

Aculab digital network access cards
Telephony software installation guide

MAN1761 Revision 6.7.2



PROPRIETARY INFORMATION

The information contained in this document is the property of Aculab plc and may be the subject of patents pending or granted, and must not be copied or disclosed without prior written permission. It should not be used for commercial purposes without prior agreement in writing.

All trademarks recognised and acknowledged.

Aculab plc endeavours to ensure that the information in this document is correct and fairly stated but does not accept liability for any error or omission.

The development of Aculab's products and services is continuous and published information may not be up to date. It is important to check the current position with Aculab plc.

Copyright © Aculab plc. 2004-2019 all rights reserved.

Document Revision

Rev	Date	By	Detail
6.0	06.12.02	CJL	Interim release
6.1.0A	19.06.03	DJL	Controlled release of initial revision 6 software
6.1.0	15.01.04	DJL	updated installation and configuration tools
6.1.1	18.06.04	DJL	Review updates
6.2.2	07.09.04	DJL	Beta release
6.2.2	15.09.04	DJL	Full release
6.2.3	28.10.04	DJL	Updates for Linux
6.2.4	12.11.04	DJL	Addition of SS7 signalling link and ISUP bearer ACT info.
6.3.0	24.12.04	DJL	Various updates including support for new hardware
6.3.1	26.01.05	DJL	Corrections to appendix B
6.3.2	10.02.05	DJL	Removal of Solaris restrictions
6.3.3	22.02.05	DJL	Addition of Linux/Solaris configuration notes
6.3.4	19.04.05	DJL	Correction of typographical errors
6.3.5	26.05.05	DJL	Review of ACT functions
6.4.0	03.11.05	DJL	Updates for V6.4 release
6.4.1	18.01.06	DJL	Correction of ACT images/definitions
6.4.2	10.04.06	DJL	Review updates – small changes
6.4.3	06.07.06	DJL	Update to document cross reference descriptions
6.4.4	15.08.06	DJL	Update to Prosody X information
6.4.5	21.08.06	DJL	New example added – installing a Prosody X card.
6.4.6	05.10.06	DJL	Updates to Linux installation example
6.4.7	14.12.06	DJL	Updates to include Solaris specific information
6.4.8	27.09.07	EBJ	Removal of Download and install of the Aculab telephony-software and Telephony-software download and installation.
6.4.9	21.09.09	DRG	Updated Aculab Configuration Tool (ACT) screenshots.

Rev	Date	By	Detail
6.4.10	02.11.09	EBJ	Updated to corporate fonts, reformatted and minor corrections.
6.6.0	17.03.15	BWM	Updated for 6.6 release, incorporating ipv6
6.7.0	11.04.17	PGD	Updated supported O/S list.
6.7.1	06.09.18	PGD	Delete Fetex, update O/S list
6.7.2	29.11.19	PGD	Update O/S support list, correct a weblink

CONTENTS

1	Installation and Role of V6 Telephony Software.....	6
2	Installation process overview.....	7
2.1	Install the Aculab hardware.....	7
2.2	Host system startup	7
2.2.1	Prosody X PCIe cards installed in controlling host	7
2.2.2	Prosody X media processing chassis and remote ProsodyX PCIe cards	8
2.3	Downloading and installing the V6 telephony software	9
2.4	Configuring the Aculab Telephony Software	9
2.5	Aculab example code	9
3	Example card installation and configuration for a Windows environment	10
3.1	Installing the Prosody X card hardware.....	10
3.2	Starting the Prosody X card host system.....	10
3.3	Adding the card to the Aculab card list.....	11
3.3.1	Setting up the Ethernet TCP/IP card address	11
3.3.2	Updating the Aculab card list	12
3.4	Configuring the Aculab telephony-software	15
3.4.1	Card Details.....	15
3.4.2	Clocking settings.....	18
3.4.3	IP Settings	19
3.4.4	TiNG Settings	19
3.4.5	Applying the configuration.....	21
4	Example installation and configuration with command line.....	23
4.1	Linux drivers	23
4.1.1	Pre-requisites	23
4.1.2	Making Libraries and Supporting Utilities	23
4.1.3	Building, and Configuring, the Driver.....	23
4.1.4	Creating the Driver.....	24
4.1.5	Configuring the Driver	24
4.1.6	Loading and Unloading the Driver.....	24
4.1.7	Automatic Loading of the Driver.....	25
4.1.8	Removing the Driver	25
4.1.9	Setting up Prosody X Card IP Address on Linux	25
4.2	Managing Prosody X cards	25
4.2.1	Distinguishing Prosody X cards on Linux	25
4.2.2	Adding a Remote Prosody X card.....	26
4.2.3	Adding a Local Prosody X Card	26
4.2.4	Listing installed Prosody X cards	26
4.2.5	Obtaining information for an installed Prosody X card.....	27
4.2.6	Removing a Prosody X card	27
4.3	Card configuration files	27
4.3.1	Creating configuration files	28
4.3.2	Per-card configuration file description	28
4.3.3	Updating Card Configuration	33
4.4	Signalling Software Download.....	33
	Appendix A: Call Driver Installation Options.....	35

A.1	Table of Signalling System Options	35
A.2	System Specific Option definitions	36
A.2.1	-cA/B and X/Y bits configuration (DPNSS only)	36
A.2.2	-cBBY Backbusy control (CAS only)	36
A.2.3	-cCA connect acknowledgement (ETS300 only)	36
A.2.4	-cCICnnnn circuit identification codes (ISUP only)	36
A.2.5	-cCn number of CLI digits (CAS only)	37
A.2.6	-cDn number of DDI digits (CAS only)	37
A.2.7	-cDPCnnnnn signalling point code (ISUP only)	37
A.2.8	-cEn Call Charging Switch (ETS300 only)	37
A.2.9	-cEX Primary Rate Call Charging (ETS300 only)	38
A.2.10	-cFD Diversion (QSIG, ETS300, AT&T T1, and NI-2)	39
A.2.11	-cFF facility (QSIG, ETS300, and NI2)	39
A.2.12	-cFP MLPP activation (ETS300 only)	39
A.2.13	-cFR Raw Data (ETS300 and QSIG)	39
A.2.14	-cFU user to user (QSIG and ETS300)	39
A.2.15	-cIMP75	39
A.2.16	-cME	39
A.2.17	-cNA1 release link trunk (NI-2 only)	39
A.2.18	-cNCRC disable CRC4 (ISUP only)	39
A.2.19	-cNE network end config (DASS, ETS300, AT&T, and NI2)	40
A.2.20	-cOPCnnnnn[,i] (ISUP only)	40
A.2.21	-cQM/S-A/B master/slave priority (QSIG only)	40
A.2.22	-cRn Default clearing cause (all versions)	40
A.2.23	-cSLCnn signalling link code (ISUP only)	40
A.2.24	-cSO disable service message (AT&T and NI2)	41
A.2.25	-cSP stop call proceeding (ETS300 only)	41
A.2.26	-cSU stop setup acknowledge (ETS300 only)	41
A.2.27	-cSW configuration (ETS300 only)	41
A.2.28	-cTS[i]nn (ISUP only)	41

1 Installation and Role of V6 Telephony Software

Aculab Prosody X cards and media processing chassis require a controlling host running associated “V6” Telephony software to configure and manage them. Following physical card installation or attachment of media processing chassis to network, this V6 software must be installed on the controlling host system and then configured to set up use of Prosody X resources in the desired way. The same process is used to install and configure a Prosody S host media processor, and to install the services and libraries that provide API services to applications.

V6 software installation is achieved through use of Aculab Installation Tool which is freely downloadable package from the Aculab website. The package provides both a graphical user interface (AIT_GUI) and command line (AIT_CMD) tool to install V6 software either over the network from an Aculab distribution server or from a local distribution file. The resulting installed V6 software will include device drivers, management and configuration tools, firmware to be loaded to Prosody X hardware, services and libraries for interaction with applications, and licence management software. On Linux based systems some scripts will need to be run to start this software running prior to configuration, whereas with Windows based systems it will start running following successful installation.

Once V6 software has been installed and is running, in order to bring ProsodyX resources into an operational state, some operating system network related settings may need to be changed, and then Prosody X elements will require configuration either through use of the graphical Aculab Configuration Tool (ACT) or through use of configuration text files and various command line tools. The configuration process will include such steps as selection of IP addresses to be used, signalling systems for E1/T1 trunks, system clock source, and TiNG algorithms to download onto ProsodyX DSPs. Finally the same tools can be used to verify the Prosody X card or media processing chassis has up-to-date flash programming.

This document provides an overview of the above process and how it is realised for Aculab supported operating systems. It does not cover the regulatory requirements and only gives a brief description of card hardware installation. Please refer to the appropriate hardware installation guides for further details.

The V6 telephony software Aculab configuration tool (ACT) supports the following operating systems:

- Windows Server 2019
- Windows Server 2016
- Linux (64 bit platforms, please check with Aculab for confirmation of the supported distributions)

Procedures are only described for Prosody X PCIe (rev3) telephony card and Prosody X media processing chassis, older products no longer supported by current V6 software distributions are not covered. For additional information regarding Prosody S configuration using same tools see Prosody S User Guide.

2 Installation process overview

Installing and configuring Aculab hardware and any associated telephony software involves a number of key steps:

Install the Aculab hardware.

Start the Aculab hardware host system.

Download the required Aculab telephony-software distribution package.

Configure the Aculab telephony-software.

Once you have completed these steps, you will be able to utilise the Aculab card telephony software and hardware resources through the various Aculab application programming interfaces (APIs).

2.1 Install the Aculab hardware

Install the Aculab hardware in accordance with the appropriate hardware installation guide.

2.2 Host system startup

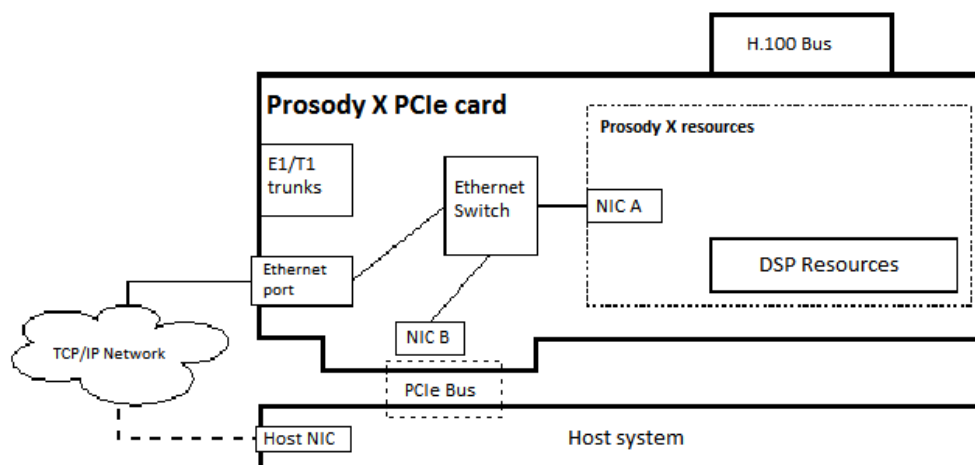
2.2.1 Prosody X PCIe cards installed in controlling host

Unlike earlier Aculab cards, which used bespoke communication between the card processing elements and PCIe (host system), with Prosody X this communication now uses TCP/IP Ethernet protocols managed through an Ethernet switch on the base card.

NOTE

Please refer to the 'card boot & discovery' section of the 'Prosody X administration guide' for further guidance on Prosody X boot-up and network configuration considerations.

1. Install the cards into the required system as detailed in the appropriate hardware manuals.
2. Power up the system
3. Once Windows has started, the system will auto-detect the new hardware as an Ethernet card.



