DAG 7.1S Card User Guide

EDM01-17

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 their 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 Technology 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 Technology Limited has taken great effort to verify the accuracy of this manual, but nothing herein should be construed as a warranty and Endace shall not be liable for technical or editorial errors or omissions contained herein.

In accordance with the Endace Technology Limited policy of continuing development, the information contained herein is subject to change without notice.

Website

http://www.endace.com

Copyright 2005 - 2008 Endace Technology Ltd. All rights reserved.

Use of the Endace products described in this document is subject to the Endace Terms of Trade and the Endace End User License Agreement (EULA).

Contents

Overview

The Endace DAG 7.1S card provides the means to transfer data at the full speed of the network into the memory of the host computer, with zero packet loss in even worst-case conditions. Further, unlike a Network Interface Card (NIC), Endace products actively manage the movement of network data into memory while only consuming a minimal amount of the host computer'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 DAG 7.1S card is a four port, PCIe card that allows capture and transmission of data. It is a designed to provide high efficiency monitoring and transmission of ATM, POS or Bit HDLC traffic with precision timestamping capability.

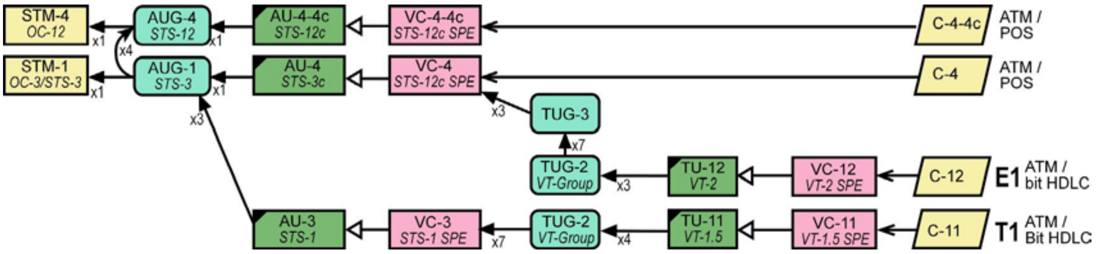
It supports the following:

- Concatenated POS/ATM receive and transmit over 4 x OC-3c/STM-1c or 4 x OC-12c/STM-4c.
- Channelized Bit HDLC/ATM receive and transmit over 4 x OC-3/STM-1 or 2 x OC-12/STM-4.

Card Features

The following features are available on this DAG card. **Note:** Different firmware images may be required. Not all features are available on each firmware image. For further information on which feature is available in what firmware image, see Firmware images.

- Network Processor AAL2 and AAL5
- Network Processor IPv4 / v6 Packet filtering
- Network Processor VCI / VPI / CID filtering
- The different encapsulations are described in the following diagram:



Purpose of this User Guide

The purpose of this User Guide is to provide you with an understanding of the DAG 7.1S card architecture, 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
- Synchronizing clock time
- Data formats

You can also find additional information relating to functions and features of the DAG 7.1S card in the following documents which are available from the Support section of the Endace website at http://www.endace.com:

- *EDM04-03 dagflood User Manual*
- *EDM04-08 Configuration and Status API Programming Guide*
- *EDM04-13 SAR API Programming Guide*
- *EDM04-11 IXP Filtering API*
- *EDM04-14 DAG IXP Filter Loader User Guide*
- *EDM04-19 DAG Programming Guide*
- *EDM04-13 SAR API Programming Guide*

This User Guide and the *EDM04-01 DAG Software Installation Guide* are also available in PDF format on the installation CD shipped with your DAG 7.1S card.

System Requirements

General

The minimum system requirements for the DAG 7.1S card are:

- A computer, with at least a Intel Xeon 1.8GHz or faster and a minimum of 1GB RAM.
- At least one free PCI-Express slot supporting at least one lane.
- Software distribution requires 60MB free space.
- For details of the supported operating systems, see one of the following documents:
  - *EDM04-01 DAG Software Installation Guide*
  - Current release notes - See the Documentation CD or the Endace support website at https://www.endace.com/support.
- Requires 12V 1A external power via 4.2mm Pitch Mini-Fit Jr , 6 way, (PCI Express Card Power Connector).

Operating System

This document assumes you are installing the DAG 7.1S card in a computer which already has an operating system installed. To install refer to *EDM04-01 DAG Software Installation Guide*. All related documentation is included on the CD shipped with the DAG 7.1S card.

Other Systems

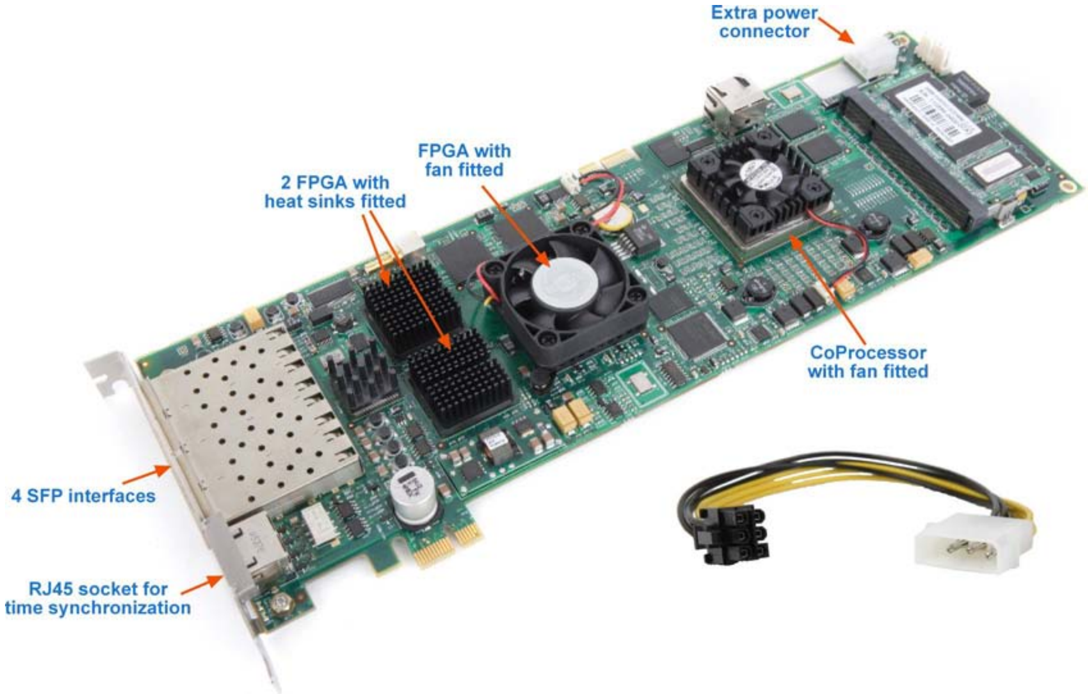
For advice on using an operating system that is substantially different from any of those specified above, please contact Endace Customer Support at support@endace.com.

Card Description

The DAG 7.1S SDH/SONET Network Monitoring Card provides either four STM-1 (OC3) or two STM-2 (OC12) interfaces supporting concatenated or channelized ATM or Packet over SONET (PoS) networks.

The DAG 7.1S has four optical transceivers which can be operated simultaneously.

The DAG 7.1S Card has an additional power supply socket. This must be connected to a 12 V power supply using the supplied cable.


Extra power connector
FPGA with fan fitted
2 FPGA with heat sinks fitted
CoProcessor with fan fitted
4 SFP interfaces
RJ45 socket for time synchronization

The key features of the card are:

- Four interfaces allow full line rate capture and processing for 4 x STM-1/OC-3 or 2 x STM-2/OC-12.
- Fully programmable Intel IXP Network Processor
- PCI Express bus interface.
- 1244Mps raw transmit and receive bandwidth.
- Combined FPGA and network processor architecture.
- Channelized and concatenated support.
- ATM AAL2 and AAL5 segmentation and reassembly.
- PoS IP filtering.

Battery removal – don’t do it!

Removing the battery from a DAG card voids your warranty.

Removing the battery from a DAG card will cause the loss of encryption key used to decode the DAG card's firmware. Once the encryption key is lost the DAG card must be returned to Endace for reprogramming.

The battery in this product is expected to last a minimum of 10 years.

Caution

Risk of explosion if the battery is replaced by an incorrect type.
Dispose of used batteries carefully.

Card Architecture

Serial SONET/SDH optical data is received by four optical interfaces, and passed through deserializers.

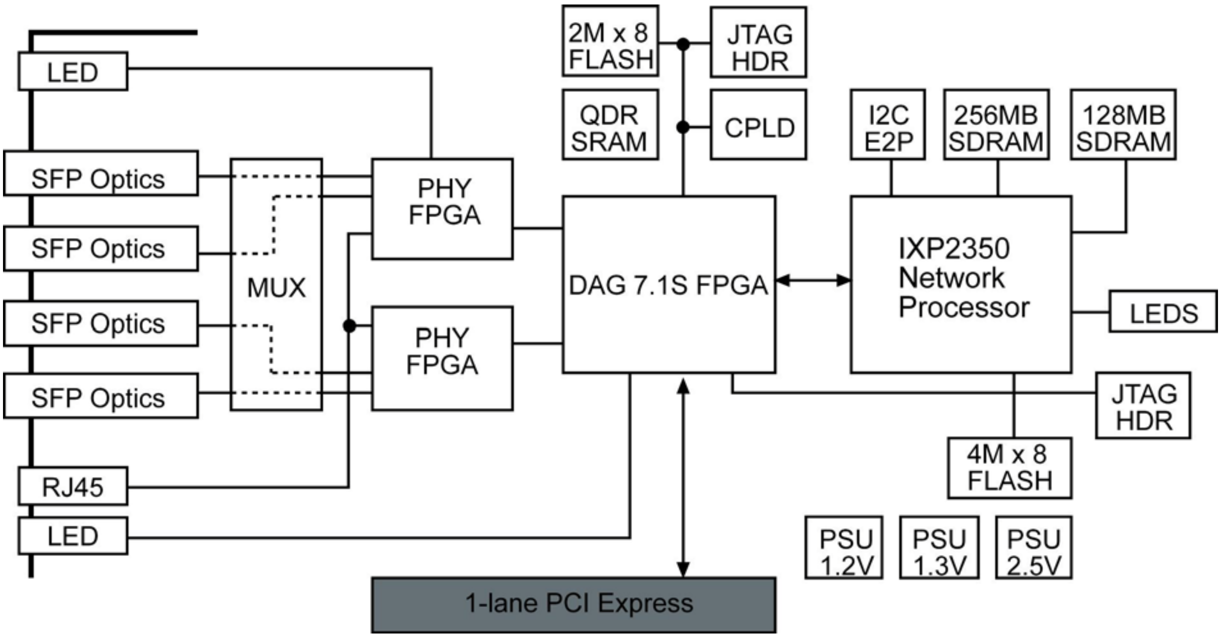
The network data feeds immediately into two physical layer FPGAs. The SONET/SDH payload data is then sent to the main FPGA.

The FPGA contains the packet record processor, PCI Express interface logic and the DAG Universal Clock Kit (DUCK) timestamp engine. The DUCK provides high resolution per packet timestamps which can be accurately synchronized..

Note: For further information on the DUCK and time synchronizing, see Synchronizing Clock Time (page 61) later in this User Guide.

An Intel IXP network processor is logically located next to the main FPGA. The main FPGA can route packets to either the IXP network processor for additional processing before routing onto the host or directly to the host via the PCI-Express port.

The following diagram shows the card’s major components and the flow of data.



Line Types

It is important that you understand the physical characteristics of the network to which you want to connect. If your configuration settings do not match your network, the DAG 7.1S card will not function as expected.

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

Supported Line Types

The line characteristics supported by the DAG 7.1S card are described below.

Type	Description
PoS/ATM	Packet over SONET/Asynchronous Transfer Mode.
OC-3/OC-3c	A SONET network line with transmission speeds up to 155.52 Mbit/s using fiber optics. Also called STM-1 (SDH).
OC-12/OC-12c	A SONET network line with transmission speeds up to 622.08 Mbit/s using fiber optics. Also called STM-4 (SDH).
STM-1/STM-1c	An SDH line equivalent to OC-3 (SONET).
STM-4/STM-4c	An SDH line equivalent to OC-12 (SONET).

Extended Functions

In addition to standard packet capture the DAG 7.1S card also provides TCP/IP Filtering and Classification and ATM Segmentation and Reassembly.

TCP/IP Filtering and Classification

This feature allows you to classify packets into arbitrary categories which then drop, retransmit or capture a packet to the host based upon the result. You can also change filter rules "on the fly" without any loss of data.

The specifications for the IP filtering/packet classification are:

- Packets are classified and filtered by IP header (both IPv4 and IPv6) and/or UDP/TCP/SCTP port number.
- Up to 1024 IP header classification rules.
- Up 254 UDP/TCP/SCTP port or ICMP type rules can set per IP header classification.
- Classification rules are assigned a user-defined 14-bit identifier.
- Packets matching classification rules are assigned the matching rule's identifier.
- Programmable actions may be associated with each rule identifier. For example the packet should either be dropped, or presented to the host.
- Packets presented to the host include the rule-match identifier in the record header.

AAL2/AAL5 Reassembly

This feature allows you to eliminate the significant CPU load associated with AAL2/AAL5 reassembly on a busy ATM link by offloading this process to the DAG 7.1S card. It also provides the ability to reduce volume of captured data to only what is required by filtering on VPI/VCI pairs.

The Reassembler specifications are:

- Supports up to 8160 simultaneously active VCI/VPI/CIDs.
- Supports simultaneous reassembly of AAL2 and AAL5 frames up to 8kB long.
- VPI/VCI scanning.
- Supports up to full STM-4/OC-12 cell rate on two interfaces simultaneously (approx 2.8 million cells/sec), or four full STM-1/OC-3 interfaces for AAL 5 reassembly.
- Supports 2 x STM-1/OC-3 cell rate on combined four interfaces (approx 0.8 million cells/sec) for AAL2 reassembly.
- Optional ATM cell filtering prior to reassembly.

Introduction

A DAG 7.1S card can be installed in any free PCIe slot.

The DAG 7.1S card operates on a single lane PCIe, this interface is capable of providing a maximum throughput of 1.8Gigabits per second for both receive and transmit.

You can run multiple DAG 7.1S cards on one bus. By default, the DAG driver supports up to four DAG cards in one system.

DAG Software package

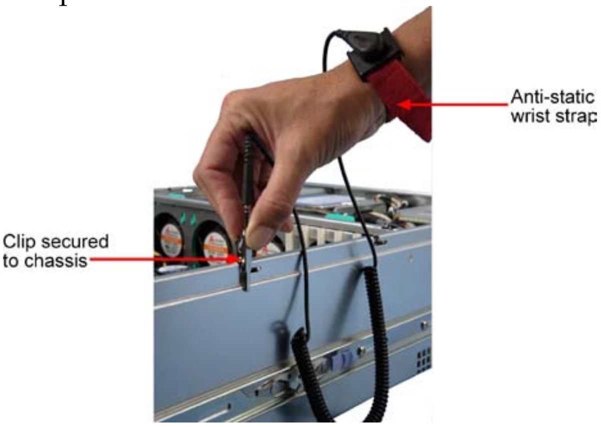
The latest DAG Software package must be installed before you install the DAG 7.1S card itself. See *EDM04-01 DAG Software Installation Guide*, which is included on the CD shipped with the DAG 7.1S card.

Inserting the DAG Card

Caution:

It is very important to protect both the computer and the DAG 7.1S card from damage by electro-static discharge (ESD). Failure to do so could cause damage to components and subsequently cause the card to partially or completely fail.

1. Turn power to the computer OFF.
2. Remove the PCIe bus slot screw and cover.
3. Using an approved ESD protection device attach the end with the strap to your wrist and pull or clip firmly so there is firm contact with your wrist.
4. Securely attach the clip on the other end of the strap to a solid metal area on the computer chassis as shown below.



5. Insert the DAG 7.1S card into PCIe bus slot ensuring it is firmly seated.
6. If this DAG card requires an external power supply, complete the following steps:
   a. Connect the supplied (or equivalent) power cable to the external power connector on the DAG card.
   b. Connect the cable to the appropriate power connector on your server's power supply unit.
7. Check the free end of the card fits securely into the card-end bracket that supports the weight of the card.
8. Secure the card with the bus slot cover screw.
9. Turn power to the computer ON.
10. Ensure the blue (FPGA successfully programmed) LED on the DAG card illuminates.

Port Connectors

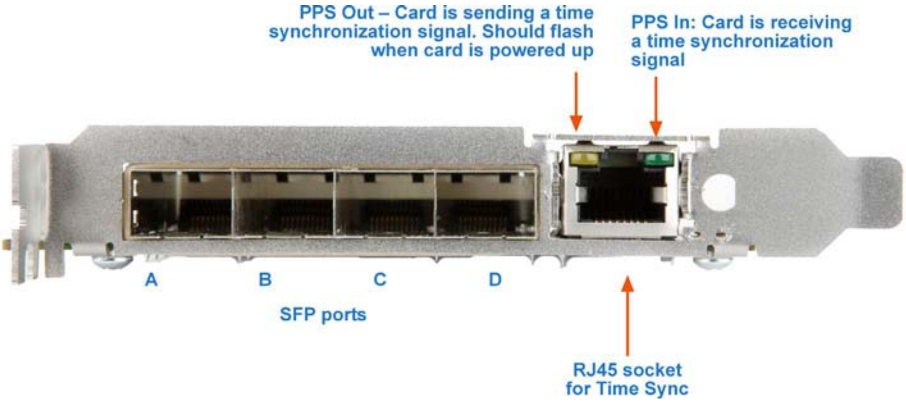
The DAG 7.1S has 4 SFP socket connectors. Each connector consists of an optical fiber transmitter and receiver.

