endace

accelerated

DAG 3.7T Card
User Guide
EDM01-12

Published by:

Endace Measurement Systems® Ltd

Building 7
17 Lambie Drive

PO Box 76802
Manukau City 1702
New Zealand

Phone: +64 9 262 7260

Fax: +64 9 262 7261

support@endace.com

www.endace.com

International Locations

New Zealand	**Americas**	**Europe, Middle East & Africa**
Endace Technology® Ltd	Endace USA® Ltd	Endace Europe® Ltd
Level 9	Suite 220	Sheraton House
85 Alexandra Street	11495 Sunset Hill Road	Castle Park
PO Box 19246	Reston	Cambridge CB3 0AX
Hamilton 2001	Virginia 20190	United Kingdom
New Zealand	United States of America	
Phone: +64 7 839 0540	Phone: ++1 703 382 0155	Phone: ++44 1223 370 176
Fax: +64 7 839 0543	Fax: ++1 703 382 0155	Fax: ++44 1223 370 040

Protection Against Harmful Interference

When present on equipment this manual pertains to, the statement "This device complies with part 15 of the FCC rules" specifies the equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the Federal Communications Commission [FCC] Rules.

These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment.

This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications.

Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Extra Components and Materials

The product that this manual pertains to may include extra components and materials that are not essential to its basic operation, but are necessary to ensure compliance to the product standards required by the United States Federal Communications Commission, and the European EMC Directive. Modification or removal of these components and/or materials, is liable to cause non compliance to these standards, and in doing so invalidate the user's right to operate this equipment in a Class A industrial environment.

Disclaimer

Whilst every effort has been made to ensure accuracy, neither Endace Measurement Systems Limited nor any employee of the company, shall be liable on any ground whatsoever to any party in respect of decisions or actions they may make as a result of using this information.

Endace Measurement Systems Limited has taken great effort to verify the accuracy of this manual, but assumes no responsibility for any technical inaccuracies or typographical errors.

In accordance with the Endace Measurement Systems policy of continuing development, design and specifications are subject to change without notice.

Table of Contents

Chapter 1: Introduction

Overview

Endace DAG 3.7T card provides the means to transfer data at the full speed of the network into the memory of the host PC, with zero packet loss guaranteed in even worst-case conditions. Further, unlike a NIC, Endace products actively manage the movement of network data into memory without consuming any of the host PC's resources. The full attention of the CPU remains focused on the analysis of incoming data without a constant stream of interruptions as new packets arrive from the network. For a busy network link, this feature has a turbo-charging effect similar to that of adding a second CPU to the system.

The DAG3.7T is a Network Monitoring Interface Card specifically designed to perform high efficiency monitoring and transmission with precision timestamping capability on up to sixteen T1/E1 network links.

Purpose of this User Guide

The purpose of this User Guide is to provide you with an understanding of the DAG card architecture and functionality and to guide you through the following:

- Installing the card and associated software and firmware,
- Configuring the card for your specific network requirements,
- Running a data capture session,
- Synchronising clock time,
- Data formats.

You can also find additional information relating to functions and features of the DAG 3.7T card in the following documents which are available from the Support section of the Endace website at www.endace.com:

- EDM04-08 Configuration and Status API Programming Guide,
- EDM04-12 DAG 3.7T HDLC Filtering Guide,
- EDM04-13 SAR API Programming Guide.

This User Guide and the Linux and Window Guides are also available in PDF format on the Installation CD shipped with your DAG 3.7T card.

System Requirements

General

The minimum system requirements for the DAG 3.7T card are :

- PC, at least Intel Xeon 1.4GHz or faster,
- 256 MB RAM,
- At least one free PCI 2.1 slot supporting 33MHz operation,
- Software distribution free space of 30MB.

Operating System

This User Guide assumes you are installing the DAG card in a PC which already has an operating system installed.

However for convenience, a copy of Debian Linux 3.1 (Sarge) is provided as a bootable ISO image on the CDs that is shipped with the DAG card.

To install either the Linux/FreeBSD or Windows operating system please refer to the following documents which are also included on the CD that is shipped with the DAG card.

- EDM04-01 Linux FreeBSD Software Installation Guide
- EDM 04-02 Windows Software Installation Guide

Other Systems

For advice on using an operating system that is substantially different from either of those specified above, please contact Endace Customer Support at support@endace.com

Card Description

The DAG 3.7T cards are PCI bus cards designed for cell and packet capture and generation over IP networks. The key features of the card are:

- Support for an external pod housing 16 RJ-45 T1/E1 network interfaces,
- A Spartan III FPGA supporting high-performance Endace firmware,
- An Intel 80321 XScale IO processor which supports AAL2/AAL5 reassembly or inverse multiplexing over ATM (IMA) and filtering services,
- Support for receiving and sending channelised, unchannelised, and fractional T1/E1, HDLC and non-HDLC data traffic,
- Support for data traffic filtering.
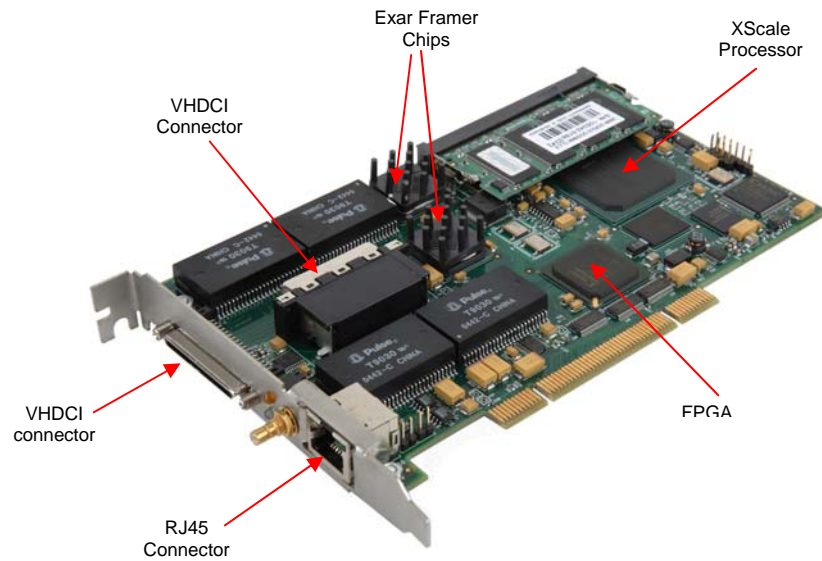
Pod Front View

16 x RJ-45 connectors

Pod Rear View

VHDCI Connector

Card Architecture

T1 or E1 data is received on up to 16 x RJ-45 interfaces, and passes through line interface units. It then feeds immediately into the FPGA for deframing and demapping into ATM or HDLC frames.

The FPGA contains an Ethernet processor and the DAG Universal Clock Kit (DUCK) timestamp engine. The DUCK provides high resolution per packet timestamps which can be accurately synchronised. Time stamped packet records are then stored in the lower FIFO.

> **Note:** For further in formation on the DUCK and time synchronising please refer to *Chapter 8: Synchronising Clock Time* later in this User Guide.

An Intel 80321 XScale processor is logically located next to the main FPGA. The XScale processor provides the facility to pre-process data before it is presented to the host, or before being transmitted over an E1/T1 link. It can also facilitate hostless operation via an embedded Linux kernel.

The main FPGA can route packets to either the XScale processor before routing onto the host, or directly to the host via the PCI port.

The following diagram shows the card's major components and the flow of data.
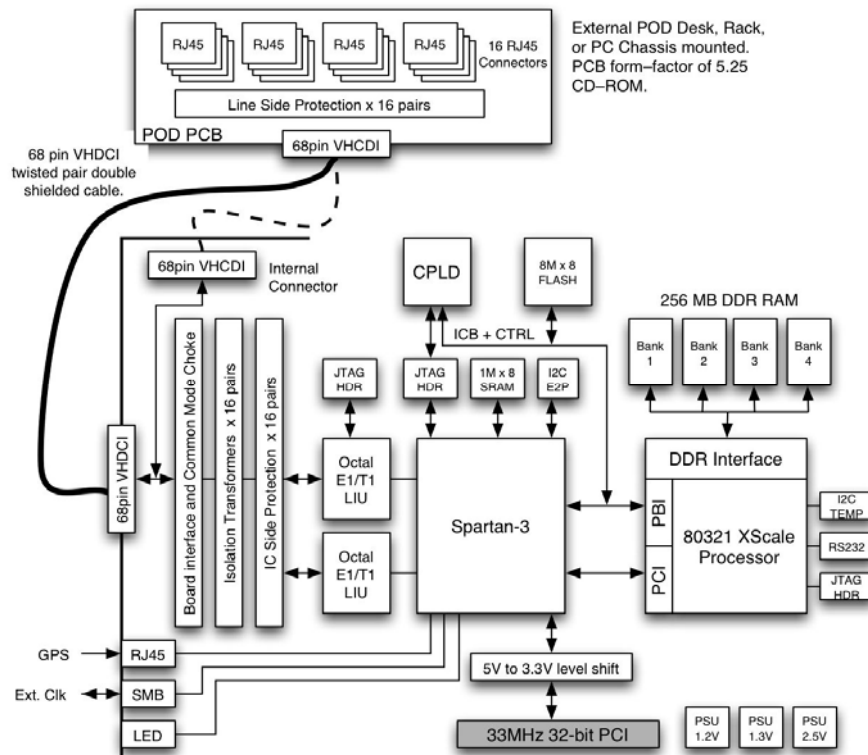
Extended Functions

The DAG 3.7T card supports the following extended functions:

- AAL2/AAL5 segmentation and reassembly,
- Inverse Multiplexing ATM (IMA),
- HDLC Filtering.

These functions are described in more detail in *Chapter 7: Capturing Data* later in this User Guide and also in the following documents available from Endace Customer Support at support@endace.com:

- SAR API Programming Guide,
- IMA API Programming Guide,
- HDLC Filtering Guide.