The Prosody X card is detected by the host system as a network interface card (shown as NIC B). Communication with the Prosody X card media DSP and signalling resources (integrated PMX) is also via an Ethernet connection (shown as NIC A).

Configuration of NIC B is via the host system; for example, using Microsoft Windows network connections.

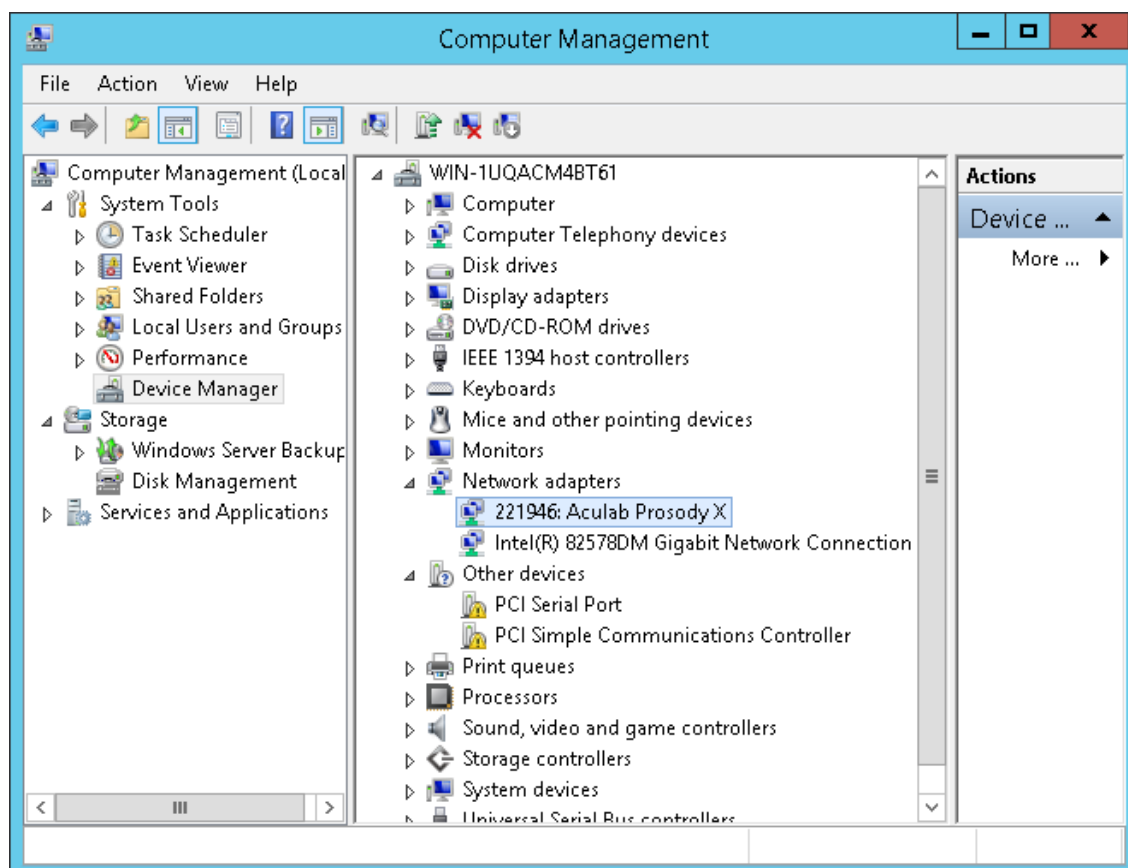
Configuration of NIC A is via either the ACT or the `prosody_ip_card_mgr` command.

The Prosody X card contains an Ethernet switch, which connects NIC A with both the local host system via NIC B, and a TCP/IP network via the cards Ethernet port.

Management of the Prosody X card resources can therefore be carried out either by the local host via NIC B, or by a remote host via the TCP/IP network Ethernet port.

NOTE

As Windows recognises the Prosody X card as a network adapter (NIC) and not an Aculab Computer Telephony device, it will not appear automatically in the ACT card list. It must be manually added to the list before it can be configured.



2.2.2 Prosody X media processing chassis and remote ProsodyX PCIe cards

These just need to be powered up and be reachable on the same subnet as the designated controlling host. Assignment of chassis/card to a particular controlling host and selection of the chassis/card IP addresses is achieved either through the ACT or the `prosody_ip_card_mgr` command where the chassis/card will be referred to by its serial number.

2.3 Downloading and installing the V6 telephony software

The required V6 telephony software is available for download from the Aculab company web site at www.aculab.com. To assist you with downloading and installation, Aculab provide a GUI tool called the Aculab installation tool (AIT). The tool and its usage instructions for the AIT application can be found in the Windows/Linux tools tab of the download page on the Aculab website. The non-windows command line tool, AIT_CMD, is detailed in section 4.1 of this document.

The AIT application can be run on most systems, however Internet access would normally be required to enable you to select and download the individual V6 telephony software components required for your particular system configuration. Should you require to run the Aculab installer on a stand alone system, the AIT has an option to create an Aculab Package File (APF), which can be used to install software on a system without Internet connectivity.

AIT application installation options

AculabInstaller.msi - running this version will install a copy of the AIT application only onto your system. The default location being `C:/Program Files/aculab/installer`

Package*.tgz - For Linux systems, uncompress the content of this file into a suitable location

An example of the use of the AIT is detailed in section 3.3 of this document

NOTE

AIT Telephony-software packages contain some utilities and applications that are included either for use by the Aculab tools or for Aculab diagnostic and test purposes. Documentation for them is not supplied because it is intended that they be used only as instructed by Aculab Support.

2.4 Configuring the Aculab Telephony Software

The process of configuring firmware required to utilise the Aculab card resources is detailed in the [Aculab configuration tool](#) (ACT) user guide.

Sections 3 and 4 of this document provide examples of a typical Aculab card installation and configuration.

2.5 Aculab example code

The AIT includes a package called 'example code'. This package contains generic examples of applications developed for test purposes only. These examples may be of assistance when getting started with the Aculab APIs. For further information, please see the Aculab SDK documentation that accompanies the example code files.

3 Example card installation and configuration for a Windows environment

For this example, we will demonstrate the installation of a Prosody X PCIe card into a host system running the Windows operating system. Similar processes will apply to Linux.

3.1 Installing the Prosody X card hardware.

Proper electrostatic discharge (ESD) procedures should be maintained throughout.

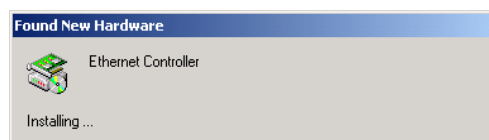
- Step 1. MAKE A NOTE OF THE CARD SERIAL NUMBER
- Step 2. Remove the power from the host system and disconnect any mains leads
- Step 3. Remove the host computer covers.
- Step 4. Locate a vacant full-length PCIe slot and, if required, remove the blanking plate.
- Step 5. Fit the card to the PCIe slot and screw the bracket to the chassis.
- Step 6. Ensure that adjacent devices/cards do not foul the Prosody X card.
- Step 7. If required, fit H.100 CTBus ribbon cable.
- Step 8. Replace the covers on the host computer.
- Step 9. Attach the mains leads ready to apply power.

CAUTION

In order to retain a good earth connection to the card, it is important that you ensure that the card retaining screw, as detailed in step 5 above, is securely fitted.

3.2 Starting the Prosody X card host system.

- Step 10. Turn on the host system, once Windows has started it will automatically detect the new hardware:



and present a **Found New Hardware Wizard** dialog:



Step 11. As the Aculab cards driver is Aculab specific, it will not be available until you have downloaded the Aculab telephony software, therefore select **Cancel**.

For further guidance, please refer to the *boot & discovery* section of the *Prosody X administration guide*.

3.3 Adding the card to the Aculab card list

As previously mentioned, the Prosody X card is recognised by the host system as a network interface card (NIC) and not an Aculab resource card. Before you can configure the Prosody X card resources, you must first make it known to the Aculab card list, which is referenced by the [Aculab configuration tool](#) (ACT). The ACT has an option to add Prosody X cards to the card list using the cards IP address and card serial number.

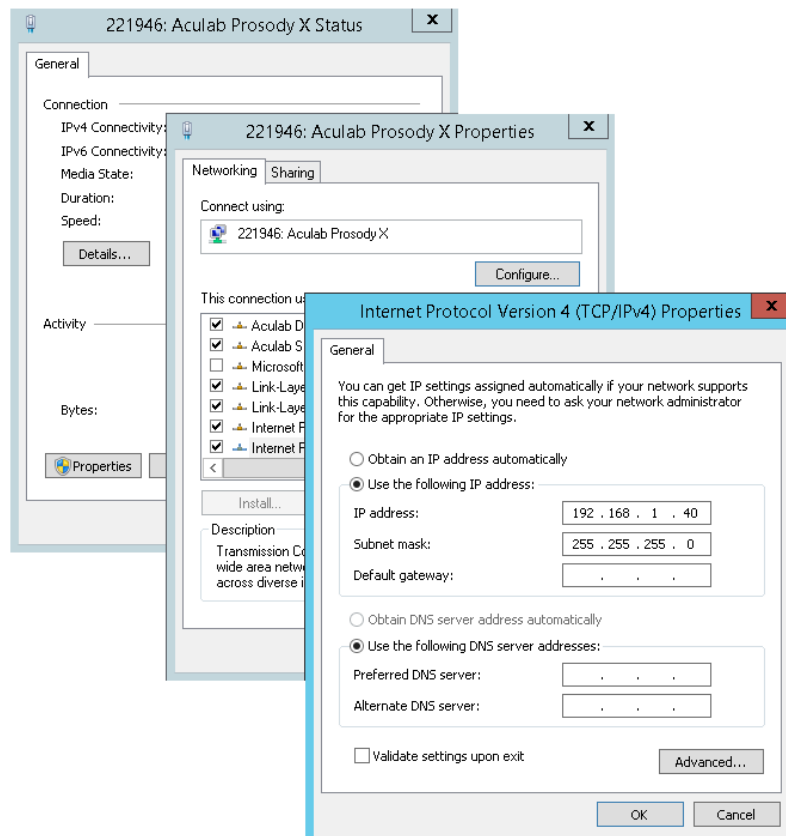
For further guidance, please refer to the *boot & discovery* section of the *Prosody X card Administration guide* for the booting and discovery of cards.

3.3.1 Setting up the Ethernet TCP/IP card address

For guidance on choosing an IP address, plus clarification on other issues such as network connectivity, network bandwidth, and fragmentation, please refer to the *network setup considerations* section of the *Prosody X card Administration guide*.

For this example, we will assume that we are not using DHCP, and that a range of IP addresses (192.168.1.40 to 49) within a network have been reserved for use by IP gateways, such as our Prosody X card. We will use the first of these addresses, (192.168.1.40) for the NIC and the next address (192.168.1.41) for the Prosody X card resources.

