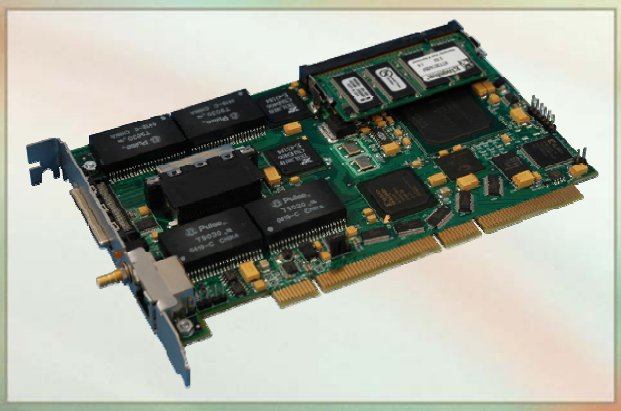
DAG 3.7T Card User Manual
2.5.5r1

EDM01.05-12r1



Endace Measurement Systems | New Zealand | www.endace.com

Leading Network Intelligence

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
Endace Technology® Ltd
Level 9
85 Alexandra Street
PO Box 19246
Hamilton 2001
New Zealand
Phone: +64 7 839 0540
Fax: +64 7 839 0543
support@endace.com
www.endace.com

Americas
Endace USA® Ltd
Suite 220
11495 Sunset Hill Road
Reston
Virginia 20190
United States of America
Phone: ++1 703 382 0155
Fax: ++1 703 382 0155
support@endace.com
www.endace.com

Europe, Middle East & Africa
Endace Europe® Ltd
Sheraton House
Castle Park
Cambridge CB3 0AX
United Kingdom
Phone: ++44 1223 370 176
Fax: ++44 1223 370 040
support@endace.com
www.endace.com

Typographical Conventions Used in this Document

- Command-line examples suitable for entering at command prompts are displayed in `mono-space courier font`.

  Results generated by example command-lines are also displayed in `mono-space courier font.`

- Information relating to functions not implemented in this beta version of this product are underlined

Protection Against Harmful Interference

When present on product this manual pertains to and indicated by product labelling, 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.

Table of Contents

1.0 PREFACE

Introduction

The installation of the Endace DAG 3.7T card on a PC begins with installing the operating system and the Endace software. This is followed by fitting the card and connecting the ports.

The installation process will also include confidence testing of the DAG 3.7T 16-interface T1/E1 HDLC/ATM PCI cards.

Viewing this document

This document, DAG 4.2GE Card User Manual is available on the installation CD.

In this chapter

This chapter covers the following sections of information.

- User Manual Purpose
- DAG 3.7T Card Product Description
- DAG 3.7T Architecture
- DAG 3.7T Card Extended Functions
- DAG 3.7T Card System Requirements

1.1 User Manual Purpose

Description

The purpose of the DAG 3.7T Card User Manual is to identify and explain:

- Installing DAG 3.7T Card
- Confidence Testing
- Running Data Capture Software
- Synchronizing Clock Time
- Data Formats Overview

Pre-requisite

This document presumes the DAG card is being installed in a PC already configured with an operating system.

A copy of the Debian Linux 3.1 (Sarge) is available as a bootable ISO image on one of the CD's shipped with the DAG card.

To install on the Linux/FreeBSD operating system, follow the instructions in the document EDM04.05-01r1 Linux FreeBSD Installation Manual, packaged in the CD shipped with the DAG card.

To install on a Windows operating system, follow the instructions in the document EDM04.05-02r1 Windows Installation Manual, packaged in the CD shipped with the DAG card.

1.2 DAG 3.7T Card Product Description

Description

The DAG 3.7T cards are PCI bus cards designed for cell and packet capture and generation on telecommunication networks. The card's key features include:

- 16 T1 or E1 Network Interfaces
- A Spartan III FPGA supporting high-performance Endace Firmware
- An Intel XScale IO Processor
- Support for receiving and sending Channelised, Unchannelised, and Fractional T1/E1, HDLC and non-HDLC data traffic.
- Support for data traffic filtering.

Figure

Figure 1-1 shows the DAG 3.7T Card.

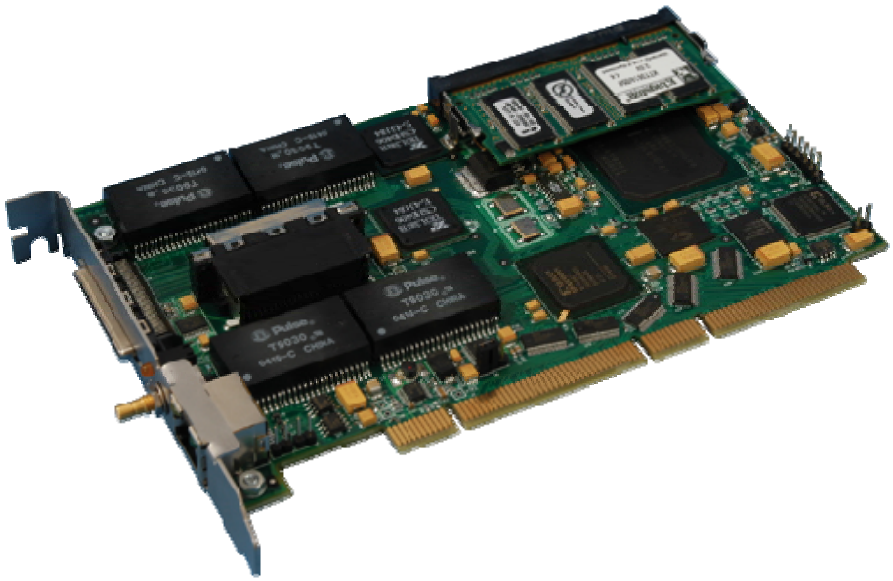


Figure 1-1. DAG 3.7T Card.

1.3 DAG 3.7T Architecture

Description

The TDM T1 or E1 data is received by the 16 RJ45 interfaces, and passed through line interface units. The data is then fed immediately into the FPGA for deframing and demapping into HDLC frames.

