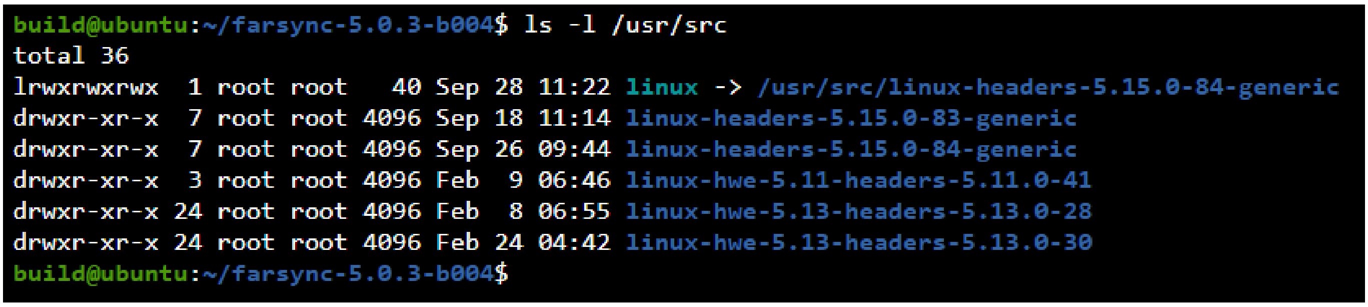
```
build@ubuntu:~/farsync-5.0.3-b004$ sudo ln -s /usr/src/linux-headers-5.13.0-30-generic /usr/src/linux
build@ubuntu:~/farsync-5.0.3-b004$
```

And check that the link has been created successfully with the **ls** command:

---

[1] In some distributions of Linux this link may already exist

**ls -l /usr/src**

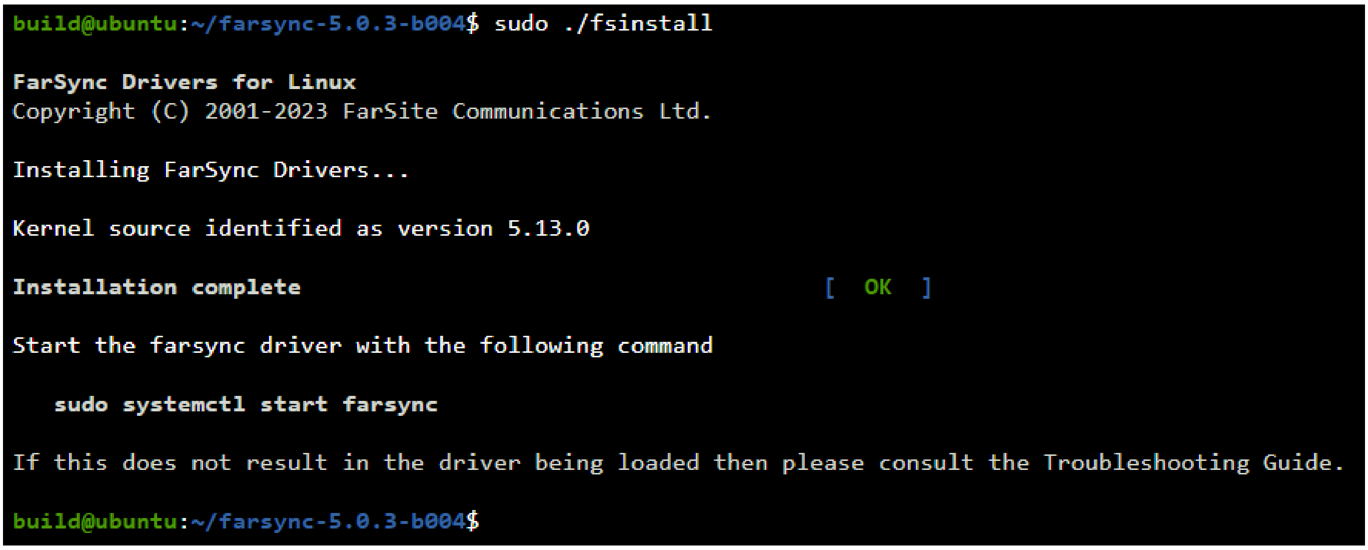


```
build@ubuntu:~/farsync-5.0.3-b004$ ls -l /usr/src
total 36
lrwxrwxrwx  1 root root   40 Sep 28 11:22 linux -> /usr/src/linux-headers-5.15.0-84-generic
drwxr-xr-x  7 root root 4096 Sep 18 11:14 linux-headers-5.15.0-83-generic
drwxr-xr-x  7 root root 4096 Sep 26 09:44 linux-headers-5.15.0-84-generic
drwxr-xr-x  3 root root 4096 Feb  9 06:46 linux-hwe-5.11-headers-5.11.0-41
drwxr-xr-x 24 root root 4096 Feb  8 06:55 linux-hwe-5.13-headers-5.13.0-28
drwxr-xr-x 24 root root 4096 Feb 24 04:42 linux-hwe-5.13-headers-5.13.0-30
build@ubuntu:~/farsync-5.0.3-b004$
```

## 4.2 Running the Install Script (fsinstall)

As root, run the fsinstall script from the top level directory created when extract the install files from the tar.gz (see Section 4.1.2):

**sudo ./fsinstall**



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo ./fsinstall

FarSync Drivers for Linux
Copyright (C) 2001-2023 FarSite Communications Ltd.

Installing FarSync Drivers...

Kernel source identified as version 5.13.0

Installation complete                                      [  OK  ]

Start the farsync driver with the following command

   sudo systemctl start farsync

If this does not result in the driver being loaded then please consult the Troubleshooting Guide.

build@ubuntu:~/farsync-5.0.3-b004$
```

The installation script will detect the Kernel version using the /usr/src/linux link as setup in Section 4.1.4, and perform some other checks before installing the software.

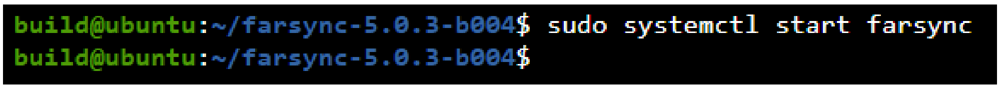
The support utilities will be added to **/sbin** and support/configuration files placed in **/etc/farsite/farsync**.

In the case of any warning or errors being reported during the running of the install script, please consult the Troubleshooting appendices in Section A9

## 4.3 Start the Drivers

The next step is to start the drivers as follows:

**sudo systemctl start farsync**



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo systemctl start farsync
build@ubuntu:~/farsync-5.0.3-b004$
```

Once the start request has completed you can run one or more of the following commands:

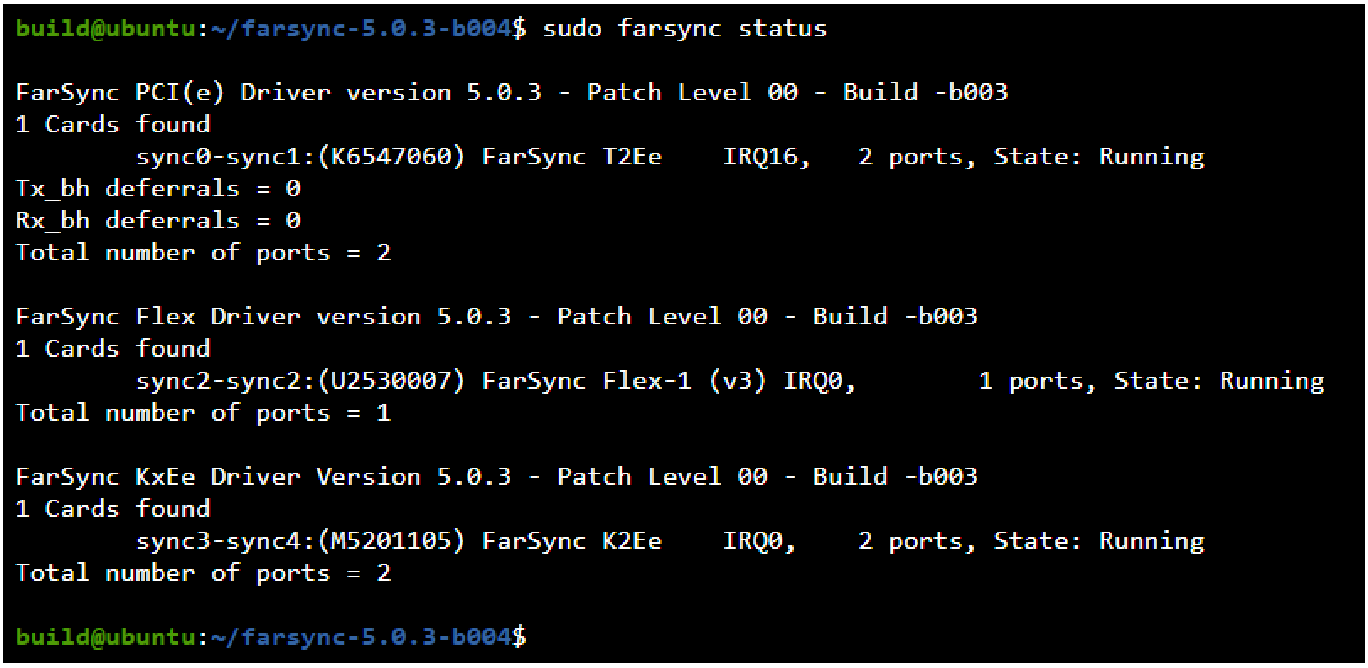
| Device type | Driver name | Linux command to view status |
|---|---|---|
| FarSync T-Series cards (FarSync T2Ue, T4Ue, T2Ee) | farsync | **cat /proc/farsync** |
| FarSync USB devices (FarSync Flex) | fsflex | **cat /proc/fsflex** |
| FarSync K-Series card (FarSync K2Ee) | farsynck | **cat /proc/farsynck** |

to check that the card/device-type-specific interfaces are running.

You can also use the following command to see the status of all three drivers, but if you only have one device type then only one of the drivers will provide useful information:

**sudo farsync status**

to view them all:
e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farsync status

FarSync PCI(e) Driver version 5.0.3 - Patch Level 00 - Build -b003
1 Cards found
        sync0-sync1:(K6547060) FarSync T2Ee    IRQ16,   2 ports, State: Running
Tx_bh deferrals = 0
Rx_bh deferrals = 0
Total number of ports = 2

FarSync Flex Driver version 5.0.3 - Patch Level 00 - Build -b003
1 Cards found
        sync2-sync2:(U2530007) FarSync Flex-1 (v3) IRQ0,        1 ports, State: Running
Total number of ports = 1

FarSync KxEe Driver Version 5.0.3 - Patch Level 00 - Build -b003
1 Cards found
        sync3-sync4:(M5201105) FarSync K2Ee    IRQ0,    2 ports, State: Running
Total number of ports = 2

build@ubuntu:~/farsync-5.0.3-b004$
```

## 4.4 Reboot (Optional)

As a final test of the installation, you could reboot the system to ensure that the drivers are loaded when the system starts up.  If not already done this is a good time to power off and install the hardware.

As the system boots look for and check the farsync, fsflex and farsynck driver startup messages
e.g.

```
[    1.730680] usb 3-1: New USB device found, idVendor=1afd, idProduct=7621, bcdDevice= 2.00
[    1.730691] usb 3-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[    1.730697] usb 3-1: Product: FarSync K2Ee USB 2.0
[    1.730722] usb 3-1: Manufacturer: FarSite Communications
[    1.730727] usb 3-1: SerialNumber: M5201105

[    2.016089] usb 1-9: New USB device found, idVendor=1afd, idProduct=0721, bcdDevice= 2.00
[    2.016100] usb 1-9: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[    2.016105] usb 1-9: Product: FarSync Flex USB 2.0
[    2.016109] usb 1-9: Manufacturer: FarSite Communications
[    2.016112] usb 1-9: SerialNumber: U2530007

[    3.042510] FarSync OEM driver 4.0.0-00-64bit-b400 (c) 2001-2019 FarSite Communications Ltd.

[    3.060791] sync0-sync1: (K6547060) FarSync T2Ee     IRQ16, 2 ports
[    3.060828] fst: fst_min_dma_len set to 64
[    3.060829] fst: fst_dmathr set to dd00dd00
[    3.060830] fst: fst_iocinfo_version set to 5
[    3.060831] fst: fst_allow_ioctl set to 0

[    5.077035] Flex USB OEM driver 5.0.1-00-64bit-b500
           (C) 2001-2022 FarSite Communications Ltd.
[    5.077713] sync2-sync2: (U2530007) FarSync Flex-1 (v3), 1 ports
[    5.077752] USB flex device now attached to interface minor 1
[    5.078280] usbcore: registered new interface driver fsflex
[    5.078282] fst: fst_iocinfo_version set to 5
[    5.078283] fst: fst_allow_ioctl set to 0

[    5.082599] FarSync KxEe Driver 5.0.1-00-64bit-b000
           Copyright (C) 2001-2022 FarSite Communications Ltd.
[    5.082628] fst: Interval value on int endpoint not as expected 4
[    5.086580] sync3-sync4: (M5201105) FarSync K2Ee    , 2 ports
[    5.086907] KxEe device now attached to interface minor 2
[    5.090607] usbcore: registered new interface driver farsynck
[    5.090609] fst: fst_iocinfo_version set to 5
[    5.090609] fst: fst_allow_ioctl set to 0

[    7.133861] fst: 0: Card initialisation OK
```

This output can be viewed, after the system has booted, by running

```
dmesg -T
```

You can confirm that the drivers have found all the FarSync card/devices and check the serial numbers by looking in the /proc/farsync, /proc/fsflex and /proc/farsynck files as detailed above.

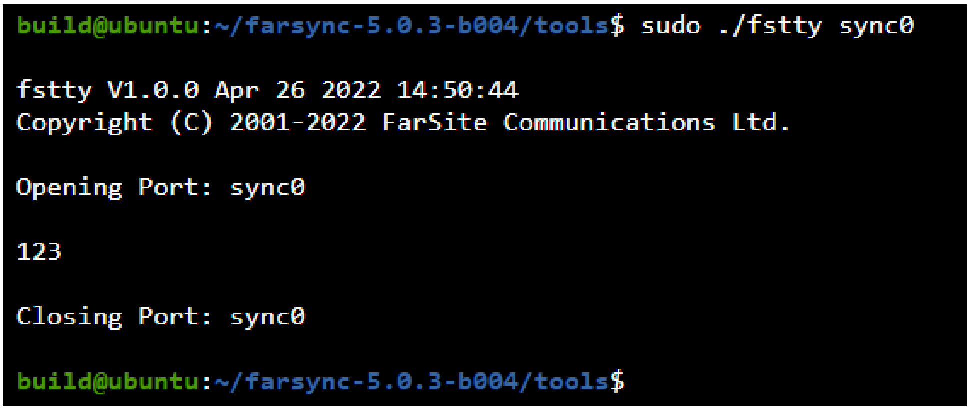
# 5 Verification

## 5.1 fstty

Once the driver(s) have been installed, the operation of the FarSync device/card(s) and their associated drivers can be verified using the supplied **fstty** utility.

**fstty** is installed into /sbin as part of the FarSync install sequence.

When running the application, you can **optionally** specify the logical device associated with the required FarSync port to be used e.g. sync0. By default, if the parameter is not specified, the value of sync0 shall be used.
e.g.



```
build@ubuntu:~/farsync-5.0.3-b004/tools$ sudo ./fstty sync0

fstty V1.0.0 Apr 26 2022 14:50:44
Copyright (C) 2001-2022 FarSite Communications Ltd.