6

Chapter 2: Installation

Introduction

A DAG 3.7T card can be installed in any free PCI slot. It is 5V tolerant and operates only in 32-bit 33MHz PCI mode.

If you install the card into a slot that is rated for higher speeds it will cause the bus to automatically change to 33MHz. This will also affect any other devices which may be sharing the bus.

You can run multiple DAG 3.7T cards on one bus. By default, the DAG driver supports up to four DAG cards in one system.

DAG Device Driver

The DAG device driver must be installed before you install the DAG card itself.

If you have not already completed this please follow the instructions in *EDM04-01 Linux FreeBSD Software Installation Guide* or E*DM 04-02 Windows Software Installation Guide* as appropriate, which are included on the CD shipped with the DAG card.

Inserting the DAG Card

To insert the DAG card in the PC follow the steps described below:

- Turn power to the computer OFF,
- Remove the PCI bus slot screw and cover,
- Insert DAG card into PCI bus slot ensuring that it is firmly seated in the slot,
- Check the free end of the card fits securely into the card-end bracket that supports the weight of the card,
- Secure the card with the bus slot screw,
- Turn power to the computer ON.

Mounting the Pod

! **Note:** You can connect only one pod to the card at any one time

You can mount the 3.7T breakout pod either internally in a spare 5.25 inch drive bay in the PC chassis, or sit it separately outside of the PC chassis.

The pod and the DAG card connect via a VHDCI cable. The card has one VHDCI connector located on the PCI bracket and another one on the card itself. The pod has one VHDCI connector located externally on the rear of the casing.

If the pod is mounted internally in the PC you will need to connect to the card using the connector located on the card itself.

If the pod is external to the PC chassis you will need to connect to the card using the connector located on the card PCI bracket.

Connecting the Pod to the PC

Connect the pod to the DAG card using the 68-pin VHDCI cable supplied with the card as shown below.

The picture below shows the pod and DAG card connected using the external connector on the card. Use this configuration when the pod is <u>not</u> mounted in the PC chassis.

The picture below show the pod and the DAG card connected using the connector located on the card itself. Use this configuration when the pod is mounted in the PC chassis.

Connecting the Interfaces

The RJ45 connector cables for connecting the network interfaces to the pod are not supplied with the DAG card or the pod. You must source these yourself.

If you have an Endace pod you will be able to use standard E1/T1 cables available from your local electronic stockist.

If you have another type of pod you will need to make the cables up yourself to match the pod pinouts shown in the following tables.

> **Note:** You can identify the standard Endace pod by the web address "www.endace.com" written on the front of the casing above the top row of RJ45 sockets.

Connecting the Interfaces (cont.)

The physical pinouts of the RJ-45 connectors for both the Endace and other types of pod are shown below:

Endace Pod			Other Pod	
1	TX Tip		1	
2	TX Ring		2	
3			3	TX Ring
4	RX Tip		4	TX Tip
5	RX Ring		5	RX Ring
6			6	RX Tip
7			7	
8			8	

Connecting to the Network

Once you have connected the interfaces to the pod you must connect the interfaces to the network via a tap as shown in the diagram below:

Endace Pod
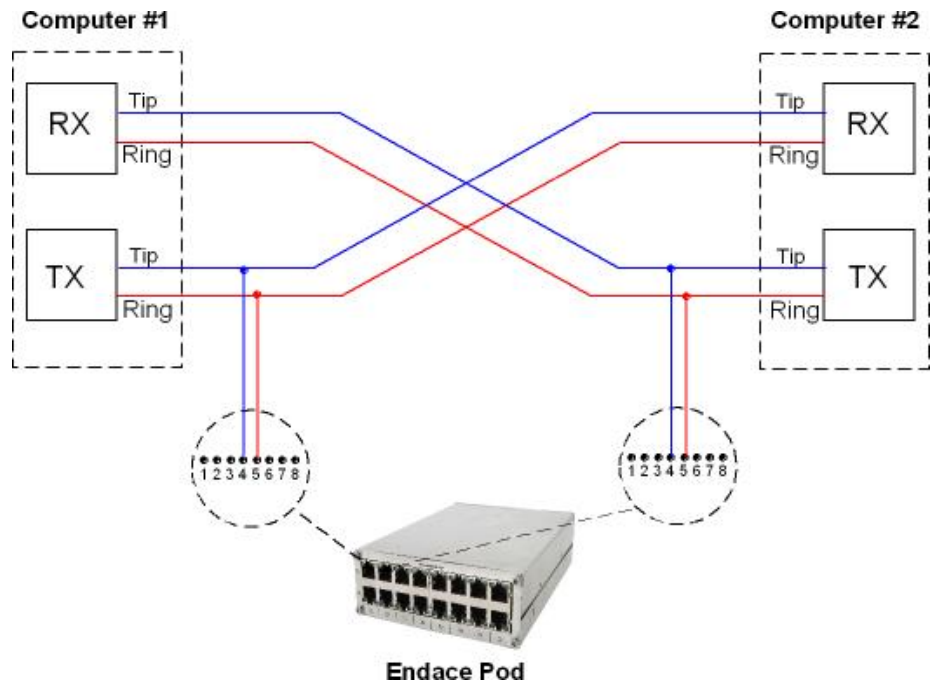
You must use a separate tap for each interface.

> **Note:** For further information about using taps to connect to the network please consult your network administrator.

When you have connected the pod to the DAG Card and the network interfaces to the network you must configure the card for your specific requirements. This process is described next in *Chapter 3: Card Configuration.*

10

Chapter 3:
Configuring the Card

Introduction

Configuring the DAG card for data capture involves the following steps:

- Loading an image and programming the FPGA,
- Setting the link,
- Checking the link,
- Configuring the connections,
- Capturing data.

The DAG 3.7T card uses two integrated Exar Octal T1/E1 Framer Chips which each provide support for up to 8 EI/T1interfaces. The `dagthree` tool which is supplied with the DAG card allows you to configure the card via the Exar chips for a range of physical line characteristics according to your network requirements. `dagthree` also provides access to interface statistics.

The `dagchan` tool which is also supplied with the DAG card allows you to configure channel characteristics. Sample `dagthree` and `dagchan` outputs are shown later in this chapter.

Line Characteristics

Overview

It is important that you understand the physical characteristics of the network to which you want to connect before you begin configuring the card.

The flexibility provided by the Exar chips means that the card will accept a wide range of settings. However if they are not the correct settings for your network the card will not function as expected.

Note: If you are unsure about which of the options listed below apply to your network, please contact your Network Administrator for further information before proceeding with configuring the card.

Supported Options

The line characteristics supported by the DAG 3.7T card are described below.

Line Type:
- **E1:** European digital standard 2 Mbps,
- **E1 CRC:** E1 with cyclic redundancy check,
- **T1**: North American digital standard 1.544 Mbps,
- **T1 SF:** Super frame, (also called D4 framing). An SF is 12 frames long,
- **T1 ESF:** Extended super frame,(also called D5 framing), includes CRC and bandwidth for a data link channel. An ESF is 24 frames long.

Line Characteristics (cont)

Encoding Type:

- **B8ZS:** Bipolar with Eight Zero Substitution (T1 only)/**HDB3** Hi-density Bipolar Three zeros(E1 only),
- **AMI** Alternate Mark Inversion.

Cable Type:

- Externally terminated,
- 75Ω unbalanced coaxial cable (E1 only),
- 100Ω balanced twisted pair (T1 only),
- 120Ω balanced twisted pair (E1 only).

Signal Attenuation:

- <-15dB short haul,
- <-36dB long haul,
- <-43dB extended long haul.

LEDs and Inputs

Before you begin to configure the DAG card it is important to understand the function of the various LEDs associated with the card, as well as the sockets on the PCI bracket.
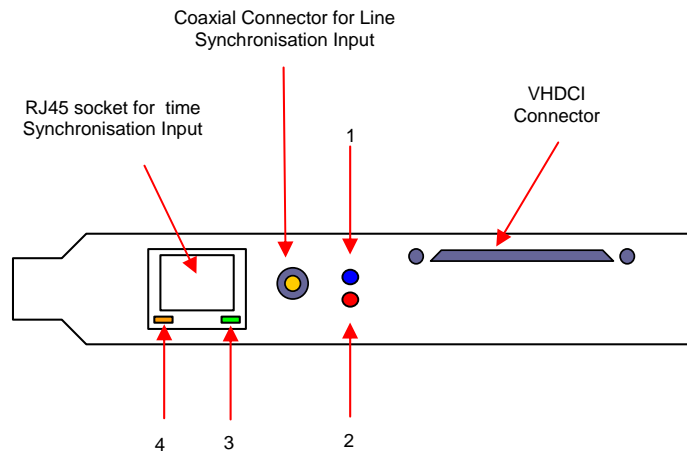
The LED functions are described below:

LED	Description
1	FPGA successfully programmed.
2	Data capture in progress
3	PPS (pulse per second) Out: indicates the card is sending a clock synchronisation signal
4	PPS (pulse per second) In: indicates the card is sending a clock synchronisation signal

Load the FPGA Image

Note: Before you can configure the card you must first load the card with the appropriate FPGA image for the type of data you want to capture. It is important you understand the protocol used by the network to which you want to connect. If you do not load the correct image for the protocol you will not be able to capture data.

- For High Level Data Link Control (HDLC) capture use:

    ```
    dagrom –rvp –d dag0 –f xilinx/dag37tpci-hdlc-erf.bit
    ```

- For Asynchronous Transfer Mode (ATM) capture use:

    ```
    dagrom –rvp –d dag0 –f xilinx/dag37tpci-atm-erf.bit
    ```

- For ATM <u>and</u> HDLC capture use:

    ```
    dagrom –rvp –d dag0 –f xilinx/dag37tpci-mixed-erf.bit
    ```

    Note: The mixed FPGA image does not support transmit.

Display Current Configuration

- Once you have loaded the appropriate image you should run the `dagthree` tool without arguments to display the current card configuration using:

    `dagthree –d dag0` (where `dag0` is the device number of the DAG card)

An explanation of an example `dagthree` output is shown below:

Resets or clears the framer status

Unsets (nofcl) or sets (fcl) facility loopback. **Note:** fcl loops back to the line.