The upper connector of each pair is used for the transmit signal. These can be connected to daisy-chain systems if you have facility loopback (fcl) set on the card. You can also connect them if you are using a data generation program.

The bottom connector of each pair is used for the received signal.

There is an 8-pin RJ-45 socket located below the optical port connectors on the car bracket. This is available for connection to an external time synchronization source.

Caution: Never connect an Ethernet network or telephone line to the RJ-45 sockets.


PPS Out – Card is sending a time synchronization signal. Should flash when card is powered up
PPS In: Card is receiving a time synchronization signal
A
B
C
D
SFP ports
RJ45 socket for Time Sync

External power supply

The DAG 7.1S PCB has an additional power supply socket. This must be connected to a 12 V power supply using the supplied cable.

Caution:

Prolonged operation of the DAG 7.1S card without this extra power supply connected could damage parts of the card.

The current draw through this extra power connector is up to 1A.

The external power supply connector is a 4.2mm Pitch Mini-Fit Jr , 6 way, (PCI Express Card Power Connector).

Some modern PSU units have connectors on the power supply designed to fit directly to this. Otherwise use the Endace supplied adapter cable to connects this input to a standard 5.25" Drive power connector on the system power supply unit.

Pluggable Optical Transceivers

Overview

The DAG 7.1S card uses industry standard Small Form-factor Pluggable (SFP) optical transceivers.

The transceiver consists of two parts:

- Mechanical chassis attached to the circuit board
- Transceiver unit which may be inserted into the chassis

Note: You must select the correct transceiver type to match the optical parameters of the network to which you want t connect. Configuring the card with the wrong transceiver type may damage the card.

You can connect the transceiver to the network via LC-style optical connectors.

For further information on Pluggable Optical Transceiver, please refer to the Endace website at http://www.endace.com.

Setting Power

The optical power range depends on the particular SFP module that is fitted to the DAG card.

Optical power is measured in dBm. This is decibels relative to 1 mW where 10 dB is equivalent to a factor of 10 in power. The optical power is always a negative value, indicating power that is less than 1 mW. The most sensitive devices can work at power levels down as low as –30dBm or 1µW.

The DAG 7.1S card optical power module configuration for Multi Mode Fibre (MMF) and Single Mode Fiber (SMF) is shown below:

Part #	Fibre	Data Rate	Max Pwr	Min Pwr	Nom Pwr	Mode
FTRJ1322	SMF	622	-8dBm	-28dBm	-	OC-12 Single Mode
FTRJ1323	SMF	155	-8dBm	-28dBm	-	OC-3 Single Mode
TRPD12MM3EAS	MMF	622	-	-	-	OC-12 Multi Mode
TRPD03MM3EAS	MMF	155	-	-	-	OC-3 Multi Mode

Caution:

Ensure that you insert **only** OC-3/OC-12 modules. **Do not** use OC-48 or GiG-E modules as these will destroy the PHY FPGA

The card supports 1310 nanometer singlemode and multimode fibre attachments with optical signal strength between 0 dBm and -22 dBm.

If there is doubt, check card receiver ports light levels are correct using an optical power meter.

The card receiver ports are the lower of each dual-LC-style connectors, the closest to the PCI-Express slot.

Cover card transmit ports with LC-style plugs to prevent dust and mechanical hazards damaging optics if not in use.

Note: If you remove the optics modules for any reason they will not automatically power on when re-inserted. You will need to turn them on using the `sfppwr` option when you configure the card channelized or concatenated operation.

Power Input

Note: The optical power input to the DAG card must be within the receiver's dynamic range. See the previous table for details. If it is slightly outside of this range it will cause an increased bit error rate. If it is significantly outside of this range the system will not be able to lock onto the signal.

When power is above the upper limit the optical receiver saturates and fails to function. When power is below the lower limit the bit error rate increases until the device is unable to obtain lock and fails. In extreme cases, excess power can damage the receiver.

When you set up the DAG card you should measure the optical power at the receiver and ensure that it is within the specified power range. If it is not, adjust the input power as follows:

- Insert an optical attenuator if power is too high, or
- Change the splitter ratio if power is too high or too low.

Splitter Losses

Splitters have the insertion losses either marked on their packaging or described in their accompanying documentation. General guidelines are:

- A 50:50 splitter will have an insertion loss of between 3 dB and 4 dB on each output
- 90:10 splitter will have losses of about 10 dB in the high loss output, and <2 dB in the low loss output

Note: Endace recommends that you do not use a combination of single mode and multi mode fibers and optics modules on the same link, as the quality of the received signal cannot be guaranteed.

If you have no choice but to mix single mode and multi mode you should be aware that a single mode input connected to a multi mode fiber will have some attenuation but may still be acceptable. However a multi mode input connected to a single mode fiber will likely have large and unpredictable losses.

Concatenated Configuration

Introduction

Configuring the DAG 7.1S card ready for capturing data requires the following steps:

- Setting up the FPGA (page 12)
- Preparing the DAG card for use (page 14)
- Display Current Configuration (page 15)
- Interface Statistics (page 20)
- Verify Configuration (page 22)

Once the DAG 7.1S is configured you can start capturing data, see Using your DAG card to capture data (page 47) for details on capturing data.

Before configuring the DAG card

Before configuring the FPGA, you should ensure that:

- `dagmem` has been run and memory allocated to each installed DAG card.
- `dagload` has been run so that all DAG drivers have been installed.

Refer to the *Installing the drivers* section for the required Operating system in *EDM04-01 DAG Software Installation Guide* for the further details.

Available Configurations

The available concatenated configurations are shown below:

Number of Ports	Line Type	VC Type and Number	Protocol
4	OC-3c/STM-1c	4 x VC-4	PoS/ATM
4	OC-12c/STM-4c	4 x VC-4-4c	PoS/ATM

Setting up the FPGA

All DAG cards have at least one Field-Programmable Gate Array (FPGA). The FPGA contains the firmware for the DAG card. The firmware defines how the DAG card operates when capturing data and contains the specific configuration.

Note: Some DAG cards have multiple FPGA's.

For each FPGA there are two firmware images:

- a factory image - contains fixed basic functionality for operating the DAG card.
- a user image - contains an upgradable version of the DAG card firmware. Additional functionality for the DAG card is available via the user image. Different user images may be available with different functionality, i.e. TERF, DSM etc.

Firmware images are loaded into DAG card flash ROM in the factory. The image is programmed into the FPGA each time the DAG card is powered up. The user image can then be programmed into the FPGA either manually or via a script.

Programming the FPGA

Before configuring the DAG card for capture, you must load and program the DAG card with the appropriate FPGA image.

Note: For information about the `dagrom` options, see dagrom (page 13).

The following explains how to program the DAG 7.1S:

where "0" is the device number of the DAG card you wish to capture data from

1. Place the card in "`eql`" mode to prevent any erroneous signals interfering with the images during loading using :
   ```
   dagconfig –d0 eql
   ```
2. Load the images using the following commands:
   - PCI-Express FPGA image:
     ```
     dagrom -rvp –d0 -f dag71spci-conc-terf.bit
     ```
   - PHY FPGA image:
     ```
     dagld –x –d0 dag71spp-conc-terf.bit: dag71spp-conc-terf.bit
     ```

   The filename of the FPGA image may differ from the above depending on the version required.

dagrom

`dagrom` is a software utility that enables you to configure the FPGA on Endace DAG cards. The following is a list of options available in `dagrom`.

Option	Description
`-a,--alternate-half`	Use alternate (stable) half. [Default is current half.] Factory / User.
`-A,--entire-rom`	Entire ROM. [Default is current half only.]
`-b,--swid-rom-check`	Check if there is a SWID on the ROM.
`-c,--cpu-region <region>`	Access CPU region: c=copro, b=boot, k=kernel, f=filesystem.
`--continue`	Continue on erase error.
`-d,--device <device>`	DAG device to use.
`-e,--erase`	Erase ROM. [Default is read.]
`-F,--disable-cfi-fast`	Disable fast program option for CFI mode.
`-f,--file <filename>`	File to be read when programming ROM. There are multiple FGPA images per DAG card, covering the different versions, ERF, TERF DSM etc.
`--force`	Force loading firmware. Dangerous.
`-g,--rom-number <rom>`	Access specified ROM controller. [Default is 0.]

`-h,--help`
`-?,--usage`
	This page.

`-i,--halt-ixp`	Halt the embedded IXP Processor (DAG 7.1S only).
`--image-table-fpga <image table fpga>`	Specify the Power On image selection table FPGA number
`--image-table-image <image table image>`	Specify the Power On image selection table Image number
`-j,--swid-rom-check-key <key>`	Check the ROM SWID key with the one supplied.
`-l,--hold-bus`	Hold PBI bus from XScale (DAG 3.7T only).
`-m,--swid-key <key>`	Hexadecimal key for writing the Software ID (aka SWID).
`-o,--swid-rom-read`	Read SWID from ROM.
`-p,--program-current`	Program current User 1 Xilinx image into FPGA.
`-q,--image-number <image number>`	Specify the image number to write or to program the card.[0 - 3]. 0 factory image, 1 user image 1, 2 user image 2, 3 user image 3. (7.5G2/G4 only)
`--swid-write <swid>`	Write given SWID. The key must be supplied with the -m option, requires a valid running XScale ROM Image. (3.7T, 3.7D, 3.8S and 7.1S only)
`-r,--reprogram`	Reprogram ROM (may imply erase and write).
`--reset-method <reprogram method>`	Specify the method to reprogram the card.[1.Ringo 2.George 3.Dave]
`-s,--swid-rom-write <swid>`	Write given SWID to ROM. The key must be supplied with the -m option.
`-t,--swid-read-bytes <bytes>`	Read <bytes> of SWID, requires a valid running XScale ROM image (3.7T only)
`-u,--swid-erase`	Erase SWID from ROM.
`--unknown`	Force loading firmware. Dangerous.
`-v,--verbose`	Increase verbosity.
`-V,--version`	Display version information.
`-w,--write`	Write ROM (implies erase). [Default is read.]
`--write-out <filename>`	Write the contents of the ROM to a file.
`-x,--list-revisions`	Display Xilinx revision strings (the default if no arguments are given).
`-y,--verify`	Verify write to ROM.
`-z,--zero`	Zero ROM. [Default is read.]

All commands apply to the current image portion of the ROM, unless one of the options `-a`, `-A`, `-c` is specified.

Note: Not all commands are supported by all DAG cards.

To view the FPGA image revision strings, type the following:

```
dagrom -d0 -x
```

where "0" is the device number of the DAG card you wish to capture data from

Output:

```
user:    dag71spci_oc12c_pci_v2_39 2vp30ff1152 2007/06/13 11:23:02 (active)
factory: dag71spci_oc12c_pci_v2_35 2vp30ff1152 2007/02/02 14:59:50
```

dagld

`dagld` is a software tool that enables you to load the PHY FPGA images into DAG cards.

The following is a list of options available in `dagld`:

Option	Description
-d,--device <device>	DAG device to use.
-?,--usage -h,--help	This page.
--force	Force loading firmware. Dangerous.
-r,--strongarm	Load and run StrongARM binary (DAG 3.5S).
--unknown	Force loading firmware. Dangerous.
-v,--verbose	Increase verbosity.
-V,--version	Display version information.
-x,--xilinx	File contains Xilinx FPGA image.

Loading new firmware images onto a DAG Card

New DAG card FPGA images are released regularly by Endace as part of software packages. They can be downloaded from the Endace website at https://www.endace.com/support.

Endace recommends you use the `dagrom -r` command when loading images from the computer to the ROM on the DAG card.

The `-r` option invokes a comparison of images on the computer and in the DAG card. Newer versions are automatically loaded onto the DAG card and programmed into the FPGA. See dagrom (page 13). This eliminates unnecessary reprogramming of the ROM and extends its life.

Preparing the DAG card for use

Before configuring the DAG 7.1S card you must run the following dagconfig command to set the default parameters in the DAG card. This ensures the DAG 7.1S card functions correctly once you begin capturing data.

Note: Ensure you run this command each time the FPGA is reprogrammed.

```
dagconfig -d0 default
```

The current DAG 7.1S configuration displays and the firmware is verified as correctly loaded. See dagconfig (page 45) for more information.

Display Current Configuration

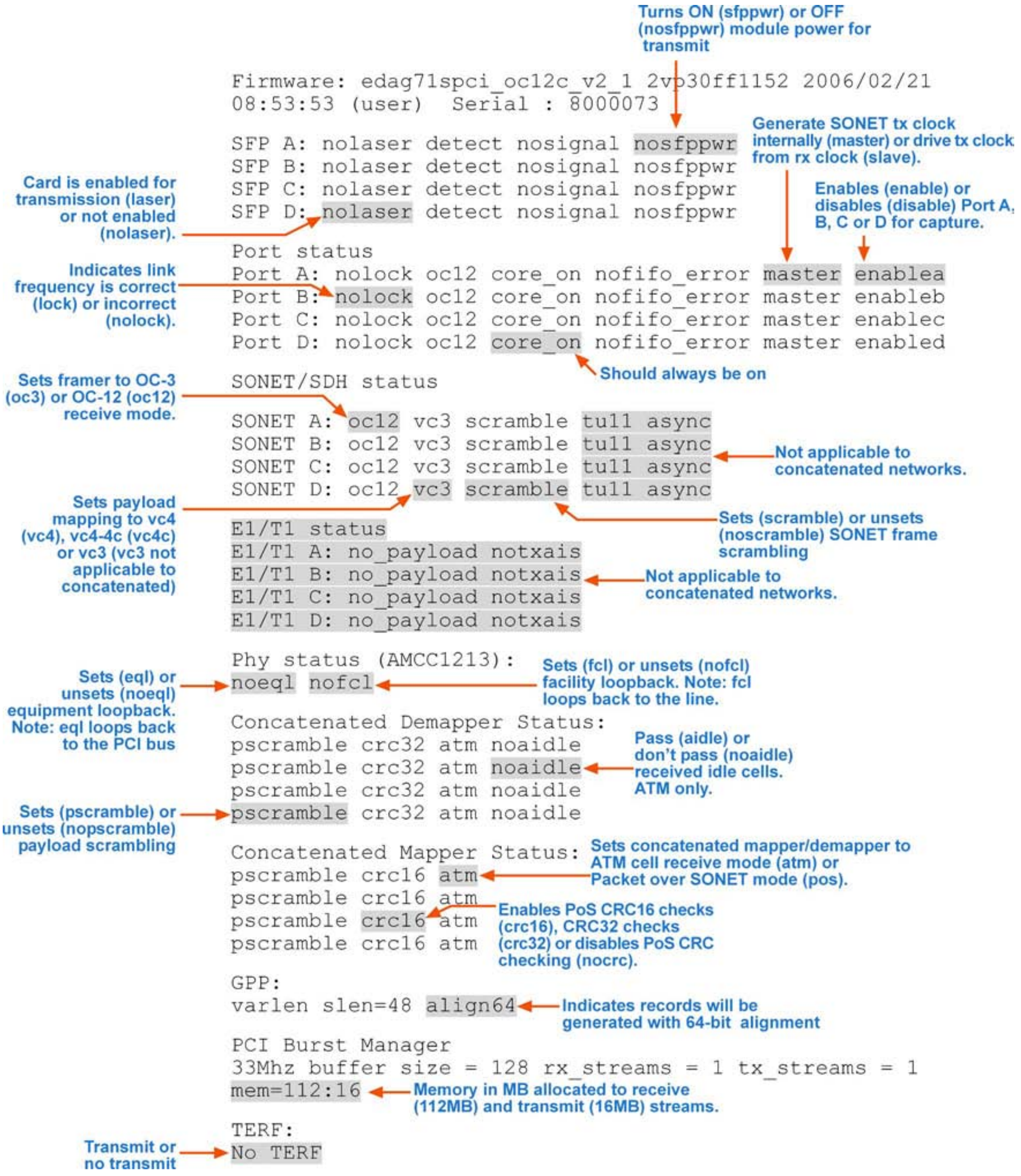
Once you have loaded the appropriate images you should run the `dagconfig` tool without arguments to display the current card configuration and verify the firmware has loaded correctly, using:

```
dagconfig -d0
```

(where “0” is the device number of the DAG card)

For an explanation of the default `dagconfig` output see dagconfig. (page 39)

Note: The `dagconfig` default output may include some information that is not applicable to concatenated networks and these are indicated in the example.



```
Firmware: edag71spci_oc12c_v2_1 2vp30ff1152 2006/02/21
08:53:53 (user)  Serial : 8000073

SFP A: nolaser detect nosignal nosfppwr
SFP B: nolaser detect nosignal nosfppwr
SFP C: nolaser detect nosignal nosfppwr
SFP D: nolaser detect nosignal nosfppwr

Port status
Port A: nolock oc12 core_on nofifo_error master enablea
Port B: nolock oc12 core_on nofifo_error master enableb
Port C: nolock oc12 core_on nofifo_error master enablec
Port D: nolock oc12 core_on nofifo_error master enabled

SONET/SDH status

SONET A: oc12 vc3 scramble tu11 async
SONET B: oc12 vc3 scramble tu11 async
SONET C: oc12 vc3 scramble tu11 async
SONET D: oc12 vc3 scramble tu11 async

E1/T1 status
E1/T1 A: no_payload notxais
E1/T1 B: no_payload notxais
E1/T1 C: no_payload notxais
E1/T1 D: no_payload notxais

Phy status (AMCC1213):
noeql nofcl

Concatenated Demapper Status:
pscramble crc32 atm noaidle
pscramble crc32 atm noaidle
pscramble crc32 atm noaidle
pscramble crc32 atm noaidle

Concatenated Mapper Status:
pscramble crc16 atm
pscramble crc16 atm
pscramble crc16 atm
pscramble crc16 atm

GPP:
varlen slen=48 align64

PCI Burst Manager
33Mhz buffer size = 128 rx_streams = 1 tx_streams = 1
mem=112:16

TERF:
No TERF
```

Turns ON (sfppwr) or OFF (nosfppwr) module power for transmit

Generate SONET tx clock internally (master) or drive tx clock from rx clock (slave).