This FPGA contains an Ethernet processor and the DUCK timestamp engine. Because of component close association, packets or cells are time-stamped accurately. Time stamped packet records are then stored in the lower FIFO.

Continued on next page

1.3 DAG 3.7T Architecture, continued

Figure

Figure 1-2 shows the DAG 3.7T Card major components and data flow.

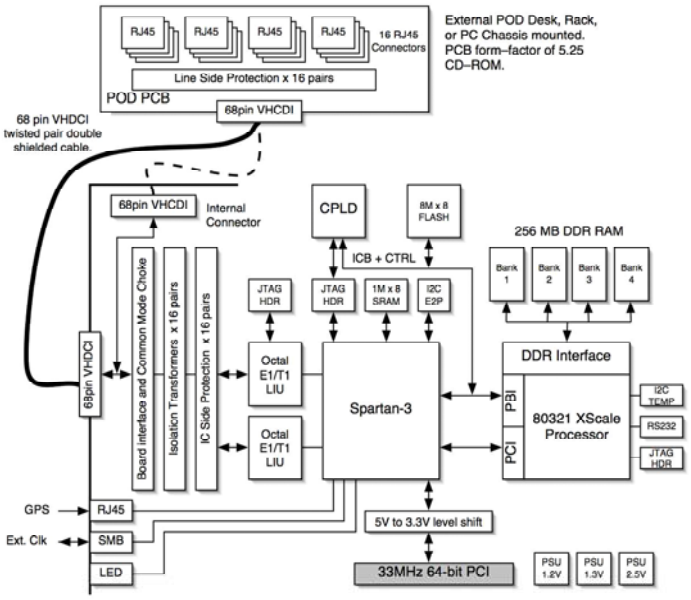


Figure 1-2. DAG 3.7T Card Major Components and Data Flow.

1.4 DAG 3.7T Card Extended Functions

Description

The DAG 3.7T functionality can be extended in many ways.

A physical transmit path is provided on the DAG 3.7T card. Special FPGA images are required in order to enable the transmit function.

Contact the Endace customer support team at support@endace.com to enable effective use of extended functions.

1.5 DAG 3.7T Card System Requirements

Description

The DAG 3.7T card and associated data capture system minimum operating requirements 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

For convenience, the Debian 3.1 [Sarge] Linux system is included on the Endace Software Install CD. Endace currently supports Windows XP, Windows Server 2000, Windows Server 2003, FreeBSD, RHEL 3.0, and Debian Linux operating systems.

Different system

For advice on using a system substantially different from that specified above, contact Endace support at support@endace.com

USE THIS SPACE FOR NOTES

2.0 INSTALLING DAG 3.7T CARD

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 placed into a slot rated for higher speeds the bus will automatically change to 33MHz, including any other devices sharing the bus.

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

In this chapter

This chapter covers the following sections of information.

- Installation of Operating System and Endace Software
- Insert DAG 3.7T Card into PC
- DAG 3.7T Card Port Connectors

2.1 Installation of Operating System and Endace Software

Description

If the DAG device driver is not installed, before proceeding with the next chapter, install the software by following the instructions in the appropriate Endace Software Installation Manual.

Go to the next chapter of information when the DAG device driver is installed.

2.2 Insert DAG 3.7T Card into PC

Description

Inserting the DAG 3.7T card into a PC involves accessing the PCI bus slot, fitting the card, and replacing bus slot screw.

Procedure

Follow these steps to insert the DAG 3.7T card.

Step 1. Access bus Slot

Power computer down.

Remove PCI bus slot cover.

Step 2. Fit Card

Insert DAG 3.7T card into PCI bus slot.

Step 3. Replace bus Slot Screw

Secure card with screw.

Step 4. Power Up Computer

Continued on next page

2.3 DAG 3.7T Card Port Connectors

Description

There is one VHDCI connector on the PCI bracket, and there is also a VHDCI connector on the DAG 3.7T board itself.

The connector on the board allows a DAG 3.7T Pod to be used internally in a PC chassis in a spare 5.25 inch drive bay.

Use the connector on the bracket if the POD is to be used outside of the chassis Only one Pod may be connected to the DAG 3.7T card at a time.

Connector pin-out

The DAG 3.7T Pod has the 16 RJ45 T1/E1 connectors. The pin-out is:

1	-
2	-
3	TX Ring
4	TX Tip
5	RX Ring
6	Rx Tip
7	-
8	-

3.0 CONFIDENCE TESTING

Introduction

The confidence testing is a process to determine the DAG 3.7T card is functioning correctly.

The process also involves a card capture session, and demonstrates configuration in the style of 'What You See You Can Change'.

Interface statistics are also inspected during this process.

In this chapter

This chapter covers the following sections of information.

- Interpreting DAG 3.7T Card LED Status
- DAG 3.7T Card LED Display Functions
- DAG 3.7T Card Capture Sessions
- Configuration in WYSYCC Style
- Configuration Options Supported
- Inspect Interface Statistics
- Configuring HDLC Channels
- Configuring HDLC RAW Connections
- Configuring ATM Channels
- Reporting Problems

3.1 Interpreting DAG 3.7T Card LED Status

Description

The DAG 3.7T has 4 status LEDs with one blue, one red, one orange and one green.

On the DAG 3.7T card the blue LED 1 for example displays when the FPGA is successfully programmed.

Figure

Figure 3-1 shows the DAG 3.7T card status LEDs.