Receive and transmit termination, encoding and line type as defined by the selected *Mode*.

Corresponds toRJ-45 connections 0-15 on pod

Unsets (noeql) or sets (eql) equipment loopback. **Note:** eql loops back to the PCI bus.

```
links 0-7      clear   E1
links 8-15     noreset T1
link 0  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 1  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 2  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 3  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 4  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 5  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 6  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 7  mode=29 rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 E1
link 8  mode=8  rxpkts txpkts nofcl eql rxterm120 txterm120 b8zs/hdb3 T1
link 9  mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
link 10 mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
link 11 mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
link 12 mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
link 13 mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
link 14 mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
link 15 mode=8  rxpkts txpkts nofcl eql rxterm100 txterm100 b8zs/hdb3 T1
pci     33MHz 32-bit buf=32MiB rxstreams=1 txstreams=1 mem=24:8
Firmware:dag37tpci_erf-atm_pci_v2_33 3s1500fg456 2006/03/28 18:14:59
(user)
time stamp      head
```

Indicates the specific line characteristics as described in the *Mode* table below

Enables(rxpkts) or disables (norxpkts) the receive stream

Enables (txplts) or disables (notxpkts) the transmit stream

Timestamp is recorded at the start (head) or end (tail) of a received packet.

Total memory available to the DAG card in Windows this 32MB. In Linux it is 128mMB

Memory in MiB allocated to receive (24MiB) and transmit (8MiB) streams.

Mode Options

The table below describes the different line characteristics associated with each of the supported modes.

Note: Ensure that the mode you select matches the physical characteristics of the network to which you want to connect.

Mode	Type	Tx LBO	Cable	Coding
0	T1 Long Haul/36dB	0dB	100Ω/ TP	B8ZS
1	T1 Long Haul/36dB	-7.5dB	100Ω/ TP	B8ZS
2	T1 Long Haul/36dB	-15dB	100Ω/ TP	B8ZS
3	T1 Long Haul/36dB	-22.5dB	100Ω/ TP	B8ZS
4	T1 Long Haul/45dB	0dB	100Ω/ TP	B8ZS
5	T1 Long Haul/45dB	-7.5dB	100Ω/ TP	B8ZS
6	T1 Long Haul/45dB	-15dB	100Ω/ TP	B8ZS
7	T1 Long Haul/45dB	-22.5dB	100Ω/ TP	B8ZS
8	T1 Short Haul/15dB	0-133 ft./ 0.6dB	100Ω/ TP	B8ZS
9	T1 Short Haul/15dB	133-266 ft./ 1.2dB	100Ω/ TP	B8ZS
10	T1 Short Haul/15dB	266-399 ft./ 1.8dB	100Ω/ TP	B8ZS
11	T1 Short Haul/15dB	399-533 ft./ 2.4dB	100Ω/ TP	B8ZS
12	T1 Short Haul/15dB	533-655 ft./ 3.0dB	100Ω/ TP	B8ZS
13	T1 Short Haul/15dB	Arbitrary Pulse	100Ω/ TP	B8ZS
14	T1 Gain Mode/29dB	0-133 ft./ 0.6dB	100Ω/ TP	B8ZS
15	T1 Gain Mode/29dB	133-266 ft./ 1.2dB	100Ω/ TP	B8ZS
16	T1 Gain Mode/29dB	266-399 ft./ 1.8dB	100Ω/ TP	B8ZS
17	T1 Gain Mode/29dB	399-533 ft./ 2.4dB	100Ω/ TP	B8ZS
18	T1 Gain Mode/29dB	533-655 ft./ 3.0dB	100Ω/ TP	B8ZS
19	T1 Gain Mode/29dB	Arbitrary Pulse	100Ω/ TP	B8ZS
20	T1 Gain Mode/29dB	0dB	100Ω/ TP	B8ZS
21	T1 Gain Mode/29dB	-7.5dB	100Ω/ TP	B8ZS
22	T1 Gain Mode/29dB	-15dB	100Ω/ TP	B8ZS
23	T1 Gain Mode/29dB	-22.5dB	100Ω/ TP	B8ZS
24	E1 Long Haul/36dB	ITU G.703/Arbitrary	75Ω/ coax	HDB3
25	E1 Long Haul/36dB	ITU G.703/Arbitrary	120Ω/ TP	HDB3
26	E1 Long Haul/43dB	ITU G.703/Arbitrary	75Ω/ coax	HDB3
27	E1 Long Haul/43dB	ITU G.703/Arbitrary	120Ω/ TP	HDB3
28	E1 Short Haul	ITU G.703/Arbitrary	75Ω/ coax	HDB3
29	E1 Short Haul	ITU G.703/Arbitrary	120Ω/ TP	HDB3
30	E1 Gain Mode	ITU G.703/Arbitrary	75Ω/ coax	HDB3
31	E1 Gain Mode	ITU G.703/Arbitrary	120Ω/ TP	HDB3

Default Configuration

Before configuring the card for your specific requirements Endace recommends that you return the card to the default settings using:

```
dagthree -d1 default
```

The default output is shown below:

```
links 0-7        noreset E1
links 8-15       noreset E1
link 0  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 1  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 2  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 3  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 4  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 5  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 6  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 7  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 8  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 9  mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 10 mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 11 mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 12 mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 13 mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 14 mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
link 15 mode= 0 rxpkts txpkts nofcl noeql rxterm=ext txterm=ext b8zs/hdb3 E1
pci     33MHz 32-bit buf=32MiB rxstreams=1 txstreams=1 mem=16:16
Firmware: dag37tpci_erf-atm_pci_v2_33 3s1500fg456 2006/03/28 18:14:59
(user)
time stamp      head
```

> **Note:** The default configuration will display for all 16 links even if there is no interface physically connected to the pod.

Returning the card to the default configuration means that the links will have the same settings. You then only need to re-configure those you want to change from the default.

Configuring the Links

Overview

Configure the links that you want to change from the default settings according to your specific requirements, referring to the *Mode* and *Supported Line Characteristics* information provided earlier.

You can change multiple settings at the same time and you can also configure more than one link at the same time. However the new settings will be applied in the order in which you specify them.

Examples

To change link 1 to T1 using mode 8 and eql, use:

```
dagthree -d dag 0 link=1 mode=8 eql
```

To change link 2 to mode 29 and link 3 to mode 29 but receive only, use:

```
dagthree -d dag0 link=2 mode=29 link=3 mode=29 notxpkts
```

If you do not specify a link the changes will be applied to all links. In the example below all links will be changed to mode 29:

```
dagthree -d dag0 mode=29
```

Configuring the Links (cont.)

Using "default"

If you use "default" any settings you specify earlier in the command line will be returned to the default factory setting regardless of the setting specified. In addition you can not return individual links to the default settings. The default settings will be applied to <u>all</u> links <u>unless</u> they are specified after "default" in the command line.

In the example below only link 4 will not have the default settings. All other links including links 1, 2 and 3 will be returned to the default settings:

```
dagthree -d dag0  link=1 mode=29 link=2 mode=29 eql link=3
default link=4 notxpkts
```

Interface Statistics

Overview

When you have configured the card according to your specific requirements you can view the interface statistics to check the status of each of the links using:

```
dagthree -d dag0 -si
```

An example output is shown below:

Line Interface Unit (LIU) conditions

Framer conditions

if	los	ais	lcv	fls	dmo	clos	-	rx0	rx1	tx0	tx1	crc	ais	ferr	lcd	up
0	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
1	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
2	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
3	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
4	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
5	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
6	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
7	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
8	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
9	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
10	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
11	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
12	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
13	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
14	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1
15	0	0	0	0	0	0	-	1	1	1	1	0	0	0	0	1

Interface 0-15. Corresponds to RJ-45 connections 0-15 on pod.

The output shows the condition status of each of the links.

Note: "1" indicates the condition is present on the link.
"0" indicates the condition is not present on the link.

The output will include a number of status bits as they have occurred since the last read. In our example, the read interval is set 1 sec via the -i option.

See the *Definitions* section next in this chapter for a full description of each of the status conditions.

Definitions

A definition of each of the status bits is described below:

Condition	Definition
if	Interface or port number (0-15)
los	LIU Loss of Signal. There is not enough voltage swing on the input line.
ais	LIU Alarm Indication Signal. The input is receiving only the value "1"
lcv	LIU Line Code Violation. Data coding is in valid and does not comply with the AMI/HDB3 or B8ZS standard.
fls	LIU FIFO Limit Status. The data recovery FIFO has overflowed.
dmo	LIU Drive Monitor Output. Limits the transmit output voltage.
clos	LIU Cable Loss (dB +/- 1dB). The loss seen on the cable relative to 0dB.
rx0	The framer has received the value "0". This is useful to ensure the line toggles correctly.
rx1	The framer has received the value "1". This is useful to ensure the line toggles correctly.
tx0	The framer has transmitted the value "0". This is useful to ensure the line toggles correctly.
tx1	The framer has transmitted the value "0". This is useful to ensure the line toggles correctly.
crc	Framer CRC error. The framer has detected a CRC error. Only valid for E1 CRC.
ais	Framer Alarm Indication Signal. The framer (internally) is only receiving the value "1" as payload.
ferr	Framer Error. A framing error has been detected.
lcd	Loss of Cell Delineation. The ATM demapper experienced a problem locking to the ATM cell stream.
up	Framer Up The Framer is locked to the frame sequence.

Interpretation

The statistics display which dagthree outputs provides you with detailed information about the status of each of the individual links that you have configured on the card. If any of the conditions on any of the links are not as you expect you may need to review your configuration options and change them accordingly

As a general rule you should use the lowest possible gain control value (eg. short haul) to avoid over amplifying the signal which increases the chance of signal errors.

For Example: For a twisted pair E1 line you could use Mode 29 (minimum gain control, signal attenuated less < 15dB). However if dagthree reports a poor signal you could increase the amplification by using Mode 25 (long haul/36dB) or even Mode 27 (extended long haul/ 43 dB).