Card is enabled for transmission (laser) or not enabled (nolaser).

Enables (enable) or disables (disable) Port A, B, C or D for capture.

Indicates link frequency is correct (lock) or incorrect (nolock).

Should always be on

Sets framer to OC-3 (oc3) or OC-12 (oc12) receive mode.

Not applicable to concatenated networks.

Sets payload mapping to vc4 (vc4), vc4-4c (vc4c) or vc3 (vc3 not applicable to concatenated)

Sets (scramble) or unsets (noscramble) SONET frame scrambling

Not applicable to concatenated networks.

Sets (fcl) or unsets (nofcl) facility loopback. Note: fcl loops back to the line.

Sets (eql) or unsets (noeql) equipment loopback. Note: eql loops back to the PCI bus

Pass (aidle) or don’t pass (noaidle) received idle cells. ATM only.

Sets (pscramble) or unsets (nopscramble) payload scrambling

Sets concatenated mapper/demapper to ATM cell receive mode (atm) or Packet over SONET mode (pos).

Enables PoS CRC16 checks (crc16), CRC32 checks (crc32) or disables PoS CRC checking (nocrc).

Indicates records will be generated with 64-bit alignment

Memory in MB allocated to receive (112MB) and transmit (16MB) streams.

Transmit or no transmit

Component not found

If the firmware has not loaded correctly the `dagconfig` output will indicate "`component not found`" as the `SONET/SDH status` and `E1/T1 status` as shown below:

```
SONET/SDH status
SONET A:  component not found.
SONET B:  component not found.
SONET C:  component not found.
SONET D:  component not found.
E1/T1 status
E1/T1 A:  component not found.
E1/T1 B:  component not found.
E1/T1 C:  component not found.
E1/T1 D:  component not found.
```

In this case you should perform the following steps:

1. Check the firmware image that you are using and ensure that it is correct for your network configuration and protocol.
2. Then reload the firmware as described in Preparing the DAG card (page 14) for use earlier in this chapter.

If after performing these steps `dagconfig` still displays "`component not found`" please contact Endace Customer Support at support@endace.com for further assistance.

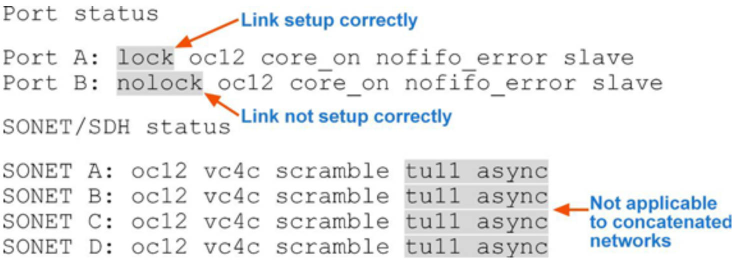
Verify Mapping/Framing Setup

Parameters

As shown on the previous page, the `dagconfig` output displays whether or not mapping and framing is setup correctly.

For correct setup both ends of the link must match the framing and mapping parameters. The main parameters involved in setting the framing are:

- Link speed `(oc3/oc12)`
- Clock master `(master/slave)`
- Payload mapping `(vc4c/vc4)`



```
Port status

Port A: lock oc12 core_on nofifo_error slave
Port B: nolock oc12 core_on nofifo_error slave

SONET/SDH status

SONET A: oc12 vc4c scramble tu11 async
SONET B: oc12 vc4c scramble tu11 async
SONET C: oc12 vc4c scramble tu11 async
SONET D: oc12 vc4c scramble tu11 async
```

Link setup correctly

Link not setup correctly

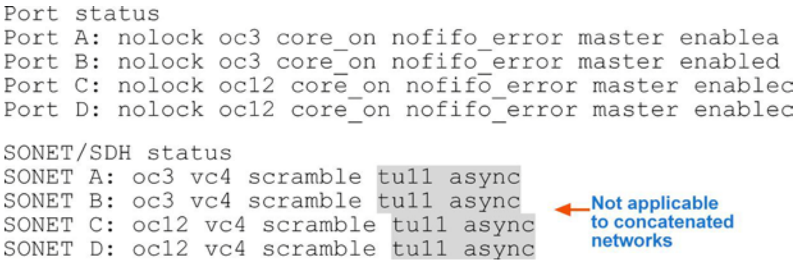
Not applicable to concatenated networks

Different Port Configurations

The DAG 7.1S card allows different ports to have different configurations. For example if you want to configure only Ports A and B for OC-3c VC-4c you could do so using the following command:

```
dagconfig –d0 default -1 -4 oc3 vc4
```

Output



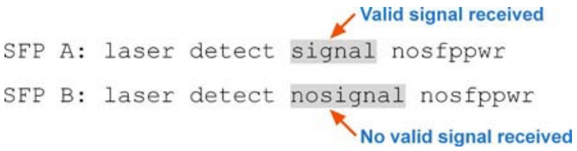
```
Port status
Port A: nolock oc3 core_on nofifo_error master enablea
Port B: nolock oc3 core_on nofifo_error master enabled
Port C: nolock oc12 core_on nofifo_error master enablec
Port D: nolock oc12 core_on nofifo_error master enablec

SONET/SDH status
SONET A: oc3 vc4 scramble tu11 async
SONET B: oc3 vc4 scramble tu11 async
SONET C: oc12 vc4 scramble tu11 async
SONET D: oc12 vc4 scramble tu11 async
```

Not applicable to concatenated networks

Verify Optical Signal

As previously shown, the `dagconfig` output displays whether or not a port is receiving a valid optical signal.



```
SFP A: laser detect signal nosfppwr

SFP B: laser detect nosignal nosfppwr
```

Valid signal received

No valid signal received

To receive a valid optical signal, both ends of the link must use be using the same type of optical transceivers. In addition the optical fiber used must match the requirements of the optical transceiver and be in good condition.

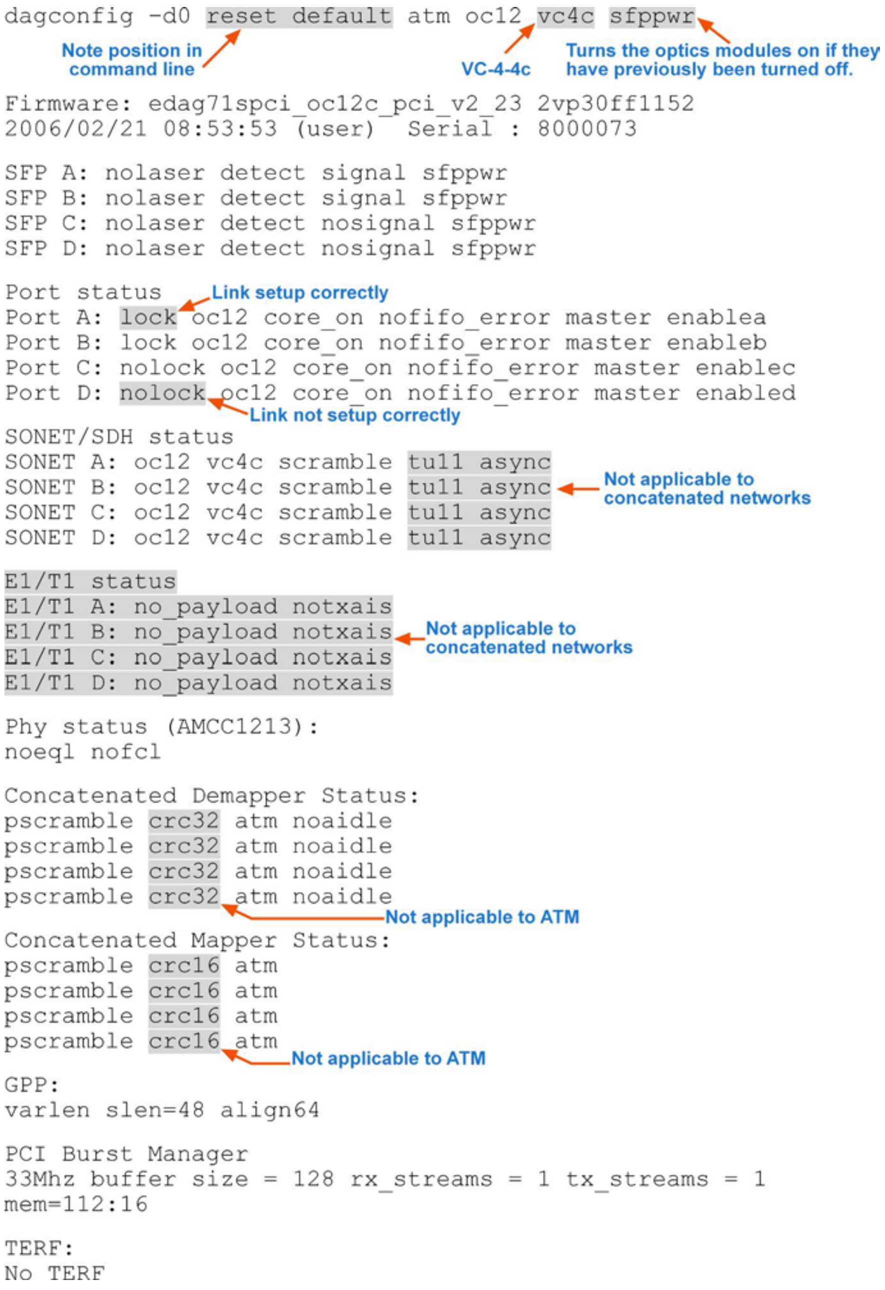
ATM Mode

When configuring the DAG 7.1S card for ATM mode ensure you configure it to match the characteristics of the network you wish to monitor.

Please refer to Display Current Configuration (page 15)earlier in this chapter for a full explanation of `dagconfig` outputs.

Note: The order in which the configuration options appear in the command line is important. Therefore you should use `default` and `reset` before other options in the command line.

The example below has been configured for OC-12c VC-4-4c on ports A and B with ports C and D unused:



```
dagconfig -d0 reset default atm oc12 vc4c sfppwr

Firmware: edag71spci_oc12c_pci_v2_23 2vp30ff1152
2006/02/21 08:53:53 (user)  Serial : 8000073

SFP A: nolaser detect signal sfppwr
SFP B: nolaser detect signal sfppwr
SFP C: nolaser detect nosignal sfppwr
SFP D: nolaser detect nosignal sfppwr

Port status
Port A: lock oc12 core_on nofifo_error master enablea
Port B: lock oc12 core_on nofifo_error master enableb
Port C: nolock oc12 core_on nofifo_error master enablec
Port D: nolock oc12 core_on nofifo_error master enabled

SONET/SDH status
SONET A: oc12 vc4c scramble tu11 async
SONET B: oc12 vc4c scramble tu11 async
SONET C: oc12 vc4c scramble tu11 async
SONET D: oc12 vc4c scramble tu11 async

E1/T1 status
E1/T1 A: no_payload notxais
E1/T1 B: no_payload notxais
E1/T1 C: no_payload notxais
E1/T1 D: no_payload notxais

Phy status (AMCC1213):
noeql nofcl

Concatenated Demapper Status:
pscramble crc32 atm noaidle
pscramble crc32 atm noaidle
pscramble crc32 atm noaidle
pscramble crc32 atm noaidle

Concatenated Mapper Status:
pscramble crc16 atm
pscramble crc16 atm
pscramble crc16 atm
pscramble crc16 atm

GPP:
varlen slen=48 align64

PCI Burst Manager
33Mhz buffer size = 128 rx_streams = 1 tx_streams = 1
mem=112:16

TERF:
No TERF
```

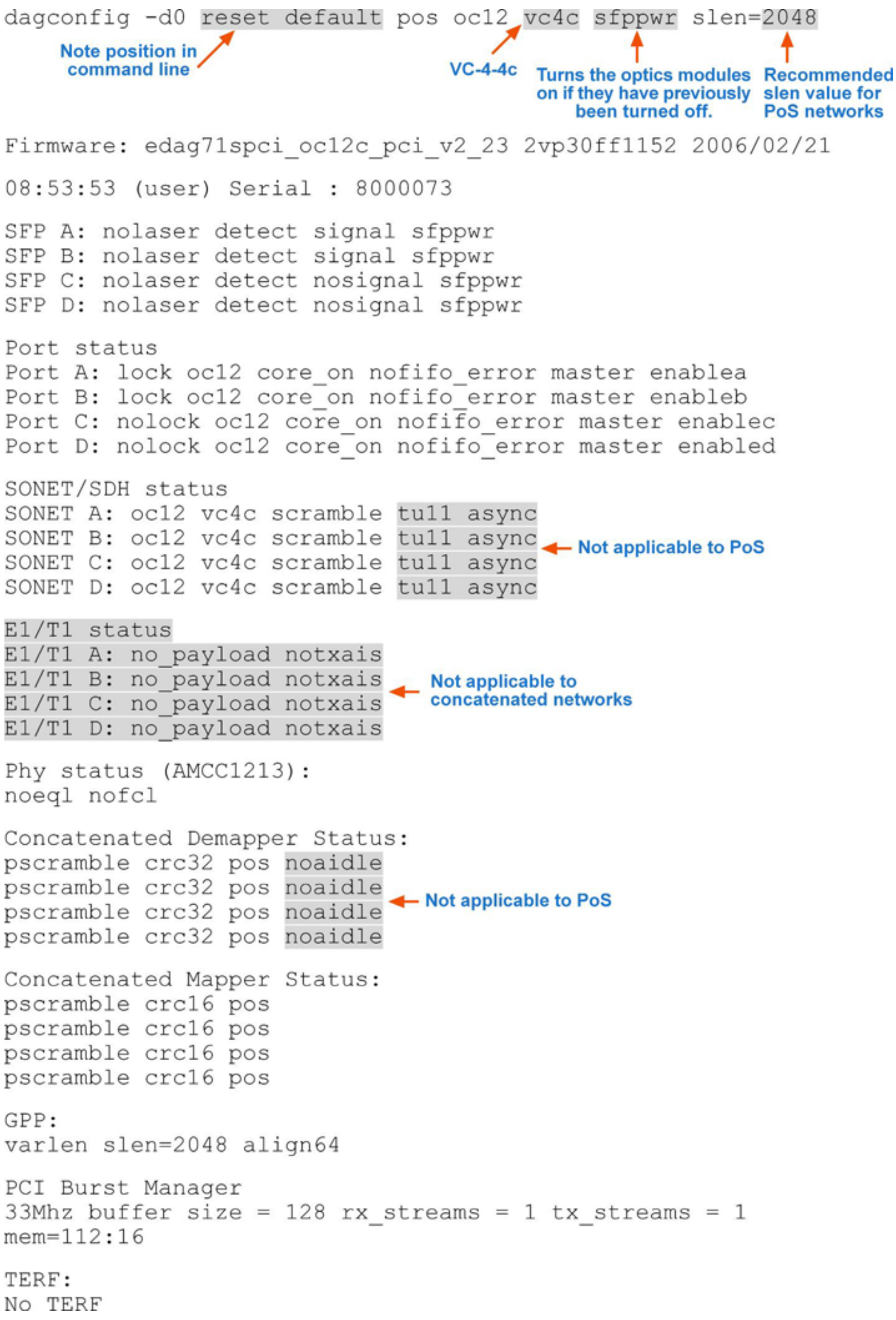
Note position in command line
VC-4-4c
Turns the optics modules on if they have previously been turned off.
Link setup correctly
Link not setup correctly
Not applicable to concatenated networks
Not applicable to concatenated networks
Not applicable to ATM
Not applicable to ATM

PoS Mode

To configure the DAG 7.1S card for PoS mode you must replace oc12 and vc4c to match your network configuration as shown below:

Note: The order in which the configuration options appear in the command line is important. Therefore you should use default and reset before other options in the command line.