Step 12. To set the static IP address, use the standard Windows `Network and Dial-up Connection` properties dialogs:

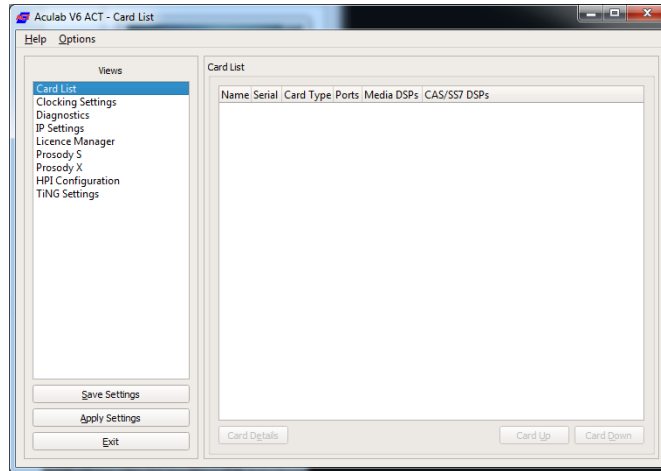


3.3.2 Updating the Aculab card list

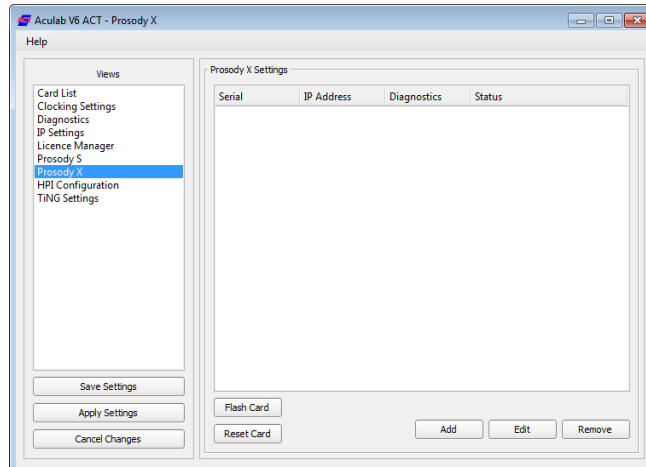
For this example, we will update the card list using the Aculab configuration tool (ACT). For command line options, please refer to section 4.3.2. For further information on using the ACT, please refer to the [Aculab configuration tool \(ACT\)](#) user guide.

Step 13. To open the ACT, select **Start – programs – Aculab – V6 – ACT**, or run the `C:\Program Files\Aculab\V6\ACT.exe` application, then select the **Card List** view.

The card list view will not show the Prosody X card until it has been added to the Aculab card list using the Prosody X view functions.

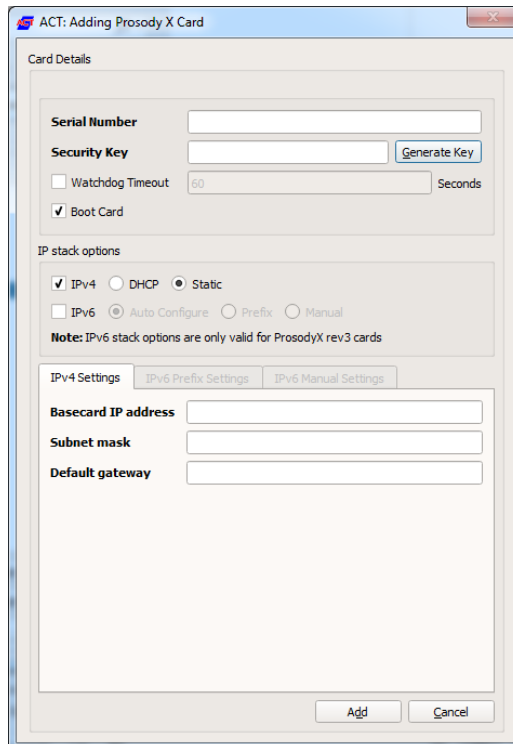


Step 14. To add the Prosody X card to the **Card List**, first select the **Prosody X** view.

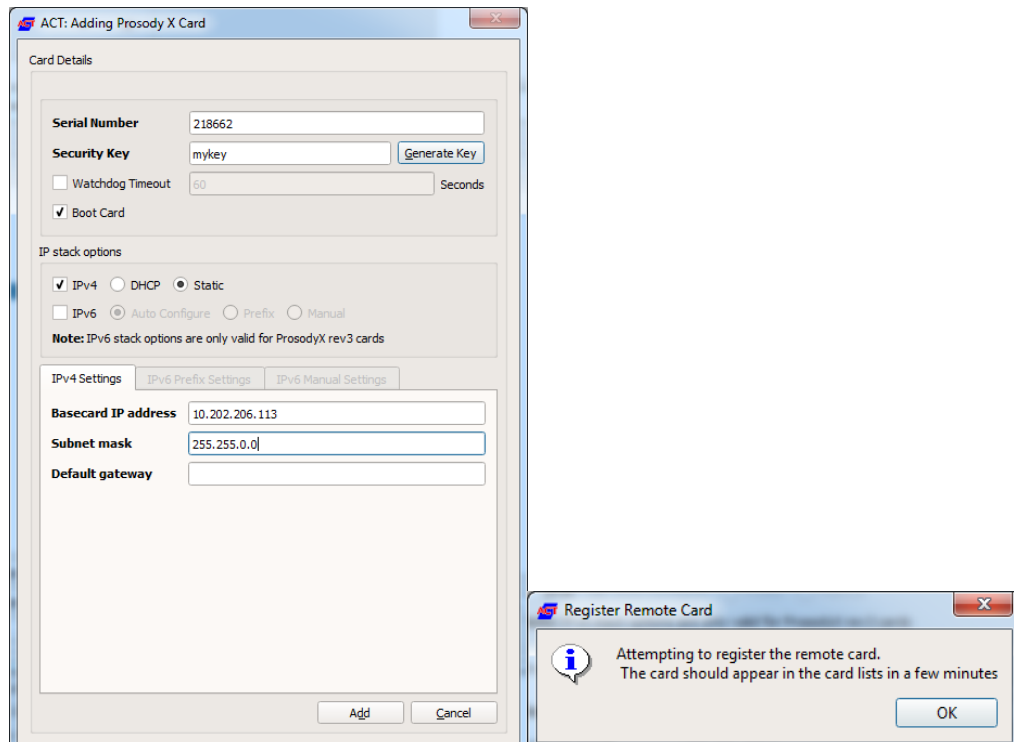


NOTE

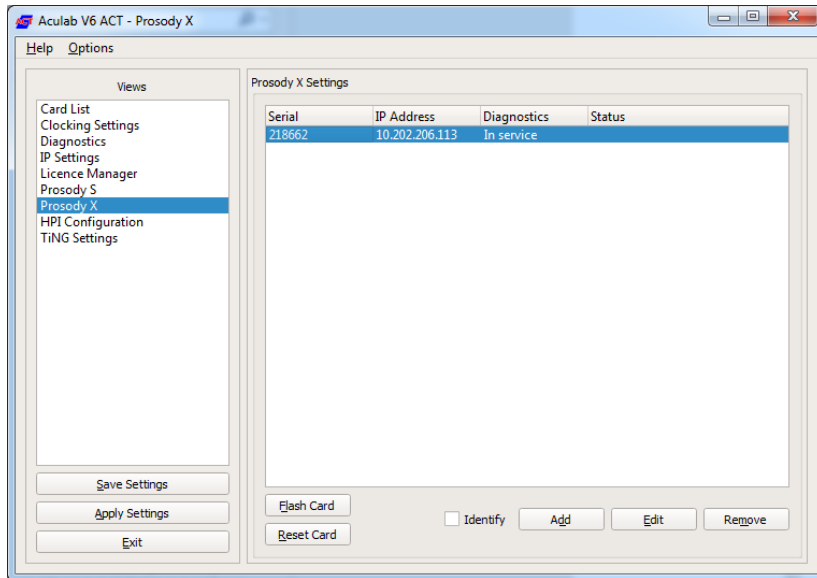
Under certain conditions, the Prosody X cards may appear in the list but will not show an *IP Address* or active *Status*. In these instances you should select the card entry followed by **Edit**. Otherwise:

Step 15. Select Add


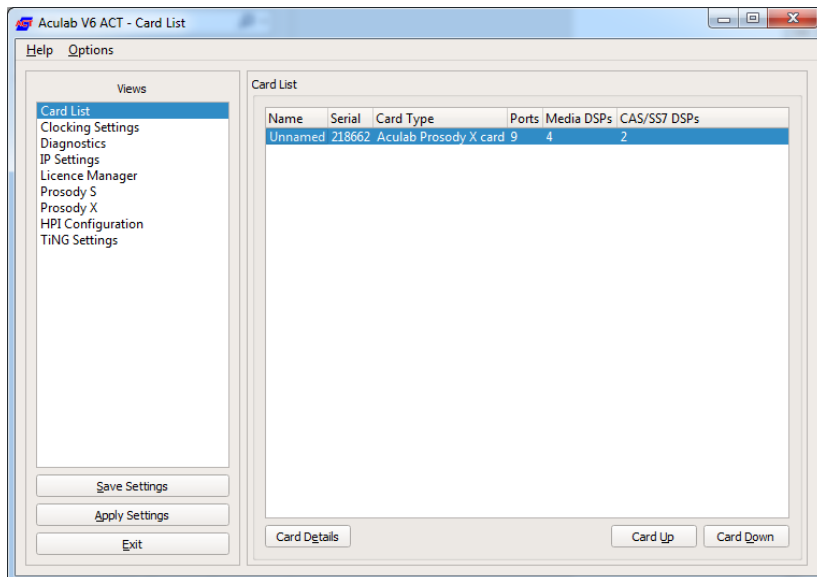
If the card is to be configured from another system, you only need to add the *Serial Number*, select IPv4 or IPv6 then fill out the *IP Address*. If the card is to be booted and configured by the local host, you must select *Boot Card* to enable entry of the required parameter.

Step 16. Select Boot Card then enter the appropriate Serial Number, IP Address type, IP Address and Netmask parameters.


- Step 17. Enter a unique **Security Key**, or select **Generate Key**, to create a key for the card. This key is used to restrict access to the card from approved hosts. For further details, please see the *ACT and Prosody X card administration guides*.
- Step 18. Select **Add** followed by **OK** for the **Register Remote Card** dialog prompt. The card will go through various *Diagnostics* and *Status* changes prior to being added to the *Prosody X* page list.



- Step 19. Selecting the **Card List** view dialog should now show the **Prosody X** card.



3.4 Configuring the Aculab telephony-software.

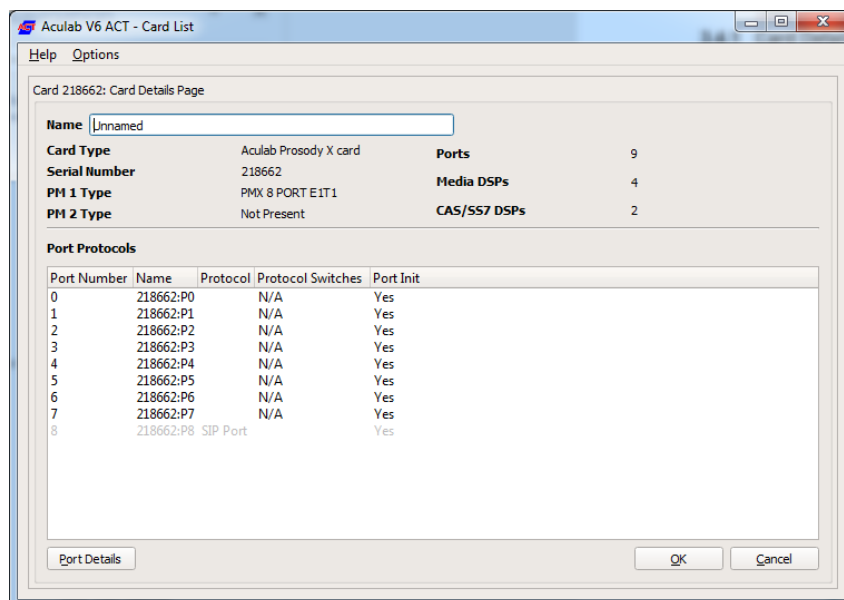
We will now configure:

- Card Details - Set up the E1/T1 port protocols
- Clock Settings - Set the clock master
- IP Settings - Set up for SIP
- TiNG Settings - Set up various speech processing features

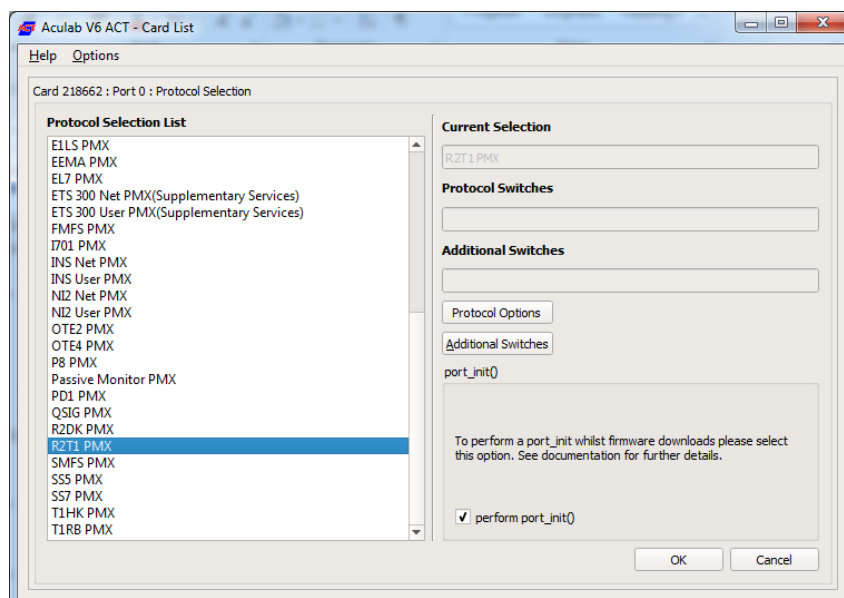
3.4.1 Card Details

For our example, we are going to set half of the ports for R2 (CAS), and the other half of the ports for ETS 300.