Help

If assistance with configuring the DAG card or any other aspect of configuration you can use print a help file by using:

```
dagthree -h
```

Chapter 4:
Configuring HDLC Connections

Overview

You can define the HDLC physical channels for the card by using the `dagchan` tool to read a channel configuration file and then create each of the channels defined in that file. `Dagchan` is one of the standard tools shipped with your Endace software.

> **Note:** To use `dagchan` to configure HDLC channels you must have either the HDLC firmware or the mixed firmware loaded on the card.

You can create a channel configuration file using any common text editor such as Notepad, VI Editor, Emacs etc.

By default `dagchan` deletes all existing channel definitions on the card, and <u>then</u> creates the new channel definitions. Use this option if you are configuring a new card or if you want to remove all existing definitions. If the card you are configuring has existing channel definitions that you want to retain, just add new definitions by using the "`-r`" option in the command line.

Configuration File

You can use the configuration file to do the following:

- Configure HDLC channel connection characteristics,
- Enable/disable RAW Rx,
- Designate channels to either receive or transmit.

> **Note:** The type of connections shown in the configuration file reflect the data that is present on the network to which you are connected. Please ensure you understand the type of connection you wish to monitor before beginning to configure the interfaces.

Each line in the configuration file represents a different command. Each command begins with a letter which designates the connection type, followed by a sequence of one or more numbers as shown in the examples later in this chapter. The following letters are valid

- 'c' = timeslot connection,
- 'd' = delete connection,
- 'h' = hyper-channel connection,
- 'l' = line connection,
- 'r' = RAW line connection,
- 's' = sub-channel connection,
- 'lr' = RAW line connection,
- 'hr' = RAW hyper-channel connection,
- 'sr' = RAW sub-channel connection,
- 'cr' = RAW channel connection.

Multiple Interfaces

If you wish to configure connections on multiple interfaces you will need to repeat the configuration procedure for each interface you want to monitor.

Receive/ Transmit

By default all new connections will receive. To designate specific connections to either receive or transmit you must type "transmit" on the line of the file immediately before the configuration line for that connection.

All subsequent connections in the file will also transmit unless you type "receive" after which all subsequent connections will receive. This is shown in the sample *Hyper-Channel Connection (h)* chan.config file later in this chapter.

> **Note:** If you do not type either "receive" or "transmit" into the configuration file all connections will default to receive.

Connection ID

HDLC Connections

The first HDLC connection defined on the board is assigned connection ID 16. Each subsequent connection is assigned a successive ID up to the limit of 496 connections.

A typical HDLC data stream is bit stuffed with a zero bit after every five consecutive one bits to prevent confusion with the HDLC packet delimiter (0x7e or 0111 1110).

RAW Connections

Raw connections are assigned connection ID 0 through to ID 15. On a RAW connection there is no bit unstuffing.

Output Record Formats

The output from HDLC connections depends upon the connection type as shown below:

> **Note:** Please refer to *Chapter 9: Data Formats* later in this User Guide for detailed descriptions of ERF record formats.

Connection Type	Output ERF Type	Data Type
(-h), (-s), (-c), (-l)	Type 5	unbitstuffed unframed
(-r)	Type 6	bitstuffed unframed
(-lr), (-hr), (-sr), (-cr)	Type 8	bitstuffed framed

Connection Types

Hyper-Channel Connection [h]

A hyper-channel connection is an HDLC connection which occupies one or more timeslots on one interface. This line would look like:

```
h chan_format ifc_num ts_num ts_num ts_num
```

Note: You can use up to 32 `ts_num` identifiers

To configure a connection on interface 3, timeslots 0, 1, 2, 26, 27, 28, 29, the line would look like:

```
h  0   3   0   1   2   26   27   28   29
```

An example of a `chan.cfg` file for 16 PCM 30 E1 hyper-channels is shown below:

All connections below this designator will operate in receive mode.

Interface 0-9. Corresponds to RJ-45 connections 0-9 on pod.

Timeslot Connections See note below:

Channel Type: 0 = default (HDLC when the HDLC image is used)

```
receive
h 0  0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  1 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  2 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  3 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  4 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  5 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  6 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  7 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  8 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  9 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
transmit
h 0 10 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0 11 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0 12 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0 13 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0 14 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0 15 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
```

All connections below this designator will operate in transmit mode.

Hyper-Channel Designator

Interface 10-15. Corresponds to RJ-45 connections 10-15 on pod.

Note: Timeslots 0 and 16 are not available as PCM 30 uses these for framing and signaling.

Connection Types (cont.)

Timeslot Connection [c]

A timeslot connection is a connection which occupies only one timeslot on one interface at 64 kbps. The line would look like:

```
c chan_format ifc_num ts_num
```

To configure a connection on interface 5, timeslot 16, it the line would look like:

```
c      0      5      16
```

Sub-channel Connection [s]

A sub-channel connection is a connection which occupies one, two, or four consecutive bits within a timeslot on one interface. The line would look like:

```
s chan_format ifc_num ts_num ts_mask
```

To configure a connection on interface 6, timeslot 2, bits 0-3, the line would look like:

```
S   0   6   2   0 0 0 0 1 1 1 1
```

One bit equals 8 kbps. For example, one timeslot at 64k consists of 8kbps in a sub-channel.

An 8kbps time channel extends 2 x 8k to 16k, 4 x 8k to 32k, or 7 x 8k to 56k.

> **Note:** The DAG 3.7T does not support HDLC transmit on sub-channels.

Line Connection [l]

A line connection is a connection which occupies all timeslots (i.e. full line rate) on one interface. The line would look like:

```
l chan_format ifc_num
```

To configure a line connection on interface 0 it would look like:

```
l     0     0
```

RAW Connection [r]

A RAW connection is a connection which occupies all timeslots on one interface and is in RAW mode. RAW mode is when there is no de-framing, or bitunstuffing, only the raw data from the timeslots read by the firmware. The line would look like:

```
r ifc_num
```

To configure a RAW connection on interface 4 it would look like:

```
r 4
```

Connection Types (cont.)

Line RAW Connection [lr]

A line RAW connection with HDLC firmware is a connection which occupies all timeslots on one interface. The line would look like:

```
lr chan_format ifc_num
```

To configure a line connection on interface 0 the line would look like:

```
lr      0       0
```

Channel RAW Connection [cr]

A channel RAW connection with HDLC firmware is a connection which occupies only one timeslot on one interface. The line would look like:

```
cr chan_format ifc_num ts_num
```

To configure a connection on interface 5, timeslot 16, the line would look like:

```
cr      0       5       16
```

Hyper-Channel RAW Connection [hr]

A hyper-channel RAW connection is a connection which occupies one or more timeslots on one interface. The line would look like:

```
hr chan_format ifc_num ts_num ts_num ts_num
```

The `bit_offset` field is reserved for future use and must be set to 0.

To configure a connection on interface 3, timeslots 0, 1, 2, 26, 27, 28, 29, the line would look like:

```
hr  0  3  0  1  2  27  28  29
```

Sub-Channel RAW Connection [sr]

A sub-channel RAW connection is a connection which occupies one, two, or four consecutive bits within a timeslot on one interface.   The line would look like:

```
sr chan_format ifc_num ts_num ts_mask
```

The `bit_offset` field is reserved for future use and must be set to 0. To configure a connection on interface 6, timeslot 2, bits 4-7, the line would look like:

```
sr  0  6  2   0 0 0 0 1 1 1 1
```

Delete Connection [d]

Before you can delete a connection the connection must already exist. When you first start `dagchan` there are no connections, so all connections that you want to create or delete must exist in the `.cfg` file. The line would look like:

```
d conn_num
```

To delete connection number 17, the line would look like:

```
d 17
```

Chapter 5:
Configuring ATM Connections

Overview

You can define the ATM physical channels for the card by using the `dagchan` tool to read a channel configuration file and then create each of the channels defined in that file. `dagchan` is one of the standard tools shipped with your Endace software and can be found in the `tools` sub-directory.

> **Note:** To use `dagchan` to configure ATM channels you must have either the ATM firmware or the mixed firmware loaded on the card.

You can create a channel configuration file using any common text editor such as Notepad, VI Editor, Emacs etc.

By default `dagchan` deletes all existing channel definitions on the card, and <u>then</u> creates the new channel definitions. Use this option if you are configuring a new card or if you want to remove all existing definitions. However if the card you are configuring has existing channel definitions that you want to retain you can just add new definitions by using the "`-r`" option in the command line.

Configuration File

You can use the configuration file to do the following:

- Configure ATM channel connection characteristics,
- Designate channels to either receive or transmit.

> **Note:** The type of connections shown in the configuration file reflect the data that is present on the network to which you are connected. Please ensure you understand the type of connection you wish to monitor before beginning to configure the interfaces.

Each line in the configuration file represents a different command. Each command begins with a letter, followed by a sequence of one or more numbers. The following letters are valid line beginnings:

- 'c' = simple connection,
- 'd' = delete connection,
- 'h' = hyper-channel connection,
- 'l' = line connection,
- 't' = ATM scrambling on interface,
- 'f' = HEC correction.

Multiple Interfaces

If you wish to configure connections on multiple interfaces you will need to repeat the configuration procedure for each interface you want to monitor.

Receive/ Transmit

By default all new connections will receive. To designate specific connections to either receive or transmit you must type "transmit" on the line of the file immediately before the configuration line for that connection. All subsequent connections in the file will also transmit unless you type "receive" after which all subsequent connections will receive. This is shown in the sample *Hyper-Channel Connection (h)* chan.config file later in this chapter.

> **Note:** If you do not type either "receive" or "transmit" into the configuration file all connections will default to receive.

Output Record Formats

The output from ATM connections depends upon the connection type as shown below:

Please refer to *Chapter 9: Data Formats* later in this User Guide for further information on ERF record formats.