See Display Current Configuration (page 15) earlier in this chapter for a full explanation of dagconfig outputs.



```
dagconfig -d0 reset default pos oc12 vc4c sfppwr slen=2048

Firmware: edag71spci_oc12c_pci_v2_23 2vp30ff1152 2006/02/21

08:53:53 (user) Serial : 8000073

SFP A: nolaser detect signal sfppwr
SFP B: nolaser detect signal sfppwr
SFP C: nolaser detect nosignal sfppwr
SFP D: nolaser detect nosignal sfppwr

Port status
Port A: lock oc12 core_on nofifo_error master enablea
Port B: lock oc12 core_on nofifo_error master enableb
Port C: nolock oc12 core_on nofifo_error master enablec
Port D: nolock oc12 core_on nofifo_error master enabled

SONET/SDH status
SONET A: oc12 vc4c scramble tu11 async
SONET B: oc12 vc4c scramble tu11 async
SONET C: oc12 vc4c scramble tu11 async
SONET D: oc12 vc4c scramble tu11 async

E1/T1 status
E1/T1 A: no_payload notxais
E1/T1 B: no_payload notxais
E1/T1 C: no_payload notxais
E1/T1 D: no_payload notxais

Phy status (AMCC1213):
noeql nofcl

Concatenated Demapper Status:
pscramble crc32 pos noaidle
pscramble crc32 pos noaidle
pscramble crc32 pos noaidle
pscramble crc32 pos noaidle

Concatenated Mapper Status:
pscramble crc16 pos
pscramble crc16 pos
pscramble crc16 pos
pscramble crc16 pos

GPP:
varlen slen=2048 align64

PCI Burst Manager
33Mhz buffer size = 128 rx_streams = 1 tx_streams = 1
mem=112:16

TERF:
No TERF
```

Note position in command line
VC-4-4c
Turns the optics modules on if they have previously been turned off.
Recommended slen value for PoS networks
Not applicable to PoS
Not applicable to concatenated networks
Not applicable to PoS

Interface Statistics

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:

```
dagconfig -d0 -s
```

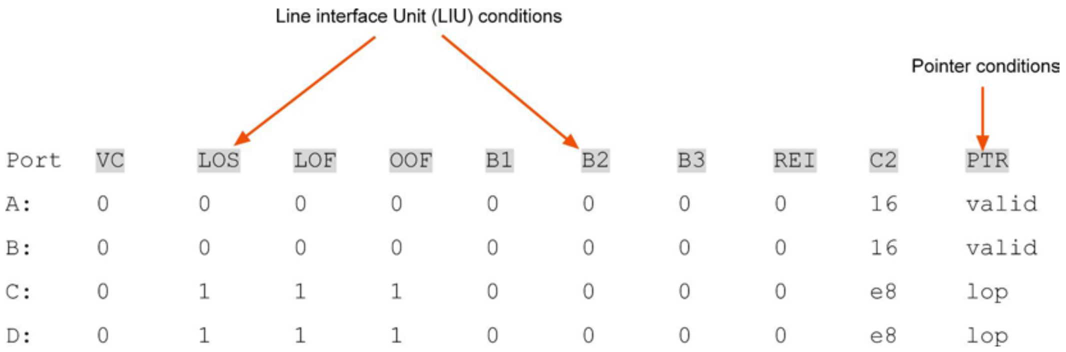
where "0" is the device number of the DAG card you wish to capture data from

Example outputs are shown below:

Note: “1” indicates the condition is present on the link “0” indicates the condition is not present on the link. See Status Conditions (page 21) later in this chapter for a full description of each of the status conditions.

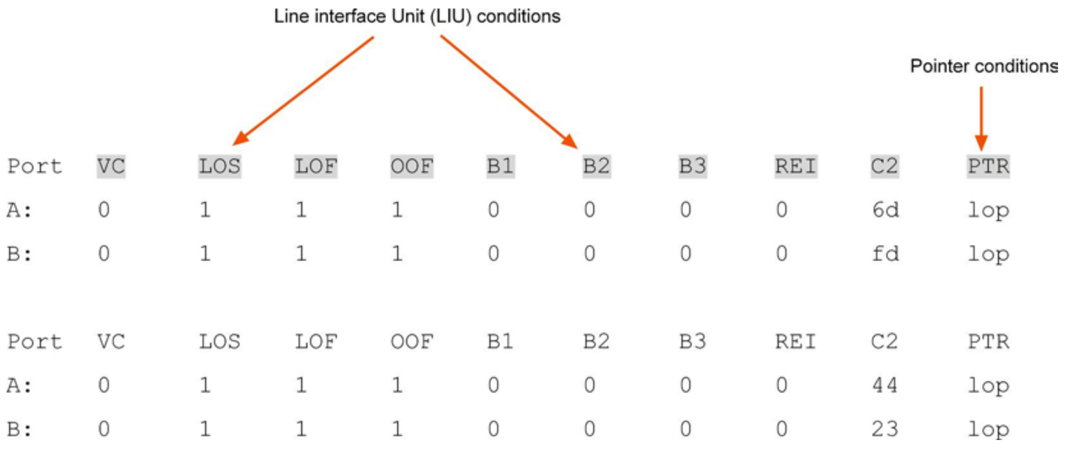
PoS OC12 Stream

In the example below the card is locked to a PoS OC-12 stream on ports A and B, ports C and D are unused:



```
Port  VC   LOS   LOF   OOF   B1   B2   B3   REI   C2   PTR
A:    0    0     0     0     0    0    0    0     16   valid
B:    0    0     0     0     0    0    0    0     16   valid
C:    0    1     1     1     0    0    0    0     e8   lop
D:    0    1     1     1     0    0    0    0     e8   lop
```

In the example below the card is set to OC-3 PoS while the line carries OC-12 PoS:

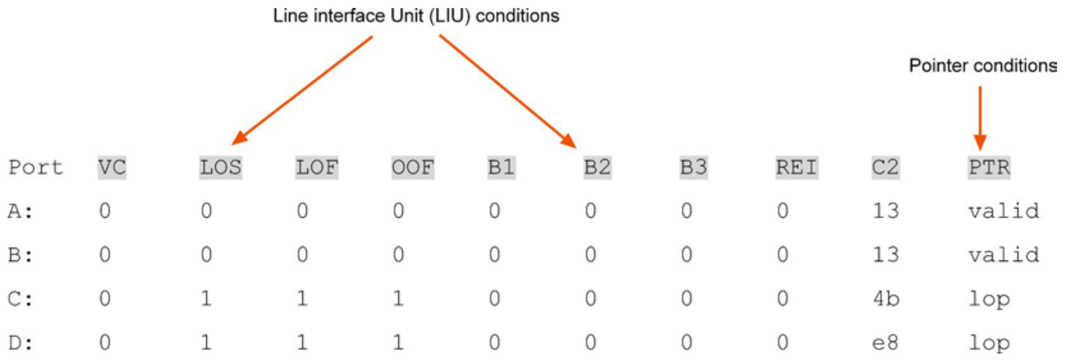


```
Port  VC   LOS   LOF   OOF   B1   B2   B3   REI   C2   PTR
A:    0    1     1     1     0    0    0    0     6d   lop
B:    0    1     1     1     0    0    0    0     fd   lop

Port  VC   LOS   LOF   OOF   B1   B2   B3   REI   C2   PTR
A:    0    1     1     1     0    0    0    0     44   lop
B:    0    1     1     1     0    0    0    0     23   lop
```

ATM Cell Stream

In the example below the card is locked to an ATM OC-12 stream on ports A and B, ports C and D are unused:



```
Port  VC   LOS   LOF   OOF   B1   B2   B3   REI   C2   PTR
A:    0    0     0     0     0    0    0    0     13   valid
B:    0    0     0     0     0    0    0    0     13   valid
C:    0    1     1     1     0    0    0    0     4b   lop
D:    0    1     1     1     0    0    0    0     e8   lop
```

Status Conditions

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

Condition	Definition
`B1, B2, B3`	Bit interleaved parity error as reported by SONET B1, B2 and B3 overhead octets. Indicates the connection between the card and the network is impaired. • If `OOF` and `LOF` conditions are **also** set the OCx carrier configuration is incorrect. • If `OOF` and `LOF` conditions are **not** set the there is a signal problem related to low light levels reaching the optical receivers, or there are true SONET-level errors on the equipment operating the link.
`C2`	Path signal label. Reflects the content of SONET C2 overhead octet. Typical settings are: • 13ATM • 16 PPP w/SPE scrambling • CF PPP wo/SPE scrambling Changing values for this condition indicate a SONET level error.
`LOF`	Loss of Frame. Indicates Out of Frame (`OOF`) condition has been asserted for more than 2 milliseconds.
`LOS`	Loss of Signal. There is either no signal at the receiver or the optical signal strength is too low for the card to recognize.
`OOF`	Out of Frame. The section overhead processor is not locked to the SONET stream and may indicate incorrect OC-3/OC-12 setting.
`PTR`	Indicates if the pointer processing logic has locked to the SONET stream. Possible values are: • `valid`: The pointer has locked to the SONET stream. • `lop` (loss of pointer): The pointer has not locked to the SONET stream and may indicate incorrect OC- 3/OC-12 setting. • `conc`: The card is configured for channelised data but the network is concatenated.
`REI`	Remote Error Indicator.
`VC`	Virtual container number. Should always be "`0`" for concatenated networks.

ATM Network

If the card is connected to an ATM network and you have loaded the correct firmware for the network configuration the following values apply:

- C2 should be "`13`"
- PTR should be "`valid`"
- Remaining statistics should be "`0`"

PoS Network

If the card is connected to an PoS network and you have loaded the correct firmware for the network configuration the following values apply:

- C2 should be "`16`" for PPP or "`cf`" for HDLC
- PTR should be "`valid`"
- Remaining statistics should be "`0`"

Verify Configuration

You can verify the card configuration by checking settings and path label for any errors as follows:

- Ensure `LOS` is 0, otherwise check light levels.
- Ensure `OOF` and `LOF` are 0, otherwise change OC-3 settings to OC-12 or vice versa.
- Ensure `PTR` is `valid`
- Ensure no bit interleaved parity errors occur, otherwise check cabling and light levels.
- Ensure `C2` is correct for the payload
- Ensure PoS scrambling and CRC settings are correct.

Channelised Configuration

Introduction

Configuring the DAG 7.1S card ready for capturing data requires the following steps:

- Setting up the FPGA (page 24)
- Preparing the DAG card for use (page 26)
- Display Current Configuration (page 27)
- Interface Statistics (page 20)
- Verify Configuration (page 22)

Once the DAG 7.1S is configured you can start capturing data, see Using your DAG card to capture data (page 47) for details on capturing data.

Before configuring the DAG card

Before configuring the FPGA, you should ensure that:

- `dagmem` has been run and memory allocated to each installed DAG card.
- `dagload` has been run so that all DAG drivers have been installed.

Refer to the *Installing the drivers* section for the required Operating system in *EDM04-01 DAG Software Installation Guide* for the further details.

Available Configurations

The available channelized configurations are shown below:

Number of Ports	Line Type	VC Type and Number	Protocol
4	OC-3/STM-1	252 x VC-12 (E1) 336 x VC-11 (T1)	ATM/Bit-HDLC/RAW
2	OC-12/STM-4	504 x VC-12 (E1) 672 x VC-11 (T1)	ATM/Bit-HDLC/RAW

Setting up the FPGA

All DAG cards have at least one Field-Programmable Gate Array (FPGA). The FPGA contains the firmware for the DAG card. The firmware defines how the DAG card operates when capturing data and contains the specific configuration.

Note: Some DAG cards have multiple FPGA's.

For each FPGA there are two firmware images:

- a factory image - contains fixed basic functionality for operating the DAG card.
- a user image - contains an upgradable version of the DAG card firmware. Additional functionality for the DAG card is available via the user image. Different user images may be available with different functionality, i.e. TERF, DSM etc.

Firmware images are loaded into DAG card flash ROM in the factory. The image is programmed into the FPGA each time the DAG card is powered up. The user image can then be programmed into the FPGA either manually or via a script.

Programming the FPGA

Before configuring the DAG card for capture, you must load and program the DAG card with the appropriate FPGA image.

Note: For information about the `dagrom` options, see dagrom (page 13).

The following explains how to program the DAG 7.1S:

(where "0" is the device number of the DAG card you wish to capture data from)

1. Place the card in “`eql`” mode to prevent any erroneous signals interfering with the images during loading using:
   ```
   dagconf –d0 eql
   ```
2. Load the images using the following commands:
   - PCI-Express FPGA image:
     ```
     dagrom -rvp –d0 -f xilinx/dag71spci-chan-terf.bit
     ```
   - PHY FPGA image:
     ```
     dagld –x –d0 dag71spp-chan-terf.bit: dag71spp-chan-terf.bit
     ```

   The filename of the FPGA image may differ from the above depending on the version required.

dagrom

`dagrom` is a software utility that enables you to configure the FPGA on Endace DAG cards. The following is a list of options available in `dagrom`.

Option	Description
`-a,--alternate-half`	Use alternate (stable) half. [Default is current half.] Factory / User.
`-A,--entire-rom`	Entire ROM. [Default is current half only.]
`-b,--swid-rom-check`	Check if there is a SWID on the ROM.
`-c,--cpu-region <region>`	Access CPU region: c=copro, b=boot, k=kernel, f=filesystem.
`--continue`	Continue on erase error.
`-d,--device <device>`	DAG device to use.
`-e,--erase`	Erase ROM. [Default is read.]
`-F,--disable-cfi-fast`	Disable fast program option for CFI mode.
`-f,--file <filename>`	File to be read when programming ROM. There are multiple FGPA images per DAG card, covering the different versions, ERF, TERF DSM etc.
`--force`	Force loading firmware. Dangerous.
`-g,--rom-number <rom>`	Access specified ROM controller. [Default is 0.]

`-h,--help`
`-?,--usage`
	This page.

`-i,--halt-ixp`	Halt the embedded IXP Processor (DAG 7.1S only).
`--image-table-fpga <image table fpga>`	Specify the Power On image selection table FPGA number
`--image-table-image <image table image>`	Specify the Power On image selection table Image number
`-j,--swid-rom-check-key <key>`	Check the ROM SWID key with the one supplied.
`-l,--hold-bus`	Hold PBI bus from XScale (DAG 3.7T only).
`-m,--swid-key <key>`	Hexadecimal key for writing the Software ID (aka SWID).
`-o,--swid-rom-read`	Read SWID from ROM.
`-p,--program-current`	Program current User 1 Xilinx image into FPGA.
`-q,--image-number <image number>`	Specify the image number to write or to program the card.[0 - 3]. 0 factory image, 1 user image 1, 2 user image 2, 3 user image 3. (7.5G2/G4 only)
`--swid-write <swid>`	Write given SWID. The key must be supplied with the -m option, requires a valid running XScale ROM Image. (3.7T, 3.7D, 3.8S and 7.1S only)
`-r,--reprogram`	Reprogram ROM (may imply erase and write).
`--reset-method <reprogram method>`	Specify the method to reprogram the card.[1.Ringo 2.George 3.Dave]
`-s,--swid-rom-write <swid>`	Write given SWID to ROM. The key must be supplied with the -m option.
`-t,--swid-read-bytes <bytes>`	Read <bytes> of SWID, requires a valid running XScale ROM image (3.7T only)
`-u,--swid-erase`	Erase SWID from ROM.
`--unknown`	Force loading firmware. Dangerous.
`-v,--verbose`	Increase verbosity.
`-V,--version`	Display version information.
`-w,--write`	Write ROM (implies erase). [Default is read.]
`--write-out <filename>`	Write the contents of the ROM to a file.
`-x,--list-revisions`	Display Xilinx revision strings (the default if no arguments are given).
`-y,--verify`	Verify write to ROM.
`-z,--zero`	Zero ROM. [Default is read.]

All commands apply to the current image portion of the ROM, unless one of the options `-a`, `-A`, `-c` is specified.

Note: Not all commands are supported by all DAG cards.

To view the FPGA image revision strings, type the following:

```
dagrom -d0 -x
```

where "0" is the device number of the DAG card you wish to capture data from

Output:

```
user:    dag71spci_oc12c_pci_v2_39 2vp30ff1152 2007/06/13 11:23:02 (active)
factory: dag71spci_oc12c_pci_v2_35 2vp30ff1152 2007/02/02 14:59:50
```

dagld

`dagld` is a software tool that enables you to load the PHY FPGA images into DAG cards.

The following is a list of options available in `dagld`:

Option	Description
-d,--device <device>	DAG device to use.
-?,--usage -h,--help	This page.
--force	Force loading firmware. Dangerous.
-r,--strongarm	Load and run StrongARM binary (DAG 3.5S).
--unknown	Force loading firmware. Dangerous.
-v,--verbose	Increase verbosity.
-V,--version	Display version information.
-x,--xilinx	File contains Xilinx FPGA image.

Loading new firmware images onto a DAG Card

New DAG card FPGA images are released regularly by Endace as part of software packages. They can be downloaded from the Endace website at https://www.endace.com/support.

Endace recommends you use the `dagrom -r` command when loading images from the computer to the ROM on the DAG card.

The `-r` option invokes a comparison of images on the computer and in the DAG card. Newer versions are automatically loaded onto the DAG card and programmed into the FPGA. See dagrom (page 13). This eliminates unnecessary reprogramming of the ROM and extends its life.