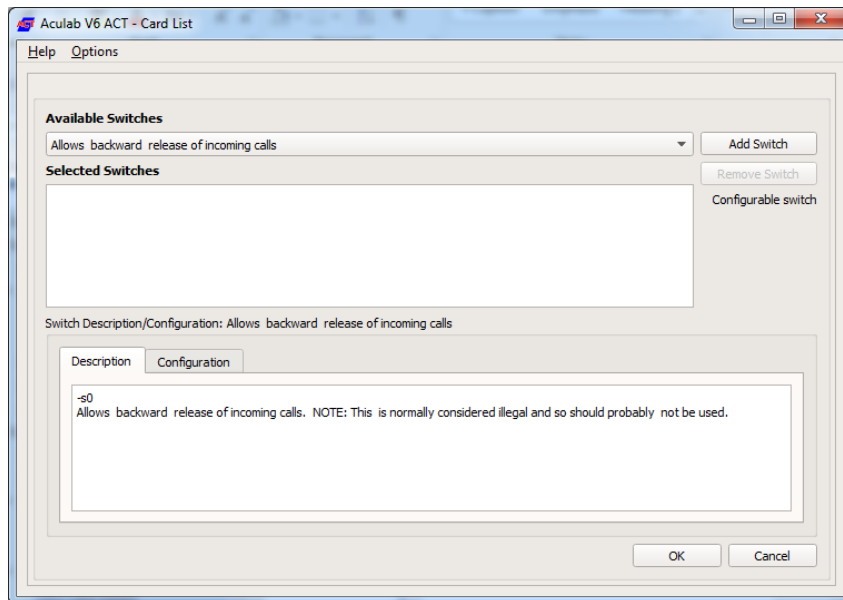
Step 20. In the `Card List` View, select the Prosody X card entry followed by `Card Details`, or double click on the Prosody X card entry, to open the `Card Details` Page dialog.



Step 21. In the `Port Protocols` field, select the first port entry followed by `Port Details`, or double click on the first port entry, to open the `Port * Protocol Selections` dialog.



Step 22. In the Protocol Selection List, select R2T1 PMX. Then select Protocol Options to open the R2T1 PMX Switch Options dialog.

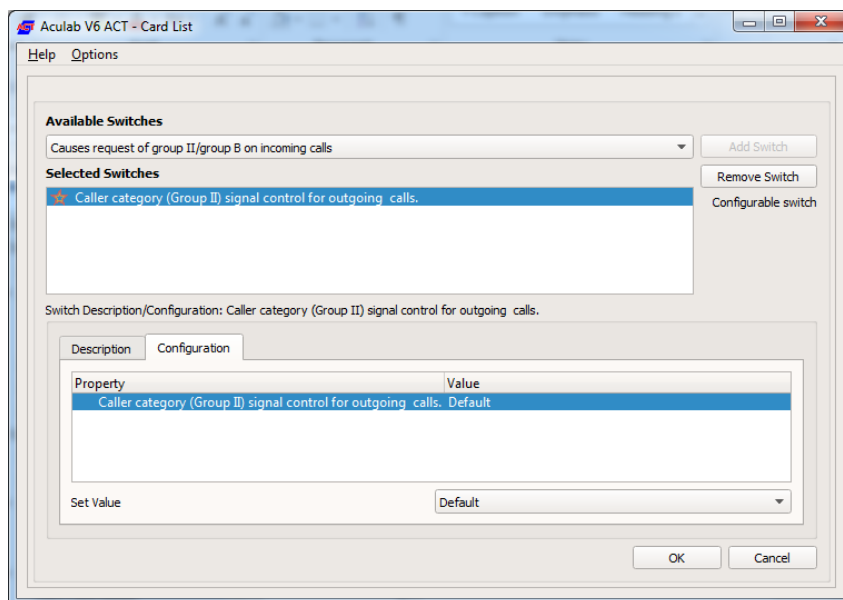


The *Available Switches* pull down menu details the available options for the selected protocol. The *Switch Description* field displays details for a selected switch. To browse the options, select options as you scroll up or down the list.

Step 23. Select a required option from the *Available Switches* field followed by **Add Switch**.

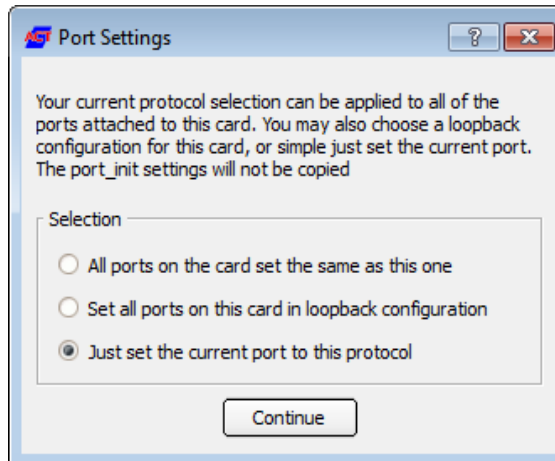
Step 24. Repeat for all required options

Step 25. For configurable switches, select the required *Selected Switches* entry then use the *Switch Configuration* tab to set the required value.

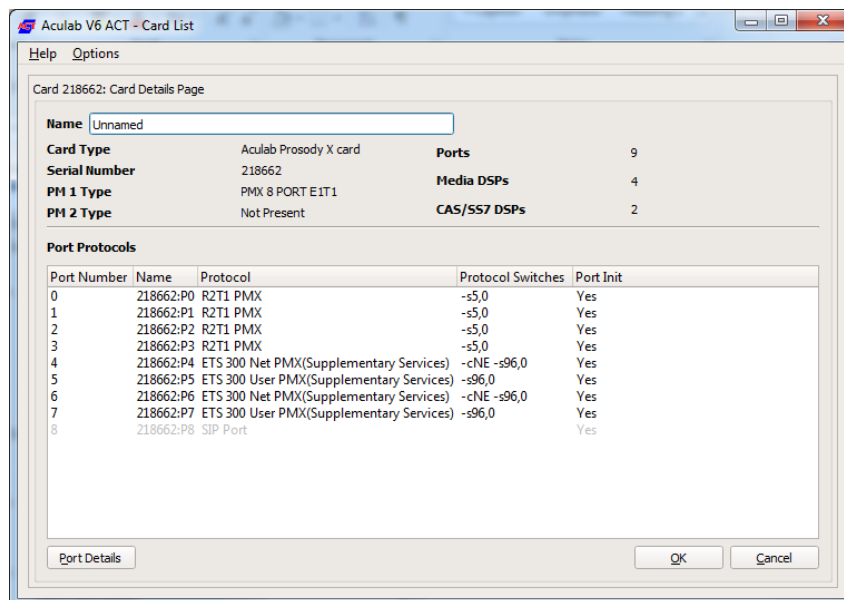


Step 26. Once you have completed the setting up of your required switch options, Select **OK** to close the Switch Options dialog and return to the Protocol Selection dialog. Then select **OK** to implement your *Protocol Selections* and return to the Card Details Page dialog.

- Step 27. You will be presented with an option to set *'All ports on the card set the same as this one'*, Ignore this option and select *'Just set the current port to this protocol'* followed by **Continue**.



- Step 28. Repeat steps 21 to 27 for each port. In the following example, we have set the first four ports for R2T1 and the second four ports for ETS300, both network and user end types.

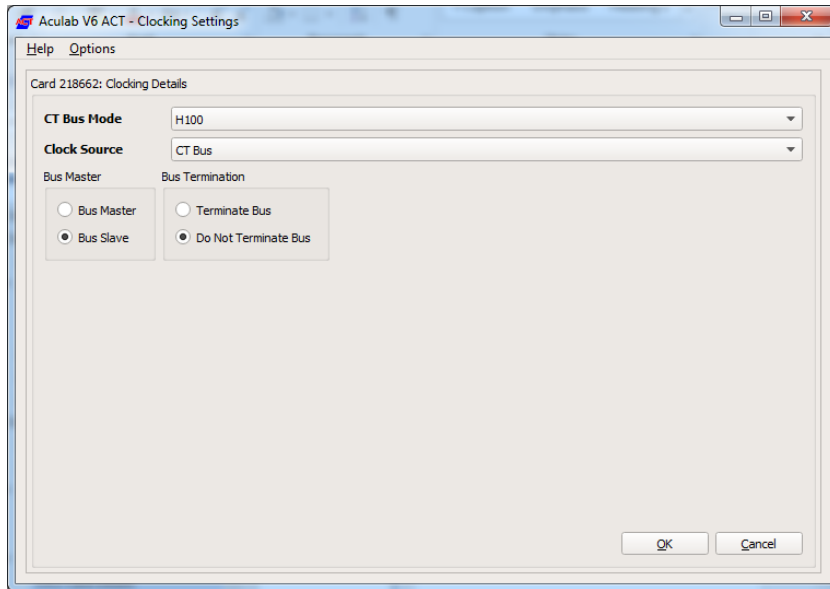


- Step 29. Select **OK** to close the Card Details Page dialog and return to the Card List primary dialog.

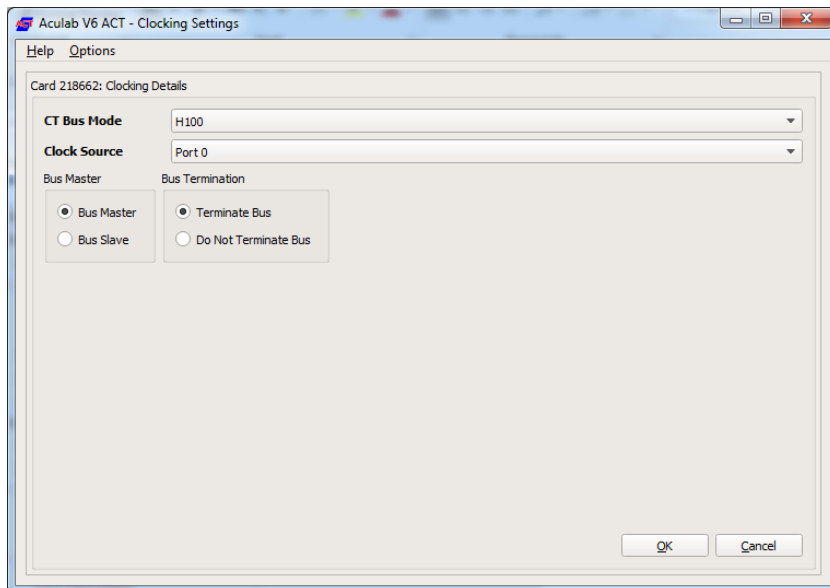
3.4.2 Clocking settings

As this example is for a single card, and will not be using the H.100 bus (CTBus), we only need to set the Clock master source; this will be set to be derived from Port 0. Details of setting up clocking for more than one card, including bus termination, is detailed in the [Aculab configuration tool.pdf](#), which is available from the support area of the company web site at www.aculab.com.