Connection Type	Output ERF Type
(−c), (−d), (−h), (−l),(-t), (-f)	Type 7
AAL5 Reassembly	Type 9
AAL2 Reassembly	Type 12

Connection Types

Hyper-Channel Connection [h]
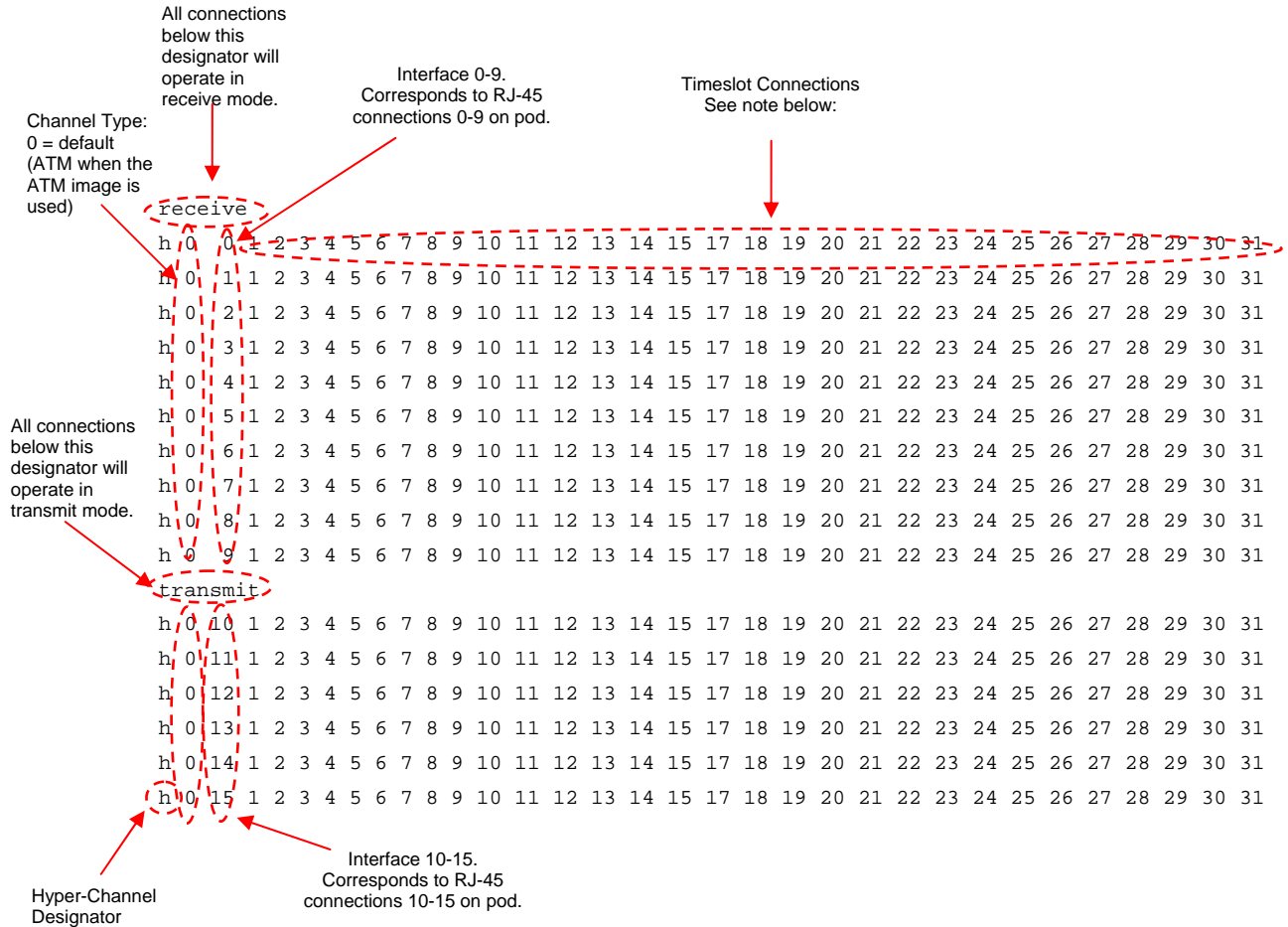
A hyper-channel connection is a connection which occupies one or more timeslots on one interface. The line would look like:

```
h chan_format ifc_num ts_num ts_num ts_num
```

To configure a connection on interface 3, timeslots 0, 1, 2, 26, 27, 28, 29, the line would look like:

```
h 0 3 0 1 2 26 27 28 29
```

An example of a chan.cfg file for 16 PCM 30 E1 hyper-channels is shown next:

Channel Type:
0 = default
(ATM when the
ATM image is
used)

All connections
below this
designator will
operate in
receive mode.

Interface 0-9.
Corresponds to RJ-45
connections 0-9 on pod.

Timeslot Connections
See note below:

```
receive
h  0   0  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   1  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   2  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   3  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   4  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   5  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   6  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   7  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   8  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0   9  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
transmit
h  0  10  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0  11  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0  12  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0  13  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0  14  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h  0  15  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
```

All connections
below this
designator will
operate in
transmit mode.

Hyper-Channel
Designator

Interface 10-15.
Corresponds to RJ-45
connections 10-15 on pod.

Note: Timeslots 0 and 16 are not available as PCM 30 uses these for framing and signaling.

Timeslot Connection [c]

A timeslot connection is a connection which occupies only one timeslot on one interface. The line would look like:

```
c chan_format ifc_num ts_num
```

The field is reserved for future use and must be set to 0.

To configure a connection on interface 5, timeslot 16, the line would look like:

```
c     0     5     16
```

Line Connection [l]

A line connection is a connection which occupies all timeslots (i.e. full line rate) on one interface. The line would look like:

```
l chan_format ifc_num
```

To configure a line connection on interface 0, the line would look like:

```
l     0     0
```

Connection Types (cont.)

ATM Scrambling on Interface [t]

The DAG 3.7T card is equipped to scramble any interface using the self synchronising scrambling polynomial specified in ITU I.432. When an interface is scrambled, all connections on the interface will be scrambled.

To enable scrambling on an interface, the line would look like:

```
t ifc_num
```

In order to configure interface 3 to scramble ATM cells, the line would look like:

```
t     3
```

HEC Connection on Interface [f]

The DAG 3.7T card can correct single bit errors in the HEC field as detailed in the ITU I.432 specification. When an interface is correcting HECs, all connections on the interface will be corrected.

To enable HEC correction on an interface, the line would look like this:

```
f ifc_num
```

To configure interface 3 to perform HEC correction, the line would look like:

```
f     3
```

Delete Connection [d]

Before you can delete a connection the connection must already exist. When you first start dagchan there are no connections, so all connections that you want to create or delete must exist in the .cfg file. The line would look like:

```
d conn_num
```

To delete connection number seventeen, the line would look like:

```
d 17
```

Chapter 6:
Configuring Mixed ATM and HDLC Connections

Using Mixed Firmware

The mixed firmware image allows you to capture both ATM and HDLC data on the same DAG card.

Note: The mixed firmware does <u>not</u> support transmit. If you wish to transmit data as well as receive you must use <u>either</u> the ATM <u>or</u> the HDLC firmware image.

You can use `dagchan` to set the `chan_format` for individual connections to either HDLC or ATM. By default the mixed firmware image is set to HDLC.

Valid channel types are :

- `0` = Default (HDLC for mixed image)

- `1` = HDLC

- `2` = ATM

An example a `chan.config` file for mixed HDLC and ATM Hyper-Channel connections occupying all timeslots is shown below:

Channel Format:
0 = default
1= HDLC
2= ATM

Interface 0-15.
Corresponds to RJ-45
connections 0-15 on

Timeslot connections
0-31

```
h 0  0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  2 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 0  3 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 1  4 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 1  5 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 1  6 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 1  7 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2  8 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2  9 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2 10 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2 11 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2 12 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2 13 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2 14 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
h 2 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
```

Hyper-Channel
Designator

Note: Because the mixed firmware does not support transmit you do not need to set individual connections to receive or transmit. All connections will automatically default to receive only.

For detailed information on configuring specific HDLC and ATM connections please refer to *Chapter 4: Configuring HDLC Connections* and *Chapter 5: Configuring ATM Connections* earlier in this User Guide.

Chapter 7:
Capturing Data

Starting a Session

For a typical data capture session follow the steps listed below:

- Move to the `dag` directory,
- Load the appropriate driver,
- Then load the appropriate FPGA image to each DAG card. For example, for HDLC capture with one DAG 3.7T card installed use:

  ```
  dagrom -rvp –d dag0 -f xilinx/dag37tpci-hdlc-erf.bit
  ```

- Set the integrity of the card's physical layer and check the integrity of the physical layer to each DAG card. For example:

  ```
  dagthree –d0 dag0 default
  ```

- Start the capture session using:

  ```
  dagsnap -d dag0 –v -o tracefile
  ```

  Note: You can use the `-v` option to provide user information during a capture session, although you may want to omit it for automated trace runs.

By default `dagsnap` will run indefinitely but can be stopped using *CTRL+C*. You can also configure `dagsnap` to run for a fixed time period then exit.

High Load Performance

Overview

As the DAG 3.7T card captures packets from the network link, it writes a record for each packet into a large buffer in the host PC's main memory.

Avoiding Packet Loss

To avoid packet loss, the user application reading the record, such as `dagsnap`, must be able to read records out of the buffer faster than they arrive. If not the buffer will eventually fill and packet records will be lost.

If the user process is writing records to hard disk, it may be necessary to use a faster disk or disk array. If records are being processed in real-time, a faster host CPU may be required.

In Linux and Free BSD, when the PC buffer fills, the following message displays on the PC screen:

```
kernel: dagN: pbm safety net reached 0xNNNNNNNNN
```

The same message is also printed to `log /var/log/messages`. In addition, when the PC buffer fills the "Data Capture" LED on the card will flash or flicker, or may go OFF completely.

In Windows no screen message displays to indicate when the buffer is full. Please contact Endace Customer Support at support@endace.com for further information on detecting buffer overflow and packet loss in Windows

High Load Performance (cont.)