Preparing the DAG card for use

Before configuring the DAG 7.1S card you must run the following dagconfig command to set the default parameters in the DAG card. This ensures the DAG 7.1S card functions correctly once you begin capturing data.

Note: Ensure you run this command each time the FPGA is reprogrammed.

```
dagconfig -d0 default
```

where "0" is the device number of the DAG card you wish to capture data from.
The current DAG 7.1S configuration displays and the firmware is verified as correctly loaded. See dagconfig (page 45) for more information.

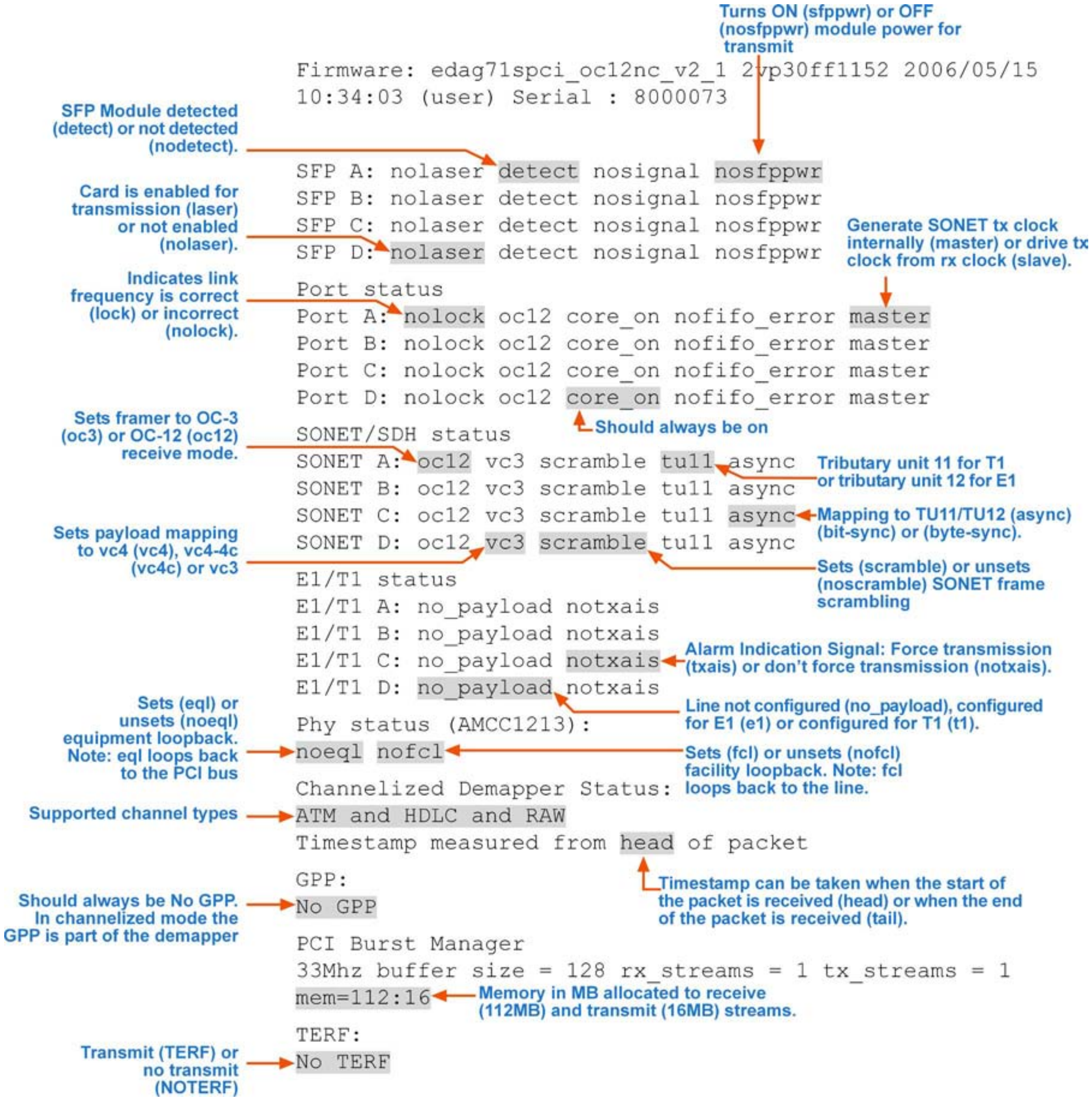
Display Current Configuration

Once you have loaded the appropriate images you should run the `dagconfig` tool without arguments to display the current card configuration and verify the firmware has loaded correctly, using:

```
dagconfig -d0
```

(where “`0`” is the device number of the DAG card)

For an explanation of the default `dagconfig` output see dagconfig. (page 39)



```
Firmware: edag71spci_oc12nc_v2_1 2vp30ff1152 2006/05/15
10:34:03 (user) Serial : 8000073

SFP A: nolaser detect nosignal nosfppwr
SFP B: nolaser detect nosignal nosfppwr
SFP C: nolaser detect nosignal nosfppwr
SFP D: nolaser detect nosignal nosfppwr

Port status
Port A: nolock oc12 core_on nofifo_error master
Port B: nolock oc12 core_on nofifo_error master
Port C: nolock oc12 core_on nofifo_error master
Port D: nolock oc12 core_on nofifo_error master

SONET/SDH status
SONET A: oc12 vc3 scramble tu11 async
SONET B: oc12 vc3 scramble tu11 async
SONET C: oc12 vc3 scramble tu11 async
SONET D: oc12 vc3 scramble tu11 async

E1/T1 status
E1/T1 A: no_payload notxais
E1/T1 B: no_payload notxais
E1/T1 C: no_payload notxais
E1/T1 D: no_payload notxais

Phy status (AMCC1213):
noeql nofcl

Channelized Demapper Status:
ATM and HDLC and RAW
Timestamp measured from head of packet

GPP:
No GPP

PCI Burst Manager
33Mhz buffer size = 128 rx_streams = 1 tx_streams = 1
mem=112:16

TERF:
No TERF
```

Turns ON (sfppwr) or OFF (nosfppwr) module power for transmit

SFP Module detected (detect) or not detected (nodetect).

Card is enabled for transmission (laser) or not enabled (nolaser).

Generate SONET tx clock internally (master) or drive tx clock from rx clock (slave).

Indicates link frequency is correct (lock) or incorrect (nolock).

Should always be on

Sets framer to OC-3 (oc3) or OC-12 (oc12) receive mode.

Tributary unit 11 for T1 or tributary unit 12 for E1

Mapping to TU11/TU12 (async) (bit-sync) or (byte-sync).

Sets payload mapping to vc4 (vc4), vc4-4c (vc4c) or vc3

Sets (scramble) or unsets (noscramble) SONET frame scrambling

Alarm Indication Signal: Force transmission (txais) or don’t force transmission (notxais).

Line not configured (no_payload), configured for E1 (e1) or configured for T1 (t1).

Sets (eql) or unsets (noeql) equipment loopback. Note: eql loops back to the PCI bus

Sets (fcl) or unsets (nofcl) facility loopback. Note: fcl loops back to the line.

Supported channel types

Timestamp can be taken when the start of the packet is received (head) or when the end of the packet is received (tail).

Should always be No GPP. In channelized mode the GPP is part of the demapper

Memory in MB allocated to receive (112MB) and transmit (16MB) streams.

Transmit (TERF) or no transmit (NOTERF)

Component not found

If the firmware has not loaded correctly the dagconfig output will indicate `“component not found”` as the `SONET/SDH status` and `E1/T1 status` as shown below:

```
SONET/SDH status
SONET A:  component not found.
SONET B:  component not found.
SONET C:  component not found.
SONET D:  component not found.
E1/T1 status
E1/T1 A:  component not found.
E1/T1 B:  component not found.
E1/T1 C:  component not found.
E1/T1 D:  component not found.
```

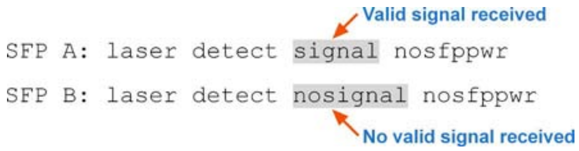
In this case you should perform the following steps:

1. Check the firmware image that you are using and ensure that it is correct for your network configuration and protocol.
2. Then reload the firmware as described in Preparing the DAG card for use earlier in this chapter.

If after performing these steps dagconfig still displays “`component not found`” please contact Endace Customer Support at support@endace.com for further assistance.

Verify Optical Signal

As previously shown, the `dagconfig` output displays whether or not a port is receiving a valid optical signal.



Valid signal received

```
SFP A: laser detect signal nosfppwr
SFP B: laser detect nosignal nosfppwr
```

No valid signal received

To receive a valid optical signal, both ends of the link must use be using the same type of optical transceivers. In addition the optical fiber used must match the requirements of the optical transceiver and be in good condition.

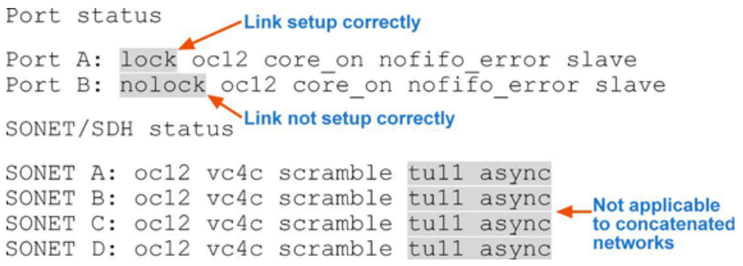
Verify Mapping/Framing Setup

Parameters

As shown on the previous page, the dagconfig output displays whether or not mapping and framing is setup correctly.

For correct setup both ends of the link must match the framing and mapping parameters. The main parameters involved in setting the framing are:

- Link speed (`oc3/oc12`)
- Clock master (`master/slave`)
- Payload mapping (`vc4/vc3`)



```
Port status

Port A: lock oc12 core_on nofifo_error slave
Port B: nolock oc12 core_on nofifo_error slave

SONET/SDH status

SONET A: oc12 vc4c scramble tu11 async
SONET B: oc12 vc4c scramble tu11 async
SONET C: oc12 vc4c scramble tu11 async
SONET D: oc12 vc4c scramble tu11 async
```

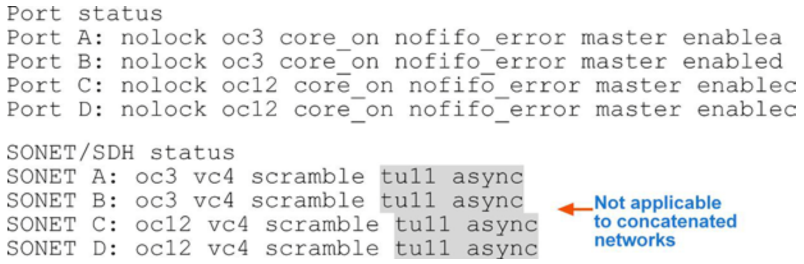
Link setup correctly

Link not setup correctly

Not applicable to concatenated networks

Different Port Configurations

The DAG card allows different ports to have different configurations. For example if you want to configure only Ports A and B for OC-3c VC-4c you could do so using the following command:



```
Port status
Port A: nolock oc3 core_on nofifo_error master enablea
Port B: nolock oc3 core_on nofifo_error master enabled
Port C: nolock oc12 core_on nofifo_error master enablec
Port D: nolock oc12 core_on nofifo_error master enablec

SONET/SDH status
SONET A: oc3 vc4 scramble tu11 async
SONET B: oc3 vc4 scramble tu11 async
SONET C: oc12 vc4 scramble tu11 async
SONET D: oc12 vc4 scramble tu11 async
```

Not applicable to concatenated networks

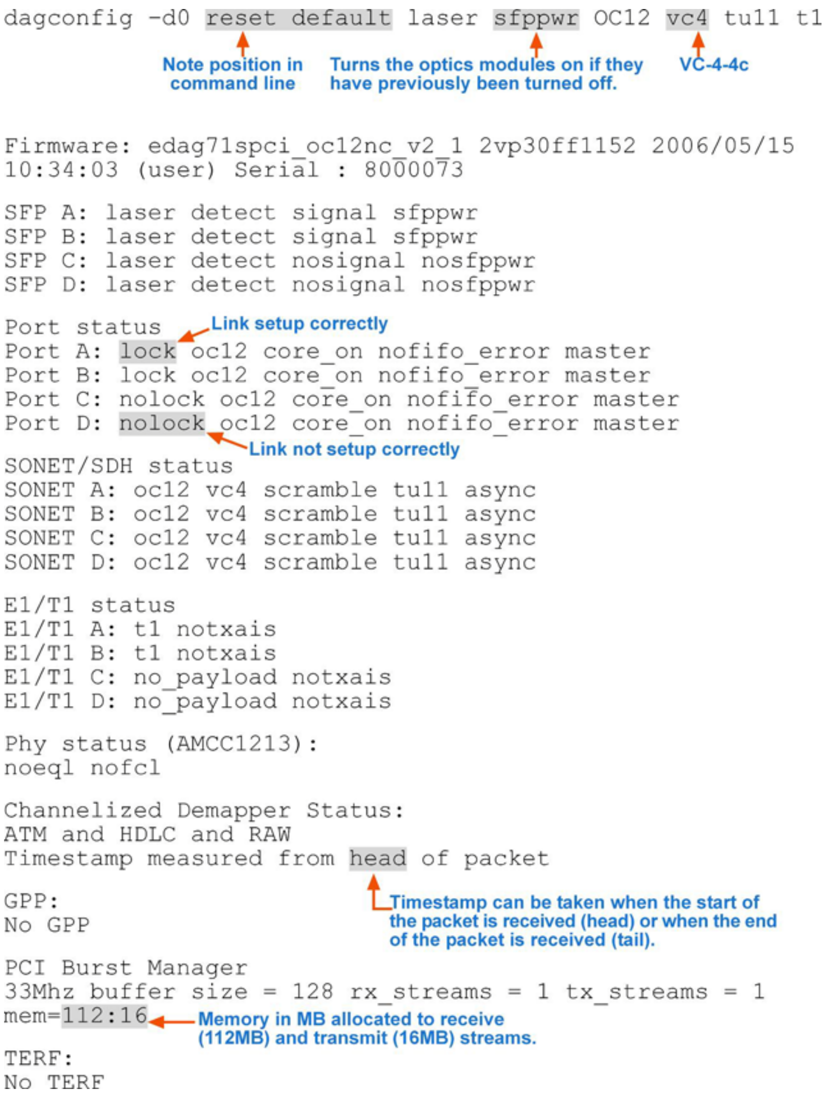
Configure Line Type

Before you configure the DAG 7.1S for specific ATM, Bit HDLC or RAW data capture you must configure the card for the correct line type and speed.

See Display Current Configuration (page 27) for a full explanation of dagconfig outputs.

Note: The order in which the configuration options appear in the command line is important. Therefore you should use `default` and `reset` before other options in the command line.

The example below has been configured for OC-12 VC-4 on ports A and B with ports C and D unused:



```
dagconfig -d0 reset default laser sfppwr OC12 vc4 tu11 t1

Firmware: edag71spci_oc12nc_v2_1 2vp30ff1152 2006/05/15
10:34:03 (user) Serial : 8000073

SFP A: laser detect signal sfppwr
SFP B: laser detect signal sfppwr
SFP C: laser detect nosignal nosfppwr
SFP D: laser detect nosignal nosfppwr

Port status
Port A: lock oc12 core_on nofifo_error master
Port B: lock oc12 core_on nofifo_error master
Port C: nolock oc12 core_on nofifo_error master
Port D: nolock oc12 core_on nofifo_error master

SONET/SDH status
SONET A: oc12 vc4 scramble tu11 async
SONET B: oc12 vc4 scramble tu11 async
SONET C: oc12 vc4 scramble tu11 async
SONET D: oc12 vc4 scramble tu11 async

E1/T1 status
E1/T1 A: t1 notxais
E1/T1 B: t1 notxais
E1/T1 C: no_payload notxais
E1/T1 D: no_payload notxais

Phy status (AMCC1213):
noeql nofcl

Channelized Demapper Status:
ATM and HDLC and RAW
Timestamp measured from head of packet

GPP:
No GPP

PCI Burst Manager
33Mhz buffer size = 128 rx_streams = 1 tx_streams = 1
mem=112:16

TERF:
No TERF
```

Note position in command line

Turns the optics modules on if they have previously been turned off.

VC-4-4c

Link setup correctly

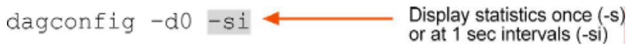
Link not setup correctly

Timestamp can be taken when the start of the packet is received (head) or when the end of the packet is received (tail).

Memory in MB allocated to receive (112MB) and transmit (16MB) streams.