- Step 30. Select the **Clocking Settings** view, and then double click on the **Prosody X** card entry, or select the **Prosody X** card entry followed by **Clocking Details**, to open the **Clocking Details** dialog.



- Step 31. Select **Bus Master** then from the **Clock Source** pull down menu, set the source to **Port 0**.

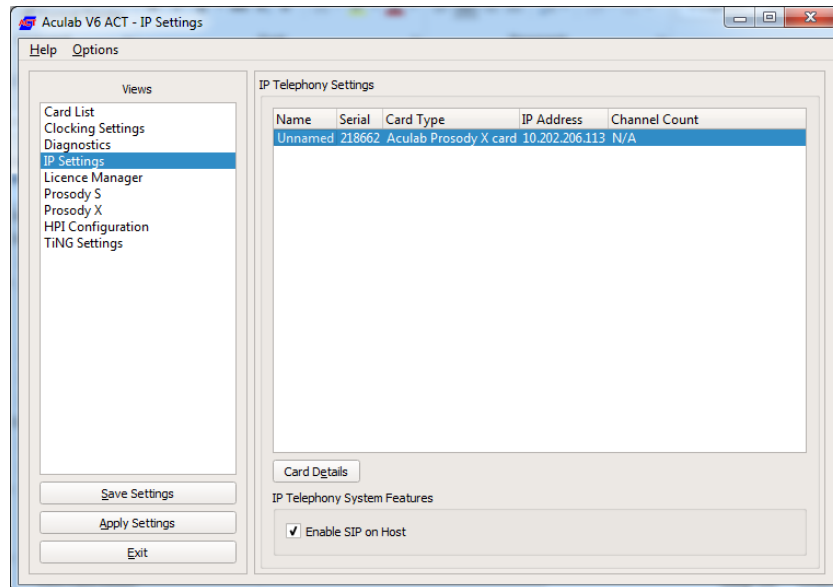


- Step 32. Now set **Bus Termination** to **Terminate Bus** followed by **OK** to complete the clocking settings and return to the **Clocking Settings** primary dialog.

3.4.3 IP Settings

This option allows you to set specific SIP port parameters as well as generic card details. For further details on setting these parameters, please refer to the [Aculab configuration tool \(ACT\)](#) user guide, available from the support area of the company web site at www.aculab.com. For our example, we will accept the default values.

Step 33. Select the IP Settings view.



Step 34. Check the Enable SIP on Host option.

3.4.4 TiNG Settings

The TiNG firmwares to be loaded onto the Prosody Modules will depend on what functionality you plan to support. For our example, we will assume that we are going to be providing some form of IVR and voicemail solution to users accessing the system from any of the R2, ETS, or IP ports.

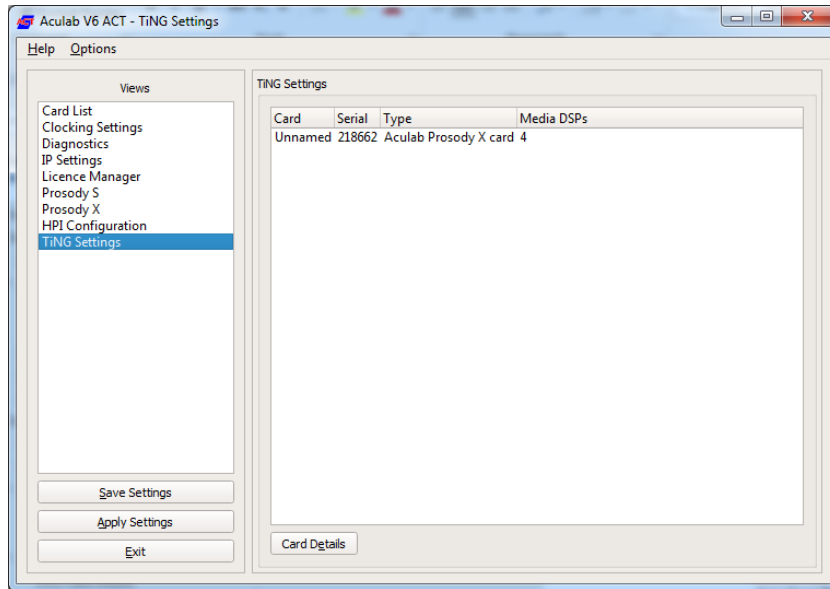
Prosody X normally requires the Datafeed module. Additional modules may include `inchan`, `outchan`, some form of `play` and `record` for playing prompts and recording messages, `vmprx` and `vmptx` for the transmitting and receiving of voice media over IP, etc.

When ProsodyX resources are to be use with legacy applications that are built to provide SIP through the limited Media Handler Plugin (MHP) generic timeslot mode, it is necessary to select the TRM option and specify the appropriate TRM file.

The layout file option may be specified if a TiNG layout file has previously been prepared with details of a set of firmware modules to be loaded together with specific loading requirements.

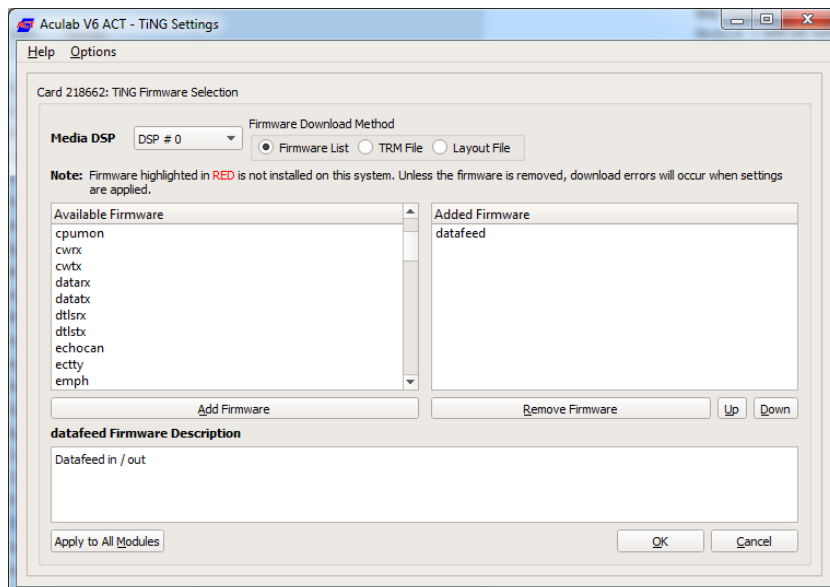
For further details, please see [Ting](#) (Prosody speech API guides) documentation. Information on using the `use TRM file` option is detailed in the [ACT](#) user guide.

Step 35. Select the **TiNG Settings** view.

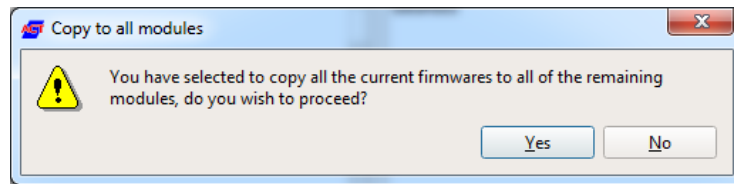


Step 36. Select the **Prosody X** card entry, then select **Card Details**, or double click on the **Prosody X** card entry, to open the **TiNG Firmware Selection** dialog. By default, **Module 0** will be selected.

Step 37. Scroll down the **Available Firmware** list, selecting and adding the required firmwares, for example, select **datafeed** followed by **Add Firmware** to add **datafeed** to the **Added Firmwares** list.



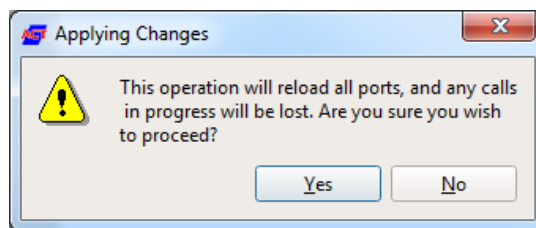
Step 38. Once you have made all the required selections, select **Apply to All Modules**. You will then be asked to confirm your selection.



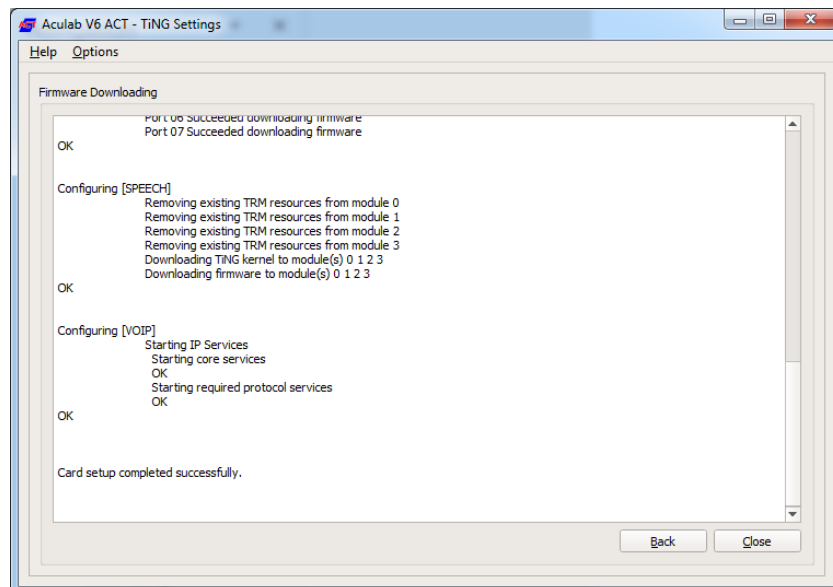
Step 39. Select **Yes** to close the prompt, then select **OK** to close the **TiNG Firmware Selection** dialog and return to the primary **TiNG Settings** dialog.

3.4.5 Applying the configuration

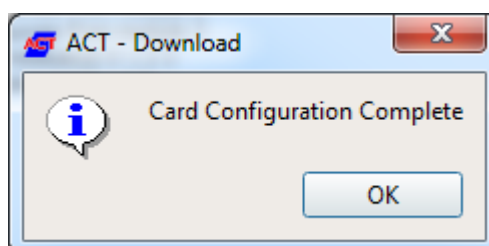
Step 40. Once you have completed all your required configurations, select **Apply Settings** to implement the configuration. You will then receive a confirmation dialog.



Step 41. Select **Yes** to continue. The **Firmware Download** dialog will display the progress status.



Once completed, you will receive a confirmation dialog.



Step 42. Select **OK**.

Step 43. Now select **Close** to close the download dialog and exit the ACT.

NOTE

You should now be ready to use the Application Programming Interface (API) functions to interface between your application and the Prosody X card resources. If you are unfamiliar with the Aculab APIs, the AIT packages contain some code examples that may be of assistance.

4 Example installation and configuration with command line

The following sections describe how to install Aculab software on a system using only command line tools. Although this example is mainly of interest to users running Linux without the X-Windows system installed, most of the tools are generic, and may be used on Windows command prompts. As indicated, some tools are specific to Linux.

NOTE

It is assumed that the Aculab card has already been installed.