Detecting Packet Losses

Once the buffer fills, any new packets arriving will be discarded by the DAG card until some data is read out of the buffer to create free space.

You can detect any such losses by observing the Loss Counter (`lctr` field) of the Extensible Record Format [ERF]. See *Chapter 9: Data Formats* later in this User Guide for more information on the Endace ERF.

Increasing Buffer Size

You can increase the size of the host PC buffer to enable it to cope with bursts of high traffic load on the network link.

By default the `dagmem` driver reserves 32MB of memory per DAG card in the system. However if you are capturing at OC-12/STM-4 (622Mbps) rates or above, you may require a larger buffer.

For Linux/BSD, please refer to the *Linux FreeBSD Software Installation Guide,* which is shipped on the installation CD with the DAG3.7T card, for further information on increasing buffer size.

For Windows the upper limit is 32MB. This is usually sufficient, however if you do need to increase the amount of reserved memory please contact Endace customer support at support@endace.com for more information

The `dsize` option sets the amount of memory used per DAG card in the system.

> **Note:** For 32-bit Linux kernels, the value of `dsize` multiplied by the number of DAG cards in the system must be <u>less than</u> the amount of physical memory installed, as well as <u>less than</u> 890MB.

Configuring Extended Functions

Overview

The embedded XScale processor provides the means to perform the following extended functions:

- Reassembly of AAL2/AAL5 cells,
- Inverse Multiplexing (IMA) monitoring over ATM,
- Frame filtering over HDLC.

  Note: You can run IMA and HDLC filtering simultaneously however this "mixed" feature does not provide transmit functionality.

Before you can make use of these extended functions you must ensure you have completed the following steps as described in *Chapter 3: Configuring the Card* earlier in this User Guide:

- Load the ATM, HDLC or mixed firmware image depending upon the function you want to perform,
- Set the correct physical layer characteristics required to receive data,
- Configure the channels for the type of data you want to capture,
- Check (using `dagsnap`) that you are able to capture ATM or HDLC traffic or both as appropriate.

Loading the Images

To make use of the extended functionality provided by the XScale, you must load the appropriate XScale images. You can do this using the `dagrom.` tool and the region (`-c`) switch. There are three images required. They are:

> - Bootloader,
> - Kernel, and
> - Filesystem.

- Load the bootloader image using:

  ```
  dagrom -rvl -cb -f d37t_bootloader.rom
  ```

- Then load the kernel using:

  ```
  dagrom -rvl -ck -f d37t_kernel
  ```

- Then load the appropriate file system depending upon the function you want to use:

  for AAL2/AAL5 reassembly load:

  ```
  dagrom -rvl -cf -f d37t_AAL_filesystem.ext2.gz
  ```

  for IMA monitoring load:

  ```
  dagrom -rvl -cf -f d37t_IMA_filesystem.ext2.gz
  ```

  for HDLC filtering load:

  ```
  dagrom -rvl -cf -f d37t_HDLC_filesystem.ext2.gz
  ```

  for IMA and HDLC filtering load:

  ```
  dagrom -rvl -cf -f d37t_mixed_filesystem.ext2.gz
  ```

  Note: The mixed image does not support transmit

Configuring Extended Functions (cont.)

Starting the XScale

You can start the XScale using one of the following "demo" tools with the "–x" option depending upon the function you want to perform:

- `dagaal5demo` for AAL2/AAL5 reassembly
- `dagimademo` for IMA
- `daghdlcdemo` for HDLC filtering

   Note: You can also make a connection to the embedded processor by using the `dagema` library.**:** For more information on the `dagema` library please contact Endace Customer Support at <u>support@endace.com</u>.

Directing Data to the XScale

In normal circumstances when you run `dagthree`, data is directed from the line directly to the host computer.

However to enable any of the extended functions the data must be directed from the line <u>then</u> to the XScale processor, <u>then</u> to the host computer.

All the "demo" tools listed in the section above do this by default. Alternatively you can use the `dag_set_mux()` function from the DAG3.7T library.

Using the AAL Reassembler

The AAL Reassembler allows you to reassemble AAL5 or AAL2 frames on the DAG 3.7T card without involving the host computer in processing. The reassembler receives ATM traffic from the line and then sends it to the host either unchanged, dropped or reassembled into AAL5 or AAL2 frames depending upon the configuration used.

You can use different configurations on different virtual connections, allowing only the required data to be reassembled and other data to be either preserved or rejected.

- To configure an ATM connection to reassemble either AAL2 or AAL5 frames use `dagaal5demo` ensuring you use the –x option to start the XScale on the <u>first</u> connection:


```
dagaal5demo –d dag0 –x –p 1 –c 33 –C16 –m1 –n1
```

Option to start XScale

AAL Mode:
0 = ATM
1 = AAL2
2 = AAL5

Net Mode:
0 = NNI
1 = UNI

- Configure as many additional connections to reassemble AAL frames as required. You do not need to use the –x option on additional connections.
- Start capturing the data using:
   ```
   dagsnap –d dag0 –v –o output_filename
   ```

For more information on the specific configuration options available for the AAL2/AAL5 reassembler please refer to the *SAR API Programming Guide* available from Endace Customer Support at <u>support@endace.com</u>.
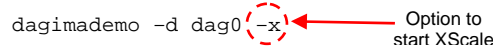
Using the IMA Monitor

Note: The IMA Monitor is intended for use in wire-tap or monitoring mode. You can not use it to terminate an IMA connection.

The IMA Monitor implements a subset of the IMA standard that allows you to monitor/tap an IMA connection and receive fully de-multiplexed ATM cells, as if it were the receiver.

By default the IMA Monitor blocks ATM cells on all interfaces unless you configure a group specifically or use auto-configuration. However running `dagimademo` sets the monitor to auto-configuration mode. In this mode the monitor will auto-configure any groups that it detects on the line.

- To run the IMA Monitor in auto-configuration mode on all interfaces use:

```
dagimademo –d dag0 –x
```
Option to start XScale

- If the IMA monitor detected groups on one or more of the interfaces you will be able to start capturing the re-ordered cell data using:

```
dagsnap –d dag0 –v –o output_filename
```

For more information on the specific configuration options available for the IMA Monitor please refer to the *IMA API Programming Guide* available from Endace Customer Support at support@endace.com.
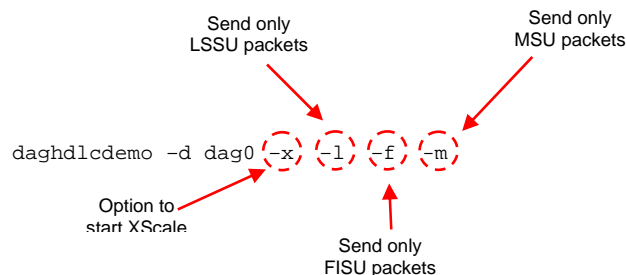
Using the HDLC Filter

The HDLC filter is designed to allow Layer 2 filtering and filtering on HDLC packets on the DAG 3.7T card without involving the host computer in processing. The filter receives HDLC traffic from the line and then either sends it to the host unchanged or dropped, depending upon the filter configuration used.

By default the filter drops the packets that much the filter and sends all other packets to the host. If you do not setup any filters all packets will be sent to the host. You can set up filters using one r more of the "`-l`", "`-f`" or "`-m`" options.

- To run the HDLC Filter on all interfaces with filters setup use:

Send only LSSU packets
Send only MSU packets

```
daghdlcdemo –d dag0 –x –l –f –m
```
Option to start XScale
Send only FISU packets

- Start capturing the filtered data using:

```
dagsnap –d dag0 –v –o output_filename
```

For more information on the specific configuration options available for the HDLC Filter please refer to the *HDLC Filtering API Programming Guide* available from Endace Customer Support at support@endace.com.

Using HDLC and IMA Together

You can use the IMA Monitor and the HDLC Filter at the same time by running `dagimademo` and `daghdlcdemo` consecutively.

> **Note:** Ensure you do not use the "`-x`" option with the second `demo` tool that you run. If you do you will restart the XScale and overwrite the configuration options set by the first `demo` tool.

To do this you must have the "mixed" FPGA image loaded which allows you to capture both HDLC and ATM traffic on the one DAG Card. See *Load the FPGA Image* in *Chapter 3: Configuring the Card* earlier in this user guide for further information.

> **Note:** If you load the "mixed" FPGA image you will be able to <u>receive only</u>. If you want to be able to receive and transmit you must load <u>either</u> the ATM image <u>or</u> the HDLC image and use them one at a time.

Chapter 8:
Synchronizing Clock Time

Overview   The Endace DAG cards have sophisticated time synchronisation capabilities, which allow for high quality timestamps, optionally synchronized to an external time standard.

The core of the DAG synchronisation capability is known as the DAG Universal Clock Kit (DUCK).

An independent clock in each DAG card runs from the PC clock. The card's clock is initialised using the PC clock, and then free-runs using a crystal oscillator.

Each card's clock can vary relative to a PC clock, or other DAG cards.

DUCK Configuration   The DUCK is designed to reduce time variance between sets of DAG cards or between DAG cards and coordinated universal time [UTC].

You can obtain an accurate time reference by connecting an external clock to the DAG card using the time synchronisation connector. Alternatively you can use the host PCs clock in software as a reference source without any additional hardware.

Each DAG card can also output a clock signal for use by other cards.

Common Synchronization   The DAG card time synchronisation connector supports a Pulse-Per-Second (PPS) input signal, using RS-422 signalling levels.

Common synchronisation sources include GPS or CDMA (cellular telephone) time receivers.

Endace also provides the TDS 2 Time Distribution Server modules and the TDS 6 units that enable you to connect multiple DAG cards to a single GPS or CDMA unit.

For more information please refer to the Endace website at http://www.endace.com/accessories.htm , or the *TDS 2/TDS 6 Units Installation Manual*.

Timestamps

ERF files contains a hardware generated timestamp of each packet's arrival. The arrival time can be either the point at which the start of the packet arrives (head) or the point at which the end of the packet arrives (tail).