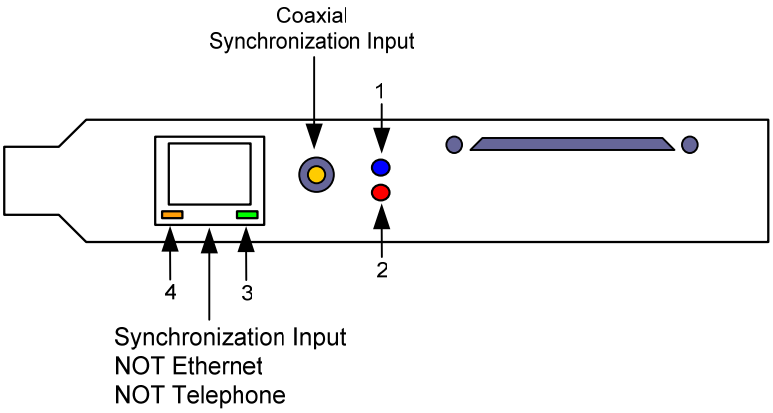


Figure 3-1. DAG 3.7T Card Status LEDs.

3.2 DAG 3.7T Card LED Display Functions

Description The DAG 3.7T LED display functions indicate a number of process conditions. When the card is powered the blue LED 1 should be lit.

Figure Figure 3-2 shows the DAG 3.7T card without network connections.

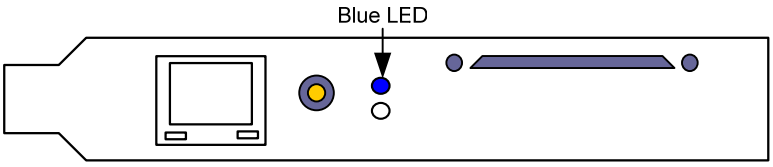


Figure 3-2. DAG 3.7T Card without Network Connections.

LED definitions The following table describes the LED display definitions:

LED	Description
LED 1	FPGA successfully programmed
LED 2	Data capture in progress
LED 3	PPS Out: Pulse Per Second Out – indicates the card is sending a clock synchronization signal
LED 4	PPS In: Pulse Per Second In – indicates the card is receiving an external clock synchronization signal

NOTE: The LED 4 is on when the Loss of Pointer or Loss of Framing conditions are true.

3.3 DAG 3.7T Card Capture Sessions

Description The DAG 3.7T uses two integrated Exar Octal T1/E1 Framers which enable support for both T1 and E1 interfaces. They also enable soft configuration of the line termination and gain, and physical layer configuration such as the selection of B8ZS or AMI zero code suppression options.

Because of its flexibility, the correct link layer configuration needs to be supplied to the card to function as expected.

For both configuration and statistics of the DAG 3.7T framers the dagthree tool is supplied. Calling `dagthree` without arguments will list the current settings. `dagthree -h` will print a help listing on the usage of the tool.

Continued on next page

3.3 DAG 3.7T Card Capture Sessions, continued

Procedure Follow these steps for the DAG 3.7T card capture session.

Step 1. Connect Pod

Connect the Pod to the DAG 3.7T using the supplied 68-pin VHDCI connector.

Step 2. Connect Integrated Exar Octal T1/E1 Framers

Using a correctly wired connector, connect the T1 or E1 line to an RJ45 socket on the Pod.

NOTE: Up to 16 lines can be connected.

Step 3 Check FPGA Images

Before starting to configure the card, make sure the most recent pair of FPGA images has been loaded onto the card. Load the newest available PCI FPGA image.

For HDLC capture:

```
dag@endace:~$ dagrom -rvp –d dag0 -f xilinx/dag37tpci-hdlc-erf.bit
```

For ATM capture:

```
dag@endace:~$ dagrom -rvp –d dag0 -f xilinx/dag37tpci-atm-erf.bit
```

Step 4. Configure DAG 3.7T Card

Configure the card according to local settings.

Check through the physical layer statistics that the card has frame synchronization and is locked to the T1 or E1 links.

Step 5. Display Card Configuration

Run the `dagthree` tool without arguments to display the card configuration.

Continued on next page

3.3 DAG 3.7T Card Capture Sessions, continued

Procedure,continued

```
dag@endace:~$ dagthree -d dag0
links 0-7       noreset E1
links 8-15      noreset E1
link 0  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 1  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 2  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 3  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 4  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 5  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 6  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 7  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 8  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 9  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 10 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 11 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 12 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 13 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 14 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
link 15 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 E1
pci     33MHz 32-bit buf=32MB rxstreams=1 txstreams=1 mem=32:0
```

3.4 Configuration in WYSYCC Style

Description

Configuration in WYSYCC is the 'What You See You Can Change' style. Configuration includes E1 Mode, other options, and individual port settings.

Running the command `dagthree` alone shows the current configuration.

In this section

This section covers the following topics of information.

- Card E1/T1 Mode Configuration
- Configuring Card for Other Options
- Change Card Individual Port Settings

3.4.1 Card E1/T1 Mode Configuration

Description If the DAG 3.7T card is configured for E1 mode, and T1 is required, type:

```
dag@endace:~$ dagthree -d dag0 T1
links 0-7       noreset T1
links 8-15      noreset T1
link 0  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 1  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 2  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 3  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 4  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 5  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 6  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 7  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 8  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 9  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 10 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 11 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 12 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 13 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 14 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 15 mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
pci     33MHz 32-bit buf=32MB rxstreams=1 txstreams=1 mem=32:0
```

3.4.2 Configuring Card for Other Options

Description For other DAG 3.7T card configuration options, removing or adding the "no" prefix will change the setting:

```
dag@endace:~$ dagthree -d dag0 noeql
links 0-7       noreset T1
links 8-15      noreset T1
link 0  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 1  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 2  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 3  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 4  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 5  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 6  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 7  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 8  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 9  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 10 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 11 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 12 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 13 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 14 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 15 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
pci     33MHz 32-bit buf=32MB rxstreams=1 txstreams=1 mem=32:0
```

3.4.3 Change Card Individual Port Settings

Description To change an individual port settings for the DAG 3.7T card, the "link=<port>" option is used:

```
dag@endace:~$ dagthree -d dag0 link=0 term100 link=2 eql
links 0-7       noreset T1
links 8-15      noreset T1
link 0  mode=29 rxpkts txpkts nofcl noeql term100 b8zs/hdb3 T1 ESF
link 1  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 2  mode=29 rxpkts txpkts nofcl eql term120 b8zs/hdb3 T1 ESF
link 3  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 4  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 5  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 6  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 7  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 8  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 9  mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 10 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 11 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 12 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 13 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 14 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
link 15 mode=29 rxpkts txpkts nofcl noeql term120 b8zs/hdb3 T1 ESF
pci     33MHz 32-bit buf=32MB rxstreams=1 txstreams=1 mem=32:0
```

3.5 Configuration Options Supported

Description The following is the complete list of configuration options supported.