Interface Statistics

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



```
dagconfig -d0 -si
```

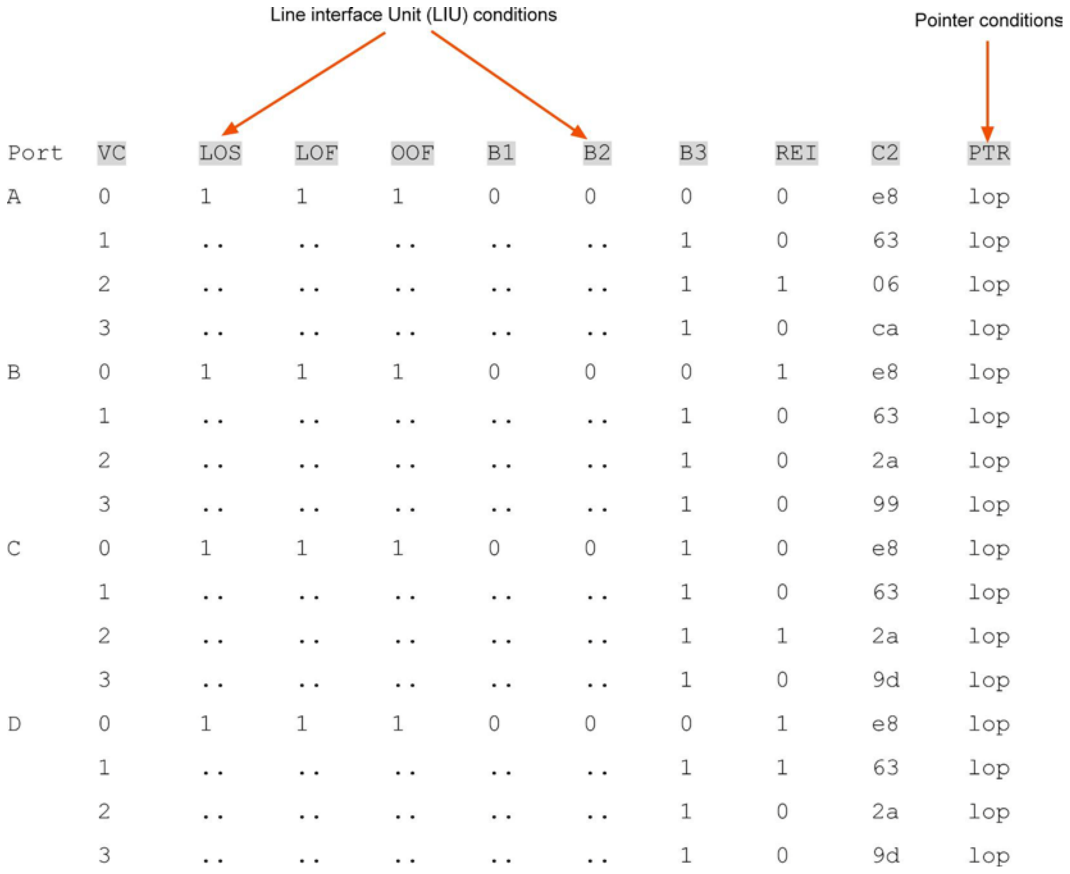
where "0" is the device number of the DAG card you wish to capture data from

Example outputs are shown below:

Note: “1” indicates the condition is present on the link “0” indicates the condition is not present on the link. See Status Conditions (page 33) later in this chapter for a full description of each of the status conditions.

E1 OC-12

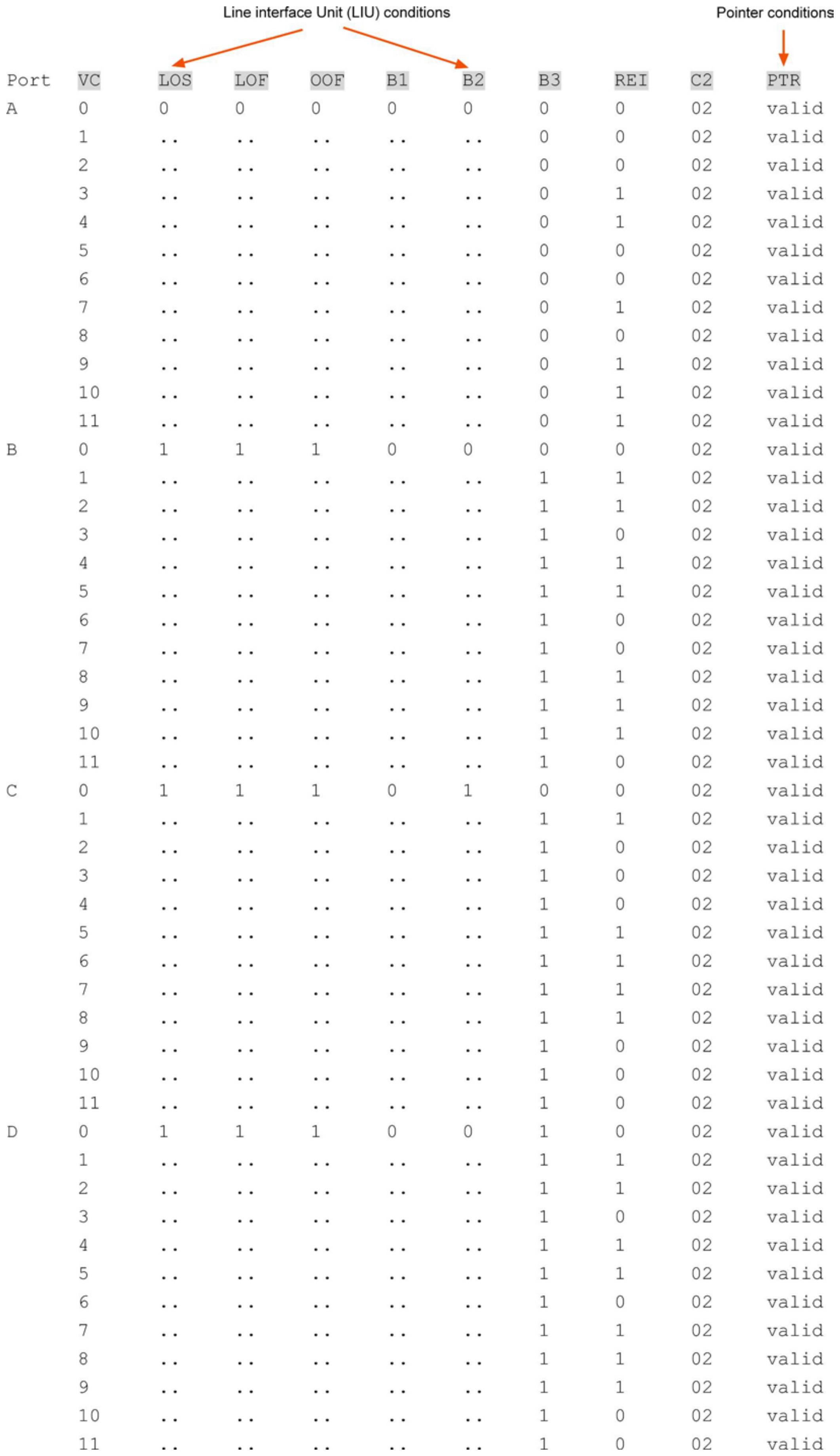
In the example below the card is set to `OC-12 vc4 tu12 e1` on all ports. The card is not receiving any traffic:



```
Port  VC   LOS  LOF  OOF  B1   B2   B3   REI  C2   PTR
A     0    1    1    1    0    0    0    0    e8   lop
      1    ..   ..   ..   ..   ..   1    0    63   lop
      2    ..   ..   ..   ..   ..   1    1    06   lop
      3    ..   ..   ..   ..   ..   1    0    ca   lop
B     0    1    1    1    0    0    0    1    e8   lop
      1    ..   ..   ..   ..   ..   1    0    63   lop
      2    ..   ..   ..   ..   ..   1    0    2a   lop
      3    ..   ..   ..   ..   ..   1    0    99   lop
C     0    1    1    1    0    0    1    0    e8   lop
      1    ..   ..   ..   ..   ..   1    0    63   lop
      2    ..   ..   ..   ..   ..   1    1    2a   lop
      3    ..   ..   ..   ..   ..   1    0    9d   lop
D     0    1    1    1    0    0    0    1    e8   lop
      1    ..   ..   ..   ..   ..   1    1    63   lop
      2    ..   ..   ..   ..   ..   1    0    2a   lop
      3    ..   ..   ..   ..   ..   1    0    9d   lop
```

T1 OC-12

In the example below the card is set to `oc12 vc3 tu11 t1` on all ports:



```
Port  VC     LOS    LOF    OOF    B1     B2     B3     REI    C2     PTR
A     0      0      0      0      0      0      0      0      02     valid
      1      ..     ..     ..     ..     ..     0      0      02     valid
      2      ..     ..     ..     ..     ..     0      0      02     valid
      3      ..     ..     ..     ..     ..     0      1      02     valid
      4      ..     ..     ..     ..     ..     0      1      02     valid
      5      ..     ..     ..     ..     ..     0      0      02     valid
      6      ..     ..     ..     ..     ..     0      0      02     valid
      7      ..     ..     ..     ..     ..     0      1      02     valid
      8      ..     ..     ..     ..     ..     0      0      02     valid
      9      ..     ..     ..     ..     ..     0      1      02     valid
      10     ..     ..     ..     ..     ..     0      1      02     valid
      11     ..     ..     ..     ..     ..     0      1      02     valid
B     0      1      1      1      0      0      0      0      02     valid
      1      ..     ..     ..     ..     ..     1      1      02     valid
      2      ..     ..     ..     ..     ..     1      1      02     valid
      3      ..     ..     ..     ..     ..     1      0      02     valid
      4      ..     ..     ..     ..     ..     1      1      02     valid
      5      ..     ..     ..     ..     ..     1      1      02     valid
      6      ..     ..     ..     ..     ..     1      0      02     valid
      7      ..     ..     ..     ..     ..     1      0      02     valid
      8      ..     ..     ..     ..     ..     1      1      02     valid
      9      ..     ..     ..     ..     ..     1      1      02     valid
      10     ..     ..     ..     ..     ..     1      1      02     valid
      11     ..     ..     ..     ..     ..     1      0      02     valid
C     0      1      1      1      0      1      0      0      02     valid
      1      ..     ..     ..     ..     ..     1      1      02     valid
      2      ..     ..     ..     ..     ..     1      0      02     valid
      3      ..     ..     ..     ..     ..     1      0      02     valid
      4      ..     ..     ..     ..     ..     1      0      02     valid
      5      ..     ..     ..     ..     ..     1      1      02     valid
      6      ..     ..     ..     ..     ..     1      1      02     valid
      7      ..     ..     ..     ..     ..     1      1      02     valid
      8      ..     ..     ..     ..     ..     1      1      02     valid
      9      ..     ..     ..     ..     ..     1      0      02     valid
      10     ..     ..     ..     ..     ..     1      0      02     valid
      11     ..     ..     ..     ..     ..     1      0      02     valid
D     0      1      1      1      0      0      1      0      02     valid
      1      ..     ..     ..     ..     ..     1      1      02     valid
      2      ..     ..     ..     ..     ..     1      1      02     valid
      3      ..     ..     ..     ..     ..     1      0      02     valid
      4      ..     ..     ..     ..     ..     1      1      02     valid
      5      ..     ..     ..     ..     ..     1      1      02     valid
      6      ..     ..     ..     ..     ..     1      0      02     valid
      7      ..     ..     ..     ..     ..     1      1      02     valid
      8      ..     ..     ..     ..     ..     1      1      02     valid
      9      ..     ..     ..     ..     ..     1      1      02     valid
      10     ..     ..     ..     ..     ..     1      0      02     valid
      11     ..     ..     ..     ..     ..     1      0      02     valid
```

Status Conditions

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

Condition	Definition
B1, B2, B3	Bit interleaved parity error as reported by SONET B1, B2 and B3 overhead octets. Indicates the connection between the card and the network is impaired. • If OOF and LOF conditions are **also** set the OCx carrier configuration is incorrect. • If OOF and LOF conditions are **not** set the there is a signal problem related to low light levels reaching the optical receivers, or there are true SONET-level errors on the equipment operating the link.
C2	Path signal label. Reflects the content of SONET C2 overhead octet. Typical settings are: • If PTR is valid – "02" • If PTR is AIS – "FF" • If PTR is conc – "13", "16" or "cf" • If PTR is lop – any value Changing values for this condition indicate a SONET level error.
LOF	Loss of Frame. Indicates Out of Frame (OOF) condition has been asserted for more than 2 milliseconds.
PTR	Indicates if the pointer processing logic has locked to the SONET stream. Possible values are: • valid: The pointer has locked to the SONET stream. • lop (loss of pointer): The pointer has not locked to the SONET stream and may indicate incorrect OC-3/OC-12 setting. • conc: The line is configured for concatenated data but the network is channelized.
LOS	Loss of Signal. There is either no signal at the receiver or the optical signal strength is too low for the card to recognize.
OOF	Out of Frame. The section overhead processor is not locked to the SONET stream and may indicate incorrect OC-3/OC-12 setting.
REI	Remote Error Indicator.
VC	Virtual container number. Should always be "0" for concatenated networks.

Verify Configuration

If the card is connected to a network and you have loaded the correct firmware for the network configuration the following values apply:

- C2 should be "02"
- PTR should be "valid"
- Remaining statistics should be "0"

You can verify the card configuration by checking settings and path label for any errors as follows:

- Ensure LOS is 0, otherwise check light levels.
- Ensure OOF and LOF are zero, otherwise change OC-3 settings to OC-12 or vice versa.
- Ensure no bit interleaved parity errors occur, otherwise check cabling and light levels.
- Ensure C2 is correct for the payload.
- Ensure ATM LCD is off and SYNC set.
- Ensure PoS scrambling and CRC settings are correct.

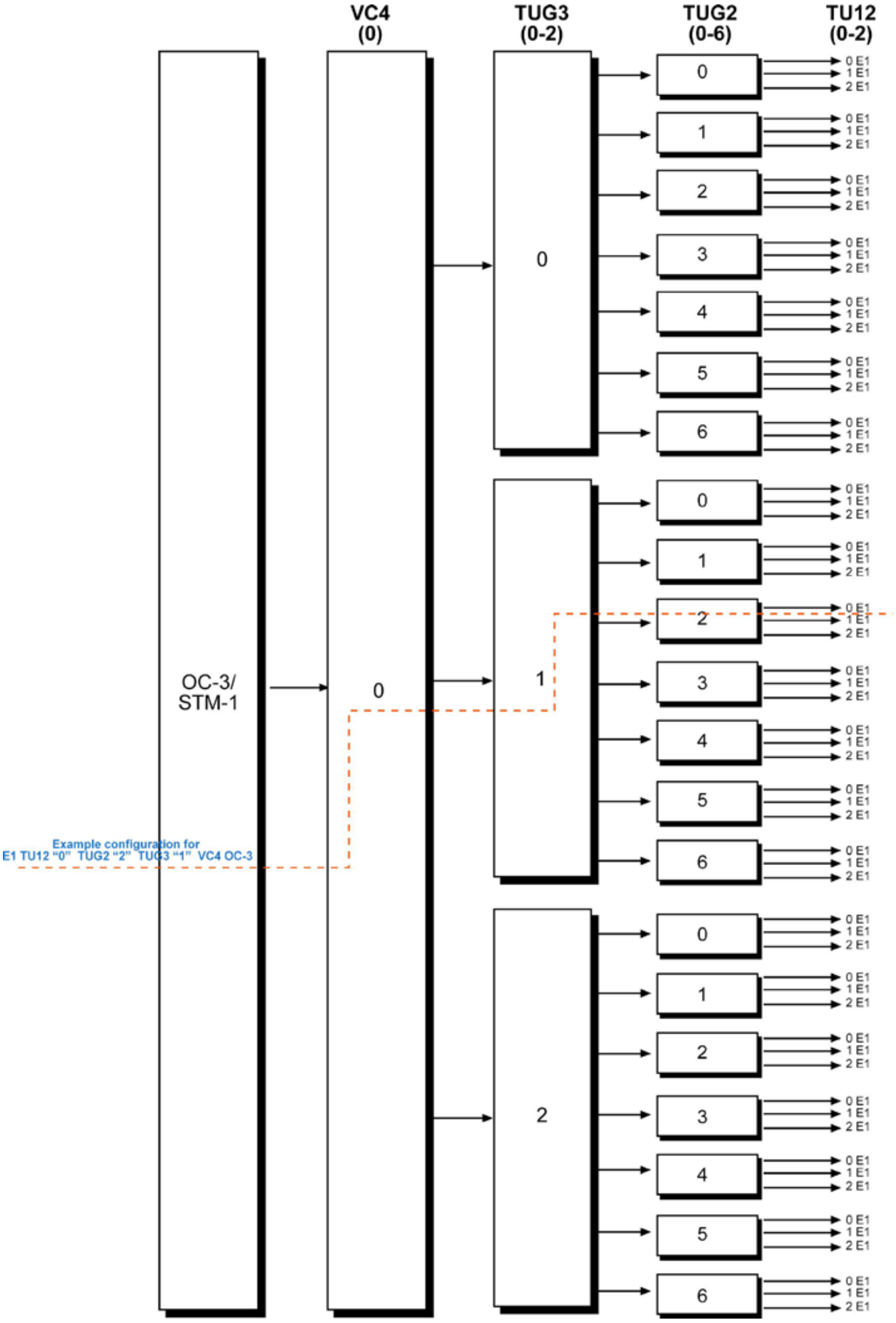
Configuring Channels

Valid Configurations

E1 OC-3/STM-1 and OC-12/STM-4

There are 63 valid channel configuration options for E1 OC-3/STM-1 and these are shown in the diagram below.