Three basic areas are covered:

1. Installing the Aculab software on a machine
2. Loading the drivers
3. Configuring the cards in the system.

4.1 Linux drivers

4.1.1 Pre-requisites

Under Linux, the Kernel headers need to be installed and configured. Your Linux distribution documentation will explain how to do this.

NOTE

See `$ACULAB_ROOT/driver/readme.install` for distribution-specific information before building the Aculab drivers on Linux.

NOTE

Linux installation scripts are kept in `$ACULAB_ROOT/driver`

4.1.2 Making Libraries and Supporting Utilities

Newer TiNG distributions are supplied with a pre-built shared object as well as the source code. This section is applicable only if you would like to link your applications against a static TiNG library.

If a static TiNG library is needed, then the `$ACULAB_ROOT/driver/makelibs` script file must be run before building the driver. This creates the appropriate supporting utilities to build and run the Prosody 2 (TiNG) driver component. `makelibs libonly` may be run to build only `libTiNG.a` and not the assorted test tools that are not needed for every-day operation. This will greatly reduce the time taken to build TiNG.

4.1.3 Building, and Configuring, the Driver

If Prosody X cards are being used in the system, then the distribution's kernel headers package must be installed. Consult the distribution's documentation for information on this.

`ACULAB_ROOT` must be set to the root directory of the Aculab distribution (e.g., `/usr/local/aculab/v6`). `setV6.sh` is a script that will set the all of the environment variables correctly. To run:

```
source setV6.sh
```

This file may also be modified and copied into `/etc/profile.d` to ensure settings are

loaded for each bash shell.

Before the driver can be loaded, it has to be fully built. Due to the many variations of Linux distributions, there cannot be a "one binary for all" situation. To cope with this, the driver kit contains precompiled objects with the final compilation of the kernel-specific code and the linking together of the loadable module, completed on the host machine. The script files `dacpinst` for Linux are used to automate this process and configure the driver to your needs.

To avoid compilation errors, Please ensure that the sources for the intended kernel have been installed, prior to building the driver. For most distributions, this required package would be called `kernel-source`.

The commands for `dacpinst` are as follows:

```
./dacpinst build      - builds, then configures, the driver.
./dacpinst clean     - removes the objects, allowing a clean rebuild.
./dacpinst version   - gives the version of dacpinst.
```

At each prompt, `dacpinst` will often give options (i.e., "yes/no") in brackets "(...)", and the default actions in square brackets "[...]". The default actions will predict the desired input according to a typical build. If you'd like the default input, hit return, or else type in what you'd like, instead.

4.1.4 Creating the Driver

Running "`./dacpinst build`" asks you what components you would like to include. Select the components as necessary. If an invalid combination is selected, you will be asked to enter an alternative combination.

Please follow the on-screen instructions.

4.1.5 Configuring the Driver

If you have successfully completed the previous part there should be a loadable module created that represents the driver for your system.

You will be asked for a non-privileged user to run the Aculab tools. The Aculab Resource Manager (`acuresmgr`) will be run as root, but will spawn any tools as this user minimizing the amount of root access required for the Aculab tools.

4.1.6 Loading and Unloading the Driver

After successfully configuring the driver, you will be given a new script file called `aculab_dacp` for Linux. This script file is compatible with those in `/etc/init.d` and `/etc/rc*.d` or `/etc/rc.d/rc?.d` (as used in most distributions of Linux) and can be used to automatically load the driver at boot-up.

The commands for Linux `aculab_dacp` are as follows:

```
./aculab_dacp start  - loads, then starts, the driver.
./aculab_dacp stop   - stops, then unloads, the driver.
./aculab_dacp restart - restarts the driver.
```

`aculab_dacp` also creates the device nodes (in `/dev`) required for correct operation.

4.1.7 Automatic Loading of the Driver

The `aculab_dacp` and `aculab_v6` script files can be used to automatically load the driver at the boot-up sequence of Linux. For example, if you are defaulting to using Linux runlevel 5 (usually boots up to X-Windows in Redhat), then you can symbolically link `aculab_dacp` into `/etc/rc.d/rc5.d` using similar superuser commands to:

```
cd /etc/rc.d/rc5.d
ln -s /usr/local/aculab/v6/install/aculab_dacp ./S91aculab_dacp
```

NOTE

The location and style of your boot-up scripts may differ on your system.

4.1.8 Removing the Driver

To unload the driver, ensure that no other process is using it, and then enter:

```
./aculap_dacp stop for Linux.
```

4.1.9 Setting up Prosody X Card IP Address on Linux

Like any other NIC card in the system, locally installed Prosody X cards need to have their host NIC interface assigned an IP address. Consult the distribution's documentation for information on how to do this.

Further details can be found in the *card boot & discovery* section of the *Prosody X administration guide*.

NOTE

Further O/S specific information can be found in
`$ACULAB_ROOT/driver/readme.install`

NOTE

Ensure `$ACULAB_ROOT/log` is writable by any users (including root) that are likely to generate trace before using `trace_mode` to generate trace.

4.2 Managing Prosody X cards

Both Windows and Linux recognise the Prosody X card as a network adapter (NIC) and not an Aculab Computer Telephony device. It will not appear automatically in the ACT card list and must be manually added to the list. This can be achieved either through the ACT using the *Prosody X cards...* option, or by using the command line tool `prosody_ip_card_mgr` (section 4.3.2).

4.2.1 Distinguishing Prosody X cards on Linux

When you add multiple Prosody X cards to a Linux system, they will have been assigned names, for example `eth2`, `eth3`, and `eth4`. This does not tell you which is which, which is essential if you want to plug an external network connection into any of them. The `prosody_ip_card_mgr` utility maybe used with the `info` command to determine the MAC address associated with a Prosody X card and hence allow correlation with associated network device.

4.2.2 Adding a Remote Prosody X card

If the Prosody X card is installed in a machine other than the machine on which the Aculab software is running, or is installed in the same machine but the host NIC has not been assigned an IP address, as in sections 4.2.9 or 3.4, then the card is a remote card. To add a card use one of the following commands:

```
prosody_ip_card_mgr add <serial_no> <card_key> <ip_address_of_card>
<netmask> <gateway_address>
```

to provide an explicit configuration or:

```
prosody_ip_card_mgr add <serial_no> <card_key> dhcp
```

to configure using DHCP.

The `card_key` is a password which ensures security of the system and which must be specified. More than one system may use resources on a Prosody X card, but they must all use the same `card_key` to talk to the card.

For example, If you want to give your Prosody X card (serial number 179779, card key theKey), an IP address of 192.168.1.42 and there's no gateway, you would use:

```
prosody_ip_card_mgr add 179119 theKey 192.168.1.42 255.255.255.0
```

4.2.3 Adding a Local Prosody X Card

If the Prosody X card is installed in the machine on which the Aculab software is running, and the host NIC interface has been assigned an IP address as in sections 4.2.9 or 3.4, then the card is a local card. Use the same commands as in section 4.3.1 but with “`prosody_ip_card_mgr configure persistent`” instead of “`prosody_ip_card_mgr add`”. For example:

```
prosody_ip_card_mgr configure persistent 179119 theKey dhcp
```

NOTE

The IP address supplied to the `prosody_ip_card_mgr` tool must be different to the IP address assigned to the host NIC.

NOTE

Should you receive an error message `ERR_CARD_NOT_FOUND`, check that your Prosody X card is running the latest flash revision. See the ‘Flash upgrade process’ section of the ‘Prosody X administration guide’ for further details.

NOTE

To see if your card has been detected, run ‘`swcmd -e`’. It may take a few seconds to show up so you may need to try a few of times.

4.2.4 Listing installed Prosody X cards

To list all Prosody X cards currently installed in the system use the following command:

```
prosody_ip_card_mgr list
```

This will return the serial number and IP address of any cards, (this example is for a single card):

```
1 Prosody IP cards registered on this system:
179779 (192.168.1.42)
```

4.2.5 Obtaining information for an installed Prosody X card

To obtain card details for a specific card, use the following command:

```
prosody_ip_card_mgr info <serial_no>
```

For example, to obtain information for card serial number 221946, use the following:

```
prosody_ip_card_mgr info 221946
```

This will return the following information:

```
Card 221946:
Type:          Aculab Prosody X card
Version:       V3.1
MAC address:   00:02:1F:00:C3:4B

Local NIC:     p4p2
Configure:     Yes
Remote:        Yes
IPv4:          Dynamic
IPv6:          /0
Key:           mysitekey
Status:        In service

2 ethernet ports:
  0: Connected at 1000Mbit full duplex and active
  1: Disconnected

Device 0:
Model:         AC5700 Prosody X
Serial number: 221946
Version:       3.1
Bootloader:    U-Boot 2010.03 1.2.3 (Prosody-X Rev3.x Oct 17 2011 - 12:12:
Firmware:     Aculab Prosody IP Firmware 1.0.142.314
3.3V:         3.394V
12V:          12.281V
Temperature 0: 39C
Temperature 1: 29C
```

4.2.6 Removing a Prosody X card

To remove a card use the following command:

```
prosody_ip_card_mgr remove <serial_no>
```

4.3 Card configuration files

The default configuration of Aculab cards is determined using a series of configuration files.

The Aculab Resource Manager looks for *.cfg files in the ACULAB_ROOT/Cfg directory, where ACULAB_ROOT is the directory in which the Aculab package is installed. These files must be generated either automatically by the Aculab Configuration Tool (ACT), via the command line using CFGTemplates.exe (on Windows) or cfgtemplates (on Linux), or manually using an ASCII text editor.

NOTE

The default location of "ACULAB_ROOT" for Windows is "C:/Program Files/Aculab/V6/"

All *.cfg files are in the form <serial_no>.cfg. For example, a card with the serial number 12345 would have a corresponding cfg file of 12345.cfg.

4.3.1 Creating configuration files

`CFGTemplates.exe` is a command line application that will create a `cfg` template file for a given Aculab card. It is found in the `bin` directory of in the `ACULAB_ROOT` folder. To run this application the user must pass a serial number into the application, for example,

```
cfgtemplates 131865
```

The above example will create files with the file names `131865.cfg` and

This will not be a fully configured file; certain fields have been left blank for the user to edit using a standard ASCII text editor. For example, the call section will have no firmware selected for the call ports. Also the Speech section will not have any TiNG firmware specified.

Example of running the application

The following is an example run on a Windows system. There are no differences in executing the tool on Windows or Linux.

```
Run cfgtemplates 131865
```

If the card is already in the system, you will be asked if you wish to automatically detect the card type.

You will then be asked to specify the bus mode; H100 should be specified.

Next you will be asked if the card is a "Bus Master" i.e. is this card to supply the clock source. If yes, then the clock source will be requested. If No, the card will take the CTBus Mode as the clock source.

Now you will be asked to confirm if the card is to be bus terminated (at either end of the CTBus).

If the card is not in the system, you will be asked to specify the card type. This will be followed by a series of questions relating to the number of ports, Prosody modules and so on as appropriate.

Once all the questions have been answered, the file will be written to the directory that the application was executed in. You can now manually add any firmwares to the file as required and copy the file into `ACULAB_ROOT/cfg` together with any additionally tool created file `voip_rm.cfg`, containing your selections for IP Telephony services.

4.3.2 Per-card configuration file description

File layout

The `.cfg` files are split into a number of blocks corresponding to the switch, call, and speech components a card may have. [`<component_name>`] and [`End<component_name>`] delimit each of the components. The components are:

```
[General]
[EndGeneral]
[Call]
[EndCall]
[Switch]
[EndSwitch]
[Speech]
[EndSpeech]
[VoIP]
[EndVoIP]
```

NOTE

Only the general, call and switch options are common to all cards, speech and VoIP are optional.