`link=<port>`	configure a specific line. <link> = {0 to 15}
`default`	set line(s) to default E1, mode 0
`e1`	set line(s) to E1
`e1_crc`	set line(s) to E1 CRC
`t1_esf`	set line(s) to T1 ESF
`t1_sf`	set line(s) to T1 D4 SF
`t1`	set line(s) to T1 ESF
`termext`	line is externally terminated (for monitoring)
`term75`	enable 75 ohm termination
`term100`	enable 100 ohm termination
`term120`	enable 120 ohm termination
`b8zs`	enable B8ZS/HDB3 zero code suppression
`Ami`	enable AMI (disable B8ZS/HDB3) (un)set HDB3 zero code suppression (E1)
`Clear`	clear framer status
`Reset`	reset framer
`mode=<mode>`	set EXAR line mode
`[no]fcl`	(un)set facility loopback
`[no]eql`	(un)set equipment Loopback
`mem=X:Y`	configure memory allocated to streams 0,1, …
`Rxonly`	assign all buffer memory to receive streams.
`Txonly`	assign all buffer memory to transmit streams.
`Rxtx`	assign buffer memory to transmit and receive streams.

Continued on next page

3.5 Configuration Options Supported, continued

Process

The following is the complete list of EXAR line modes.Mode is defined by the following parameters:

Process	Description
Physical line	E1 or T1
Signal attenuation	< -15dB (short haul), < -36dB (long haul) or < -43dB (extended long haul)
Transmit power	In T1 there a different recommendations for pulse depending of distance between equipment (transmitter -> receiver). Because a user is not usually transmitting traffic, there is little to be concerned about with Transmit LBO.
Cabling	There is two options in E1: • 75ohm unbalanced coaxial cable or, • 120ohm balanced twisted pair. In T1 only standard is 100ohm balanced twisted pair.
Coding	T1 coding is B8ZS which is called HDB3 in E1. This is not relevant because dagthree can set coding separately to ami or b8zs/hdb3.
Example	If there is a twisted pair E1 line to be monitored, the mode 29 can be used, which is the minimum gain control value (signal attenuated less than 15dB). If dagthree reports a poor signal, an increase to amplifying can be tried by setting mode to 25 (long haul/36dB) or even to 27 (extended long haul/43dB). A user should try to use lowest possible gain control value (eg. start with short haul) because if signal is amplified too much it easily creates signal errors.

Continued on next page

3.5 Configuration Options Supported, continued

Process, continued

Mode	Type	Transmit 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

3.6 Inspect Interface Statistics

Description

Once the card has been configured as expected, the interface statistics should be inspected to see if the interfaces are locked to the data stream.

```
dag@endace:~$ dagthree -d dag0 –si
if los ais lcv fls dmo clos - rx0 rx1 tx0 tx1 crc ais ferr up
 0   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 1   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 2   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 3   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 4   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 5   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 6   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 7   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 8   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
 9   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
10   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
11   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
12   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
13   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
14   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
15   0   0   0   0   0    0 -   1   1   1   1   0   0    0   1
```

Status bits display

The tool will display a number of status bits as they have occurred since the last time read. In our example, the interval is set to one second via the -i option.

`if`	Number (0-15)
`los`	LIU Loss of signal
`ais`	LIU Alarm Indication Signal
`lcv`	LIU Line code violation
`fls`	LIU FIFO limit status
`dmo`	LIU Drive monitor output
`clos`	LIU Cable loss (dB +/- 1dB)
`rx0`	Framer ever received 0
`rx1`	Framer ever received 1
`tx0`	Framer ever sent 0
`tx1`	Framer ever sent 1
`crc`	Framer CRC error
`ais`	Framer Alarm Indication Signal
`ferr`	Framer error
`up`	Framer up

3.7 Configuring HDLC Channels

Description

The `dagchan` tool is used to define the physical channels for the DAG 3.7T card. It reads a channel configuration file and then creates each of the channels defined in the file.

The default operation of `dagchan` is to delete all existing channel definitions on the board, and then add the new channel definitions.

An alternative mode is to use the '`-r`' option and preserve the channel definitions on the board and add the new channel definitions to them.

The `dagchan` tool can be used to configure the DAG 3.7T card if it is running HDLC firmware.

Configuration file

The configuration file is used to:

- Configure HDLC channel connections
- Enable/disable RAW Rx.

Each line in the file is a different command. They each start with a letter, followed by a sequence of 1 or more numbers. The following letters are valid line beginnings:

- **'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

In this section

This section covers the following topics of information.

3.7.1 Configuring DAG 3.7T Card for Receive and Transmit

Description Ensure the DAG 3.7T card has the appropriate transmit firmware image loaded.

The DAG 3.7T does not support HDLC transmit in this release.

3.7.2 Timeslot Connection ['c']

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

```
c bit_offset ifc_num ts_num
```

The `bit_offset` field is always `0`, it is reserved for future use.