See *Default Configuration* in *Chapter 3: Card Configuration* earlier in this user guide for more information on configuring the timestamp head/tail option

The format of this timestamp is a single little-endian 64-bit fixed point number, representing the number of seconds since midnight on the January 1970.

The high 32-bits contain the integer number of seconds, while the lower 32-bits contain the binary fraction of the second. This allows an ultimate resolution of 2-32 seconds, or approximately 233 picoseconds.

The ERF timestamp allows you to find the difference between two timestamps using a single 64-bit subtraction. You do not need to check for overflows between the two halves of the structure as you would need to do when comparing Unix time structures.

Different DAG cards have different actual resolutions. This is accommodated by the lowermost bits that are not active being set to zero. In this way the interpretation of the timestamp does not need to change when higher resolution clock hardware is available.

Example

Below is example code showing how a 64-bit ERF timestamp (erfts) can be converted into a struct timeval representation (tv):

```
unsigned long long lts;
struct timeval tv;

lts = erfts;
tv.tv_sec = lts >> 32;
lts = ((lts & 0xffffffffULL) * 1000 * 1000);
lts += (lts & 0x80000000ULL) << 1;      /* rounding */
tv.tv_usec = lts >> 32;
if(tv.tv_usec >= 1000000) {
    tv.tv_usec -= 1000000;
    tv.tv_sec += 1;
    }
```

Configuration Tools

The DUCK is very flexible, and can be used with or without an external time reference. It can accept synchronisation from several input sources, and also be made to drive its synchronisation output from one of several sources.

Synchronisation settings are controlled by the `dagclock` utility.

Note: You should only run `dagclock` after you have loaded the appropriate Xilinx images. If at any stage you reload the Xilinx images you must rerun `dagclock` to restore the configuration.

```
 dagclock -h
Usage: dagclock [-hvVxk] [-d dag] [-K <timeout>] [-l <threshold>] [option]
```

-h	--help,--usage	this page
-v	--verbose	increase verbosity
-V	--version	display version information
-x	--clearstats	clear clock statistics
-k	--sync	wait for duck to sync before exiting
-d	dag	the DAG device
-K	timeout	sync timeout in seconds, default 60
-l	threshold	health threshold in ns, default 596

Option:

default	RS422 in, none out
none	None in, none out
rs422in	RS422 input
hostin	Host input (unused)
overin	Internal input (synchronise to host clock)
auxin	Aux input (unused)
rs422out	Output the rs422 input signal
loop	Output the selected input
hostout	Output from host (unused)
overout	Internal output (master card)
set	Set DAG clock to PC clock
reset	Full clock reset. Load time from PC, set rs422in, none out

Note: By default, all DAG cards listen for synchronisation signals on their RS-422 port, and do not output any signal to that port

```
dagclock -d dag0
muxin   rs422
muxout  none
status  Synchronised Threshold 596ns Failures 0 Resyncs 0
error   Freq -30ppb Phase -60ns Worst Freq 75ppb Worst Phase 104ns
crystal Actual 100000028Hz Synthesized 67108864Hz
input   Total 3765 Bad 0 Singles Missed 5 Longest Sequence Missed 1
start   Thu Apr 28 13:32:45 2005
host    Thu Apr 28 14:35:35 2005
dag     Thu Apr 28 14:35:35 2005
```

Card with Reference

Overview

To obtain the best timestamp accuracy you should connect the DAG card to an external clock reference, such as a GPS or CDMA time receiver.

To use an external clock reference source, the host PC's clock must be accurate to UTC to within one second. This is used to initialise the DUCK.

When the external time reference source is connected to the DAG card time synchronisation connector, the card automatically synchronises to a valid signal.

Pulse Signal from External Source

The DAG time synchronisation connector supports an RS-422 (PPS) signal from an external source. This is derived directly from an external reference source, or distributed through the Endace TDS 2 (Time Distribution Server) module which allows two DAG cards to use a single receiver. It is also possible for more than two cards to use a single receiver by "daisy-chaining" TDS-6 expansion modules to the TDS-2 module. Each TDS-6 , module provides outputs for an additional 6 DAG cards.

Synchronise to an external source as follows:

```
dagclock –d dag0
muxin rs422
muxout none
status Synchronised Threshold 596ns Failures 0 Resyncs 0
error Freq 30ppb Phase -15ns Worst Freq 2092838ppb Worst Phase
33473626ns
crystal Actual 100000023Hz Synthesized 67108864Hz
input Total 225 Bad 0 Singles Missed 1 Longest Sequence Missed 1
start Thu Apr 28 14:55:20 2005
host Thu Apr 28 14:59:06 2005
dag Thu Apr 28 14:59:06 2005
```

Connecting the Time Distribution Server

You can connect the TDS 2 module to the DAG card using standard RJ-45 Ethernet cable including existing RJ-45 building cabling. The TDS may be located up to 600m (2000ft) from the DAG card depending upon the quality of te cable used, possible interference sources and other environmental factors. Please refer to the *TDS2/TDS6 User Guide* for more in formation

Caution: Never connect a DAG card and/or the TDS 2 module to active Ethernet equipment or telephone equipment.

Testing the Signal

For Linux and FreeBSD, when a synchronisation source is connected the driver outputs messages to the console log file `/var/log/messages`.

To test the signal is being received correctly and has the correct polarity use the `dagpps` tool as follows:

```
dagpps –d dag0
```

`dagpps` measures the input state many times over several seconds, displaying the polarity and length of input pulse. The DAG 3.7T card also has an LED indicator for synchronisation (PPS) signals. See *Chapter 3: Configuring the Card* earlier in this User Guide for more information.

Single Card No Reference

When a single card is used with no external reference, the card can be synchronised to the host PC clock. Most PC clocks are not very accurate by themselves, but the DUCK drifts smoothly at the same rate as the PC clock.

If a PC is running NTP to synchronise its own clock, then the DUCK clock is not as smooth because the PC clock is adjusted in small jumps. However the DUCK clock does not drift away from UTC.

The synchronisation achieved with this method is not as accurate as using an external reference source such as GPS.

The DUCK clock is synchronized to a PC clock by setting input synchronization selector to overflow as follows:

```
dagclock –d dag0 none overin
muxin    overin
muxout   none
status   Synchronised Threshold 11921ns Failures 0 Resyncs 0
error    Freq 1836ppb Phase 605ns Worst Freq 143377ppb Worst
Phase 88424ns
crystal Actual 49999347Hz Synthesized 16777216Hz
input    Total 87039 Bad 0 Singles Missed 0 Longest Sequence
Missed 0
start    Wed Apr 27 14:27:41 2005
host     Thu Apr 28 14:38:20 2005
dag      Thu Apr 28 14:38:20 2005
```

Two Cards No Reference

Overview

If you are using two DAG cards in a single host PC with no reference clock, you must synchronise the cards using the same method if you wish to compare the timestamps between the two cards. You may wish to do this for example if the two cards monitor different directions of a single full-duplex link. You can synchronise the cards in two ways:

- One card can be a clock master for the second. This is useful if you want both cards to be accurately synchronised with each other, but not so for absolute time of packet time-stamps, or

- One card can synchronise to the host and also act as a master for the second card

Two Cards No Reference (cont.)

Synchronising with Each Other

Although the master card's clock will drift against UTC, the cards will still be locked together. This is achieved by connecting the time synchronisation connectors of both cards using a standard RJ-45 Ethernet cross-over cable.

Configure one of the cards as the master so that the other defaults to being a slave as follows:

```
dagclock –d dag0 none overout
muxin   none
muxout  over
status  Not Synchronised Threshold 596ns Failures 0 Resyncs 0
error   Freq 0ppb Phase 0ns Worst Freq 0ppb Worst Phase 0ns
crystal Actual 100000000Hz Synthesized 67108864Hz
input   Total 0 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Thu Apr 28 14:48:34 2005
host    Thu Apr 28 14:48:34 2005
dag     No active input - Free running
```

Note: The slave card configuration is not shown as the default configuration will work.

Synchronising with Host

To prevent the DAG card clock time-stamps drifting against UTC, the master can be synchronised to the host PC's clock which in turn utilises NTP. This then provides a master signal to the slave card.

Configure one card to synchronize to the PC clock and output a RS-422 synchronization signal to the second card as follows:

```
dagclock –d dag0 none overin overout
muxin   over
muxout  over
status  Synchronised Threshold 11921ns Failures 0 Resyncs 0
error   Freq -691ppb Phase -394ns Worst Freq 143377ppb Worst Phase
88424ns
crystal Actual 49999354Hz Synthesized 16777216Hz
input   Total 87464 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Wed Apr 27 14:27:41 2005
host    Thu Apr 28 14:59:14 2005
dag     Thu Apr 28 14:59:14 2005
```

The slave card configuration is not shown, the default configuration is sufficient.

Connector Pin-outs

Overview

DAG cards have an 8-pin RJ45 connector with two bi-directional RS422 differential circuits, A and B. The PPS signal is carried on circuit A, and the serial packet is connected to the B circuit.

Pin Assignments

The 8-pin RJ45 connector pin assignments and plugs and sockets are shown below:

1	Out A+
2	Out A-
3	In A+
4	In B+
5	In B-
6	In A-
7	Out B+
8	Out B-
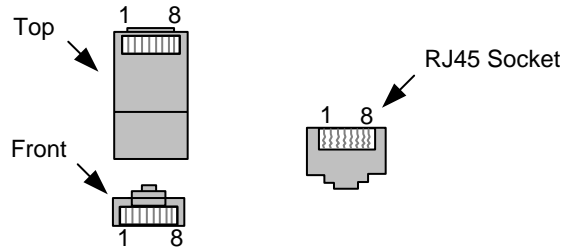
Normally you should connect the GPS input to the A channel input (pins 3 and 6).

The DAG card can also output a synchronization pulse for use when synchronizing two DAG cards without a GPS input. The synchronization pulse is output on the Out A channel (pins 1 and 2).