Opening Port: sync0

123

Closing Port: sync0

build@ubuntu:~/farsync-5.0.3-b004/tools$
```

**fstty** supports the following parameters:

**fstty [sync<X>]**

where:

**<X>** is the device index of the target FarSync port e.g. sync0

**fstty** will act as a simple terminal attached to the FarSync port, displaying characters that are received and sending any characters that are typed at the keyboard. **fstty** is typically used with a loopback plug attached to the FarSync port being verified. If the port is being used in a synchronous mode when using a loopback plug then clocking would normally need to be configured as internal.

- ⇨ Be sure to set the port's configuration prior to running **fstty**, using the **farutil** application – see Section 6.2.1

# 6 Configuration

The ports of the FarSync card/devices can be configured either

- manually using the **farutil** application
  OR
- using a custom application (as developed using the FarSync SDK)

Note that, by default, all configuration commands require root privileges. See Section 7 for details of how to configure access to the FarSync ports.

## 6.1 Port Names

Each card has one, two or four ports. Each port has a logical system “device” associated with it, each with their own system-wide, unique name. This allows applications to programmatically access specific ports by using their corresponding names.

The device interface names for these ports are of the format "syncX" where X is 0, 1, 2... etc

For example, if two cards are installed in a system with a 2 port PCIe card in the first numbered PCIe slot; a 4 port PCIe card is in the next numbered slot and the system also has a (single port) FarSync Flex plugged in to a USB port, then the associated device interface names would be:

- sync0 -- First port (A) on two port card
- sync1 -- Second port (B) on two port card
- sync2 -- First port (A) on four port card
- sync3 -- Second port (B) on four port card
- sync4 -- Third port (C) on four port card
- sync5 -- Fourth port (D) on four port card
- sync6 -- Port on the FarSync Flex

When there are FarSync PCIe, Flex and/or K2Ee devices present in the same system, the ports will be created in the following order:

- FarSync PCIe
- FarSync Flex
- FarSync K2Ee

## 6.2 Port Configuration

The configuration for each of the FarSync ports in the system can be set and examined using either the supplied **farutil** application or by a user application developed using the FarSync API as detailed in the FarSync SDK documentation. application will configure the line interfaces programmatically rather than manually using the **farutil** program. Please refer to the **FarSync® Linux® SDK User Guide** [5] for further details on setting a port config through the API. Even when using the API, **farutil** is often useful for monitoring the status of the port and checking the interface stats etc.

With **farutil** you can:

- Configure ports
- Display Card/Device Details
- Display status information
- Display Stats information
- Download firmware
- Identify a card

The **farutil** command has the following syntax:

```
farutil [-v] interface command [parameters...]
```

The -v option can be used at any time to report additional information.

Note that the naming convention for the interface used in the **farutil** command is as follows:

**syncX (sync0, sync1 etc)**

### 6.2.1 Configure ports

The syntax of the **farutil** command for setting parameters is as follows:

```
farutil sync<X> set [parameters]
```

where:

**<X>** is the device index of the target FarSync port e.g. sync0

For the list of parameters that can be used with the **farutil** command refer to Section 6.2.7.1.

Example:

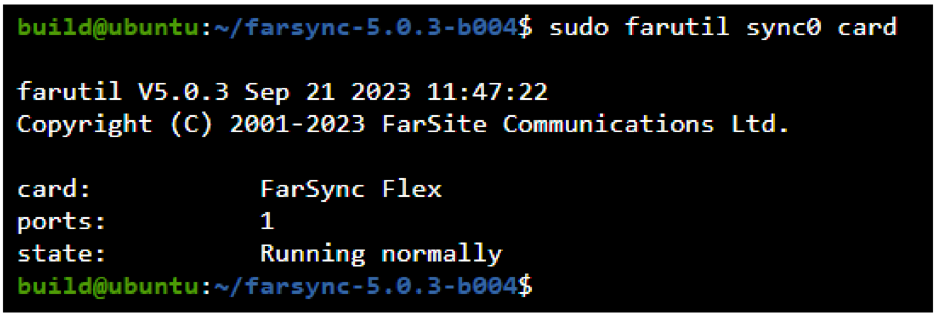
```
farutil sync0 set media x21 proto raw clock ext
```

### 6.2.2 Display Card/Device Details

To display card/device details the **farutil** command should be used in the following format:

```
farutil sync<X> card
```

e.g.



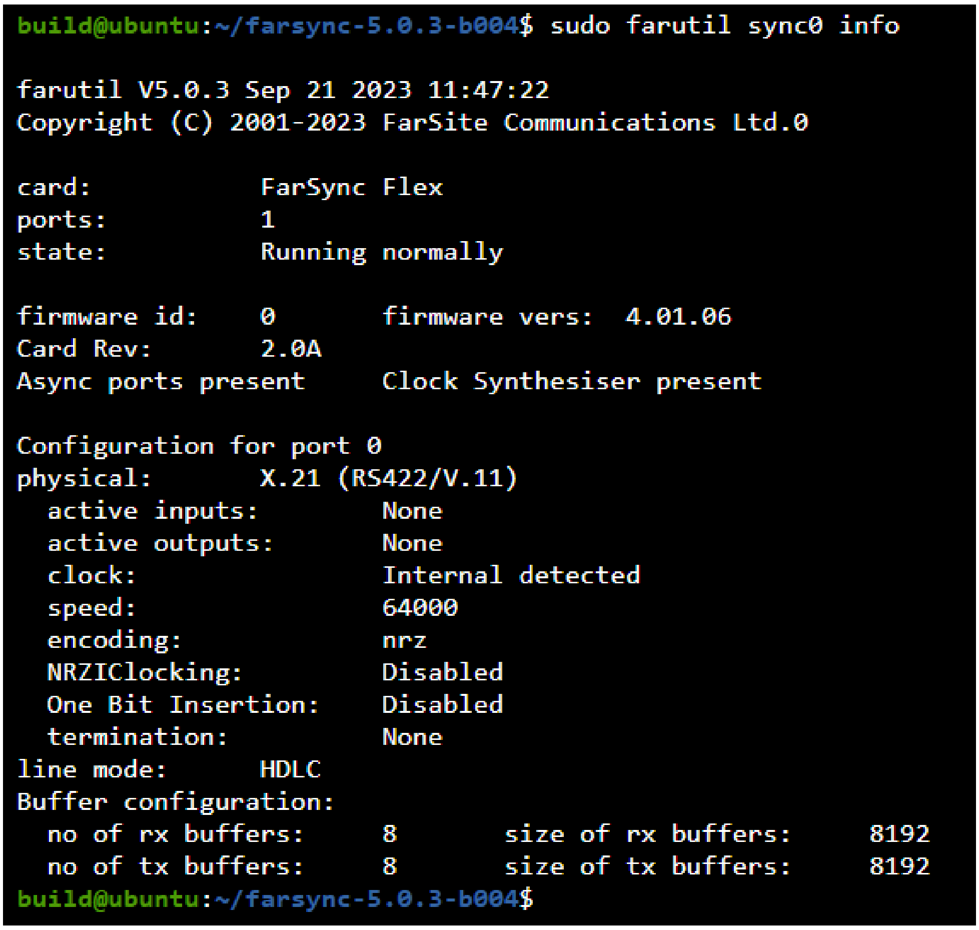
```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farutil sync0 card

farutil V5.0.3 Sep 21 2023 11:47:22
Copyright (C) 2001-2023 FarSite Communications Ltd.

card:           FarSync Flex
ports:          1
state:          Running normally
build@ubuntu:~/farsync-5.0.3-b004$
```

### 6.2.3 Display Status Information

To display port status information, the **farutil** command should be used in the following format:

**farutil sync<X> info**

e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farutil sync0 info

farutil V5.0.3 Sep 21 2023 11:47:22
Copyright (C) 2001-2023 FarSite Communications Ltd.0

card:           FarSync Flex
ports:          1
state:          Running normally

firmware id:    0       firmware vers:  4.01.06
Card Rev:       2.0A
Async ports present     Clock Synthesiser present

Configuration for port 0
physical:       X.21 (RS422/V.11)
  active inputs:        None
  active outputs:       None
  clock:                Internal detected
  speed:                64000
  encoding:             nrz
  NRZIClocking:         Disabled
  One Bit Insertion:    Disabled
  termination:          None
line mode:      HDLC
Buffer configuration:
  no of rx buffers:     8       size of rx buffers:     8192
  no of tx buffers:     8       size of tx buffers:     8192
build@ubuntu:~/farsync-5.0.3-b004$
```

Note:

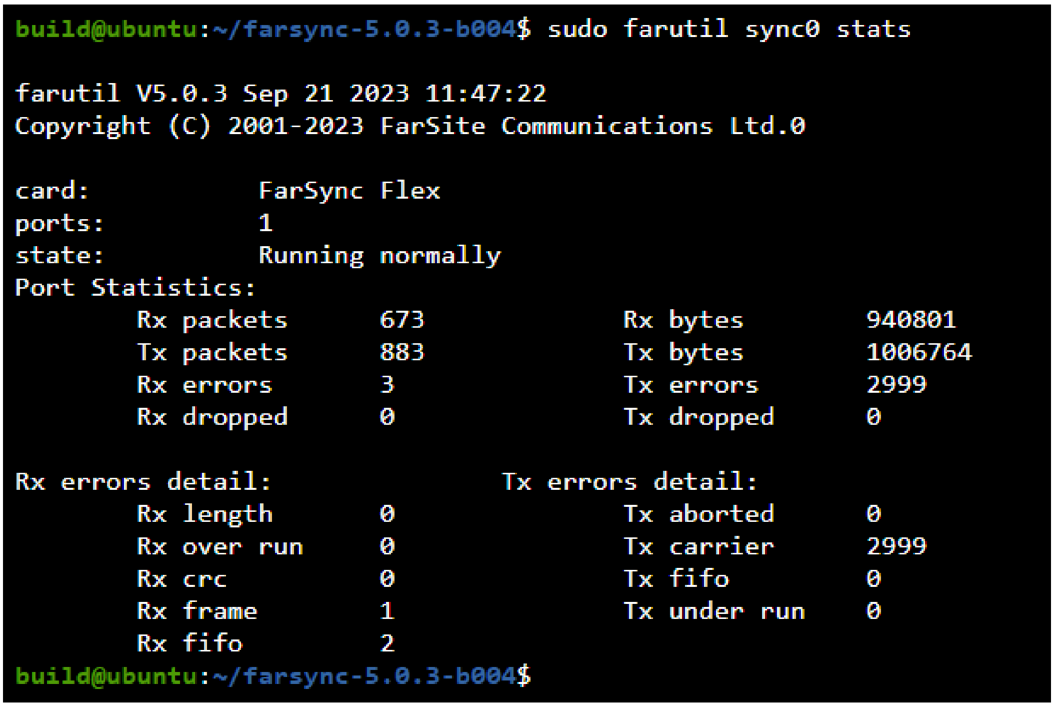
- Once dual clocking mode has been configured the physical interface will be displayed as X21d.
- When the clock phase has been inverted, the clock will be displayed as Internal/Inverted.
- Clock detection works on transmit clocks only.

### 6.2.4 Display Stats Information

To display port transmit and receive statistics the **farutil** command can be used in the following format:

**farutil sync<X> stats**

e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farutil sync0 stats

farutil V5.0.3 Sep 21 2023 11:47:22
Copyright (C) 2001-2023 FarSite Communications Ltd.0

card:           FarSync Flex
ports:          1
state:          Running normally
Port Statistics:
        Rx packets      673             Rx bytes        940801
        Tx packets      883             Tx bytes        1006764
        Rx errors       3               Tx errors       2999
        Rx dropped      0               Tx dropped      0

Rx errors detail:               Tx errors detail:
        Rx length       0               Tx aborted      0
        Rx over run     0               Tx carrier      2999
        Rx crc          0               Tx fifo         0
        Rx frame        1               Tx under run    0
        Rx fifo         2
build@ubuntu:~/farsync-5.0.3-b004$
```

### 6.2.5 Download firmware

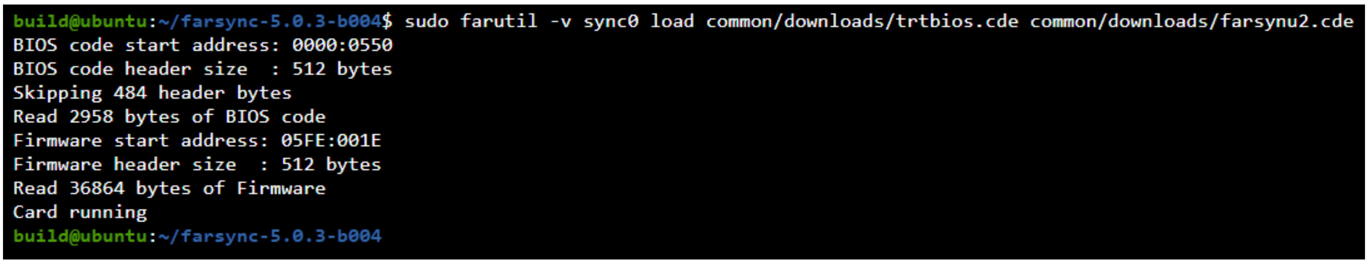
To download the card firmware the **farutil** command should be used in the following format:

**farutil sync<X> load [bios] [firmware]**

where:

**<X>** is the device index of the target FarSync port e.g. sync0
bios is the trtbios.cde file from the downloads directory
firmware is the farsync2.cde file from the downloads directory

e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farutil -v sync0 load common/downloads/trtbios.cde common/downloads/farsynu2.cde
BIOS code start address: 0000:0550
BIOS code header size  : 512 bytes
Skipping 484 header bytes
Read 2958 bytes of BIOS code
Firmware start address: 05FE:001E
Firmware header size  : 512 bytes
Read 36864 bytes of Firmware
Card running
build@ubuntu:~/farsync-5.0.3-b004
```

Note that this step is automatically carried out as part of the driver starting support that is included in the **/etc/init.d/farsite** script.

### 6.2.6 Identify a card

For PCIe card, the ordering of the PCIe slots used to identify different cards in the system does not always follow a simple mechanical sequence. Some system cases even mislabel the slots, being intended for different motherboards. This can be quite confusing. To address this problem, a card identifying facility is provided. The command:

```
farutil sync<X> set led flash
```

will identify the board to which the interface syncX is attached to by causing the board's status LEDs to flash continuously. Normal operation of the LEDs can be restored with:

```
farutil sync<X> set led normal
```

### 6.2.7 farutil Configuration Parameters

The following tables describe the parameters that can be used with **farutil** to set or view a port’s configuration.

#### *6.2.7.1 General Parameters*

<table>
<tr><th>Parameter</th><th>Typical value</th><th>Meaning</th></tr>
<tr><td>MEDIA</td><td>X21</td><td>This parameter selects the hardware interface to be used. Possible settings are:
<table>
<tr><th>Value</th><th>Selects:</th><th>Valid For (all cards except where noted):</th></tr>
<tr><td>x21</td><td>X.21 (V.11 or RS422)</td><td></td></tr>
<tr><td>x21d</td><td>X.21 as above but with the Indicate signal used as an extra clock line (see note about dual clocking below)</td><td></td></tr>
<tr><td>v24</td><td>V.24 (aka RS232C)</td><td></td></tr>
<tr><td>rs232</td><td>V.24</td><td></td></tr>
<tr><td>v35</td><td>V.35</td><td></td></tr>
<tr><td>ux35c</td><td>V.35 internal clocking with adapter cable</td><td>TxU, TxUe</td></tr>
<tr><td>rs530</td><td>RS449/530</td><td>T2U-PMC, T4E/T4E+, T2Ee, T4Ee, Flex</td></tr>
<tr><td>rs449</td><td>RS449/530</td><td>T2U-PMC, T4E/T4E+, T2Ee, T4Ee, Flex</td></tr>
<tr><td>u530</td><td>RS530 with adapter cable</td><td>TxU, TxUe</td></tr>
<tr><td>rs485</td><td>RS485</td><td>T2Ee, T4Ee, Flex</td></tr>
<tr><td>rs485_fdx</td><td>RS485 Full Duplex</td><td>T2Ee, T4Ee, Flex (v2 only)</td></tr>
</table>
</td></tr>
<tr><td>CLOCK</td><td>64000</td><td>The line speed to use. If set then the port will generate clocks (acting like a</td></tr>
</table>

<table>
<tr><td></td><td></td><td>DCE). Alternatively use the value "ext" to select the external clocks and run at the line speed determined by the remote end.<br><br>Possible clock speeds are listed here and depend on the card type.<br><br>V.24 (RS232) interfaces are limited to a maximum speed of 128000.</td></tr>
<tr><td>PHASE</td><td>normal</td><td>The required phase of the receive clock with respect to the data. The valid values are:
<table>
<tr><td>normal</td><td>no change</td></tr>
<tr><td>inverted</td><td>invert the clock</td></tr>
</table></td></tr>
<tr><td>LINEMODE</td><td>hdlc</td><td>The line discipline to be used on this port.  Possible settings are:
<table>
<tr><td>hdlc</td><td>Use synchronous hdlc framing</td></tr>
<tr><td>async</td><td>Use async character stream</td></tr>
<tr><td>transparent</td><td>Use synchronous bit stream</td></tr>
</table></td></tr>
<tr><td>EXTENDED</td><td>erx_itx</td><td>This parameter allows more flexible clocking arrangements to be configured.  The values currently defined for this parameter are as follows:
<table>
<tr><td>dce_tt</td><td>default configuration for DCE Terminal Timing (0x8B)</td></tr>
<tr><td>dte_tt</td><td>default configuration for DTE Terminal Timing (0x84)</td></tr>
<tr><td>erx_etx</td><td>set external rx and external tx clocks (0x80)</td></tr>
<tr><td>erx_itx</td><td>set external rx and internal tx clocks (0x81)</td></tr>
<tr><td>irx_etx</td><td>set internal rx and external rx clocks (0x82)</td></tr>
<tr><td>irx_itx</td><td>set internal rx and internal tx clocks (0x83)</td></tr>
<tr><td>none</td><td>disable extended clocking modes (0x00)</td></tr>
<tr><td>0xdd</td><td>a hexadecimal representation of the extendedClocking byte in struct fstioc_info<br>e.g. 0x83 is the same as irx_itx</td></tr>
</table>
Valid for T2U-PMC, T4E/T4E+, T2Ee, T4Ee, Flex v2.  The Flex v1 does not support dce_tt or dte_tt modes as it does not support terminal timing</td></tr>
<tr><td colspan="2">LED</td><td>This parameter tests the LED operation for the device.  It can help identify which port is on which card when you have a multi card installation.  This is normally used for diagnostic purposes. The values for this parameter are:
<table>
<tr><td>flash</td><td>make the leds flash on the card</td></tr>
<tr><td>normal</td><td>restore the leds to their normal state</td></tr>
</table></td></tr>
<tr><td>CARRIER</td><td>required</td><td>The default mode of operation of the driver is to ignore transmit requests unless the modem signals are present, i.e. there is a carrier.  This parameter allows this condition to be ignored and to transmit regardless.  The parameter values are:
<table>
<tr><td>ignore</td><td>allow transmit at all times</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>required</td><td>only allow transit when modem signals are present (e.g. Indicate on X.21)</td></tr></table></td></tr>
<tr><td>1BITINSERT</td><td>on</td><td>Enable or Disable the One Bit Insert feature.  This parameter is only valid for FarSync Flex ports.  The values currently defined for this parameter are as follows:<table><tr><td>on</td><td>enable One Bit Insertion feature</td></tr><tr><td>off</td><td>disable One Bit Insertion feature</td></tr></table></td></tr>
<tr><td>NUMBUFFERS</td><td>8</td><td>Each port is allocated a number of receive and transmit buffers on the device.  The number of buffers can be configured.  This parameter will configure the number of tx and rx buffers to the number supplied as the parameter value.<br>Valid values are 1, 2, 4, 8, 16, 32, 64 or 128<br>Note that for the Flex, the number of buffers cannot be set to 1</td></tr>
<tr><td>SIZEBUFFERS</td><td>8192</td><td>This parameter specifies the size of the buffers configured. It can be any value from 1 to 32k-1 but there are restrictions on the number/size combination.  The maximum buffer space per port per direction is 64K and the maximum number of buffers possible is 128. Therefore, the maximum buffer size for each permitted value of the number of buffers is as follows:<table><tr><th>Number of buffers</th><th>Maximum buffer size</th></tr><tr><td>1</td><td>32K-1</td></tr><tr><td>2</td><td>32K-1</td></tr><tr><td>4</td><td>16K</td></tr><tr><td>8</td><td>8K</td></tr><tr><td>16</td><td>4K</td></tr><tr><td>32</td><td>2K</td></tr><tr><td>64</td><td>1K</td></tr><tr><td>128</td><td>0.5K</td></tr></table>This is per port per direction. So for example on a two port card, port A could be configured for 2 x 32K-1 buffer for transmit and receive, while port B could be configured for 128 x 512 byte buffers for transmit and 4 x 16K byte buffers for receive.</td></tr>
<tr><td>NUMRXBUFFERS</td><td>8</td><td>As NUMBUFFERS but configures the rx buffers only</td></tr>
<tr><td>SIZERXBUFFERS</td><td>8192</td><td>As SIZEBUFFERS but configures the rx buffers only</td></tr>
<tr><td>NUMTXBUFFERS</td><td>8</td><td>As NUMBUFFERS but configures the tx buffers only</td></tr>
<tr><td>SIZETXBUFFERS</td><td>8192</td><td>As SIZEBUFFERS but configures the tx buffers only</td></tr>
</table>