The general component fields

This component contains a single field, `CardName`. The field value should contain the serial number of the card.

```
[General]
CardName=Card<128483>
[EndGeneral]
```

The switch component fields

There are 7 fields in the switch component. These fields map directly to the fields in the "h100_config_board_clock_params" structure, and are populated with exactly the same values (see the Switch API guide for further details).

Example switch component:

```
[Switch]
CtBusTermination=TRUE
CtBus=SWMode_CTBus_H100
Source=H100_SOURCE_H100_A
Network=0
H100Mode=H100_SLAVE
AutoFallBack=H100_FALLBACK_DISABLED
NetRefClockSpeed=H100_NETREF_8KHZ
[EndSwitch]
```

Each line in the section consists of a field name followed by equals and then the value assigned to that field.

NOTE

The fields in this section are case sensitive

H.100 configuration

CtBusTermination: This card should terminate the bus. Possible values:

- TRUE - the card should terminate the H.100 bus
- FALSE - the card should not terminate the H.100 bus

CtBus: This setting controls the bus that the card will use. For H.100use:

```
SWMODE_CTBUS_H100 - H.100 mode (the default)
```

NOTE

Changing this setting requires a restart of the drivers for the card. This will only occur the first time this setting is applied to a card - upon subsequent reboots the card will start in the correct mode.

Source - controls the source of the card's clock. This can be one of the following:

```
H100_SOURCE_INTERNAL - generate a clock on board and provide it to the CT bus
H100_SOURCE_NETWORK - use the port specified in the network field as the clock
H100_SOURCE_H100_A - take the clock from the H.100 clock master A
H100_SOURCE_H100_B - take the clock from the H.100 clock master B
```

Network - controls the network port that is used as a clock reference if *Source* is set to `H100_SOURCE_NETWORK`. The network port index is the physical index of the port on the card. Remember that for the Switch API, ports are numbered from 1.

H100Mode - controls whether the card is an H.100 primary or secondary master or a clock slave. The possible values for this field are:

```
H100_SLAVE - card will be a slave on the H.100 bus (Source must be one of
H100_SOURCE_H100_A or H100_SOURCE_H100_B)

H100_MASTER_A - card will drive the A clock on the H.100 bus (Source must be
one of H100_SOURCE_INTERNAL or H100_SOURCE_NETWORK) .

H100_MASTER_B - card will drive the B clock on the H.100 bus (Source must be
one of H100_SOURCE_INTERNAL or H100_SOURCE_NETWORK) .
```

AutoFallback - This field determines the card's behavior when an H.100 clock failure occurs. This field can contain one of the following values:

```
H100_FALLBACK_DISABLED
H100_FALLBACK_ENABLED
H100_AUTO_RETURN
H100_CHANGEOVER_TO_NETWORK
```

See the description of the `auto_fall_back` field in the documentation for the `sw_h100_config_board_clock()` function for an explanation of how these fields apply.

NetRefClockSpeed - This field is used to tell the card the speed of the `CT_NETREF` fallback clock. See the description of the `netref_clock_speed` field in the documentation for the `sw_h100_config_board_clock()` function for further details.

The call component fields

The following fields are required in the call section for each port.

```
Port=
PortName=
Firmware=
Config=
DSPA=
DSPB=
PortInit=
```

The following definitions apply:

Port

Is the port number starting at 0, if you have four ports then they are numbered 0-3.

PortName

Is a user defined 16 character text string, for example:

```
Portname=London Port1.
```

Firmware

The location from `ACULAB_ROOT` of the call firmware file, for example:

```
Firmware=/firmware/ets_usr.pmx
```

Config

The firmware switch to applied at download time, for example:

```
-CNE (ETS300 network side).
```

For details of available/valid firmware switches, please read the associated `$ACULAB_ROOT/firmware/*_switches.txt` file. For example, for ETS this would be the `ets_switches.txt` file.

DSPA & DSPB

Redundant fields that were used for earlier generation cards.

PortInit

Can be either `TRUE` to perform `PortInit`, or `FALSE` to not perform `PortInit`. For example:

```
PortInit=TRUE
```

A typical configuration would be as follows:

```
[Call]
Port=0
PortName=Port 0
Firmware=/Firmware/ets_net.pmx
Config=-cNE
DSPA=
DSPB=
PortInit=TRUE
Port=1
PortName=Port 1
Firmware=/Firmware/ets_usr.pmx
Config=
DSPA=
DSPB=
PortInit=TRUE
[EndCall]
```

The speech component fields

The speech component is split into "blocks" in a similar fashion to the call component, however in this case the "blocks" are delimited by the "Module" field.

The "Firmwares" field takes the full path from the `ACULAB_ROOT` of the Prosody firmware to be downloaded.

The Prosody2 (TiNG) firmware modules are assumed to be in the following directory:

```
ACULAB_ROOT/TiNG/starcore/gen
```

Where `ACULAB_ROOT` is the directory in which the Aculab drivers are installed.

Example speech component:

```
[Speech]
Module=0
Firmware=datafeed inchan outchan playA
Module=1
Firmware=datafeed inchan outchan playA
[EndSpeech]
```

module – the module number, starting with module 0

Firmware – the firmware to be downloaded to that module.

IP telephony

These settings are largely redundant and were used for older cards. However some settings are still used for legacy applications that are built to use SIP indirectly through the Media Handler Plugin (MHP). The following definitions apply:

IPADDRESS, SUBNETMASK, GATEWAY, - redundant settings that were used for earlier generation cards, *these should be set to 0.0.0.0 to allow the config tool to parse the files correctly*

ENCODEGAIN, DECODEGAIN, MAX_JITTER_BUFFER - redundant settings that were used for earlier generation cards.

RTP_TOS, RTCP_TOS - Used to set IPv4 type of service for RTP end points set up by Media Handler Plugin, normally should be set to zero.

DEF_JITTER, MAX_JITTER - used to set RTP jitter buffer parameters, set to zero for usual defaults

DTMFDETECT - controls if conversion of inband tones to RFC2833 is enabled

For legacy applications using MHP in generic timeslot mode with associated TRM resources, the following definitions are also applicable:

TDMENCODING - specifies type of encoding used for signal presented on timeslot, should be set to TDM_ALAW, TDM_ULAW or TDM_DATA

ECHOCANCEL, ECHOSPAN, ECHOSUPPRESSION - can be used to insert echo cancellation for echo present in signal fed to timeslot that is used for source signal of transmitted RTP

```
[VoIP]
IPAddress=0.0.0.0
SubnetMask=0.0.0.0
Gateway=0.0.0.0
EncodeGain=8192
DecodeGain=8192
RTP_TOS=0x0
RTCP_TOS=0x0
Def_Jitter=30
Max_Jitter=150
Max_Jitter_Buffer=250
TDMEncoding=TDM_ULAW
EchoCancel=EC_OFF
EchoSpan=0
EchoSuppression=ES_OFF
DTMFDetect=1

[EndVoIP]
```


Another `.cfg` file (`voip_rm.cfg`) is generated for the configuration of the shared IP Telephony services, for example:

```
[SIPService]
USE=TRUE
AddressHold=IPT_ENABLED
EarlyMedia=IPT_ENABLED
[EndSIPService]

[PortmapService]
USE=TRUE
[EndPortmapService]
```

4.3.3 Updating Card Configuration

Should you need to update a card's configuration, for example, following changes to a `.cfg` file, you can use the `config` command as follows:

```
Config <serial number> <switches>
```

Where:

`<serial number>` is the serial number of the card, for example, 248 for file `248.cfg`

`<switches>` are the following:

`-v` verbose displays what it is doing while running, e.g. `Config 248 -v`

`-m` module. To use this you have to specify a module number, this will only download prosody firmware to that module, e.g. `Config 248 -m 0`

`-p` port. AS for `-m` except for ports, e.g. `Config 248 -p 0 -p 1`

`-s` reloads the clocking setting for the card, e.g. `Config 248 -s`

All of these switch can be use together, for example:

```
Config 248 -p 0 -p 1 -p 2 -p 3 -m 0 -m 1 -s -v
```

This example will download firmware to ports 0 – 3 & modules 0 –1, reset clocking, and run verbose.

4.4 Signalling Software Download

`fwdspldr` is a firmware download utility for Aculab Prosody X cards. It is used to download the signalling protocol firmware, and to specify configuration switches simultaneously. For previous generations of Aculab cards it was also used to download signalling DSP f/w but this is now managed automatically so the dsp options are no longer relevant.

The `fwdspldr` utility uses standard Aculab API calls, the source of which can be made available on request.

Usage for `fwdspldr` is as follows:

```
fwdspldr <serial_no> <portnum> <filename> <config>
```

Where:

<code><serial_no></code>	Is the serial number of the card to download firmware to.
<code><portnumber></code>	Is the port on the card to download the firmware to.
<code><filename></code>	Is the DSP or signalling protocol firmware file to download.
<code><config></code>	Is used to specify any configuration switches to be passed to the signalling protocol firmware.

NOTE

The filename suffix is subject to the type or revision of the primary rate module (PM). For Prosody X cards files with a file suffix of .PMX should always be specified

Download Examples

The following example downloads firmware to the first two ports of a Prosody X card with first port set up for network side ETS300 and the second for user side.

```
fwdspldr 12345 0 ets_net.pmx -cNE
```

```
fwdspldr 12345 1 ets_usr.pmx
```

For more information on the configuration switches available for a specific signalling firmware, see the release note text files associated with that firmware.

Appendix A: Call Driver Installation Options

This section describes the options that can be used to configure call control on an Aculab card or module. These options are applied by supplying a configuration switch when installing a driver, or by supplying a switch when downloading firmware to the card using 'restart'. More details on how to install drivers on a particular operating system are detailed earlier in this manual. For further details of protocol firmware switches, (-s switches) please refer to the documentation (*.txt file) that accompanies each protocol firmware file.

Following is a description of each of the configuration switches and where they may be used.

A.1 Table of Signalling System Options

Here is a check box to indicate which configuration switches are supported by which signalling systems:

	CAS	DASS	DPNSS	ETS300	QSIG	AT&T	N12	ISUP
-cAX			X					
-cAY			X					
-cBBY	X							
-cBX			X					
-cBY			X					
-cCA				X				
-cCIC								X
-cCn	X							
-cDn	X							
-cDPC								X
-cEN				X				
-cEX				X				
-cFD				X	X			
-cFF				X	X		X	
-cFP				X				
-cFR				X	X			
-cFU				X	X			
-cIMP75								X
-cME		X	X					
-cNA1							X	
-cNCRC								X
-cNE		X		X		X	X	
-cOPC								X
-cQM/s A/B					X			
-cRn								X
-cSLC								X
-cSO						X	X	
-cSP				X				
-cSU				X				
-cSW				X				
-cTS								X

Only the key switches are listed, this is especially the case for ISUP, which has an extensive list. For further details, check the associated text file for the firmware, which are available from the Aculab web site download area.

A.2 System Specific Option definitions

The following list details some of the configuration options that can be used to configure a signalling port on an Aculab Digital Access card. This list should be used in conjunction with the latest release note for each protocol. The release notes are shipped with the firmware files and provide a more in depth description of available options.

A.2.1 -cA/B and X/Y bits configuration (DPNSS only)

Default setting: A/X

This option allows the dpnss A/B X/Y bits to be configured for layers 2 and 3 to enable DPNSS to DPNSS configuration

```
-cAX
-cAY
-cBX
-cBY
```

Incorrect setting of the A/B bits will result in an inactive DPNSS layer 2.

A.2.2 -cBBY Backbusy control (CAS only)

Instructs the CAS driver (and signalling system) to use 'backbusy':

```
-cBBY          configure all timeslots all ports
-cBBY, xxxxxxxx configure all timeslots indicated in the mask xxxxxxxx
```

The mask is an 8 digit hexadecimal representation of a 32bit value where each bit indicates the busy mode. A bit set to 1 will enable backbusy. A bit reset to 0 will disable backbusy. Bit 0 of the 32bit value controls timeslot 0 and bit 31 controls timeslot 31.