Ethernet Crossover Table

You can use a standard Ethernet crossover cable to connect the two cards as shown below:

TX_A+	1	3	RX_A+
TX_A-	2	6	RX-A-
RX_A+	3	1	TX_A+
RX_B+	4	7	TX_B+
RX_B-	5	8	TX_B-
RX_A-	6	2	TX_A-
TX_B+	7	4	RX_B+
TX_B-	8	5	RX_B-

44

Chapter 9:
Data Formats

Overview

DAG Cards produce trace files in their own native format called ERF (Extensible Record Format). The ERF type depends upon the type of connection you are using to capture data.

The DAG 3.7T supports the following ERF Types:

ERF Type	Description
5	TYPE_MC_HDLC: Multi-channel HDLC Frame Record
6	TYPE_MC_RAW ERF Multi-Channel Raw Link Data Record.
7	TYPE_MC_ATM Multi-Channel ATM Cell Record
8	TYPE_MC_RAW CHANNEL Multi-Channel Raw Link Data Record
9	TYPE_MC_AAL5 Multi Channel AAL5 Frame Record
12	TYPE_MC_AAL2 Multi Channel AAL2 Frame Record

The ERF file contains a series of ERF records with each record describing one packet.

An ERF file consists only of ERF records, there is no special file header which allows concatenation and splitting to be performed arbitrarily on ERF record boundaries.

Generic Header

All ERF records share some common fields. Timestamps are in little-endian (Pentium native) byte order. All other fields are in big-endian [network] byte order. All payload data is captured as a byte stream, no byte re-ordering is applied.

The generic ERF header is shown below.

Byte 3	Byte 2	Byte 1	Byte 0
timestamp			
timestamp			
type	flags	rlen	
lctr/colour		wlen	
(rlen - 16) bytes of record			

Generic Header (cont.)	**timestamp**	The time of arrival of the cell, an ERF 64-bit timestamp. See *Timestamps* in *Chapter 8: Synchronising Clock Time* earlier in this User Guide for more information on timestamps.

type

One of the following:

5:	TYPE_MC_HDLC
6:	TYPE_MC_RAW
7:	TYPE_MC_ATM
9:	TYPE_MC_AAL5
12:	TYPE_MC_AAL2

flags

This byte is divided into several fields as follows:

1-0:	Enumerates capture interface 0-3
2:	Varying record lengths
3:	Truncated record (insufficient buffer space)
4:	RX error (link layer error)
5:	DS error (internal error)
6:	Reserved
7:	General direction bit. This bit has two uses, it indicates from where a packet has arrived, either the host or line, and enables the XScale to target the packet at either the host or line. The direction bit can be interpreted in the context of either the Rx or Tx hole

In the XScale/Host Rx hole, a value of "1" indicates the ERF has arrived from the line. A value of "0" indicates the record was received from the host.

In the XScale Tx hole, a value of "1" tells the ERF Mux to direct packets to the line. A value of "0" directs packets to the host.

rlen

Record length. Total length of the record transferred over the PCI bus to storage.

lctr

Depending upon the ERF type this 16 bit field is either a loss counter of colour field. The loss counter records the number of packets lost between the DAG card and the memory hole due to overloading on the PCI bus.

wlen

Wire length. Packet length including some protocol overhead. The exact interpretation of this quantity depends on physical medium.

Type 5 Record

The Type 5 Multi-channel HDLC Frame Record is the same as the normal ERF Types but capture interface is always zero.

> **Note:** Fixed length mode is not supported. RX error is set if any MC header Error bit is set.

The record is described below:

BYTE 3	BYTE 2	BYTE 1	BYTE 0
timestamp			
timestamp			
type:5	flags	rlen	
lctr		wlen	
MC Header			
HDLC header			
(rlen – 24) bytes of packet			

The Type 5 Multi-channel HDLC Frame Record MC header is divided into several bit fields as shown below:

Bits	Attribute
0-9	Connection Number [0-511].
10-15	Reserved.
16-23	Reserved.
24	FCS Error.
25	Short Record Error [<5 Bytes].
26	Long Record Error [>2047 Bytes].
27	Aborted Frame Error.
28	Octet Error. The closing flag was not octet aligned after bit stuffing.
29	Lost Byte Error. The internal data path had an unrecoverable error.
30	1st Rec. This is the first record received since this connection was configured.
31	Reserved

Type 6 Record

The Type 6 Multi-channel RAW Time Slot Link Data Cell Record is the same as the normal ERF Types but capture interface is always zero.

Note: Fixed length is not supported. RX error is set if any MC header error bit is set.

The record is described below:

BYTE 3	BYTE 2	BYTE 1	BYTE 0
timestamp			
timestamp			
type:6	flags	Rlen	
Lctr		Wlen	
MC Header			
(rlen – 20) bytes of raw time slot link data			

The Type 6 Multi-channel RAW Time Slot Link Data Record MC header is divided into several bit fields as shown below:.

Bits	Attribute
0-3	Physical interface [0-15].
4-15	Reserved.
16-23	Reserved.
24	Reserved
25	Short Record [<6 Bytes].
26	Long Record[>2047 Bytes].
27	Reserved.
28	Reserved.
29	Lost Bytes.
30	1st Rec. This is the first record received since this connection was configured.
31	Reserved

Type 7 Record

The Type 7 Multi-channel ATM Cell Record is the same as the normal ERF Types but capture interface is always zero.

Note: Fixed length is not supported. RX error is set if any MC header error bit is set.

The record is described below:

BYTE 3	BYTE 2	BYTE 1	BYTE 0
timestamp			
timestamp			
type:7	flags	rlen	
lctr		wlen	
MC Header			
ATM Header			
48 bytes of cell			

The Type 7 Multi-channel ATM MC header is divided into several bit fields as shown below:

Bits	Attribute
0-9	Connection number [0-511] or IMA group ID
10-14	Reserved.
15	Multiplexed from IMA into internal ATM.
16-19	Physical port [0-15] cell was captured.
20-23	Reserved.
24	Lost Byte Error. The internal data path has an unrecoverable error.
25	HEC corrected.
26	OAM Cell CRC-10 Error [Not implemented].
27	OAM Cell
28	1st Rec. This is the first record received since this connection was configured.
29	Reserved.
30	Reserved.
31	Reserved.

Type 8 Record

The Type 8 Multi-Channel Raw Link Data Record is the same as the normal ERF Types but capture interface is always zero.

Note: Fixed length is not supported. RX error is set if any MC header error bit is set.

The record is described below:

BYTE 3	BYTE 2	BYTE 1	BYTE 0
timestamp			
timestamp			
type:8	flags	rlen	
lctr		wlen	
MC Header			
(rlen - 20) bytes of data			

The Type 8 Multi-Channel Raw Link Data header is divided into several bit fields as shown below:

Bits	Attribute
0-9	Connection number (0-511).
10-28	Reserved
29	Lost Byte Error. The internal datapath had an unrecoverable error.
30	1st Rec. This is the first record received since this connection was configured.
31	Reserved

Type 9 Record

The Type 9 Multi Channel AAL5 Frame Record is the same as the normal ERF Types but capture interface is always zero.

> **Note:** Fixed length is not supported. RX error is set if any MC header error bit is set.

The record is described below:

BYTE 3	BYTE 2	BYTE 1	BYTE 0
\<- timestamp ->			
timestamp			
type9	flags	rlen	
lctr		wlen	
MC Header			
ATM Header			
(rlen - 24) bytes of AAL5 frame			

Note: This is the packet length of the ATM header and data, but does <u>not</u> include the padding required for 64 bit alignment.

Note:

The Type 9 Multi Channel AAL5 Frame header is divided into several bit fields as shown below:

Bits	Attribute
0-9	Connection number (0-1023).
10-15	Reserved
16-19	Physical port (0-15) cell was captured on
20	CRC checked
21	CRC error
22	Length Checked
23	Length Error
24-27	Reserved
28	1st Cell. This is the first cell received since this connection was configured
29	Reserved
30	Reserved
31	Reserved

Type 12 Record

The Type 12 Multi Channel AAL2 Frame Record is the same as the normal ERF Types but capture interface is always zero.

> **Note:** Fixed length is not supported. RX error is set if any MC header error bit is set.

The record is described below:

BYTE 3	BYTE 2	BYTE 1	BYTE 0
	timestamp		
	timestamp		
type12	flags	rlen	
lctr		wlen	
MC Header			
ATM Header			
(rlen - 24) bytes of AAL2 frame			

Note: This is the packet length of the ATM header and data, but does <u>not</u> include the padding required for 64 bit alignment.

The Type 12 Multi Channel AAL2 Frame header is divided into several bit fields as shown below:

Bits	Attribute
0-9	Connection number (0-1023).
10-12	Reserved for possible extra connection numbers
13-15	Reserved for indication of AAL2 type (a value of 0x0 indicates a SSSAR packet).
16-19	Physical port (0-15) cell was captured on
20	Reserved
21	1st Cell. This is the first cell received since this connection was configured.
22	MAAL Error (errnum as specified in ITU I.363.2 is copied to the data part of this record)
23	Length Error
24-31	Channel Identification Number (cid)

Chapter 10
Troubleshooting

Reporting Problems

If you have problems with a DAG card or Endace supplied software which you are unable to resolve, please contact Endace Customer Support at support@endace.com.

Supplying as much information as possible enables Endace Customer Support to be more effective in their response to you. The exact information available to you for troubleshooting and analysis may be limited by nature of the problem. However the following items will assist a quick resolution:

- DAG card[s] model and serial number.
- Host PC type and configuration.
- Host PC operating system version
- DAG software version package in use
- Any compiler errors or warnings when building DAG driver or tools
- For Linux and FreeBSD, messages generated when DAG device driver is loaded. These can be collected from command dmesg, or from log file /var/log/syslog.
- Output of daginf
- Firmware versions from dagrom –x.
- Physical layer status reported by: dagthree
- Network link statistics reported by: dagthree –si
- Network link configuration from the router where available.
- Contents of any scripts in use.
- Complete output of session where error occurred including any error messages from DAG tools. The `typescript` Unix utility may be useful for recording this information.
- A small section of captured packet trace illustrating the problem.