<table>
<tr><td rowspan="6">LOWLATENCY</td><td rowspan="6">disable</td><td colspan="2">The card normally works in a mode that reduces interrupts to the host (delayed interrupt mode), but this has the effect of introducing a small degree of latency on receive. This parameter allows the latency to be minimised. The parameter may take the following values:</td></tr>
<tr><td>rx</td><td>enter immediate interrupt mode (low latency rx)</td></tr>
<tr><td>tx</td><td>notify the card that there is a frame to transmit (low latency tx)</td></tr>
<tr><td>txrx</td><td>both of the above (low latency rx and tx)</td></tr>
<tr><td>disable</td><td>enter delayed interrupt mode</td></tr>
<tr><td colspan="2">Valid for T4U, T4E/T4E+, T4Ee</td></tr>
<tr><td rowspan="10">CODING</td><td rowspan="10">nrz</td><td colspan="2">This parameter can be used to select the line encoding methods as follows:</td></tr>
<tr><td><b>Encoding</b></td><td><b>Valid For:</b></td></tr>
<tr><td>nrz</td><td>T4E+, T2Ee, T4Ee, Flex</td></tr>
<tr><td>nrz_clk_rec</td><td>Flex (v2 only)</td></tr>
<tr><td>nrzi</td><td>Flex</td></tr>
<tr><td>fm0</td><td>T4E+, T2Ee, T4Ee, Flex</td></tr>
<tr><td>fm1</td><td>T4E+, T2Ee, T4Ee, Flex</td></tr>
<tr><td>manchester</td><td>T4E+, T2Ee, T4Ee, Flex (v2 only)</td></tr>
<tr><td>diff_manchester</td><td>T4E+, T2Ee, T4Ee, Flex (v2 only)</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td rowspan="4">TERMINATION</td><td rowspan="4">none</td><td colspan="2">This parameter sets the line termination mode. The parameter may take one of the following two values:</td></tr>
<tr><td>none</td><td>no termination</td></tr>
<tr><td>resistive</td><td>resistive termination</td></tr>
<tr><td colspan="2">Valid for T2U-PMC, T4E/T4E+, T2Ee, T4Ee, Flex</td></tr>
<tr><td rowspan="4">NRZICLOCKING</td><td rowspan="4">disable</td><td colspan="2"></td></tr>
<tr><td>enable</td><td>Use NRZICLOCKING</td></tr>
<tr><td>disable</td><td>Do not use NRICLOCKING</td></tr>
<tr><td colspan="2">Valid for T4E+ and Flex</td></tr>
<tr><td>SYNTH</td><td><string></td><td colspan="2">This parameter can be used to define an internal clock rate that isn't already provided as one of the standard rates. The string parameter is generated by using some additional tools – see [A4<br>(valid for T2Ue, T2U-PMC, T4E/T4E+, T2Ee, T4Ee, Flex v2/v3, K2Ee)<br><br>Note that when a custom clock rate is set with this command, and then <b>farutil</b> is used to assign the new internal clock rate to the port it may warn that the clock rate was unknown and that the nearest rate will be selected. In this case this message can be disregarded.</td></tr>
</table>

### Note on Dual Clocking

In order to be able to provide a clock with data in each direction on the X.21 interface requires that the Indicate signal line be reassigned as the second clock line. This means that the signal will always appear to be ON. The Control signal is not used in this configuration. In order to use this mode, appropriate changes are required to the X.21 cable.  Note that the Dual Clocking feature is only available on the T2Ue and T4Ue.

#### *6.2.7.2 Async Parameters*

<table>
<tr><th>Parameter</th><th>Typical value</th><th>Meaning</th></tr>
<tr><td>flow_control</td><td>none</td><td>The flow control mechanism to be used.  The possible values are:
<table>
<tr><td>none</td><td>No flow control</td></tr>
<tr><td>hardware</td><td>Use hardware CTS/RTS signaling</td></tr>
<tr><td>software</td><td>Use software Xon and Xoff characters</td></tr>
</table></td></tr>
<tr><td>stop_bits</td><td>1</td><td>The number of stop bits to follow the data bits.  The possible values are:
<table>
<tr><td>1</td><td>Use 1 stop bit</td></tr>
<tr><td>1.5</td><td>Use 1 and a half stop bits</td></tr>
<tr><td>2</td><td>Use 2 stop bits</td></tr>
<tr><td></td><td></td></tr>
</table></td></tr>
<tr><td>parity</td><td>none</td><td>The parity checking mode to be used.  The possible values are:
<table>
<tr><td>none</td><td>No parity checking is performed</td></tr>
<tr><td>even</td><td>The number of bits per byte is checked to be even</td></tr>
<tr><td>odd</td><td>The number of bits per byte is checked to be odd</td></tr>
</table></td></tr>
<tr><td>word_len</td><td>8</td><td>The number of data bits used per character.  The possible values are in the range 5 to 8 bits inclusive.</td></tr>
<tr><td>xon_char</td><td>0x11</td><td>When the software flow control mechanism has been configured, this is the character used to notify the remote end that it may continue sending data again.  This character value will not normally need to be changed from its default.</td></tr>
<tr><td>xoff_char</td><td>0x13</td><td>When the software flow control mechanism has been configured, this is the character used to notify the remote end that it should stop sending data until further notice.  This character value will not normally need to be changed from its default.</td></tr>
</table>

# 7 Application Development

Once the drivers have been successfully started, the ports are available for use by applications – either previously developed or for new developments.

For new developments, if you have purchased the Software Developers Tool Kit (SDK) then you can build and try out some of the example applications.  Most of them are written to work on two ports that are looped back to each other (using a NULL modem cable for example).  The SDK also contains a set of comprehensive API manuals for use when developing your own applications.

# Appendices

# A1 Permissions

By default, all access to the FarSync Ports, from both the supplied utilities and/or user-developed applications, require root privilege.

Access to other users can be granted by

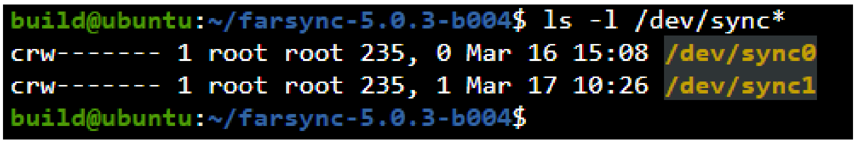
- Changing the permissions on the /dev instance representing the port e.g. /dev/sync0. This enables you to manage which users can open/read/write/close the port.

  and (optionally)

- Permitting non-root users to issue configuration/management IOCTLs to manage the port. This enables you to allow non-root users to manage the port using **farutil** for example.

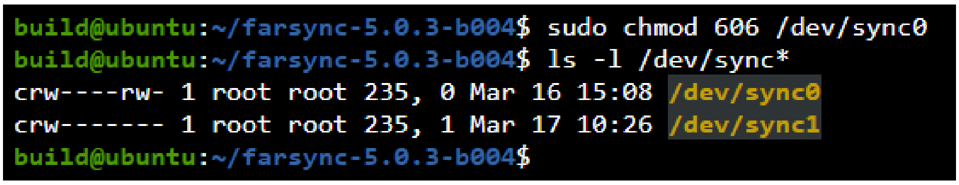
## A1.1 Device

By default, the /dev instance for each port is created with root-only access.
e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ ls -l /dev/sync*
crw------- 1 root root 235, 0 Mar 16 15:08 /dev/sync0
crw------- 1 root root 235, 1 Mar 17 10:26 /dev/sync1
build@ubuntu:~/farsync-5.0.3-b004$
```

The permissions can then be configured however you require in order to provide your target access configuration. For example,



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo chmod 606 /dev/sync0
build@ubuntu:~/farsync-5.0.3-b004$ ls -l /dev/sync*
crw----rw- 1 root root 235, 0 Mar 16 15:08 /dev/sync0
crw------- 1 root root 235, 1 Mar 17 10:26 /dev/sync1
build@ubuntu:~/farsync-5.0.3-b004$
```

enables general user access to device sync0

## A1.2 Configuration IOCTLs

Traditionally, applications that can configure low level networking parameters usually require root privilege in order to be able to perform configuration changes. To extend this capability to users without root privilege, there are two available approaches:

1) Separate the configuration from the functionality into different applications. The configuration application can be run as root, and the application that provides the functionality can be run as a normal user.

2) Allow the user level application to perform the configuration.

To support 2) the drivers can be started with a module parameter, **fst_allow_ioctl**, which is set to a numerical, bit-significant value to indicate the type of access required. This parameter can take the following values:

#define FST_ALLOW_IOCTL_GET 1 // Allow configuration parameters to be read

#define FST_ALLOW_IOCTL_SET 2 // Allow configuration parameters to be written

#define FST_ALLOW_IOCTL_CARD 4 // Allow special card functions to be executed

#define FST_ALLOW_IOCTL_ALL FST_ALLOW_IOCTL_GET + FST_ALLOW_IOCTL_SET + FST_ALLOW_IOCTL_CARD // Allow everything

The default value is 0.

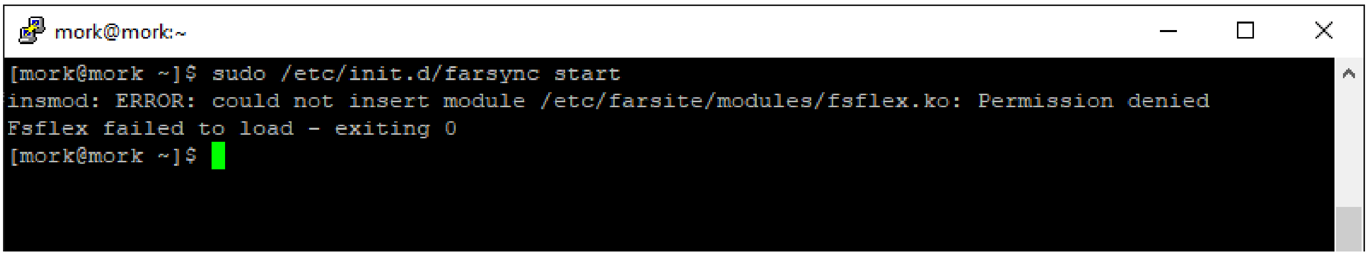
To specify a non-default value for , **fst_allow_ioctl** when starting the driver(s), edit the default /etc/init.d/farsync script and add the parameter to the **insmod** command(s) as follows:

.
.
/sbin/insmod $MODULE_DIR/farsync.ko **fst_allow_ioctl =<n>**
.
.
/sbin/insmod $MODULE_DIR/fsflex.ko $serials **fst_allow_ioctl =<n>**
.
.

where <n> is a value between 0 and 7 as described above.

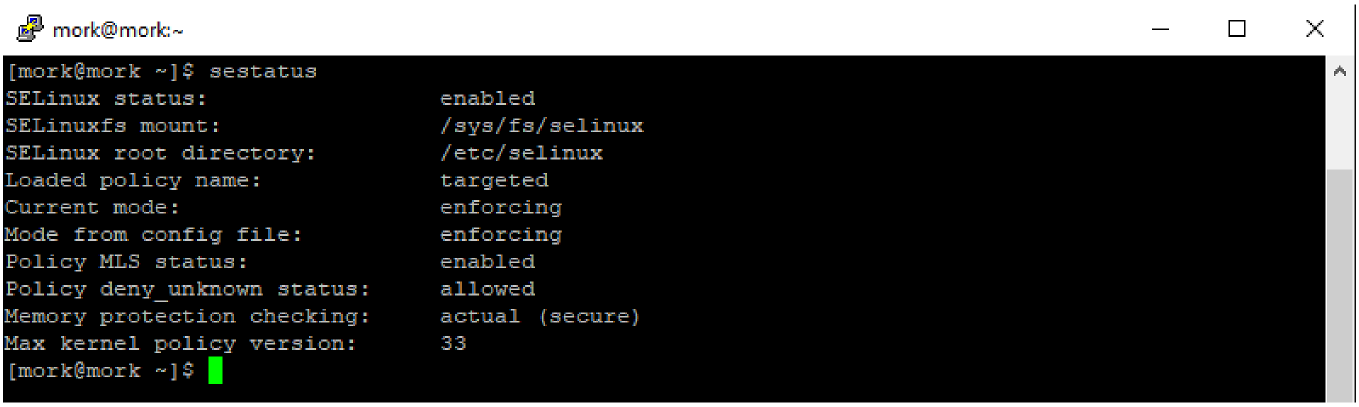
## A1.3 SELinux

If this error occurs when insmod is trying to insert a module, such as the error below, then this is normally due to SELinux:



```
[mork@mork ~]$ sudo /etc/init.d/farsync start
insmod: ERROR: could not insert module /etc/farsite/modules/fsflex.ko: Permission denied
Fsflex failed to load - exiting 0
[mork@mork ~]$
```

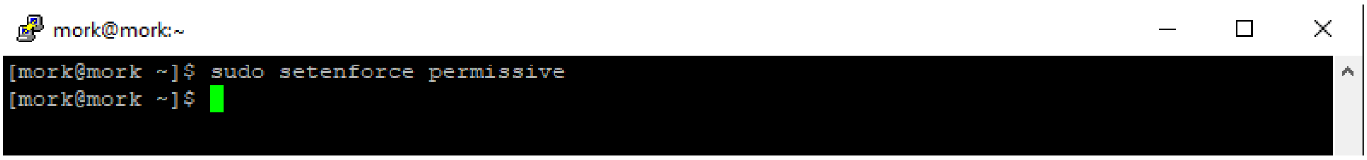
To check SELinux status use the **sestatus** command:



```
[mork@mork ~]$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
[mork@mork ~]$
```

This response indicates that SELinux is currently **enabled** and set to **enforcing**. This configuration can be set to **permissive** using:

sudo setenforce permissive

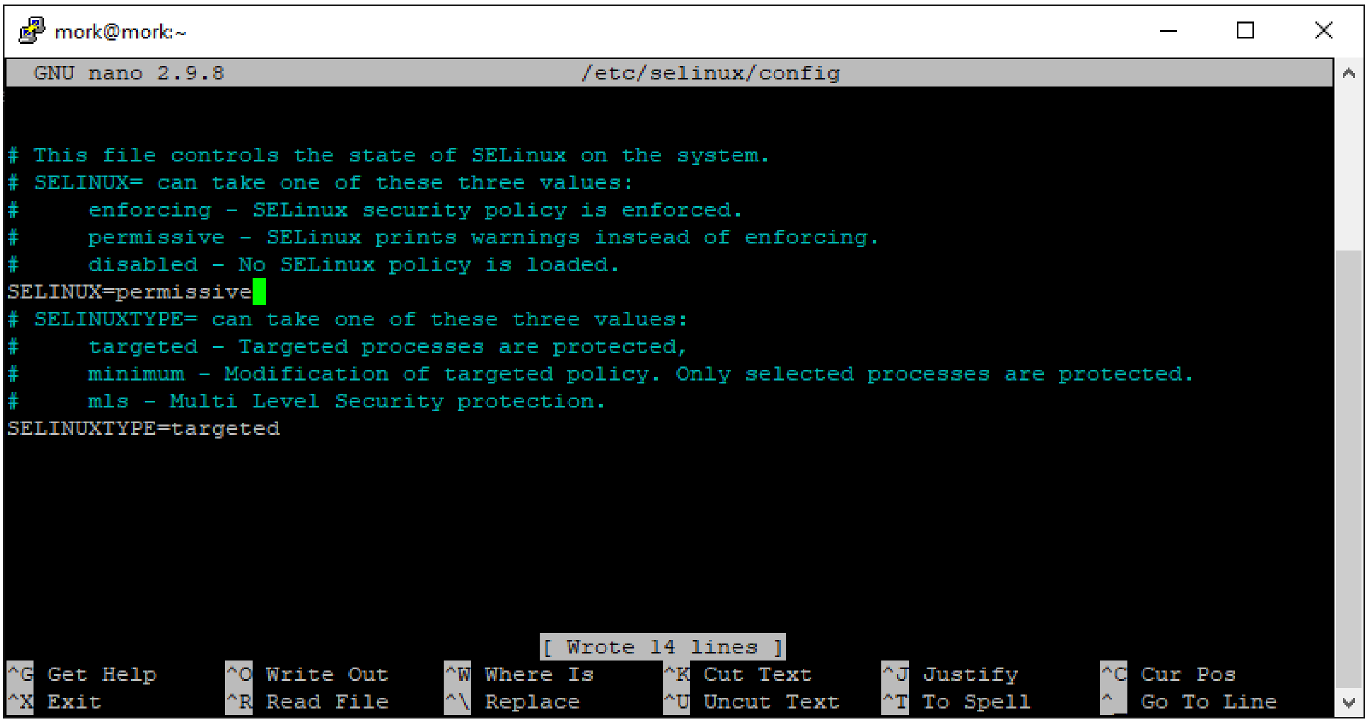


```
[mork@mork ~]$ sudo setenforce permissive
[mork@mork ~]$
```

Then it should be possible to load/unload the driver using the farsync start/stop as is.