In order to configure a connection on interface 5, timeslot 16 it would look like:

```
c    0    5   16

  0  5 16
```

3.7.3 Delete Connection ['d']

Description In order to delete a connection, the connection must already exist. This is used with `dagchan`. When `dagchan` is started there are no connections, so all connections to be created and deleted must exist in the `.cfg` file. This line would look like:

```
d conn_num
```

To delete connection number 17:

```
d 17
```

3.7.4 Hyper-channel Connection ['h']

Description A hyper-channel connection is a connection which occupies 1 or more timeslots on 1 interface. This line would look like:

```
h bit_offset ifc_num ts_num ts_num ts_num ...
```

The `bit_offset` field is always `0`. There has been no use for this at the present time. In order to configure a connection on interface 3, timeslots 0, 1, 2, 26, 27, 28, 29 it would look like:

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

3.7.5 Line Connection ['l']

Description

A line connection is a connection which occupies all timeslots, full line rate for example, on 1 interface, but is not in RAW mode. This line would look like this:

```
l bit_offset ifc_num
```

The `bit_offset` field is always `0`. There has been no use for this at the present time. In order to configure a line connection on interface `0` it would look like:

```
l    0    0
```

3.7.6 RAW Connection ['r']

Description

A RAW connection is a connection which occupies all timeslots on 1 interface and is in RAW mode. RAW mode is when no `de-framing`, or `un-bitstuffing` is done. Just the straight data from the timeslots read by the firmware. This line would look like:

```
r ifc_num
```

In order to configure a RAW connection on interface 4 it would look like:

```
r 4
```

3.7.7 Sub-channel Connection ['s']

Description

A sub-channel connection is an HDLC connection which occupies 1, 2, or 4 conectutive bits within a timeslot on one interface. The line would look like this:

```
s bit_offset ifc_num ts_num ts_mask
```

The `bit_offset` field is always 0. So far, we have no use for it. In order to configure a connection on interface 6, timeslot 2, bits 0-3 would look like this:

```
s   0   6   2   0   0   0   0   1   1   1   1
```

1 bit equals 8 kbps. For example, 1 timeslot at 64k consists of 8kbps in a sub-channel. An 8kbps time channel extends 2x8 to 16k, 4x8k to 32k.

1 Timeslot [64k]

```
8       8       8       8       8       8       8       8
```

Continued on next page

3.7.7 Sub-channel Connection ['s'], continued

Sample This is a sample `chan.cfg` file for 16 E1 sub-channels.

```
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
```

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

Raw line connections use connection ID 0 through 15. Each raw line connection captures full T1 or E1 frames for the same interface number as its connection ID.

The Endace ERF record format definitions for Type 5 Multi-channel HDLC Frame Record and Type 6 Multi-Channel Raw Link Data Record are detailed in Chapter 6 of this document.

3.8 Configuring HDLC RAW Connections

Description A typical HDLC data stream is bitstuffed with a 0 bit after every 5 consecutive 1s to prevent confusion with the HDLC packet delimiter (0x7e or 0111 1110).

Configuring for a RAW connection implies that no bit stuffing occurs.

The following section of information describes setting up channel configurations using `dagchan`.

In this section This section covers the following topics of information.

- Line RAW Connection ['lr']
- Channel RAW Connection ['cr']
- Hyper-channel RAW Connection ['hr']
- Sub-channel Raw Connection ['sr']

3.8.1 Line RAW Connection ['lr']

Description

A line RAW connection with HDLC firmware is a connection which occupies all timeslots on 1 interface. This line would look like this:
`l bit_offset ifc_num`

Captures using this configuration produce records with `erf type 8`.

The `bit_offset` field is always `0`. There has been no use for this at the present time. In order to configure a line connection on interface `0` it would look like:

```
l    0    0
```

3.8.2 Channel RAW Connection ['cr']

Description

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

```
c bit_offset ifc_num ts_num
```

Captures using this configuration produce records with `erf type 8`.

The `bit_offset` field is always `0`, it is reserved for future use.

In order to configure a connection on interface 5, timeslot 16 it would look like:

```
c    0    5   16
```

3.8.3 Hyper-channel RAW Connection ['hr']

Description

A hyper-channel RAW connection with HDLC firmware is a connection which occupies 1 or more timeslots on 1 interface. This line would look like:

```
h bit_offset ifc_num ts_num ts_num ts_num ...
```

Captures using this configuration produce records with `erf type 8`.

The `bit_offset` field is always `0`. There has been no use for this at the present time. In order to configure a connection on interface 3, timeslots 0, 1, 2, 26, 27, 28, 29 it would look like:

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

3.8.4 Sub-channel Raw Connection ['sr']

Description

A sub-channel RAW connection with HDLC firmware is an HDLC connection which occupies 1, 2, or 4 conectutive bits within a timeslot on one interface. This line would look like this:

```
s bit_offset ifc_num ts_num ts_mask
```

Captures using this configuration produce records with `erf type 8`.

The `bit_offset` field is always 0. So far, we have no use for it. In order to configure a connection on interface 6, timeslot 2, bits 0-3 would look like this:

```
s   0   6   2   0   0   0   0   1   1   1   1
```

3.9 Configuring ATM Channels

Description

The `dagchan` tool is used to define the physical channels for the DAG 3.7T card. It reads a channel configuration file and then creates each of the channels defined in the file.

The default operation of `dagchan` is to delete all existing channel definitions on the board, and then add the new channel definitions.

An alternative mode is to use the '`-r`' option and preserve the channel definitions on the board and add the new channel definitions to them.

The `dagchan` tool can be used to configure the DAG 3.7T card if it is running ATM firmware.

Configuration file

The configuration file is used to configure connections, each line in the file is a different command. Each line starts with a letter, followed by a sequence of 1 or more numbers. The following letters are valid line beginnings:

- '**c**' = simple 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

Continued on next page

3.9 Configuring ATM Channels, continued

In this section This section covers the following topics of information.

- Configuring DAG 3.7T Card for Receive and Transmit
- Timeslot Connection ['c']
- Delete Connection ['d']
- Hyper-channel Connection ['h']
- Line Connection ['l']
- Sub-channel Connection ['s']

3.9.1 Configuring DAG 3.7T Card for Receive and Transmit

Description Ensure the DAG 3.7T card has the appropriate transmit firmware image loaded.

3.9.2 Timeslot Connection ['c']

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

```
c bit_offset ifc_num ts_num
```

The `bit_offset` field is always `0`, it is reserved for future use.