A.2.3 -cCA connect acknowledgement (ETS300 only)

Some equipment require a `CONNECT_ACKNOWLEDGE` from the user end of the protocol after receipt of a `CONNECT` message. This configuration switch provides for this behaviour.

A.2.4 -cCICnnnn circuit identification codes (ISUP only)

```
-cCICnnnn
-cCICnnnn, aaaaaaaaaa
-cCICnnnn, aaaaaaaaaa, bbbbbbbb
```

Where *nnnn* is a decimal number of 1 to 4 digits that will be the CIC assigned to the first bearer timeslot, with subsequent CICs allocated sequentially (including any timeslot used for signalling, although this CIC will never be used). The OPC and DPC assigned to the ISUP circuits will be the most recent values preceding the `-cSLC` parameter itself.

If specified, *aaaaaaaa* is a CIC MAP, which allows the user to define which timeslots have CICs assigned to them. It is specified as a 32bit number expressed in hexadecimal, where bit zero corresponds to timeslot zero etc. If not specified, a value of `fffffffe` is assumed.

If specified, *bbbbbbbb* is a Circuit MAP, which allows the user to define which timeslots have may be used for ISUP call control. It is specified as a 32bit number expressed in hexadecimal, where bit zero corresponds to timeslot zero etc. If not specified, a value of `fffffffe` is assumed Any timeslots which are in use for signalling are automatically excluded from ISUP call setup, regardless of whether this is explicitly shown in the circuit map.

If `-cICnnnn` is not specified, ISUP will not be available for call setup on the timeslots of that port.

A.2.5 -cCn number of CLI digits (CAS only)

It is necessary for the device driver and signalling system firmware to know the number CLI digits supported by the trunk. The driver configuration switch available for this purpose is `-cCn`, where `n` is a one or two digit decimal value of the number of digits required.

The `-cCn` option specifies the number of CLI digits expected by the system, for example, `-cC4` instructs the driver/signalling system to expect 4 CLI digits

It is important that this initial configuration is correct or unexpected behaviour may result.

A.2.6 -cDn number of DDI digits (CAS only)

It is necessary for the device driver and signalling system firmware to know the number of DDI digits supported by the trunk. The driver configuration switch available for this purpose is `-cDn`, where `n` is a one or two digit decimal value of the number of digits required.

The `-cDn` option specifies the number of DDI digits expected by the signalling system, for example, `-cD3` instructs the driver/signalling system to expect 3 DDI digits

It is important that this initial configuration is correct or unexpected behaviour may result.

A.2.7 -cDPCnnnnn signalling point code (ISUP only)

`-cDPCnnnnn`

Where `nnnnn` is a decimal number equating to the signalling pointcode of the adjacent network component to which the link connects. The network component may be a Signalling Transfer Point, or, if using fully associated signalling, a Signalling End Point. `nnnnn` must coincide with the pointcode of an STP (`AdjacentPC`), or of an SP (`RemotePC`) in the stack configuration file as described in the SS7 installation and administration guide.

This parameter may be repeated to specify that signalling link(s) and ISUP bearer timeslots have different DPC values. See `-cSLC` and `-cCIC`.

A.2.8 -cEn Call Charging Switch (ETS300 only)

This switch allows the application to send or receive call charging information depending on the country or exchange being used at the time. For further information on the `call_put_charge()` and `call_get_charge()` functions it is advisable to consult the 'Receiving Call Charge Information' and 'Sending Call Charge Information' sections in the Call Control Driver API Guide. The value transmitted will be one of two categories of charging for EuroISDN.

1. `UNITS` - One method is to transmit the value as `UNITS`. When charging uses the `UNITS` format a value representing the number of units accrued up to that point during the call will be transmitted. This value is initially one and is incremented by one each time the `call_put_charge()` function is used. The driver does not require any parameters in the call to `call_put_charge()` by the application.
2. `CURRENCY` - The second method is `CURRENCY`. When this type of information is used by the protocol the transmitted value is the accumulated cost of the call up to that point. This information is passed to the driver in the "charge" field of `put_charge_xparms`. The format of this string is an ASCII representation of the numeric value. Example : To send the value of 100 the three ASCII characters necessary to

encode 100 as a string would be used. Printing this string to the screen should show '100' (without quotes).

Codes For Different Countries (n)

n=1 Switzerland/CURRENCY Charging will use Facility information elements based on ETS300 182. A charging string in `put_charge_xparms()` is required.

n=2 Germany/UNITS Charging will use Facility information elements based on ETS300 182. The charging string in `put_charge_xparms()` is ignored.

n=3 Holland/UNITS Charging will use Display information elements based on a country Specific specification. The charging string in `put_charge_xparms()` should be left blank.

n=4 Switzerland/CURRENCY Charging will use Display information elements based on a country Specific specification. A charging string in `put_charge_xparms()` is required.

n=5 Austria/CURRENCY Charging will use Facility information elements based on ETS300 182. A charging string in `put_charge_xparms()` is required.

n=6 Norway/CURRENCY Charging will use Facility information elements based on ETS300 182. A charging string in `put_charge_xparms()` is required.

n=7 Sweden/CURRENCY Charging will use Facility information elements based on ETS300 182. A charging string in `put_charge_xparms()` is required.

n=99 Disable Charging

A.2.9 –cEX Primary Rate Call Charging (ETS300 only)

This configuration switch is for use with ETS300 Advice of Charging where there may well be some national/network dependent differences in the way the charging information is presented.

X is a one or two digit decimal country code:

- 0 is the default
- 1 for Switzerland (type I)
- 2 for Germany
- 3 for Holland
- 4 for Switzerland (type II old style)
- 5 for Austria
- 6 for Norway
- 7 for Sweden
- 99 for disable charging

Consult the release notes supplied with the latest revision of firmware for further details.

A.2.10 -cFD Diversion (QSIG, ETS300, AT&T T1, and NI-2)

This configuration switch activates the diversion feature in the driver.

If the switch for this service is not set, any attempt to use this service through the API will fail.

Example:

If Diversion and Facility features are required for QSIG on port zero then the switches can be supplied as follows using `fwdspldr`.

```
Fwdspldr 12345 0 qsig.pmx -cFF -cFD
```

A.2.11 -cFF facility (QSIG, ETS300, and NI2)

This configuration switch activates the facility feature in the driver.

If the switch for this service is not set, any attempt to use this service through the API will fail.

Example:

If Diversion and Facility features are required for QSIG on port zero then the switches can be supplied as follows using `fwdspldr`.

```
Fwdspldr 12345 0 qsig.pmx -cFF -cFD
```

A.2.12 -cFP MLPP activation (ETS300 only)

Activates Multi-Level Precedence and Pre-emption (MLPP Q.955) in the firmware and driver.

A.2.13 -cFR Raw Data (ETS300 and QSIG)

Activates firmware to use Raw Data information. See Appendix J of the Call Control Driver API guide.

A.2.14 -cFU user to user (QSIG and ETS300)

This configuration switch activates the user-to-user feature in the driver.

User to User : `-cFU`

User to User : `-cFUN` (no negotiation)

If the switch for this service is not set, any attempt to use this service through the API will fail.

A.2.15 -cIMP75

Sets line impedance to 75 ohms (default is 120 ohms).

A.2.16 -cME

If an outbound DASS/DPNSS call is made with the sending complete flag set, then an ISRMC is used, otherwise an ISRMI will be sent. If the `-cMC` switch is applied an ISRMC will be used for all outbound calls, regardless of whether the sending complete flag is set.

A.2.17 -cNA1 release link trunk (NI-2 only)

Enables RLT (release link trunk) call transfer with NI-1 signalling. Without the switch NI-2 signalling is used for call transfers.

A.2.18 -cNCRC disable CRC4 (ISUP only)

This instructs the firmware not to use CRC4 framing at L1.

A.2.19 -cNE network end config (DASS, ETS300, AT&T, and NI2)

Default setting: `User end`

Network end configuration, sets up the device driver for network end working

`-cNE` configure port for network end

This switch applies to signalling systems with the exception of `CAS`, `DPNSS`, `QSIG` and `ISUP`.

NOTE

For correct network end operation, you will require the network end firmware for that signalling system.

A.2.20 -cOPCnnnnn[,i] (ISUP only)

Where `nnnnn` is a decimal number equating to the signalling pointcode, and optional instance, of your application. This must coincide with the pointcode (`LocalPC`) of an SS7 signalling point in the stack configuration file, as described in the SS7 installation and administration guide.

A.2.21 -cQM/S-A/B master/slave priority (QSIG only)

This configuration switch enables the QSIG Master and Slave bits, M for master and S for slave, and also the 'priority if call clash' bits A/B. Valid options are:

`-cQSB`
`-cQMA`
`-cQSA`
`-cQMB`

One end must be set to be master and the other end must be set to be slave. Incorrect setting of these bits will result in an inactive QSIG layer 2. The default configuration for QSIG is `-cQSB`.

A.2.22 -cRn Default clearing cause (all versions)

Default setting: `BUSY`

This option allows the default clearing cause to be configured, where `n` is a one or two digit hexadecimal value of the required clearing cause and must be a correct value supported by the signalling system.

`-cR3E` configure 'Call Rejected' on all ports
`-cR3En0` configure 'Call Rejected' on port 0 only
`-cR3En1` configure 'Call Rejected' on port 1 only
`-cR3En2` configure 'Call Rejected' on port 2 only
`-cR3En3` configure 'Call Rejected' on port 3 only

The above example shows the `CALL_REJECTED` cause for 1TR6.

A.2.23 -cSLCnn signalling link code (ISUP only)

Where `nn` is a decimal number equating to the MTP3 Signalling Link Code for the signalling link. The signalling link is created when the `-cSLCnn` parameter is processed, the parameters for the link must appear in this option. If `-cSLCnn` is specified more than once, multiple signalling links will be created. The OPC and DPC assigned to the signalling link will be the most recent values preceding the `-cSLC` parameter itself.

A.2.24 -cSO disable service message (AT&T and NI2)

This configuration switch disables the transmission of the 'service message' by the protocol stack. The service message brings the timeslot into service and may cause problems with some manufacturers equipment, for example,

1110 0110 0000 0010 0000 0000 0100 0000 will enable backbusy on timeslots: 31, 30, 29, 26, 25, 17, 6 and would be represented by the value: E6020040 and would therefore be configured by: -cBBY, E6020040

NOTE

On E1 systems timeslots 0 and 16 are reserved and will be ignored.

A.2.25 -cSP stop call proceeding (ETS300 only)

-cSP switch stops a CALL_PROCEEDING message from being sent automatically after a SETUP message has been received. It is useful if extra information is required to be included in the message by the use of the call_proceeding function.

- cSP configure for all ports

A.2.26 -cSU stop setup acknowledge (ETS300 only)

Stops a SETUP_ACKNOWLEDGE message from being sent automatically after a SETUP message has been received. It is useful if extra information is required to be included in the message by the use of the call_setup_ack function.

A.2.27 -cSW configuration (ETS300 only)

This configuration switch modifies the driver for use in Sweden where a call_proceeding message is required instead of a setup_acknowledge message.

A.2.28 -cTS[*I*]*nn* (ISUP only)

Where *nn* is a decimal number equating to the timeslot that will be used for the signalling link on the E1 line connected to the network port. For E1 cards that support the function, if *I* is specified, then the signalling link will face *inwards* (that is towards the switch matrix and H.100 bus), it can then be switched out on a different E1 using the Aculab switch API. Refer to [Aculab software for digital access cards - Switch API guide](#), for further details of the Aculab switch API.