As a longer term measure though you can either

1) leave SELinux in permissive mode, in which case we need to make sure the permissive setting is carried across reboots by editing the /etc/selinux/config file and setting SELINUX=permissive.



```
  GNU nano 2.9.8                        /etc/selinux/config


# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted







                                               [ Wrote 14 lines ]
^G Get Help      ^O Write Out     ^W Where Is      ^K Cut Text      ^J Justify       ^C Cur Pos
^X Exit          ^R Read File     ^\ Replace       ^U Uncut Text    ^T To Spell      ^_ Go To Line
```

OR 2) alternatively, you can set SELinux back to **enforcing** and add a custom policy module, which can either be done manually or via a script. The sample script below demonstrates creating a custom policy module and loading the newly created policy module.

```
#!/bin/bash
#
#    Installs a local SELinux policy for farsync or fshcx25
#


if [ $(id -u) != "0" ]; then
   echo "You must be root to run this script"
   exit 1
fi

if semodule -lfull | grep farsite >/dev/null; then
   echo "There is already a farsite SELinux module"
   echo "Please use sudo semodule -X300 -r farsite to remove the module and"
   echo "backup and delete any farsite.te or farsite.pp files in this directory"
   echo "before running this script again"
   exit 1
fi

if [ -e /etc/init.d/farsync ] || [ -e /etc/init.d/fshcx25]; then
   echo "Checking audit log"
   if ausearch -c 'insmod' --raw | egrep 'farsync|fsx25|fsflex|x25tap' > /dev/null; then
      echo "Entries for FarSite software found in SELinux audit log"
      echo "Generating local policy module"
      ausearch -c 'insmod' --raw | egrep 'farsync|fsx25|fsflex|x25tap' | audit2allow -M farsite >
/dev/null
      echo "Applying local policy module"
      semodule -X 300 -i farsite.pp
   else
      echo "There are no entries found for FarSite software in SELinux audit log"
      echo "You may need to stop/start your FarSite software to generate entries in the SELinux audit log"
      exit 1
   fi
else
   echo "There does not appear to be any farsite software installed"
   exit 1
fi
exit 0
```

## A2 USB Device Names

If you have a number of FarSync Flex USB devices in your system, then you may find that each time the system is restarted, or you unplug them and plug them back in again, the device name for a particular device changes. There is a way to fix the name to a device serial number to ensure that it always has the same device name.

Previously, the mapping of Flex's to specific device numbers was achieved through the use of udev, but due to the ever-evolving nature of the Linux Kernels, it was decided that the mapping functionality should be placed within the Farsync Flex driver itself.

Since release 2.0.1 of the Farsync driver, the mapping is now achieved through the use of the following configuration file:

**/etc/farsite/farsync/serials.config**

Entries in the configuration file are of the following format:

```
6 U0010009
7 U1040177
```

This example matches the Flex with serial number U0010009 to sync6 and Flex U1040177 to sync7.

The configuration file is read when the flex driver is loaded. If a device number is already in use (say for the farsync cards - which are loaded before the Flexes), then the request for the mapping will fail and the Flex will not be installed.

To see which device numbers are already in use by the Farsync cards and Flex's, type the following commands:

```
more /proc/farsync
more /proc/fsflex
```

To enable the mapping of Flex's to specific device numbers, the following steps are required:

- In the kernel-oem directory, edit the Makefile. At the end of the EXTRA_CFLAGS line add the following:

  **-DSPECIFY_DEV_NAME**

- then run:

  ```
  make clean
  make
  make install
  ```

  in the directory /etc/farsite/farsync, edit the serials.config file as desired.

- then stop and start the driver with:

  ```
  sudo systemctl restart farsync
  ```

  To check if a Flex has been mapped to a particular device number, you can do:

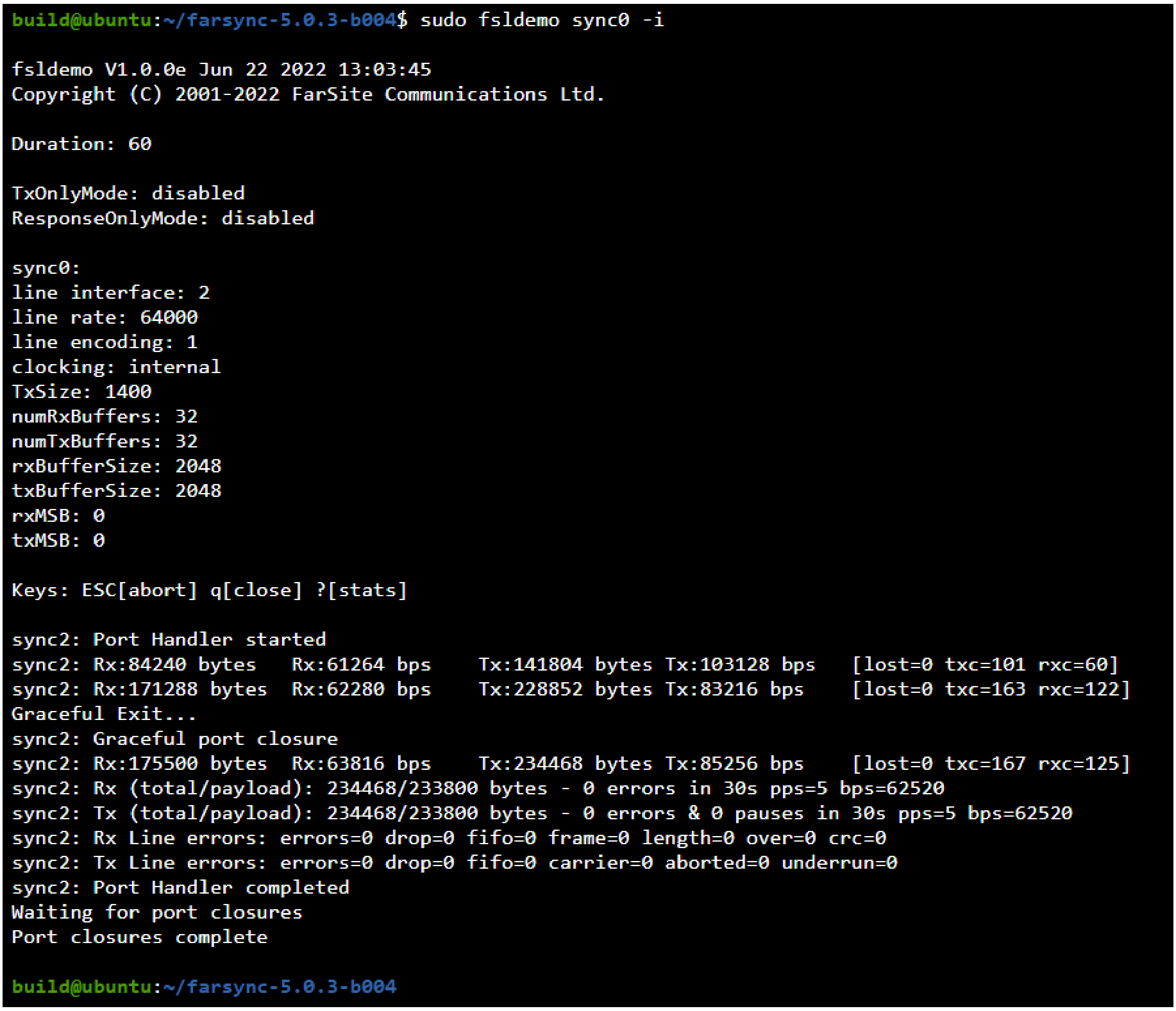
  ```
  more /proc/fsflex
  ```

# A3 Utilities

## A3.1 fsldemo

Once the driver(s) have been installed, the FarSync device/card(s) and their associated drivers can be exercised using the supplied **fsldemo** utility. **fsldemo** is installed into /sbin as part of the FarSync install sequence.

When running the application, you can **optionally** specify the logical device associated with the required FarSync port to be used e.g. sync0. By default, if the parameter is not specified, the value of sync0 shall be used. You can also optionally specify any supported, non-default configuration settings required for that particular port.
e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo fsldemo sync0 -i

fsldemo V1.0.0e Jun 22 2022 13:03:45
Copyright (C) 2001-2022 FarSite Communications Ltd.

Duration: 60

TxOnlyMode: disabled
ResponseOnlyMode: disabled

sync0:
line interface: 2
line rate: 64000
line encoding: 1
clocking: internal
TxSize: 1400
numRxBuffers: 32
numTxBuffers: 32
rxBufferSize: 2048
txBufferSize: 2048
rxMSB: 0
txMSB: 0

Keys: ESC[abort] q[close] ?[stats]

sync2: Port Handler started
sync2: Rx:84240 bytes   Rx:61264 bps    Tx:141804 bytes Tx:103128 bps   [lost=0 txc=101 rxc=60]
sync2: Rx:171288 bytes  Rx:62280 bps    Tx:228852 bytes Tx:83216 bps    [lost=0 txc=163 rxc=122]
Graceful Exit...
sync2: Graceful port closure
sync2: Rx:175500 bytes  Rx:63816 bps    Tx:234468 bytes Tx:85256 bps    [lost=0 txc=167 rxc=125]
sync2: Rx (total/payload): 234468/233800 bytes - 0 errors in 30s pps=5 bps=62520
sync2: Tx (total/payload): 234468/233800 bytes - 0 errors & 0 pauses in 30s pps=5 bps=62520
sync2: Rx Line errors: errors=0 drop=0 fifo=0 frame=0 length=0 over=0 crc=0
sync2: Tx Line errors: errors=0 drop=0 fifo=0 carrier=0 aborted=0 underrun=0
sync2: Port Handler completed
Waiting for port closures
Port closures complete

build@ubuntu:~/farsync-5.0.3-b004
```

**fsldemo** supports the following parameters:

```
fsldemo [sync<X>] [-I<interfacetype> -E<encoding> -R<rate>] [-T<txsize> -W<duration> -D<tracelevel>] [-im]
```

where:

| | |
|---|---|
| **<X>** | is the device index of the target FarSync port e.g. sync0 |
| **-I<interfacetype>** | specifies the required interface type:<br>x21, v24, v35, rs449, rs530 – default = x21 |
| **-E<encodingtype>** | specifies the required encoding type:<br>nrz, nrz_clk_rec, nrzi, fm0, fm1, manchester, diff_manchester – default = nrz |
| **-R<rate>** | specifies the required line rate – default = 64000 bps |
| **-T<txsize>** | specifies the data block tx size (in bytes) to be used – default = 1400 bytes |
| **-W<duration>** | specifies the test duration in secs - default = 60s |
| -D**<tracelevel>** | specifies the trace level {0,1,2}– default = none |
| **-i** | enables internal clocking |
| **-m** | specified MSB ordering – default = LSB |

**fsldemo** conditionally send/receives test data and then reports the throughput that is achieved during the period of running the test.

**fsldemo** can be used with a loopback plug attached to a FarSync port. In this case the port would normally be configured for **internal** clocking e.g. **sudo ./fsldemo sync0 -i** as in the example screenshot above.

**fsldemo** can be closed either immediately by pressing ESC or alternatively by pressing 'q' (i.e. quit/graceful exit). The latter option requests **fsldemo** to immediately stop transmitting and it then provides a chance for all the transmitted data to be received before the program terminates. This enables the verification of **all** the data sent. The ESC option can be used after a graceful exit request if required.

Further port configuration parameters, that are not supported explicitly by **fsldemo**, can be set prior to running **fsldemo**, by using the **farutil** application – see Section 6.2.1

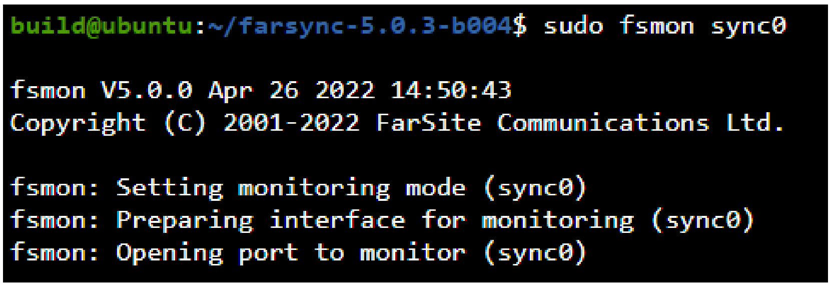
For testing multiple FarSync ports, including ports on different PCs, you can run multiple instances of **fsldemo**. In this case it is recommended that, once both sides of the test have started, the 'r' command is used to reset the stats i.e. in order to exclude any errors detected prior to the point where both sides are fully running.

## A3.2 fsmon

Data sent to and from the FarSync driver(s) can be captured using a monitoring mode as demonstrated by the **fsmon** utility.

**fsmon** is installed into /sbin as part of the FarSync install sequence.

When running the application, the logical device associated with the required FarSync port to be used, e.g. sync0, **must** be specified e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo fsmon sync0

fsmon V5.0.0 Apr 26 2022 14:50:43
Copyright (C) 2001-2022 FarSite Communications Ltd.

fsmon: Setting monitoring mode (sync0)
fsmon: Preparing interface for monitoring (sync0)
fsmon: Opening port to monitor (sync0)
```

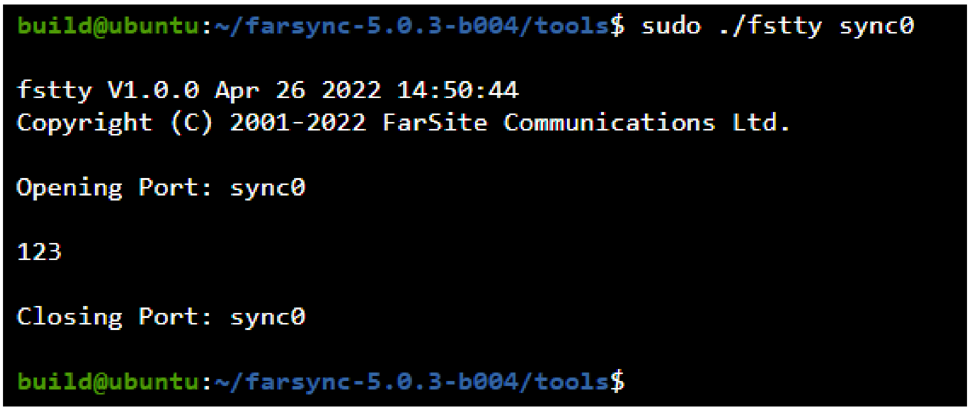
**fsmon** supports the following parameters:

**fsmon sync<X>**

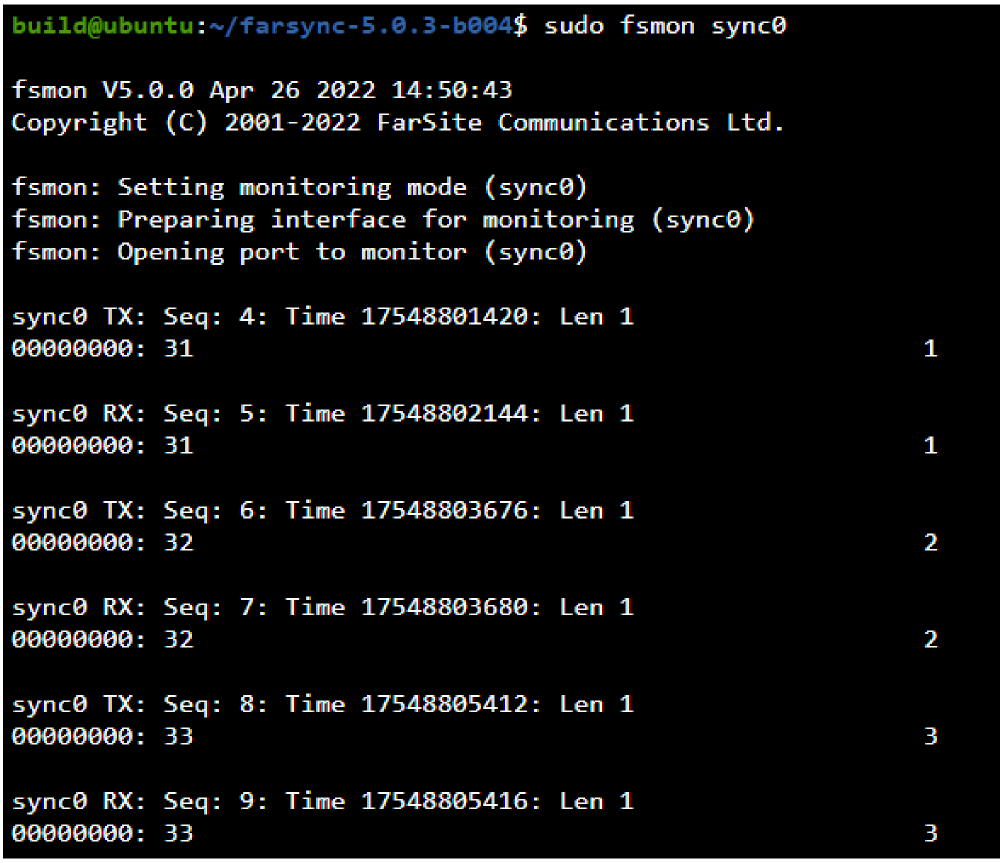
where:

**<X>** is the device index of the target FarSync port e.g. sync0

When an application is run that uses the monitored FarSync Port, such as the **fstty** application described in Section 5.1, you should see the resulting network activity displayed by **fsmon**
e.g.



```
build@ubuntu:~/farsync-5.0.3-b004/tools$ sudo ./fstty sync0

fstty V1.0.0 Apr 26 2022 14:50:44
Copyright (C) 2001-2022 FarSite Communications Ltd.

Opening Port: sync0

123

Closing Port: sync0

build@ubuntu:~/farsync-5.0.3-b004/tools$
```



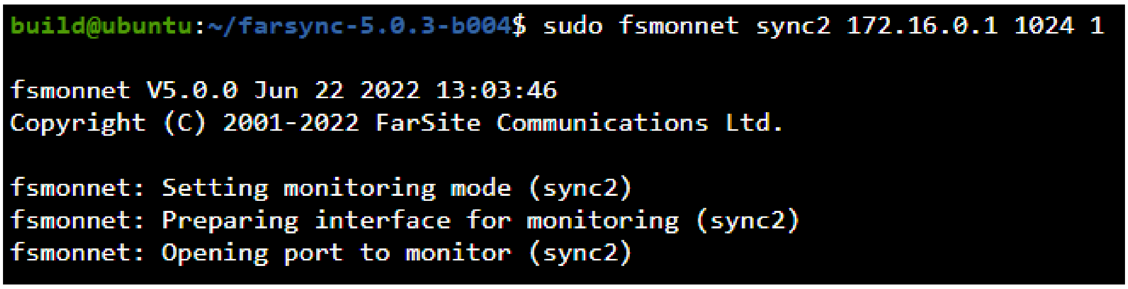
```
build@ubuntu:~/farsync-5.0.3-b004$ sudo fsmon sync0

fsmon V5.0.0 Apr 26 2022 14:50:43
Copyright (C) 2001-2022 FarSite Communications Ltd.

fsmon: Setting monitoring mode (sync0)
fsmon: Preparing interface for monitoring (sync0)
fsmon: Opening port to monitor (sync0)

sync0 TX: Seq: 4: Time 17548801420: Len 1
00000000: 31                                            1

sync0 RX: Seq: 5: Time 17548802144: Len 1
00000000: 31                                            1

sync0 TX: Seq: 6: Time 17548803676: Len 1
00000000: 32                                            2

sync0 RX: Seq: 7: Time 17548803680: Len 1
00000000: 32                                            2

sync0 TX: Seq: 8: Time 17548805412: Len 1
00000000: 33                                            3

sync0 RX: Seq: 9: Time 17548805416: Len 1
00000000: 33                                            3
```

## A3.3 fsmonnet

As an alternative to having the captured monitored data being displayed on the console by **fsmon**, it can be sent to an external application over TCP.

This can be demonstrated using the **fsmonnet** utility that is installed into /sbin as part of the FarSync install sequence.

When running the application, the following parameters **must** be specified: the logical device associated with the required FarSync port to be used e.g. sync0, the IP address/port of a Windows machine running the FarSync Line Monitor (the Windows GUI version of **fsmon**) and an id value for **fsmonnet** to use to unambiguously identify the monitored port. e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo fsmonnet sync2 172.16.0.1 1024 1

fsmonnet V5.0.0 Jun 22 2022 13:03:46
Copyright (C) 2001-2022 FarSite Communications Ltd.

fsmonnet: Setting monitoring mode (sync2)
fsmonnet: Preparing interface for monitoring (sync2)
fsmonnet: Opening port to monitor (sync2)
```

**fsmonnet** supports the following parameters:

```
fsmonnet sync<X> <ip address> <ip port> <id> [-l<n>]
```

where:

| | |
|---|---|
| **<X>** | is the device index of the target FarSync port e.g. sync0 |
| **<ip address>** | is the IP address of the PC running the Windows version of the FarSync Line Monitor (**fsmon**) |
| **<ip port>** | is the TCP port of the PC running the FarSync Line Monitor (**fsmon**) |
| **<id>** | is the id value used by the FarSync Line Monitor to identify the source of the message |
| -**l<n>** | specifies the option to zero each captured frame after n bytes (n>0) |

Note that a numerical id value must also be specified after the port number, in order to distinguish between multiple machines which could sharing a single remote **fsmon** instance.

If the data in the captured frame is potentially sensitive, a portion of it can be zeroed before being sent to **fsmon** for display/capture.  This is achieved using the -l<n> parameter where <n> specifies the offset from the start of the frame to the first byte to be zeroed. All data from that point in each captured frame, up to and including the last byte of the frame, will be zeroed.

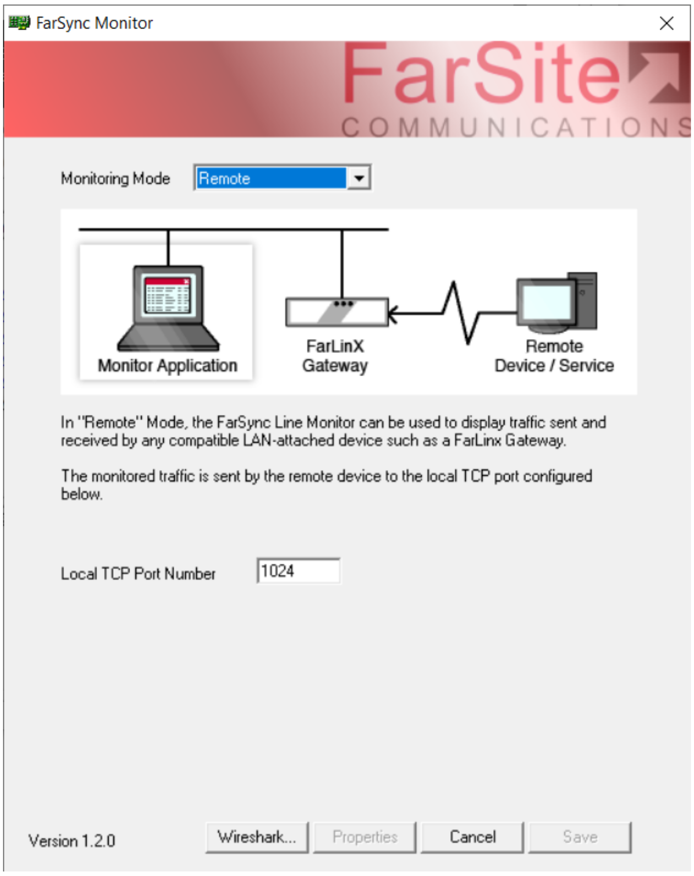
### A3.3.1 Configuring the FarSync Line Monitor (fsmon) on Windows

On the Windows PC, install the FarSync Line Monitor (**fsmon**) by running

**tools\Windows\fsmon\fsmon_install.bat**

**fsmon** can then be run simply by executing **fsmon.exe**

From the File Menu, select the **Recording Mode**. This will display the following:



Monitoring Mode: Remote

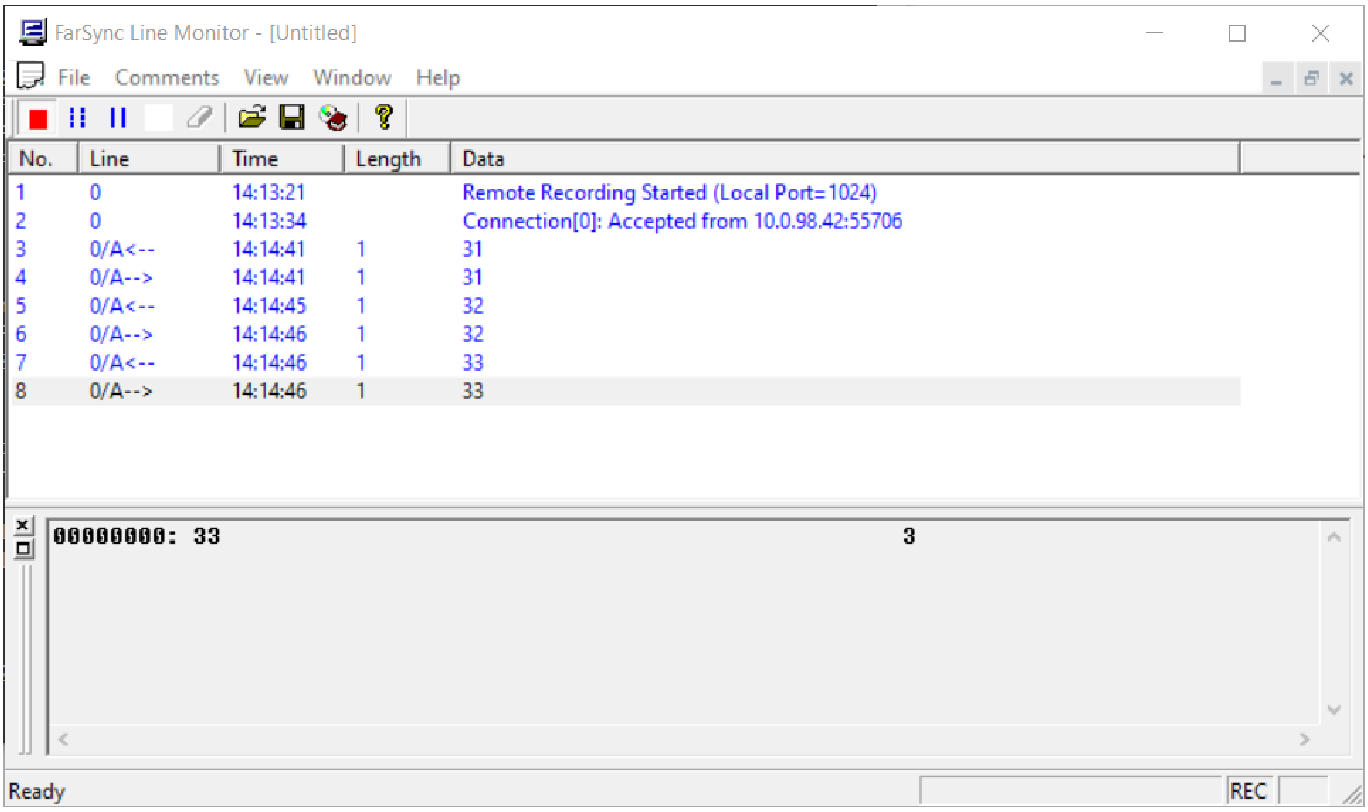
In "Remote" Mode, the FarSync Line Monitor can be used to display traffic sent and received by any compatible LAN-attached device such as a FarLinx Gateway.

The monitored traffic is sent by the remote device to the local TCP port configured below.

Local TCP Port Number: 1024

Set the Monitoring Mode to **Remote** and and specify a local IP port number for listening on. Then press Save. Stop the monitor (if it's already running) and start it again for the settings to take effect.

On the Linux machine, start **fsmonnet** as above.



| No. | Line | Time | Length | Data |
|---|---|---|---|---|
| 1 | 0 | 14:13:21 | | Remote Recording Started (Local Port=1024) |
| 2 | 0 | 14:13:34 | | Connection[0]: Accepted from 10.0.98.42:55706 |
| 3 | 0/A<-- | 14:14:41 | 1 | 31 |
| 4 | 0/A--> | 14:14:41 | 1 | 31 |
| 5 | 0/A<-- | 14:14:45 | 1 | 32 |
| 6 | 0/A--> | 14:14:46 | 1 | 32 |
| 7 | 0/A<-- | 14:14:46 | 1 | 33 |
| 8 | 0/A--> | 14:14:46 | 1 | 33 |