In order to configure a connection on interface 5, timeslot 16 it would look like:

```
c    0    5   16

  0  5 16
```

3.9.3 Delete Connection ['d']

Description In order to delete a connection, the connection must already exist. This is used with `dagchan`. When `dagchan` is started there are no connections, so all connections to be created and deleted must exist in the `.cfg` file. This line would look like:

```
d conn_num
```

To delete connection number 17:

```
d 17
```

3.9.4 Hyper-channel Connection ['h']

Description

A hyper-channel connection is a connection which occupies 1 or more timeslots on 1 interface. This line would look like:

```
h bit_offset ifc_num ts_num ts_num ts_num ...
```

The `bit_offset` field is always `0`. There has been no use for this at the present time. In order to configure a connection on interface 3, timeslots 0, 1, 2, 26, 27, 28, 29 it would look like:

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

3.9.5 Line Connection ['l']

Description

A line connection is a connection which occupies all timeslots, for example full line rate, on 1 interface, but is not in RAW mode. This line would look like this:

```
l bit_offset ifc_num
```

The `bit_offset` field is always `0`. There has been no use for this at the present time. In order to configure a line connection on interface `0` it would look like:

```
l   0   0
```

3.9.6 Sub-channel Connection ['s']

Description

A sub-channel connection is an ATM connection which occupies 1, 2, or 4 conectutive bits within a timeslot on one interface. The line would look like this:

```
s bit_offset ifc_num ts_num ts_mask
```

The `bit_offset` field is always 0. So far, we have no use for it. In order to configure a connection on interface 6, timeslot 2, bits 0-3 would look like this:

```
s   0   6   2   0   0   0   0   1   1   1   1
```

Continued on next page

3.9.6 Sub-channel Connection ['s'], continued

Sample This is a sample `chan.cfg` file for 16 E1 sub-channels.

```
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
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 32
```

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

Raw line connections use connection ID 0 through 15. Each raw line connection captures full T1 or E1 frames for the same interface number as its connection ID.

The Endace ERF record format definitions for Type 5 Multi-channel HDLC Frame Record and Type 6 Multi-Channel Raw Link Data Record are detailed in Chapter 6 of this document.

3.10 Reporting Problems

Description

If there are unresolved problems with a DAG card or supplied software, contact Endace Technical Support via the email address support@endace.com. Supplying sufficient information in an email enables effective response.

Problem checklist

The exact information available to users for trouble, cause and correction analysis may be limited by nature of the problem. The following items assist a quick problem resolution:

Ref	Item
1.	DAG card[s] model and serial number.
2.	Host PC type and configuration.
3.	Host PC operating system version.
4.	DAG software version package in use.
5.	Any compiler errors or warnings when building DAG driver or tools.
6.	For Linux and FreeBSD, messages can be generated when the DAG device driver is loaded. These can be collected from command `dmesg` or from log file `/var/log/syslog`.
7.	Output of `daginf -v`.
8.	Firmware versions from `dagrom -x`.
9.	Physical layer status reported by: `dagthree`
10.	Network link statistics reported by: `dagthree -si`
11.	Network link configuration from the router where available.
12.	Contents of any scripts in use.
13.	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.
14.	A small section of a captured packet trace illustrating the problem.

4.0 RUNNING DATA CAPTURE SOFTWARE

Introduction For a typical measurement session, the `scripts/dag37tstart` script is edited and used to operate the cards directly.

In this chapter This chapter covers the following sections of information.

- Set Capture Session
- High Load Performance

4.1 Set Capture Session

Description The various tools used for data capture are in the `tools` sub-directory. For a typical measurement session, first move to the `dag` directory, load the driver, then load the appropriate FPGA image to each DAG card. For example, for HDLC capture with one DAG 3.7T card installed:

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

The integrity of the card's physical layer is then set and checked.

Process Follow this process to set a data capture session.

Process	Description
Set and check integrity of card physical layer.	For example: `dagthree –d dag0 default E1 mode=29 term120`
Configure channels.	Configure the DAG 3.7T channels using dagchan: `dagchan –c chan.cfg`
Start capture session.	Start a capture session as follows: tools/dagsnap –v –o tracefile The option -v is used to provide user information during capture; you may want to omit it for automated trace runs. If the –o tracefile parameter is not specified the tool will write to stdout, which can be used to pipeline dagsnap with other tools from the dagtools package.

Continued on next page

4.1 Set Capture Session, continued

Process,continued

Process	Description
Stopping	By default `dagsnap` runs forever. `dagsnap` can be stopped with a signal: `killall dagsnap` `dagsnap` can also be configured to run for a fixed number of seconds and then exit using the `-s` flag.
Checking statistics and performance.	For simple statistics and performance checks, try `dagbits`: `dagbits –d dag0 –c hdlc flags` `dagbits` will report the current HDLC frame receive statistics ever second, and it ouputs a per-channel summary when exited (e.g. via <ctrl-C>).

4.2 High Load Performance

Description

As the DAG 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

In order 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, otherwise the buffer eventually fills, and packet records are lost.

For Linux and FreeBSD, when the PC buffer fills, the message:

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

is displayed on the PC screen, and printed to log `/var/log/messages`.

The "Data capture" LED also goes out. This may be visibly indicated as flashing or flickering.

Detecting packet losses

Until some data is read out of the buffer to free some space, any arriving packets subsequently are discarded by the DAG card.

Any loss is detected in-band by observing the Loss Counter `lctr` field of the Extensible Record Format [ERF]. The Endace ERF is detailed in Chapter 6 of this document.

Continued on next page

4.2 High Load Performance, continued

Increasing buffer size

The host PC buffer can be increased to deal 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. Capture at OC-12/STM-4 (622Mbps) rates and above may require a larger buffer.

128MB or more is suggested for Linux/FreeBSD.

For the DAG 3.7T card Windows operating system the upper limit is 32MB.

In Debian Linux the amount of memory reserved is changed by editing file `/etc/modules`:

```
# For DAG 3.x, default 32MB/card
dagmem
#
# For DAG 4.x or 6.x, use more memory per card, E.G.
# dagmem dsize=128m
```

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

The value of `dsize` multiplied by the number of DAG cards must be less than the amount of physical memory installed, and less than 890MB.