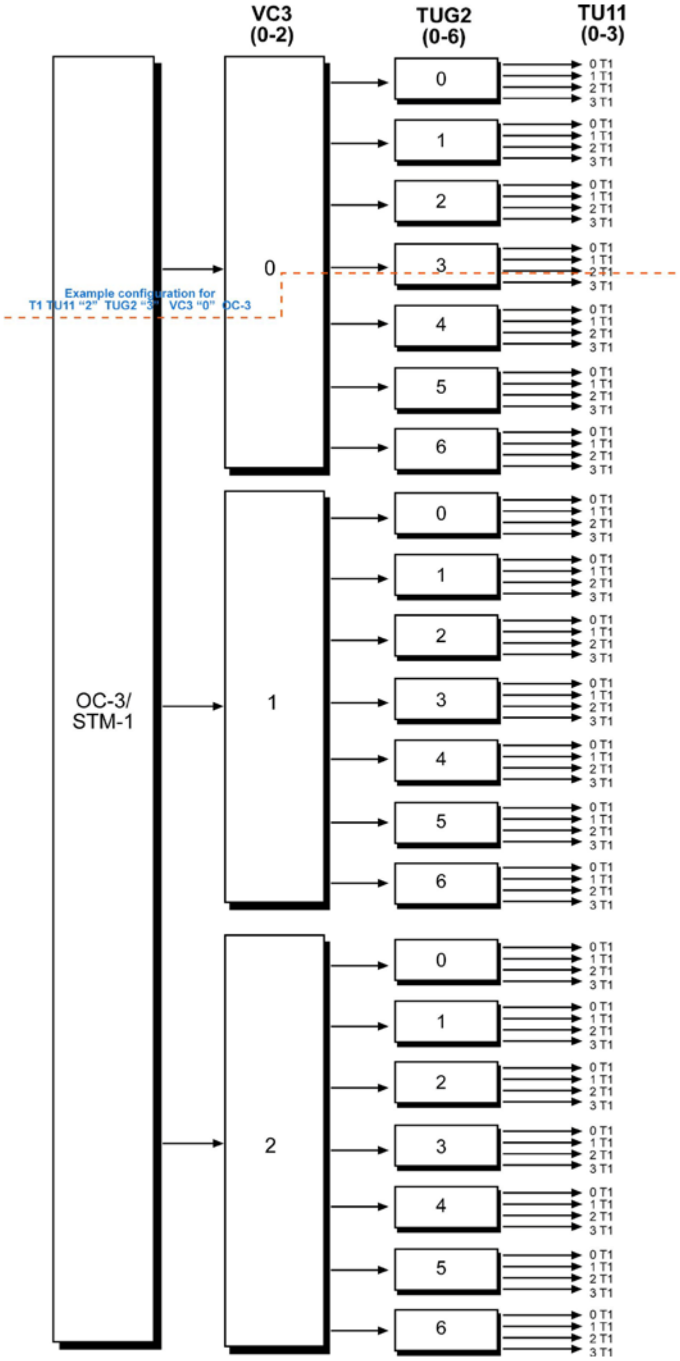
For E1 OC-12/STM-4 there are 252 (4 x 63) valid channel configuration options because OC-12/STM-4 supports 4 x VC4.



T1 OC-3/STM-1 and OC-12/STM-4

There are 84 valid channel configuration options for T1 OC-3/STM-1 and these are shown in the diagram below.

For T1 OC-12/STM-4 there are 336 (4 x 84) valid channel configuration options because T1 OC-12/STM-4 supports 12 x VC3.



Supported Channel Types

The DAG 7.1S supports capture of ATM, Bit HDLC and RAW data over the the following channel types:

Channel Type	Usage
PCM 24	E1: Timeslots `1-24` (25-31 unused) T1: Timeslots `1-24` (all available slots)
PCM 30	E1:Timeslots `1-15` and `17-31` (`0` and `16` used for framing and signalling) T1: Timeslots 1-15 and 17-24 (`16` used for signalling)
PCM 31	E1: Timeslots `1-31` (`0` is used for framing) T1: Timeslots 1-24 (all available slots)

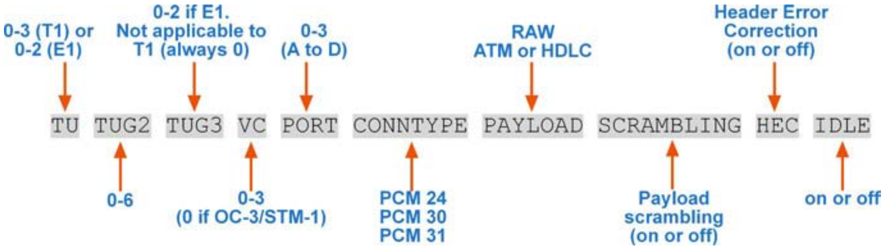
Configuration File

You can configure a specific channel type for capture of ATM, Bit HDLC or RAW data using dagconfig to read a channel configuration file and then create each of the channels defined in that file. You can create a channel configuration file using any common text editor such as Notepad, VI Editor, Emacs etc.

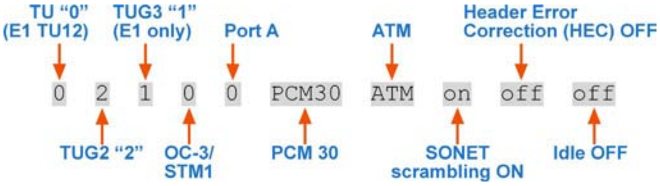
Note: The file must be a plain ASCII text file

Each line in the configuration file represents the settings for an individual channel allowing you to configure multiple channels using the one configuration file.

The file is described below:

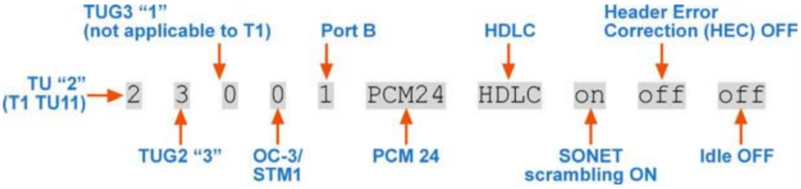


In the example line below the channel is configured for E1 channelized OC-3/STM-1, port A containing ATM over PCM-30.



This configuration is shown in the E1 OC-3/STM-1 (page 34)diagram.

In the example line below the channel is configured for T1 channelized OC-3/STM-1, port B containing HDLC over PCM-24.

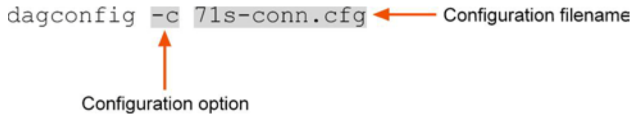


This configuration is shown in the T1 OC-3/STM-1 (page 35) diagram.

An example configuration file containing the settings for 5 channels is shown below:

```
#Channel 1:
0 0 0 0 0 PCM30 ATM on off off
#Channel 2:
1 0 0 0 0 PCM30 ATM on off off
#Channel 3:
2 0 0 0 0 PCM30 ATM on off off
#Channel 4:
0 1 0 0 0 PCM30 ATM on off off
#Channel 5:
1 1 0 0 0 PCM30 ATM on off off
```

To configure channels with the configuration file use:



```
dagconfig -c 71s-conn.cfg
```

Where `-c` is the configuration option and `71s-conn.cfg` is the configuration filename.

dagconfig tokens explained

align64

Sets the packets to be all generated as multiples of 8 bytes, 64-bit aligned, (`align64`) before being received by the host. Not a configurable option.

aidle/noaidle

When set pass through received idle cells. ATM only.

Example

```
   dagconfig aidle
   dagconfig noaidle
```

async/bit_sync/byte_sync_1/byte_sync_2

SONET allows 4 mappings, from (E1,T1) to (TU12,TU11). This flag sets which mapping the link uses.

Example

```
   dagconfig async
   dagconfig bitsync
   dagconfig bytesync1
   dagconfig bytesync2
```

Applicable to channelised networks only.

atm

Sets framer into ATM cell receive mode.

Example

```
   dagconfig atm
```

buffer_size

The buffer size=*n*MB indicates that a total of *n* MB of memory have been allocated to the DAG card in total. Memory allocation occurs when the `dagmem` driver is loaded at boot time. See *EDM04-01 DAG Software Installation Guide* for details on how to allocate memory.

core_off

Indicates the SONET framer and the firmware are disabled. The card can not capture.

crc16/crc32/nocrc

Sets the Packet over SONET (PoS) checking to None, 16 or 32 bits. SONET only.

Example

```
   dagconfig nocrc
   dagconfig crc16
   dagconfig crc32
```

detect

Indicates whether the SFP module is detected (`detect`) or undected (`nodetect`).

enable/disable

Sets whether this DAG card captures data on the defined port (a or b).

Note: DAG ports are enabled by default. You do not need to use dagconfig to enable the card in order to begin capture unless you have previously disabled it.

Example

```
dagconfig -d0 enablea
dagconfig -d0 disablea
dagconfig -d0 enableb
dagconfig -d0 disableb
```

where "0" is the device number of the DAG card you wish to capture data from

Note: On some firmware images changes to this option may not take effect.

eql/noeql

Sets or unsets equipment loopback. For testing set to `eql` mode and normal operation set to `noeql` mode.

Note: `eql` mode loops transmit data from the host back to the PCIe bus.

Example

```
dagconfig eql
dagconfig noeql
```

fcl/nofcl

Note: Sets or unsets Facility loop back. For testing set to `fcl` mode and normal operation set to `nofcl` mode.

FCL retransmits the data received and also send it to the host.

Example

```
dagconfig fcl
dagconfig nofcl
```

laser/nolaser

Enables/disables the transmit laser on for the optical transceivers.

Example

```
dagconfig laser
dagconfig nolaser
```

lock/nolock

Read only. Indicates that the card is receiving a signal at the appropriate frequency (`lock`), or there is no signal being received (`nolock`).

master/slave

Defines whether the data transmitted is clocked with the recovered clock from the input (slave) or if it uses the synthesized reference clock on the card (Master).

Setting required for SONET images only.

mem

You can split the DAG card's allocated memory between the receive and transmit stream buffers to suit your own requirements. The split is displayed as a ratio as shown below:

```
mem=X:Y
```

where:
X is the memory allocated in MB to the rx stream
Y is the memory allocated in MB to the tx stream.

If there are multiple rx or tx streams memory can be allocated to each stream:

```
mem=X:Y:X:Y:X:Y
```

Buffer_size (page 39) and rx and tx Streams (page 42) are related to mem.

Example

You can split 128MB of memory evenly between the tx and rx streams using:

```
dagconfig -d0 mem=64:64
```

Note: You can not change the stream memory allocations while packet capture or transmission is in progress.

oc3

Set framer to OC3 receive mode.

Example

```
dagconfig oc3
```

oc12

Set framer to OC12 receive mode.

Example

```
dagconfig oc12
```

overlap/nooverlap

Configures the rx and tx memory hole to be overlapped. This enables in-line forwarding without copying the data across the memory holes.

Example

```
dagconfig overlap
dagconfig nooverlap
```

Note: This option is only applicable on firmware images containing TX.

PCIe

Describes the following information about the DAG card:

- Bus speed
- buffer size

Example

```
PCI Burst Manager:
133MHz buffer_size=128
```

pscramble/nopscramble

Enables/Disables the scrambling mode on the PoS or ATM Mapper/Demapper.

Example

```
dagconfig pscramble
dagconfig nopscramble
```

nopayload/E1/T1

Indicates if the line is configured for `T1`, `E1` or not configured (`nopayload`).

rxtx

Enables both transmit and receive, and splits the memory hole for `rx` and tx.

This allocates 16MB of memory to each transmit stream, and divides the remaining memory between the receive streams.

Example

```
   dagconfig rxtx
```

Note: This option is only applicable on firmware images containing TX.

rx and tx Streams

Indicates the number of `rx` and `tx` streams are available on the DAG card. Not configurable.

Stream information relates to the setting of mem (page 41).

scramble/noscramble

Enables/Disables the scrambling mode on the SONET Framer/Deframer.

Example

```
   dagconfig scramble
   dagconfig noscramble
```

sfppwr/nosfppwr

Turns on (sfppwr) or OFF (nosfppwr) the module power for transmit.

Example

```
   dagconfig sfppwr
   dagconfig nosfppwr
```

slen

Before you begin to capture data you can set the size that you want the captured packets to be. You can do this using the `dagconfig` tool to define the packet snaplength (`slen`).

Note: The snaplength value must be a multiple of 8 and in the range 48 to 9600 per card inclusive.

By default, `slen` which is the portion of the packet that you want to capture is set to 48 per card. This means that only the first 48 bytes of each packet will be captured.

If for example you want to capture only the IP header of each packet you may want to set the length to a different value. Alternatively if you want to ensure you capture the whole packet you can set the length to a larger value.

Example

Setting up a DAG 7.1S card with a snap length of 200 bytes:

```
   dagconfig -d0 slen=200
```

Note: The ERF header is not included in the `slen` value. Therefore a `slen` of 48 will produce a 64-byte capture record made up of 48 bytes plus the number of bytes in the ERF header.

The recommended slen value is 4800.

signal/nosignal

Indicates whether the port is receiving a valid optical signal (`signal`) or not (`nosignal`).

terf_strip16/terf_strip32/noterf_strip

Strips the CRC value (16 or 32 bits) from the packet or sends packet “as is” (noterf_strip). The TERF line in the current configuration indicates the current Terf option.

Note: Only displayed if the DAG card supports transmit (i.e. has a terf image).

Example

```
dagconfig terf_strip16
dagconfig terf_strip32
dagconfig noterf_strip
```

tull

Indicates tributary unit 11 for T1 or tributary unit 12 for E1.

txais/notxais

Indicates the Alarm Indication Signal. Force transmissions (txais) or don't force transmissions (notxais).

varlen/novarlen

The DAG 7.1S card is able to capture packets in two ways. They are:

- Variable length capture (varlen)
- Fixed length capture (novarlen) - (not support on some firmware images)

In **variable length** (varlen) mode, the DAG card will capture the whole packet, providing its size is less than the slen value. Therefore to use this capture mode effectively you should set the slen value to the largest number of bytes that a captured packet is likely to contain. For more information on snaplength, see slen (page 42).

Any packet that is larger than the slen value will be truncated to that size. Any packet that is smaller than the slen value will be captured at its actual size therefore producing a shorter record which saves bandwidth and storage space.

Example

The example below shows a configuration for variable length full packet capture:

```
dagconfig -d0 varlen
```

In **fixed length** (novarlen) mode the card will capture all packets at the same length. Any packet that is longer than the slen value will be truncated to that size, in the same way as for varlen capture. However any packet that is shorter than the slen value will be captured at its full size and then padded out to the size of the slen value.

This means that in novarlen mode you should avoid large slen values because short packets arriving will produce records with a large amount of padding which wastes bandwidth and storage space.

Note: Using the novarlen option on DAG cards with an on-board Co-Processor (accelerated cards) is not recommended. It may cause excessive loss of packets.

vc3/vc 4/vc4-4c

Sets payload mapping to:

- vc3 (`vc3`)- not applicable to concatenated networks.
- vc4 (`vc4`)
- vc4-4c (`vc4c`)

Example

```
dagconfig vc3
dagconfig vc4
dagconfig vc4c
```

Version information

Details the following information about the connected DAG card:

- Firmware image programmed in the FPGA
- The DAG card serial number
- The MAC address(s) of the DAG card's ports (ethernet cards only).

dagconfig options

`dagconfig` is a software utility used to configure and display statistics.

By default all commands, unless otherwise defined, run on device 0 (`-d0`). Commands only apply to one DAG card.

The following is a list options available in `dagconfig`. Not all options listed are applicable to all cards.

Options:	Description
`-1,--porta`	Port A only (default all). Multi-port cards only.
`-2,--portb`	Port B only (default all). Multi-port cards only.
`-3,--portc`	Port C only (default all). Four-port cards only.
`-4,--portd`	Port D only (default all). Four-port cards only.
`--porte to --portp`	As above, for extra ports on the 3.7T DAG card.
`-c,--concfg <conncfg>`	Connection configuration. Used by the DAG 7.1S only.
`-C,--counters`	Outputs the counters. Verbosity levels from 0=(basic / default) to 3=(full).
`-d,--device <device>`	DAG device to use. Default is d0.
`-e,--extended`	Displays the current extended statistics (non boolean and image dependant). Verbosity levels from 0=(basic / default) to 3=(full). Note: Some images may not contain extended statistics.
`-G,--getattribute <getattribute>`	Gets individual attributes by attribute name. Use in conjunction with the --porta or --portb options to get individual only multi-port cards.
`-h,--help`	Displays the MAN pages. The information displayed is dynamically based on the DAG card and does not work correctly when there is no DAG card in the system. **Note:** There are a few commands that display even though they are not applicable.
`-i,--interval <seconds>`	Interval to repeat in seconds.
`-m,--hmon`	Outputs the hardware monitor information.
`-n,--voltages`	Outputs the DAG card voltage monitor information.
`-S,--setattribute <setattribute>`	Sets individual attributes by attribute name. Use in conjunction with the --porta or --portb options to get individual only multi-port cards.
`-s,--statistics`	Outputs the statistics for the DAG card. Verbosity levels from 0=(basic / default) to 3=(full).
`-T,--tree`	Outputs the supported Configuration and Status attributes and components with the description and name. Using the -v2 verbosity level also outputs all components and attribute codes. Verbosity levels from 0=(basic / default) to 3=(full).
`-t,--txstats`	Outputs the transmit statistics for the DAG card. Where applicable.
`-u, --ucounters`	Outputs the universal counters for the DAG card. Where applicable.
`-v,--verbose <level>`	Sets the verbosity level, from 0 (basic) to 3 (full).
`-V,--version`	Display the DAG card version information.