```
00000000: 33                                                   3
```

Note that the captured data can be saved to disk by **fsmon** in one of the following file formats:

| File Type | For use when subsequently displaying in |
|---|---|
| fmn | the GUI FarSync Line Monitor (**fsmon**) |
| txt | a plain text file viewer |
| pcap | Wireshark |

Note that if required, the captured trace can be written as a pcap file for subsequent display in Wireshark.

## A4 Custom Clock Rates

Some of the FarSync cards/devices support an on-board clock synthesiser, and it is therefore possible for non-standard clock rates to be configured.  There can be one custom clock rate per port.

The following FarSync cards/devices support custom clock rates:

- T2Ue
- T2U-PMC
- T4E(+)
- T2Ee
- T4Ee
- Flex (>=V2)
- K2Ee

These cards/devices require the use of an external tool to generate a clock string.

The process of setting the clock rate manually or programmatically is described below.

### A4.1 User Synthesizer Programming

The following describes how a user can program a FarSync Synthesiser capable device to generate non-standard user defined clock rates using the onboard synthesizer.

The FarSync device card/firmware supports a wide range of preset standard clock frequencies built in, but when non-standard frequencies are required, a design tool from the synthesizer manufacturer may be used to calculate the programming sequence.

These non-standard frequencies may be configured prior to a port being opened (static), or may be configured and changed while a port is open and running (dynamic).

Some restrictions apply to dynamic clock adjustment and are described later in this document.

#### A4.1.3 FarSync K2Ee

The external tool required for use with the FarSync K2Ee is the ON-Semi Clock Cruiser application. For further details please contact support@farsite.com.

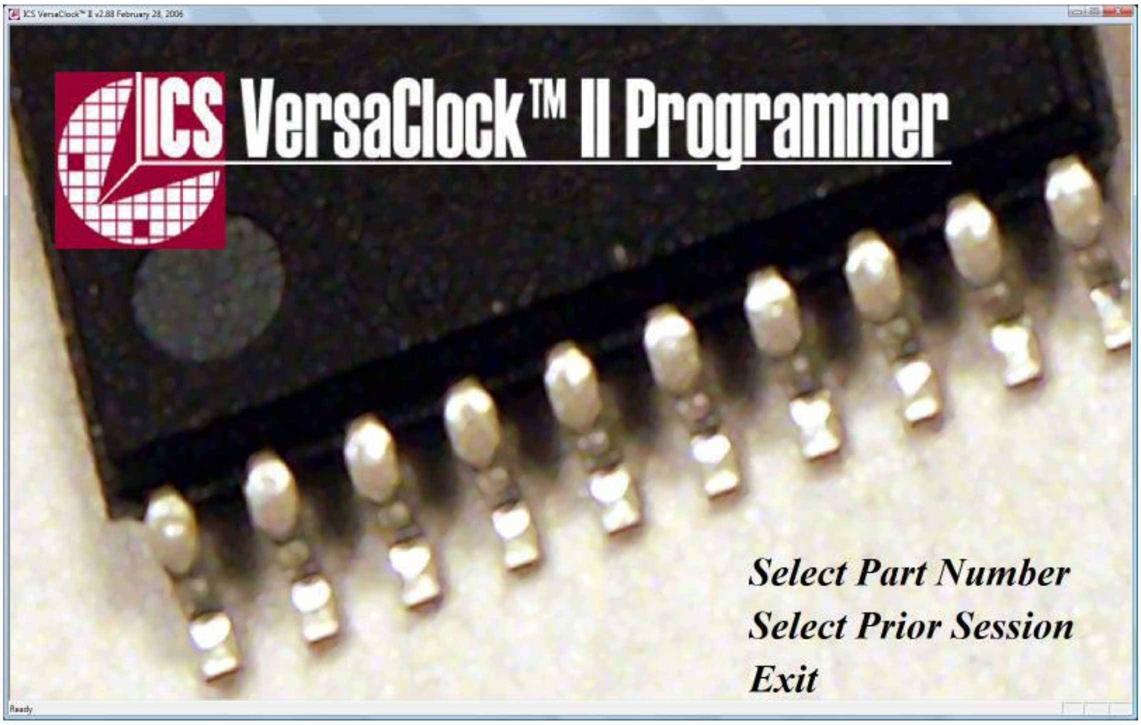
#### A4.1.4 FarSync T-Series cards and the FarSync Flex

For all other cards the required tool is the VersaClock II Programmer. You will find a copy of the tool in the VersaClock directory, but you should check with the chip vendor if there is a later release.

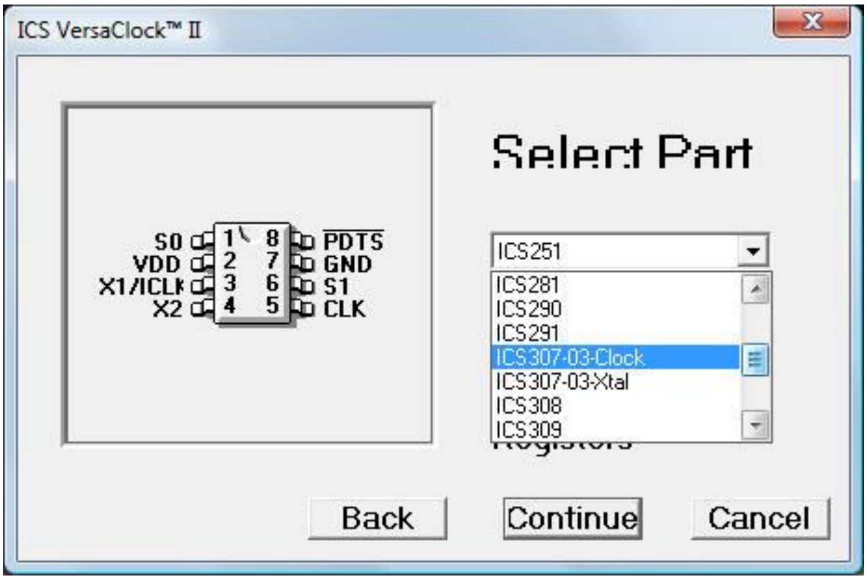
Firstly, go http://www.idt.com and search for the product page on the ICS307G-03LF device, then locate the link for the VersaClock II Desktop Software Setup and download and install the application. The tool itself comes with a full User Manual, but the following notes detail its use specifically with FarSync products.

### A4.1.5 The VersaClock II Programmer

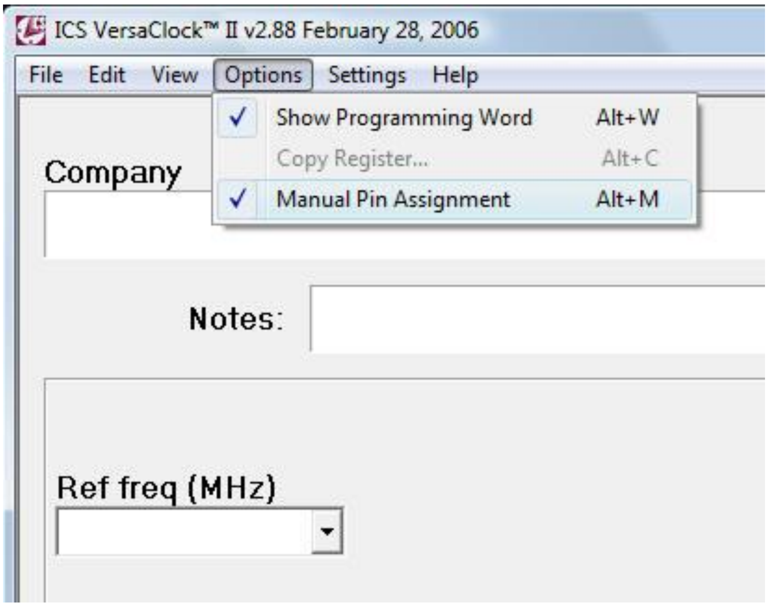
The VersaClock II Desktop application opens with the following screen:


ICS VersaClock™ II v2.88 February 28, 2006
ICS VersaClock™ II Programmer
Select Part Number
Select Prior Session
Exit
Ready

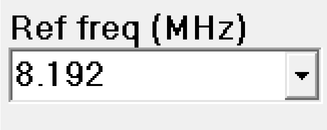
Select the Part Number as shown, then click Continue:



Select Show Programming Word and Manual Pin Assignment from the Options menu:

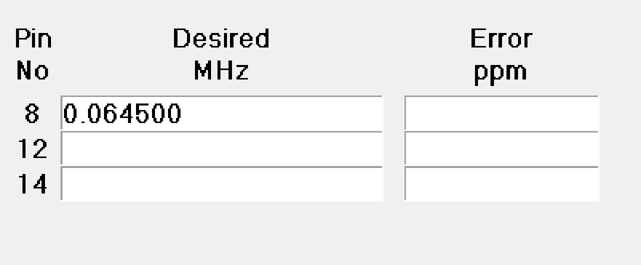

ICS VersaClock™ II v2.88 February 28, 2006
File
Edit
View
Options
Settings
Help
✓ Show Programming Word Alt+W
Copy Register... Alt+C
✓ Manual Pin Assignment Alt+M
Company
Notes:
Ref freq (MHz)

Select a Ref freq (MHz) of 8.192:



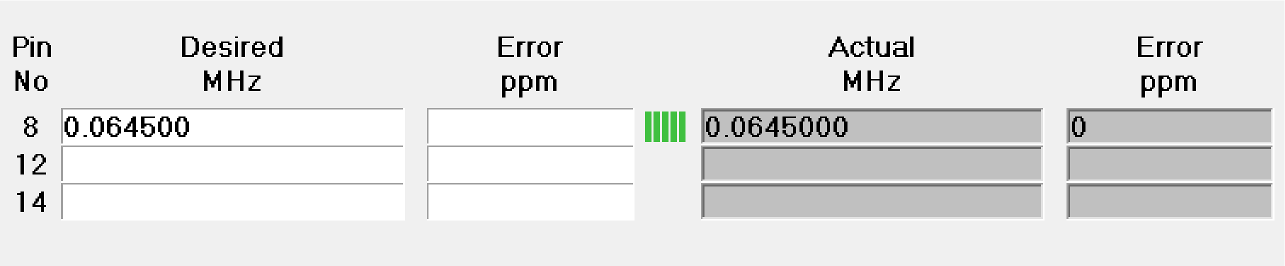
Ref freq (MHz)
8.192

Select the Desired MHz for Pin 8 (64500Hz in this example), leave the Error ppm blank:



| Pin No | Desired MHz | Error ppm |
|---|---|---|
| 8 | 0.064500 | |
| 12 | | |
| 14 | | |

Click on the Calculate Button:

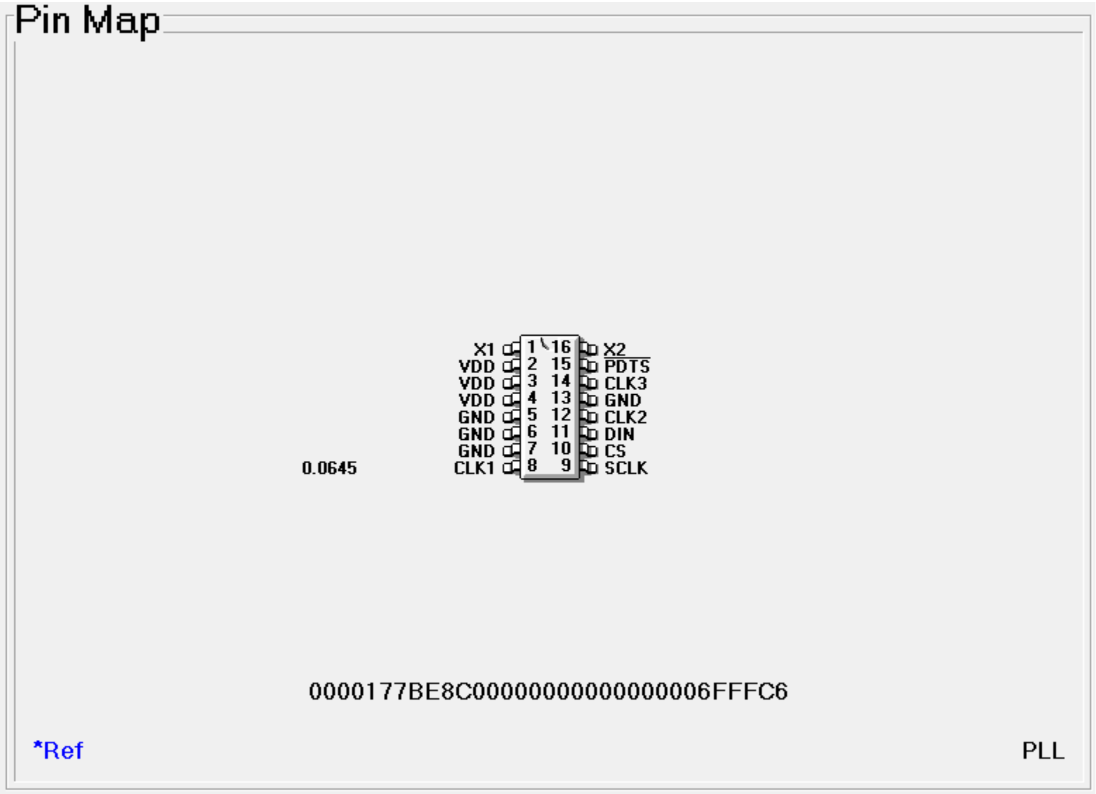


| Pin No | Desired MHz | Error ppm | Actual MHz | Error ppm |
|---|---|---|---|---|
| 8 | 0.064500 | | 0.0645000 | 0 |
| 12 | | | | |
| 14 | | | | |

For a generated clock that closely matches your requirements look for 5 green bars and 0 ppm. Less than 5 green bars and/or more than 0 ppm may be acceptable in some applications, it is left to the user to make this judgment.

NOTE: A minimum frequency of 0.032MHz (32KHz) is recommended at the synthesizer output. To generate lower frequencies, multiply the desired frequency by 256. When using dynamic clock adjustments, it is important that you do not switch between a clock that requires a divide by 256 and one that does not. For dynamic clock adjustment, in order to achieve a glitch-free transition from one clock to another, there should be no change in the synthesizer output divider configuration.

The Pin Map shows the pin out of the device and the calculated 16½ byte Programming Word:



```
0000177BE8C00000000000000006FFFC6
```

Click on the Prog. word to Clipboard button:

Please save this string in a text file for now as it will be required either in a **farutil** command to manually set the clock rate, or you can past it into you application if you are requiring the set the clock rate programmatically.

## A4.2 Setting the Custom Clock Rate Manually

To use **farutil** to set the custom clock rate manually you need to supply two pieces of information:

- the custom clock rate
- the programming word generated above

These are then used in the **farutil** command, for example, as follows:

```
farutil sync0 set clock 64500 custom_rate 0000177BE8C0000000000000006FFFC6
```

The custom clock rate will take effect the next time that the port is opened. Note that it is important that the "clock" keyword appears before the "custom_rate" keyword, as the processing of the "custom_rate" will use the current value of "clock".

## A4.3 Setting the Custom Clock Rate Programmatically

The custom clock rate can also be set through the ioctl() interface to the driver. Please see the FarSync SDK documentation for further details, and you can reference the switch_speed example application to see how this feature can be used to dynamically change the clock rate of the device.

## A4.4 Notes when using Clock Encoding Modes

If you are using and encoding mode where clock synchronisation is encoded in the data, for example FM0, FM1, Manchester etc, then a few of the steps outlined above need to change.

1. You actually need to generate a clock 16 times the rate that you actually want. Therefore, in the example above we need to generate a clock rate of 16*64500 = 1032000

2. The progword for this example is calculated as 00001433E8C0000000000000006FFFC6

3. The Multiplier now needs to be 16 instead of 1. If you are setting the clock rate manually, then **farutil** will know what the encoding mode is and automatically apply the right multiplier. If you are setting the speed programmatically then you need to change:

   ```
   custom_clock_rate.multiplier = FST_CUSTOM_CLOCK_MULTIPLIER_1;
   custom_clock_rate.rate = required_speed;
   ```

   to

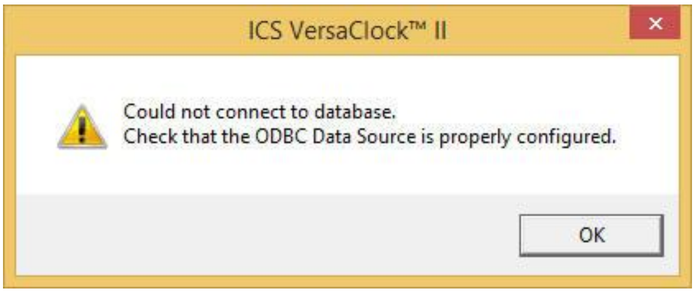
   ```
   custom_clock_rate.multiplier = FST_CUSTOM_CLOCK_MULTIPLIER_16;
   custom_clock_rate.rate[2] = required_speed*16;
   ```

---

[2] Note, however, that when subsequently switching to the required speed i.e. using FSTCMD_UPDATE_CLOCK), the actual value of **required_speed** should be specified to that cmd i.e. not: **required_speed*16**

## A4.5 Compatibility with Windows 8 and later

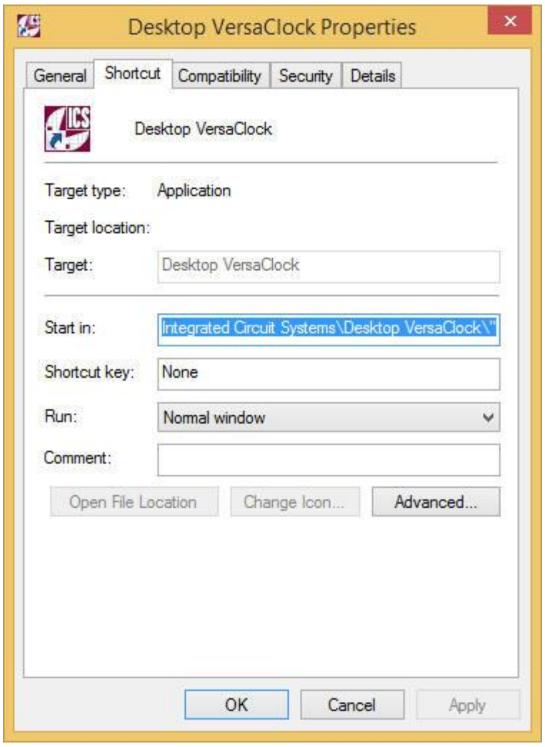
If you receive the following message when you try and run the VersaClock tool from the desktop:



Could not connect to database.
Check that the ODBC Data Source is properly configured.

Then please follow the steps below.

- Right click on the Desktop VersaClock icon on the desktop and select properties from the dropdown list



Desktop VersaClock

Target type: Application

Target location:

Target: Desktop VersaClock

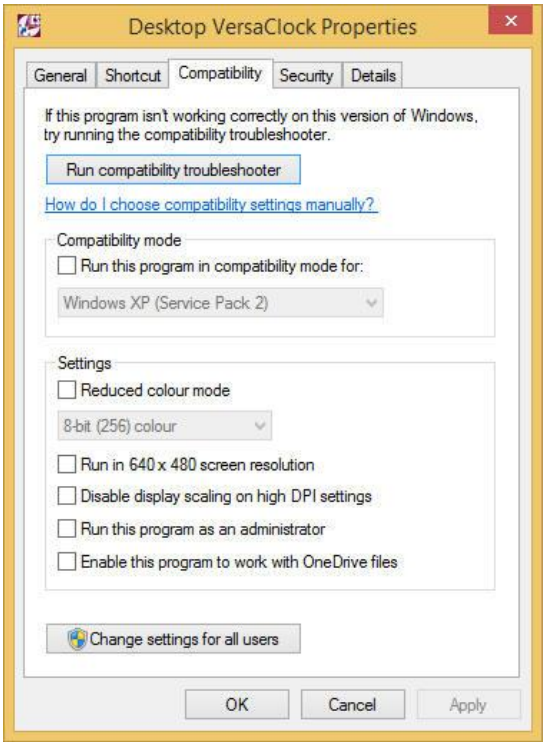
Start in: Integrated Circuit Systems\Desktop VersaClock\"

Shortcut key: None

Run: Normal window

Comment:

- Select the Compatibility tab


Desktop VersaClock Properties
General
Shortcut
Compatibility
Security
Details
Run compatibility troubleshooter
How do I choose compatibility settings manually?
Change settings for all users
OK
Cancel
Apply

If this program isn't working correctly on this version of Windows, try running the compatibility troubleshooter.

Compatibility mode

☐ Run this program in compatibility mode for:

Windows XP (Service Pack 2)

Settings

☐ Reduced colour mode

8-bit (256) colour

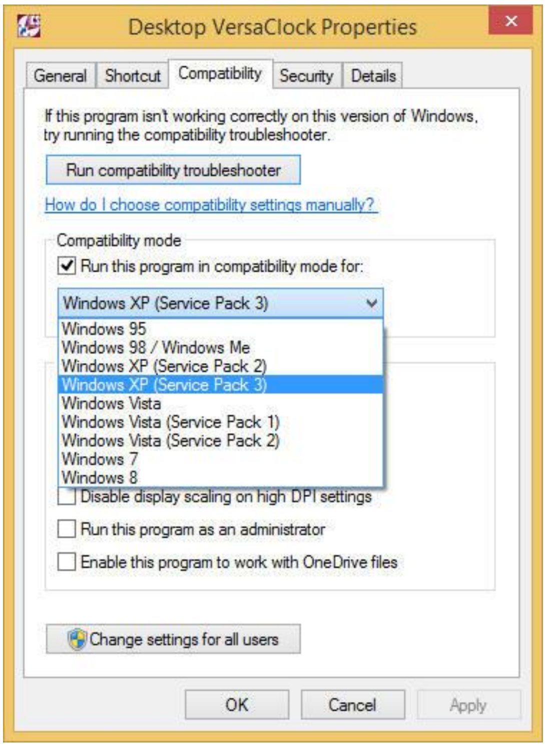
☐ Run in 640 x 480 screen resolution

☐ Disable display scaling on high DPI settings

☐ Run this program as an administrator

☐ Enable this program to work with OneDrive files

- Tick the Compatibility mode box and then make sure that Windows XP (Service Pack3) option is selected from the dropdown list


Desktop VersaClock Properties
General
Shortcut
Compatibility
Security
Details
Run compatibility troubleshooter
How do I choose compatibility settings manually?
Windows 95
Windows 98 / Windows Me
Windows XP (Service Pack 2)
Windows XP (Service Pack 3)
Windows Vista
Windows Vista (Service Pack 1)
Windows Vista (Service Pack 2)
Windows 7
Windows 8
Change settings for all users
OK
Cancel
Apply

If this program isn't working correctly on this version of Windows, try running the compatibility troubleshooter.

Compatibility mode

☑ Run this program in compatibility mode for:

Windows XP (Service Pack 3)

☐ Disable display scaling on high DPI settings

☐ Run this program as an administrator

☐ Enable this program to work with OneDrive files

- Press OK

The application can now be launched from the desktop icon.

# A5 Updating the Driver (fsupdate)

If you have installed the FarSync drivers but have then upgraded the running Kernel, for example after a system update, you will find that the driver modules no longer load. This is because they were built against the previous Kernel and not the new one. However, they just need to be rebuilt against the new Kernel header files. This situation is evident if the current **/usr/src/linux** soft link references a different version compared with that which is reported in the output produced by running the **uname -r** command (as described in Section 4.1.4).

If your system update didn't also update the Kernel header files then you will need to do this before you proceed.

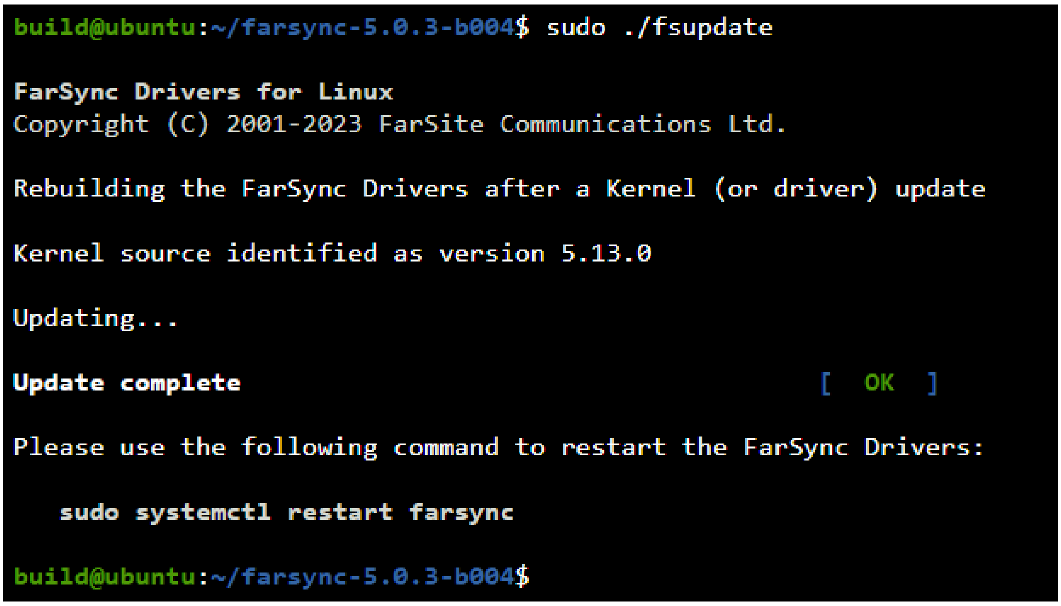
You can perform the update manually or use the supplied update script (fsupdate).

The fsupdate script will perform the appropriate update steps as well as checking that you still have all the required tools available.

Before running the fsupdate script, or performing the update manually, you should

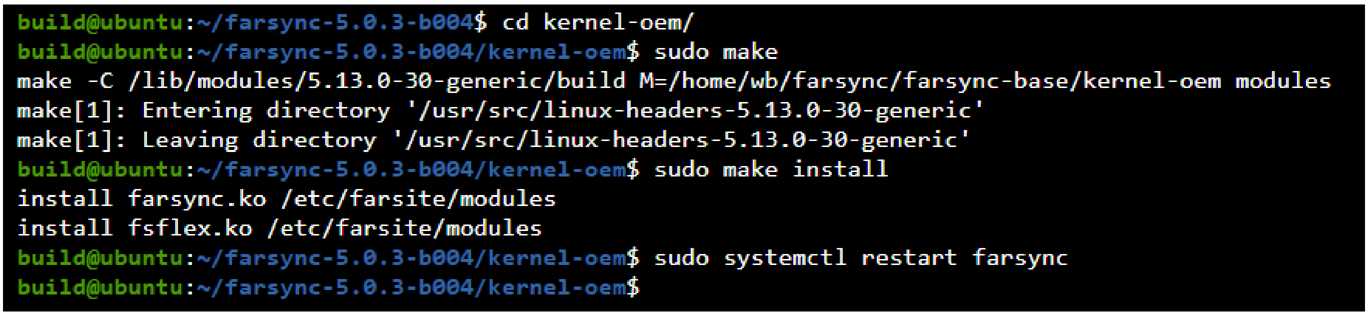
a) delete the old symbolic link (/usr/src/linux) which was pointing to the old Kernel headers or source and then
b) remake it but with it now pointing to the **new** Kernel headers or source – as described in Section 4.1.4.

For example, if you were updating to 5.13.0-30-generic you would need to use the following sequence:



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo ./fsupdate

FarSync Drivers for Linux
Copyright (C) 2001-2023 FarSite Communications Ltd.

Rebuilding the FarSync Drivers after a Kernel (or driver) update

Kernel source identified as version 5.13.0

Updating...

Update complete                                    [  OK  ]

Please use the following command to restart the FarSync Drivers:

   sudo systemctl restart farsync

build@ubuntu:~/farsync-5.0.3-b004$
```

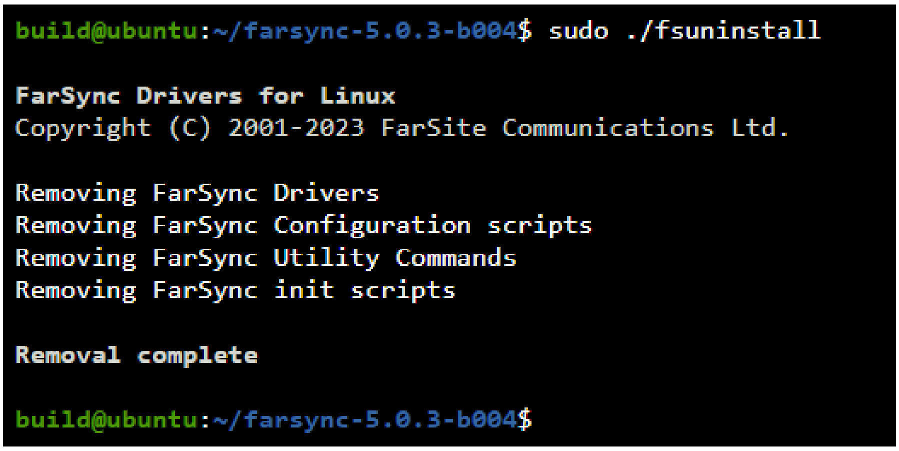
If you prefer to do the update manually then the following set of commands can be used:



```
build@ubuntu:~/farsync-5.0.3-b004$ cd kernel-oem/
build@ubuntu:~/farsync-5.0.3-b004/kernel-oem$ sudo make
make -C /lib/modules/5.13.0-30-generic/build M=/home/wb/farsync/farsync-base/kernel-oem modules
make[1]: Entering directory '/usr/src/linux-headers-5.13.0-30-generic'
make[1]: Leaving directory '/usr/src/linux-headers-5.13.0-30-generic'
build@ubuntu:~/farsync-5.0.3-b004/kernel-oem$ sudo make install
install farsync.ko /etc/farsite/modules
install fsflex.ko /etc/farsite/modules
build@ubuntu:~/farsync-5.0.3-b004/kernel-oem$ sudo systemctl restart farsync
build@ubuntu:~/farsync-5.0.3-b004/kernel-oem$
```

## A6 Uninstalling the Driver (fsuninstall)

If you wish to remove the farsync, flex and farsynck driver then you can run, as root, the fsuninstall scipt as follows:



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo ./fsuninstall

FarSync Drivers for Linux
Copyright (C) 2001-2023 FarSite Communications Ltd.

Removing FarSync Drivers
Removing FarSync Configuration scripts
Removing FarSync Utility Commands
Removing FarSync init scripts

Removal complete

build@ubuntu:~/farsync-5.0.3-b004$
```

Note that this will remove everything that was created or copied when fsinstall was run – see Section 4.2.

## A7 Updating Device Firmware

If you are installing this version of the FarSync Drivers as an upgrade to an existing installation, and you have either a FarSync Flex or K2Ee, then you may wish to update the card/device firmware in order to take advantage of any new features that are available.

The utilities that provide the firmware update support are currently Windows-based and are located in the FarSync Drivers for Linux product image as follows:

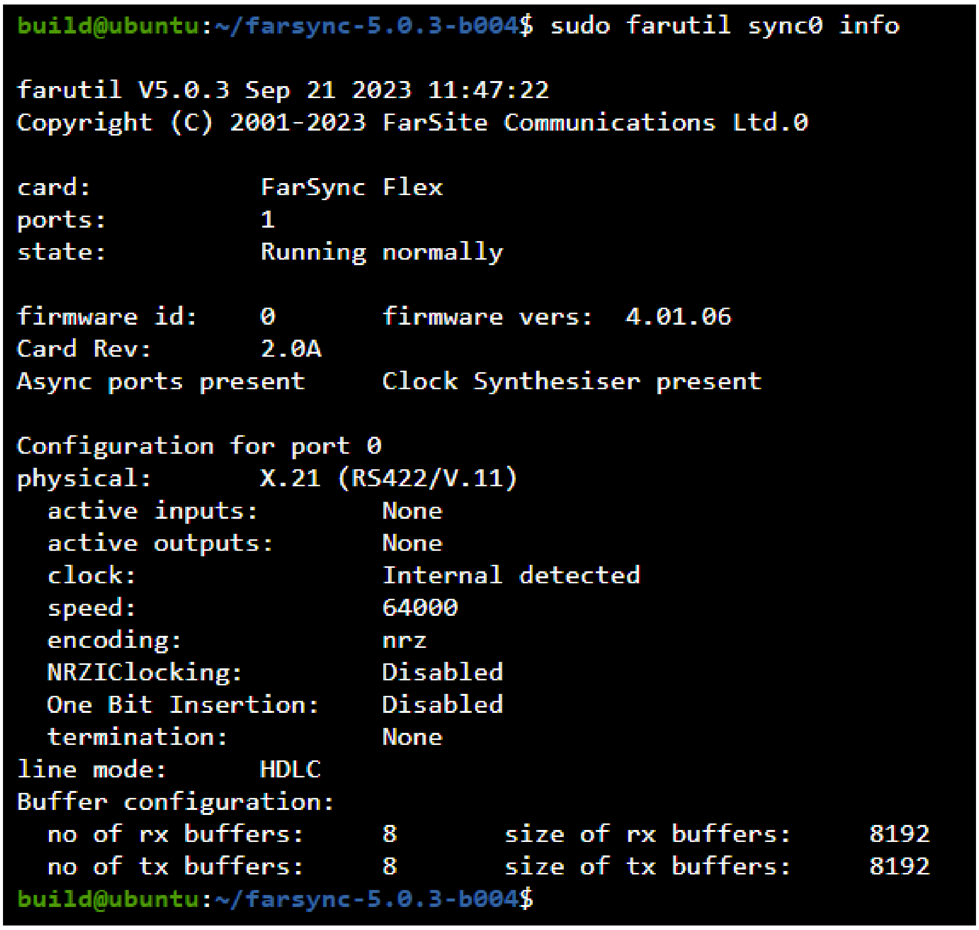
| Card/Device | Folder | Utility | Image Name Format | Instructions |
|---|---|---|---|---|
| FarSync Flex | tools\Windows\F3Flash | F3Flash.exe | FLEX_ARM-xxx-AB.hex | FarSync Flex V3 FLASH Update.pdf |
| FarSync K2Ee | tools\Windows\KFlash | KFlash.exe | FS_K2EeI.hex | FarSync KxEe FLASH Update.pdf |

You may also like to check the Support section of the FarSite Website, www.farsite.com, for a later version.

# A8 Async Operation

Not all FarSync ports support async operation. In general, it is available on the FarSync Flex, the FarSync T4E, T4E+ and T4Ee. The T4Ue and the T4U can be hardware upgraded to include support for async. The **farutil** command will normally indicate if the FarSync device you have supports ports in async mode.

e.g.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farutil sync0 info

farutil V5.0.3 Sep 21 2023 11:47:22
Copyright (C) 2001-2023 FarSite Communications Ltd.0

card:           FarSync Flex
ports:          1
state:          Running normally

firmware id:    0       firmware vers:  4.01.06
Card Rev:       2.0A
Async ports present     Clock Synthesiser present

Configuration for port 0
physical:       X.21 (RS422/V.11)
  active inputs:        None
  active outputs:       None
  clock:                Internal detected
  speed:                64000
  encoding:             nrz
  NRZIClocking:         Disabled
  One Bit Insertion:    Disabled
  termination:          None
line mode:      HDLC
Buffer configuration:
  no of rx buffers:     8       size of rx buffers:     8192
  no of tx buffers:     8       size of tx buffers:     8192
build@ubuntu:~/farsync-5.0.3-b004$
```

This section gives some guidance of using the FarSync ports in async mode.

## A8.1 Switching to Async Mode

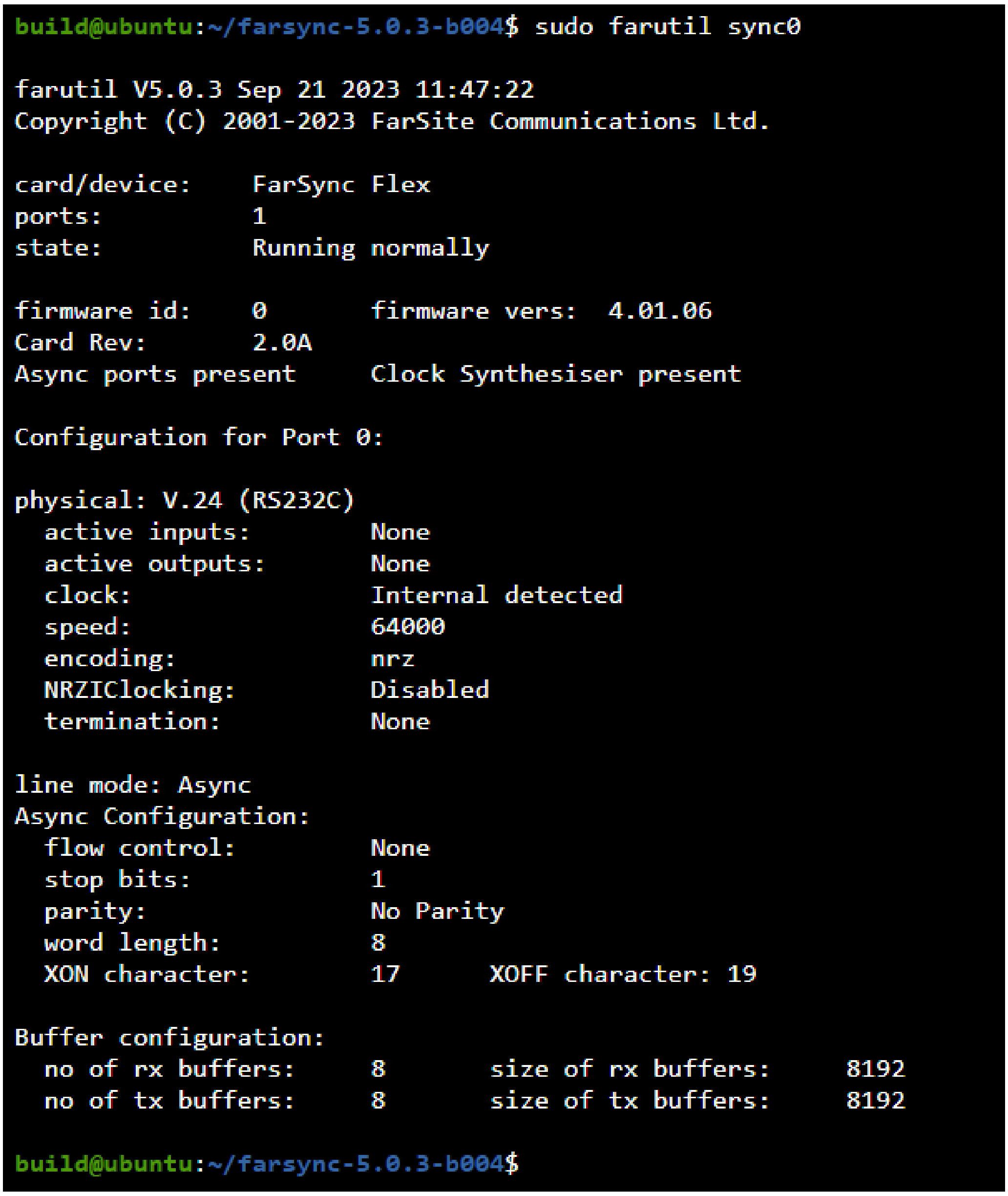
When the FarSync Flex device is first initialised it defaults to the synchronous HDLC mode of operation. In order to start using it as an asynchronous port, it must first be switched into async mode. The media type may also need to be changed to v24. This can be done with the following **farutil** command.

```
farutil <device> set linemode async media v24
```

the device name will typically be sync0.