5.0 SYNCHRONIZING CLOCK TIME

Description

The Endace DAG range of products come with sophisticated time synchronization capabilities, in order to provide high quality timestamps, optionally synchronized to an external time standard.

The system that provides the DAG synchronization capability is known as the DAG Universal Clock Kit (DUCK).

An independent clock in each DAG card runs from the PC clock. A 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 configured to avoid time variance between sets of DAG cards or between DAG cards and coordinated universal time [UTC].

Accurate time reference can be obtained from an external clock by connecting to the DAG card using the synchronization connector, or the host PCs clock can be used in software.

Common synchronization

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

Common synchronization sources include GPS or CDMA (Cellular telephone) time receivers.

Endace produces the TDS 2 Time Distribution Server modules and the TDS 6 units that enable multiple DAG cards to be connected to a single GPS or CDMA unit.

More information is on the Endace website, http://www.endace.com/accessories.htm, or the TDS 2/TDS 6 Units Installation Manual.

In this chapter

This chapter covers the following sections of information.

- Configuration Tool Usage
- Time Synchronization Configurations
- Synchronization Connector Pin-outs

5.1 Configuration Tool Usage

Description

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

Synchronization settings are controlled by the `dagclock` utility.

Example

```
dag@endace:~$ dagclock -h
Usage: dagclock [-hv] [-d dag] [option]

       -h  this page
       -v  increase verbosity
       -d  DAG device to use
  Option:
        default          RS422 in, none out
        none             None in, none out
        rs422in          RS422 input
        hostin           Host input (unused)
        overin           Internal input (synchronize 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
```

By default, all DAG cards listen for synchronization signals on their RS-422 port, and do not output any signal to their RS-422 port.

```
dag@endace:~$ dagclock –d dag0
muxin   rs422
muxout
```

5.2 Time Synchronization Configurations

Description

The DUCK is very flexible, and can be used in several ways, with or without an external time reference source.

The use includes a single card with no reference, two cards with no reference, and a card with reference.

In this section

This section covers the following topics of information.

- Single Card no Reference Time Synchronization
- Two Cards no Reference Time Synchronization
- Card with Reference Time Synchronization

5.2.1 Single Card no Reference Time Synchronization

Description

When a single card is used with no external reference, the card can be synchronized to the host PC's clock.

The clock in most PC's is not very accurate by itself, but the DUCK drifts smoothly at the same rate as the PC clock.

If a PC is running NTP to synchronize it's own clock, then the DUCK clock is less smooth because the PC clock is adjusted in small jumps. However, overall the DUCK clock does not drift away from UTC.

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

```
dag@endace:~$ dagclock –d dag0 none overin
muxin   overin
muxout
```

NOTE: `dagclock` should be run only after appropriate Xilinx images have been loaded. If the Xilinx images must be reloaded, the `dagclock` command must be rerun afterwards to restore the configuration.

5.2.2 Two Cards no Reference Time Synchronization

Description

When two DAG cards are used in a single host PC with no reference clock, the cards are to be synchronized in some way if timestamps between the two cards are to be compared. For example, if two cards monitor different directions of a single full-duplex link.

Synchronization between two DAG cards is achieved in two ways. One card can be a clock master for the second, or one can synchronize to the host and also act as a master for the second.

Synchronizing cards

If both cards are to be accurately synchronized, but not so for absolute time of packet time-stamps being correct, then one card is configured as the clock master for the other.

Locking cards together

Although the master card's clock will drift against UTC, the cards are locked together.

The cards are locked together by connecting the synchronization connector ports of both cards with a standard RJ-45 Ethernet cross-over cable.

Configure one of the cards as the master, the other defaults to being a slave.

Continued on next page

5.2.2 Two Cards no Reference Time Synchronization, continued

Preventing time-stamps drift

To prevent DAG card time-stamps drifting against UTC, one card is synchronized to the host PC's clock which in turn utilises NTP. This provides a master signal to the second card.

In this case, connect synchronization connectors with a standard RJ-45 Ethernet cross-over cable.

Configure one card to synchronize to the PC clock, and output a RS-422 synchronization signal to the second card.

```
dag@endace:~$ dagclock –d dag0 none overin overout
muxin   overin
muxout  overout
```

5.2.3 Card with Reference Time Synchronization

Description

The best timestamp accuracy occurs when DAG card is connected to an external clock reference, such as a GPS or CDMA time receiver.

Pulse signal from external sources

The DAG synchronization connector accepts a RS-422 Pulse Per Second [PPS] signal from external sources.

This is derived directly from a reference source, or distributed through the Endace TDS 2 [Time Distribution Server] module which allows a number of DAG cards to use a single receiver. The process is facilitated using TDS 6 expansion units coupled like a daisy-chain to a TDS 2 module.

Using external reference source

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.

Time accuracy is achieved using the host PC NTP. The time reference source is connected to synchronization connector, the card automatically synchronizes to a valid signal.

Connecting time distribution server

The TDS 2 module connects to any DAG card with a standard RJ-45 Ethernet cable and can be placed some distance from a DAG card and existing RJ-45 cabling infrastructure.

CAUTION: Never connect DAG and/or the TDS 2 module to active Ethernet equipment.

Continued on next page

5.2.3 Card with Reference Time Synchronization, continued

Testing signal

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

The `dagpps` tool is used to test a signal is being received correctly and is of correct polarity. To perform the test, run:

```
dagpps –d dag0.
```

The tool measures input state many times over several seconds, displaying polarity and length of input pulse.

Some DAG cards have LED indicators for synchronization (PPS) signals.

5.3 Synchronization Connector Pin-outs

Description

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 are:

1.	Out A+
2.	Out A-
3.	In A+
4.	In B+
5.	In B-
6.	In A-
7.	Out B+
8.	Out B-

Figure

Figure 6-1 shows the RJ45 plug and socket connector pin-outs.