## A8.2 Checking/Setting Async Parameters with farutil

When the port is in async mode, some additional parameters are displayed when the **farutil** command is invoked.



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo farutil sync0

farutil V5.0.3 Sep 21 2023 11:47:22
Copyright (C) 2001-2023 FarSite Communications Ltd.

card/device:    FarSync Flex
ports:          1
state:          Running normally

firmware id:    0       firmware vers:  4.01.06
Card Rev:       2.0A
Async ports present     Clock Synthesiser present

Configuration for Port 0:

physical: V.24 (RS232C)
  active inputs:        None
  active outputs:       None
  clock:                Internal detected
  speed:                64000
  encoding:             nrz
  NRZIClocking:         Disabled
  termination:          None

line mode: Async
Async Configuration:
  flow control:         None
  stop bits:            1
  parity:               No Parity
  word length:          8
  XON character:        17      XOFF character: 19

Buffer configuration:
  no of rx buffers:     8       size of rx buffers:     8192
  no of tx buffers:     8       size of tx buffers:     8192

build@ubuntu:~/farsync-5.0.3-b004$
```

The possible setting for the various parameters are as follows:

- flow control: none or hardware or software
- stop bits: 1 or 1.5 or 2
- parity: none or odd or even
- word length: 5 or 6 or 7 or 8
- XON Character: do not use
- XOFF character: do not use

To set any of these parameters with the **farutil** command use the following syntax:

**farutil <device> set flow_control <value> | stop_bits <value> | parity <value> | word_len <value>**

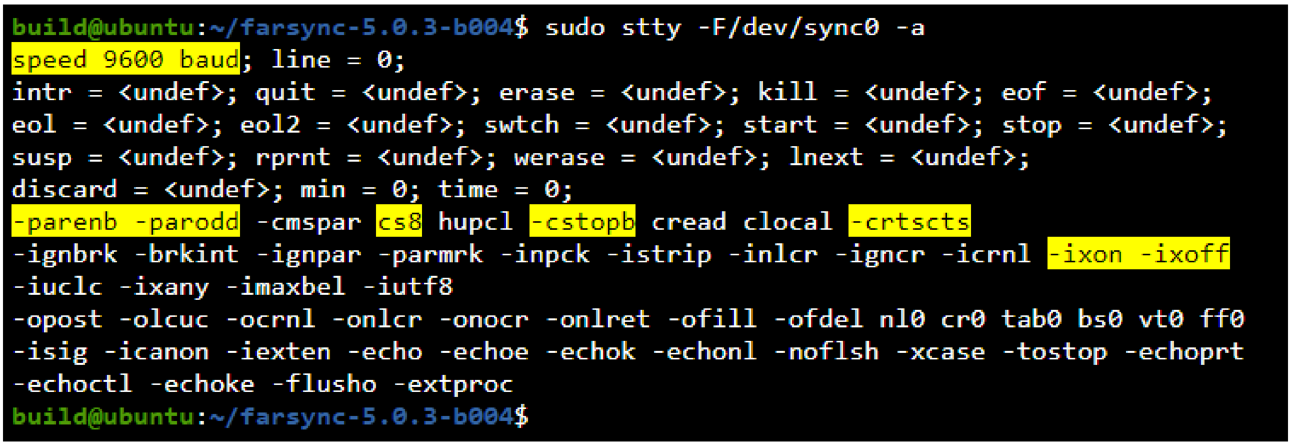
e.g.
**farutil sync0 set linemode async flow_control hardware stop_bits 1 parity none word_len 8**

## A8.3 Using stty

To set and inspect the async port parameters the **stty** command can be used.  The format of the command to see all the async port parameters would be:

```
stty -F/dev/sync0 -a
```

Which will return something like the following:



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo stty -F/dev/sync0 -a
speed 9600 baud; line = 0;
intr = <undef>; quit = <undef>; erase = <undef>; kill = <undef>; eof = <undef>;
eol = <undef>; eol2 = <undef>; swtch = <undef>; start = <undef>; stop = <undef>;
susp = <undef>; rprnt = <undef>; werase = <undef>; lnext = <undef>;
discard = <undef>; min = 0; time = 0;
-parenb -parodd -cmspar cs8 hupcl -cstopb cread clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr -icrnl -ixon -ixoff
-iuclc -ixany -imaxbel -iutf8
-opost -olcuc -ocrnl -onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
-isig -icanon -iexten -echo -echoe -echok -echonl -noflsh -xcase -tostop -echoprt
-echoctl -echoke -flusho -extproc
build@ubuntu:~/farsync-5.0.3-b004$
```

The parameters of significance to the async port are highlighted. The output shows that the line rate is set for 9600 bps and that parity is disabled. The number of bits per byte is set as 8, and there are 2 stop bits. Both Hardware and XON/XOFF flow control are disabled.

The stty command can be used to set any of these parameters. Use the "man stty" command for details on the command syntax.

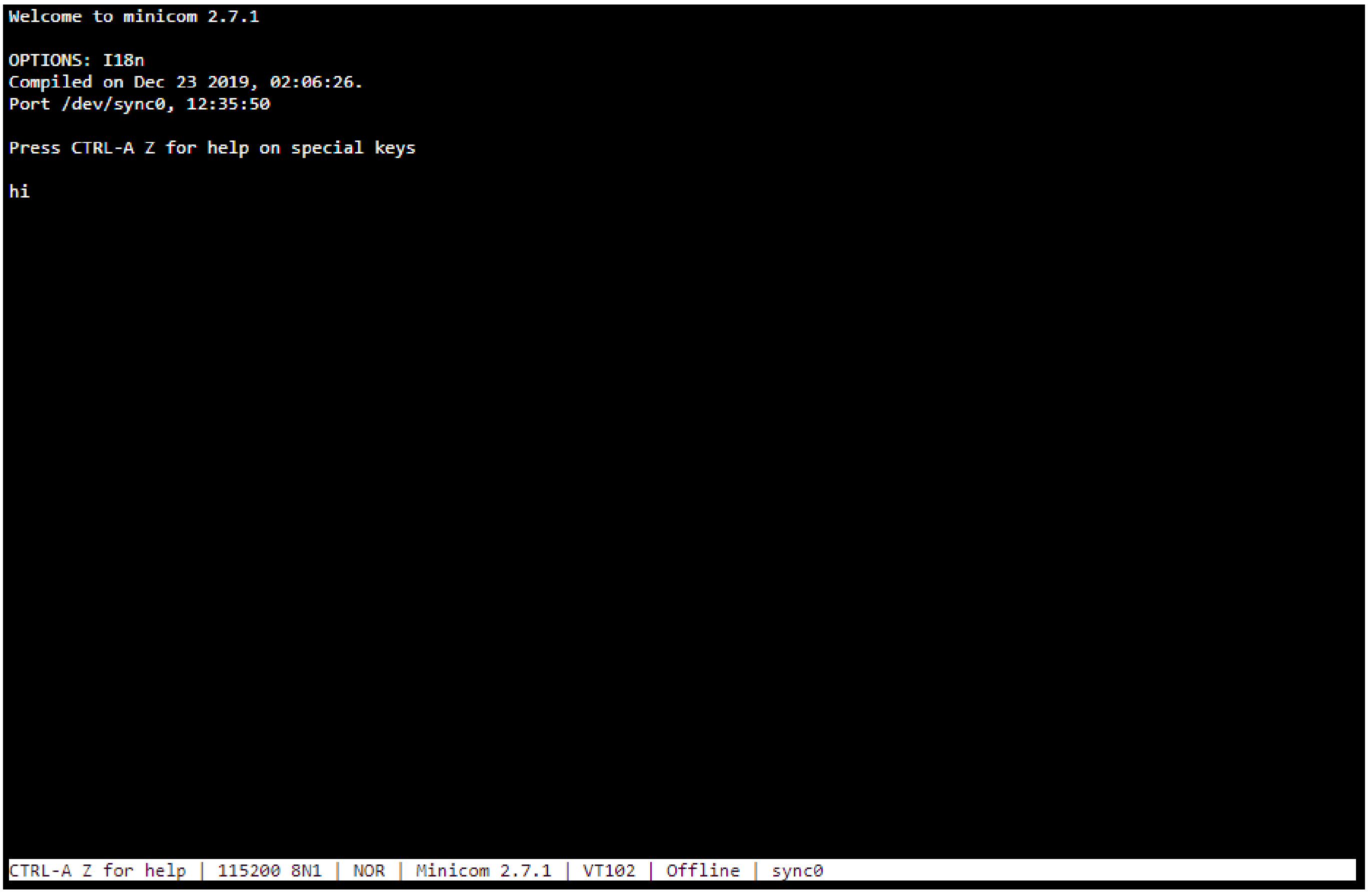
## A8.4 Using the Async port with minicom

To use the port with a Linux terminal application such as **minicom**, the port must first be set in async mode.  You may also need to set the media type to v24.  Once this has been done the **minicom** application can be started.

```
farutil sync0 set linemode async media v24
```

```
minicom -s
```

You can then use the **minicom** menus system to set the required serial port parameters.  When you have changed the settings don't forget to save them too.  Once they have been saved, **minicom** can subsequently be started specifying the corresponding configuration file in order to reuse the saved port parameter settings.



```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/sync0, 12:35:50

Press CTRL-A Z for help on special keys

hi
```

```
CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Offline | sync0
```

## A8.5 Async ioctls supported

The following "standard" serial port ioctls are supported:

| | |
|---|---|
| TCGETS | Get serial port parameters |
| TCSETS | |
| TCSETSW | Set the following serial port parameter |
| | baud |
| | char_size |
| | stop_bits |
| | Parity |
| | rtscts |
| | xonxoff |
| TCSETSF | Not yet supported |
| TIOCGSERIAL | Not yet supported |
| TIOCSSERIAL | Not yet supported |
| TIOCMGET | Get control signals |
| TIOCMBIS | Not yet supported |
| TIOCMBIC | Not yet supported |
| TIOCMSET | Set control signals |
| TCFLSH | Not yet supported |

In general, you should be able to configure the serial port parameters as described above either with the terminal application (if it supports configuration) or with the **stty** command.

# A9 Troubleshooting

## A9.1 Compilation errors

**Problem**

There are compile errors during the install process for the fsflex.c and farsync.c driver source files.

**Resolution**

From time to time the internal Kernel API's change. This usually requires patches to be applied to the drivers to resolve the issue.

Check to see if these patches are available on the support pages of the web site, or there is not patch available that fixes the error then please send the details of the error to support@farsite.com

## A9.2 Unable to load driver

**Problem**

The farsync or flex driver driver does not load e.g. because there are unresolved load time references.

**Resolution**

This normally means that the required modules haven't been loaded yet.

Use **dmesg** or **cat /var/log/messages** to identify the unresolved references. This will indicate which module hasn't been loaded that the driver requires.
Check the install documentation for details on how to load the additional modules required. This may require updating the **/etc/init.d/farsync** script that loads the driver(s).

If this does not resolve the issue, please contact support@farsite.com

## A9.3 The driver cannot find the FarSync device(s)

**Problem**

The farsync or flex driver driver does not find the FarSync Devices

**Resolution**

| | |
|---|---|
| The PCIe slot is which the FarSync PCIe adapter has been installed is damaged.<br>The USB port that the FarSync Flex adapter has been connected to is damaged. | Try installing the adapter in another PCIe slot or USB port. |
| The adapter is not making good electrical contact with the PCIe slot. | Make sure that the contacts are clean. If necessary, also try installing the adapter in another slot. |
| The FarSync driver files are somehow corrupted or missing. | Uninstall and reinstall the FarSync software. |

| | |
|---|---|
| The installation of the software failed in some way. | See further details for checking the installation. |
| The FarSync PCIe adapter or Flex device is faulty. | The FarSync cards and Flex USB devices are very reliable - the adapter or device  is very unlikely to be faulty. Try downloading and running the FarSync card test software to check the card (requires DOS). If, however, you do believe it to be faulty please contact FarSite to discuss potentially returning the adapter to FarSite for repair or replacement. |

## A9.4 farutil reports “No such device”

**Problem**

Running

```
farutil <device>
```

reports the following error:

```
farutil: Problem getting port information. No such device
```

**Resolution**

Either the required (farsync, fsflex or farsynck) driver isn’t loaded or the specific device is not physically present.

Check the required driver is available (see A9.1).

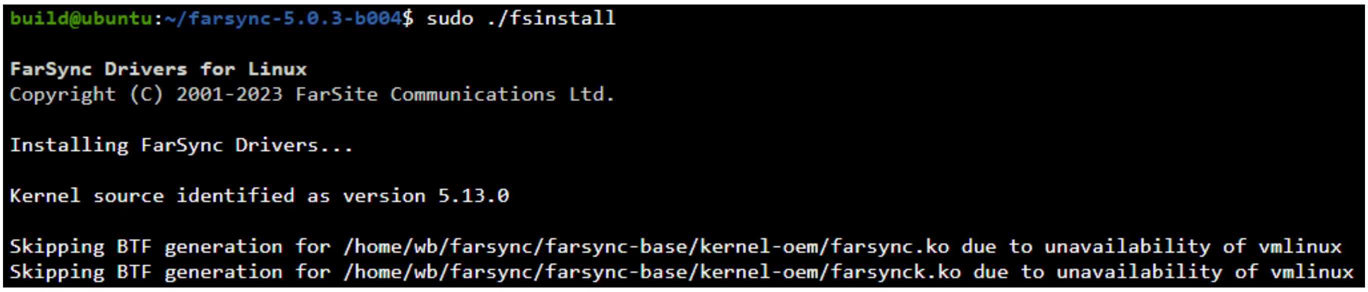
Check the driver has been loaded (see A9.2).

If the problem persists, please contact support@farsite.com

## A9.5 BTF generation error

**Problem**

The following BTF generation error is displayed, for example, when trying to install the drivers on Ubuntu 20.04



```
build@ubuntu:~/farsync-5.0.3-b004$ sudo ./fsinstall

FarSync Drivers for Linux
Copyright (C) 2001-2023 FarSite Communications Ltd.

Installing FarSync Drivers...

Kernel source identified as version 5.13.0

Skipping BTF generation for /home/wb/farsync/farsync-base/kernel-oem/farsync.ko due to unavailability of vmlinux
Skipping BTF generation for /home/wb/farsync/farsync-base/kernel-oem/farsynck.ko due to unavailability of vmlinux
```

**Resolution**

Issue the following sequence of commands:

- sudo apt install dwarves
- sudo cp /sys/kernel/btf/vmlinux /usr/lib/modules/5.13.0-30-generic/build

and then rerun fsinstall.

Note: In the above 5.13.0-30-generic is what has been returned by **uname -r**

## A9.6 IRQ Routing/APIC Mode

System IRQ Routing issues can, in some rare circumstances, result in interrupts not being able to get through to the FarSync driver. This can *potentially* be due to problems with the system's APIC (Advanced Programmable Interrupt Controller).

Try disabling the APIC, by booting the system using the **noapic** kernel parameter.

The method for doing this will depend on the Linux distribution and boot loader. Below is an example for openSUSE 15.4:

1) Open **/etc/default/grub** for editing

   Go to the line that starts with GRUB_CMDLINE_LINUX_DEFAULT =

   ```
   GRUB_CMDLINE_LINUX_DEFAULT=”splash=silent preempt=full mitigations=auto quiet security=apparmor”
   ```

2) Add **noapic** to the end of the line

   ```
   GRUB_CMDLINE_LINUX_DEFAULT=”splash=silent preempt=full mitigations=auto quiet security=apparmor noapic”
   ```

3) Save the file

4) Update the bootloader using

   ```
   sudo update-bootloader −−refresh
   ```

5) Reboot the system

Once rebooted, you can confirm that the system has booted using the noapic option, by using **cat /proc/cmdline**, which then displays the boot command line.

Once you have confirmed that the system is in non-APIC mode, determine if the interrupt issue still remains.

Please contact support@farsite.com for further assistance.

FarSite COMMUNICATIONS

+44 (0)1256 330 461

info@farsite.com

www.farsite.com