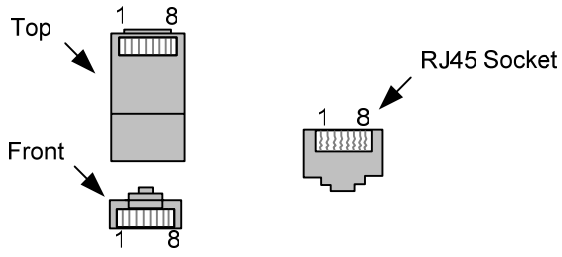


Figure 6-1. RJ45 Plug and Socket Connector Pin-outs.

Continued on next page

5.3 Synchronization Connector Pin-outs, continued

Out-pin connections

Normally the GPS input should be connected to the A channel input, pins 3 and 6. The DAG card can also output a synchronization pulse; used when synchronizing two DAG cards without a GPS input. Synchronization output is generated on the Out A channel, pins 1 and 2.

Ethernet crossover cable

A standard Ethernet crossover cable can be used to connect the two cards.

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-

Support

For cables and further advice on using GPS and CDMA time receivers email support@endace.com.

6.0 DATA FORMATS OVERVIEW

In this chapter This chapter covers the following sections of information.

- Data Formats
- Timestamps

6.1 Data Formats

Description The DAG card uses the ERF Type 5 Multi-channel HDLC Frame Record, ERF Type 6 Multi-Channel Raw Link Data Record, and ERF Type 7 Multi-channel ATM Cell Record. 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.

In this section This section covers the following topics of information.

- Generic Variable Length Record
- Type 5 Multi-channel HDLC Frame Record
- Type 6 Multi-channel RAW Link Data Record
- Type 7 Multi-channel ATM Cell Record

6.1.1 Generic Variable Length Record

Description Endace DAG Network Measurement Interface Cards [NMICs] produce trace files in their own native format, the Extensible Record Format [ERF].

The ERF format consists of a series of records. Each record describes one packet. An ERF format file consists only of ERF records; there is no special file header. This allows file concatenation and splitting to be performed arbitrarily on ERF record boundaries.

Table Table 6-1 shows the generic variable length record.

<table>
<tr><td colspan="3">timestamp</td></tr>
<tr><td colspan="3">timestamp</td></tr>
<tr><td>type</td><td>flags</td><td>rlen</td></tr>
<tr><td colspan="2">Lctr</td><td>wlen</td></tr>
<tr><td colspan="3">(rlen - 16) bytes of record</td></tr>
</table>

Table 6-1. Generic Variable Length Record.

Continued on next page

6.1.1 Generic Variable Length Record, continued

Data format An overview of the data format used is described in the following table.

Data Format	Description
type:	This field contains an enumeration of the frame subtype. If the type is zero, then this is a legacy format. 0: TYPE_LEGACY 1: TYPE_HDLC_POS: PoS w/HDLC framing 2: TYPE_ETH: Ethernet 3: TYPE_ATM: ATM Cell 4: TYPE_AAL5: reassembled AAL5 frame 5: TYPE_MC_HDLC: Multi-channel HDLC frame 6: TYPE_MC_RAW: Multi-channel Raw link data 7: TYPE_MC_ATM: Multi-channel ATM Cell
flags:	This byte is divided into 2 parts, the interface identifier, and the capture offset. 1-0: capture interface 0-3 2: varying record lengths present 3: truncated record [insufficient buffer space] 4: rx error [link error] 5: 5: ds error [internal error] 7-6: reserved
Rlen: record length	Total length of the record transferred over PCI bus to storage.
Lctr: *loss counter*	A 16 bit counter, recording the number of packets lost since the previous record. Records can be lost between the DAG card and memory hole due to overloading on PCI bus. The counter starts at zero, and sticks at 0xffff.
Wlen: *wire length*	Packet length including some protocol overhead. The exact interpretation of this quantity depends on physical medium.

6.1.2 Type 5 Multi-channel HDLC Frame Record

Description

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

Fixed length mode is not supported.

RX error is set if any MC header Error bit is set.

Table

Table 6-2 shows the Type 5 Multi-channel HDLC frame record.

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			

Table 6-2. Type 5 Multi-channel HDLC Frame Record.

MC Header attributes

The Type 5 Multi-channel HDLC Frame Record MC header is divided into several bit fields.

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

6.1.3 Type 6 Multi-channel RAW Link Data Record

Description

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

Fixed length mode is not supported.

RX error is set if any MC header Error bit is set.

Table

Table 6-3 shows the Type 6 Multi-channel RAW Link Data record.

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

Table 6-3. Type 6 Multi-Channel RAW Link Data Record.

MC Header Attributes

The Type 6 Multi-channel RAW Link Data Record MC header is divided into several bit fields.

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

6.1.4 Type 7 Multi-channel ATM Cell Record

Description

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

Fixed length mode is not supported.

RX error is set if any MC header Error bit is set.

Table

Table 6-4 shows the Type 7 Multi-channel ATM Cell record.

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

Table 6-4. Type 7 Multi-Channel ATM Cell Record.

MC Header attributes

The Type 7 Multi-channel ATM MC header is divided into several bit fields.

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.

6.1.5 Type 8 Multi-channel RAW Link Data Record

Description

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

Fixed length mode is not supported.

RX error is set if any MC header Error bit is set.

Table

Table 6-5 shows the Type 8 Multi-channel RAW Link Data record.

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

Table 6-5. Type 5 Multi-channel RAW Link Data Record.

MC Header Attributes

The Type 8 Multi-channel RAW Link Data Record MC header is divided into several bit fields:

Bits	Attribute
0-9	Connection number [0-511].
10-28	Reserved.
29	Lost Byte Error
30	1st Rec. This is the first record received since this connection was configured.
31	Reserved.

6.2 Timestamps

Description

The ERF format incorporates a hardware generated timestamp of the packet's arrival.

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

Another advantage of the ERF timestamp format is that a difference between two timestamps can be found with a single 64-bit subtraction. It is not necessary to check for overflows between the two halves of the structure as is needed when comparing Unix time structures, which are also available to Windows users from the Winsock library.

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 codes

Here is some 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;
 }
```