Note: For cards with more than 2 ports you can select the required port using: -(portnumber) or --(portletter).

Using your DAG card to capture data

Introduction

This chapter describes how to complete the following operations for a DAG card:

- Basic data capture (page 47)
- Viewing captured data (page 50)
- Converting captured data (page 52)
- Using third party applications (page 54)
- Transmitting captured data (page 54)

Basic data capture

`dagsnap` is a software utility used to write to disk the raw data captured from a DAG card.

Data is collected in Extensible Record Format (ERF) which can then be viewed using `dagbits`, or converted to other formats using `dagconvert`.

When capturing high speed data Endace recommends you use `dagsnap`, see Capturing data at high speed (page 48).

For further information on the software utilities see:

- dagsnap (page 49)
- dagbits (page 50)
- dagconvert (page 53)

Starting a capture session

To start the capture session type the following at the prompt:

```
dagsnap -d0 –v -o tracefile
```

(where "0" is the device number of the DAG card you wish to capture data from)

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` runs indefinitely. To stop, use *CTRL+C*. You can also configure `dagsnap` to run for a fixed time period then exit.

Capturing data at high speed

As the DAG 7.1S card captures packets from the network link, it writes a record for each packet into a large buffer in the host computer's main memory.

To avoid packet loss, the user application reading the record, such as dagsnap, must be able to read records out of the buffer as fast or 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 computer buffer fills, the following message displays on the computer screen:

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

The same message is also printed to log /var/log/messages file. In addition, when the computer 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.

Detecting Packet Losses

Once the buffer fills, any new packets arriving will be discarded by the DAG 7.1S 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 Data Formats (page 73) later in this User Guide for more information on the Endace ERF record format.

Increasing Buffer Size

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

For information on increasing the buffer size, see buffer_size (page 39).

dagsnap

`dagsnap` is a data capture software utility.

The following is a list of the options available in `dagsnap`.

Option	Description

`-d DEVICE`
`--device DEVICE`
	Use the DAG device referred to by DEVICE. Supported syntax includes 0, dag1, and /dev/dag3 to refer to DAG cards, and 0:2, dag1:0, and /dev/dag2:0 to refer to specific streams on cards.

`-h, -?`
`--help, --usage`
	Display usage (help) information.

`-j`	Maximize disk writing performance by only writing data to disk in chunks. This option may not be available on all operating systems.
`-m NUM`	Write at most NUM megabytes of data per call to the DAG API (default is 4 MiB).

`-o FILE`
`--fname FILE`
	Write the captured packets to FILE, in ERF format (default is standard output).

`-s NUM`
`--runtime NUM`
	Run for NUM seconds, then exit.

`-v` `--verbose`	Increase output verbosity. When `dagsnap` is run with this command three columns of data are reported every second. These columns contain 1. The cumulative total of data written out from the DAG card. 2. The buffer occupancy. Small values indicate no packet loss. 3. The rate at which data is currently being written.
`-V, --version`	Display version information.
`-w,--wait SEC`	Delay(wait) in seconds before capture and after.

Viewing captured data

Data captured in ERF format can be viewed with `dagbits`. For further details on how to use `dagbits`, see dagbits (page 50).

Note: `dagbits` decodes and displays ERF header fields and packet contents are displayed as a Hex dump only. To decode higher level protocols, Endace recommends using a third party application, see Using third party applications (page 54).

Examples

Test live traffic on dag0, stream 0 for 60 seconds running the lctr, flags and fcs tests:

```
dagbits -vvc -d0:0 -s60 lctr flags fcs
```

To read a trace log file using dagbits:

```
dagbits -vvc print -f logname.log | less
```

To check for errors in the trace:

```
dagbits -vvc ltcr flags fcs -f logname.log
```

If `dagbits` reaches the end of the traffic and prints its report then the ERF records were valid.

dagbits

`dagbits` is a software utility used to view and test ERF records. `dagbits` can receive data from either:

- directly from the DAG card (using the `-d` option), or
- a ERF data file created by `dagsnap`.

The following is a list options available in `dagbits`:

Options	Description

`-0`
`-16`
`-32`
	ERF records contain no Link-Layer CRCs.
	ERF records contain 16 bit Link-Layer CRCs (PoS).
	ERF records contain 32 bit Link-Layer CRCs (PoS and Ethernet).

`-a`	Set legacy format to ATM (this is the default).
`-b`	Treat ERF timestamps as big-endian.
`-c`	Print real-time progress reports as `dagbits` captures traffic. This is a useful indicator that a test is running correctly.
`-C NUM`	Sets CRC correction factor for BTX test (0, 16 or 32 bits).

`-d DEVICE`
`--device DEVICE`
	Use the DAG device referred to by DEVICE. Supported syntax includes 0, dag1, and /dev/dag3 to refer to DAG cards, and 0:2, dag1:1, and /dev/dag2:0 to refer to specific streams on cards.

`-D NUM`	Introduces a NUM nanosecond delay between processing each record.
`-e`	Set legacy format to Ethernet (default: ATM).
`-E NUM`	Halt operation after a maximum of NUM errors. This option prevents `dagbits` from creating extremely large output files when being redirected to a file.
`-f FILE`	Read captured data from FILE.

`-h, -?`
`--help, --usage`
	Display usage (help) information.

`-i API`	Use "API" interface for live DAG API capture. Possible options are: 0 DAG 2.4 legacy API interface [dag_offset(3)]. 1 DAG 2.5 API interface [dag_advance_stream(3)]. 2 DAG 2.5 API interface [dag_rx_stream_next_record(3)].
`-I`	Assume the ERF contains color information in the pad and offset bytes (for Ethernet ERFs) or HDLC header bytes (for PoS ERFs) and display this information as a packet classification and destination memory buffer.
`-j NUM`	Set the threshold for the jitter test to NUM microseconds.
`-m NUM`	Print the first NUM errored records only, and then continue to count errors silently for the duration of the session.

-n NUM	Expected number of packets to receive. Returns an error if the actual number is different.
-p	Set legacy format to PoS (default: ATM).
-P PARAMS	DAG 3.5S capture parameters.
-q	Quiet. This instructs dagbits to suppress summary information when terminating. Error messages are not affected by this option.
-r NUM	Set legacy format record lengths to NUM.
-R NUM	When used in conjunction with the rlen test, indicates the RLEN of ERF records to match against. NUM.
-s	Check for strictly monotonic (increasing) timestamps, rather than monotonic (non-decreasing). Affects the behavior of the mono test. With strict checking it is an error for consecutive timestamps to be equal; they must always increase.
-S NUM	Terminate dagbits after NUM seconds of capture. This option only makes sense when capturing packets from a DAG card (i.e. when used in conjunction with the -d flag).
-t NUM	Terminate dagbits if any ERF record type does not match NUM.
-U NUM	Process at most NUM records in one pass. This option enables the user to reduce the performance of dagbits for various purposes. See also -D.
-v -vv --verbose	Increase verbosity of dagbits. This option increase the amount of data displayed when printing an ERF record due to the print test or errors in other tests. -v will print payload contents, -vv will print payload contents and an accompanying ASCII dump of the packet payload.
-V, --version	Display version information.
-w	Instruct dagbits to treat all warnings as errors.
-W NUM	When used in conjunction with the wlen test, the wire length of ERF records must be exactly NUM bytes.
-z	Stop when no traffic is received for one second.

dagbits takes several options that serve as parameters to particular tests. Available tests include monotonic time-stamp increment and frame checksum (FCS, aka CRC) validation. See the dagbits help for further details.

Converting captured data

`dagconvert` is the software utility that converts captured data from ERF format to Pcap (and other formats). Once in non ERF format the data can be read using third party applications (page 54).

`dagconvert` can also be used to capture data directly into pcap format.

Examples

To read from DAG card 0 and save to a file in ERF format:

```
dagconvert -d0 -o outfile.erf
```

To read from DAG card 0 and save to a file in pcap format:

```
dagconvert -d0 -T dag:pcap -o outfile.pcap
```

To convert a file from ERF format to pcap format:

```
dagconvert -T erf:pcap -i infile.erf -o outfile.pcap
```

To convert a file from pcap format to ERF format, ensuring the ERF records are 64-bit aligned (and therefore suitable for transmission using dagflood):

```
dagconvert -T pcap:erf -A 8 -i infile.pcap -o outfile64.erf
```

To capture from DAG Card 0 using a BPF filter:

```
dagconvert -d0 -o outfile.erf -b "host 192.168.0.1 and tcp port 80"
```

To capture from DAG card 0 using ERF filtering:

```
dagconvert -d0 -o outfile.erf -f "rx,a"
```

To capture from DAG card 0 to a series of files of size 128 MB:

```
dagconvert -d0 -o outfile.erf -r 128m
```

The first file created is labeled `outfile0000.erf`, once the file size reaches 128MB, a second file is created. The second is labeled `outfile0001.erf` etc.

ATM traffic filtering (ATM, AAL5, MC_AAL5 record types) by VPI and VCI is also possible by using the SUNATM DLT, which includes VPI/VCI information. The following example shows how to use `dagconvert` to achieve this:

```
dagconvert -v -i atm.erf -o atm_filtered.erf -y SUNATM -b "vpi 10 and vci 12"
```

Dagconvert

`dagconvert` is a software utility for converting data to various file formats. Supported formats are:

File format	Description
dag	Read ERF records directly from DAG card (input only).
erf	ERF (Extensible Record Format) file (input and output).
atm	Legacy ATM files (input only).
eth	Legacy Ethernet files (input only).
pos	Legacy PoS files (input only).
null	Produces no input or output.
pcap	pcap(3) format file (input or output).
prt	ASCII text packet dump (output only).

Data can be input from a file or captured from a DAG card. `dagconvert` can be used for converting data captured from a DAG card to pcap format. This allows the trace file to be used with tools that support the pcap file format. Also the reverse is possible, where data can be converted to ERF format for use in other dag utilities. The following is a list of options available in `dagconvert`.

Options	Description
`-A NUM`	Set the record alignment of the ERF to NUM bytes (ERF only).
`-b EXPRESSION`	Specify a tcpdump(1) style BPF expression to be applied to the packets.
`-c 0|16|32`	Specify the size (in bits) of the frame checksum (FCS) (pcap(3) only).

`-d DEVICE`
`--device DEVICE`
	Use the DAG device referred to by DEVICE. Supported syntax includes 0, dag1, and /dev/dag3 to refer to DAG cards, and 0:2, dag1:1, and /dev/dag2:0 to refer to specific streams on cards.

`-f FILTERS`	A comma-delimited list of filters to be applied to the data. Supported filters are: • `rx` Filter out rx errors (link layer). • `ds` Filter out ds errors (framing). • `trunc` Filter out truncated packets. • `a,b,c,d` Filter on indicated port/interface(s).
`-F`	Select fixed length output (ERF only).
`-G NUM`	Set the GMT offset to NUM seconds (pcap(3) only).
`-h, -?, --help, --usage`	Display usage (help) information.
`-i FILES`	Name(s) of the input file(s). If more than one filename is given, the ERF records from the files will be merged in timestamp order to the output.
`-o FILE`	Name of the output file.
`-r NUM`	Rotate the output file after NUM bytes. Add `k` (kilobytes), `m` (megabytes), `g` (gigabytes) and `t` (terabytes) suffixes.
`-s NUM`	Set the snap length to NUM bytes.
`-t NUM`	Capture from the DAG card for NUM seconds.
`-T atm|dag|erf|eth |pcap|pos : erf|pcap|prt`	Input and output types. See the DESCRIPTION section above for more information about the input and output types.
`-v, --verbose`	Increase output verbosity.
`-V, --version`	Select variable length output (ERF only). Display version information.
`-y <DLT>`	This sets the pcap data link type to be used for BPF filtering (-b) and for pcap output. Previously only one DLT was mapped to each ERF type. Supported DLT types (case insensitive): EN10MB: Ethernet DOCSIS : Ethernet CHDLC : HDLC PPP_SERIAL : HDLC MTP2 : HDLC ATM_RFC1483 : ATM, AAL5 SUNATM : ATM, AAL5

Note: Not all options are applicable to all DAG cards.

Using third party applications

Once the captured data is in Pcap format you can use third party applications to examine and process the data. The third party applications include:

- Wireshark /Tshark (formerly Ethereal /Tethereal)
- TCPDump
- Libpcap
- SNORT
- Winpcap, etc.

Note: Wireshark can also read ERF formatted data.

Transmitting captured data

Configuration

The DAG 7.1S card is able to transmit as well as receive packets and can capture received traffic **while** transmitting. This allows you to use capture tools such as `dagsnap`, `dagconvert`, and `dagbits` while `dagflood` is sending packets.

To configure the DAG 7.1S card for transmission, you must allocate some memory to a transmit stream.

By default, the memory allocation is evenly split between the rx streams, with the transmit streams having no memory allocated. For information on setting the Memory allocation see Memory. (page 41)

If you wish to use the card for both transmitting and receiving, you can use the rxtx (page 42) option. This allocates 16MB of memory to each transmit stream, and divides the remaining memory between the receive streams.

Note: You can not change the stream memory allocations while packet capture or transmission is in progress.

Explicit Packet Transmission

The operating system does not recognize the DAG 7.1S card as a network interface and will not respond to ARP, ping, or router discovery protocols.

The DAG 7.1S card will only transmit packets that are explicitly provided by the user. This allows you to use the DAG 7.1S card as a simple traffic load generator.

You can also use it to retransmit previously recorded packet traces. The packet trace is transmitted as fast as possible. The packet timing of the original trace file is not reproduced.

Trace Files

You can use dagflood to transmit ERF format trace files, providing the files contain **only** ERF records of the type matching the current link configuration.

When you use DAG cards with multiple ports, ensure all ports referred to by the Trace file are active. This ensures the dagflood traffic is not blocked when trying to delivering data to an inactive port. Check the interface status output for the DAG card and ensure the link status for all required destination ports is active. See Viewing the DAG card statistics.

For further information on using dagflood please refer to the *EDM04-03 dagflood User Manual* available from Endace Customer Support at https://www.endace.com/support.

In addition the length of the ERF records to be transmitted must be a multiple of 64-bits. You can configure this when capturing packets for later transmission by setting 64-bit alignment using the dagconfig align64 command.

If packets have been captured without using the align64 option you can convert the trace files so that they can be transmitted by using dagconvert (page 53) as shown below:

```
dagconvert -v -i tracefile.erf -o tracefile.erf -A8
```

Alternatively if you are unsure if a trace file is 64-bit aligned you can test the file using dagbits (page 50) as shown below:

```
dagbits -v align64 -f tracefile.erf
```

If you do not have any ERF trace files available, you can use daggen to generate trace files containing simple traffic patterns. This allows the DAG 7.1S card to be used as a test traffic generator.

For further information on using daggen please refer to the *EDM04-06 Daggen User Guide* available from Endace Customer Support at https://www.endace.com/support.

Configuring Extended Functions

Overview

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

- Reassembly of AAL2/AAL5 frames
- IP filtering of PoS packets

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

- Load the channelized or concatenated firmware images depending upon the function you want to perform,
- Configure the card based upon your network settings,
- Check (using `dagsnap`) that you are able to capture ATM and either PoS or bit-HDLC as appropriate.

Loading the Images

To make use of the extended functionality provided by the IXP Network processor, you must load the appropriate IXP images. You can do this using the dagrom command line option tool and the region (–c) option. There are three image required. They are:

- Bootloader,

- Kernel, and

- Filesystem.

- Load the Bootloader image using:

```
dagrom -d0 -rvi -cb -f d71S_bootloader.rom
```

- Then load the kernel using:

```
dagrom -d0 -rvi -ck -f d71S_kernel.rom
```

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

  for AAL/2AAL5 reassembly on a **concatenated** network load:

```
dagrom -d0 -rvi -cf -f d71S_sar_conc_filesystem.rom
```

  for AAL2/AAL5 reassembly on a channelized network load:

```
dagrom -d0 -rvi -cf -f d71S_sar_chan_filesystem.rom
```

  for PoS IP filtering load:

```
dagrom -d0 -rvi -cf -f d71S_ipf_filesystem.rom
```

Starting the IXP

You can start the IXP using one of the following command line tools with the “`-x`” option depending upon the function you want to perform.

- `dagixp_filter_loader` for PoS IP filtering
- `dagsar_stub` for AAL2/AAL5 reassembly

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

Directing Data to the IXP

In normal circumstances when you run `dagconfig`, 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 **then** to the IXP processor, **then** to the host computer.

Both the `dagixp_filter_loader` and `dagsar_stub` tools do this by default. Alternatively you can use the "Erf-Mux" component (`kComponent ErfMux`) provided by the Endace Configuration and Status API library. For more information, please refer to the *EDM04-08 Configuration and Status API Programming Guide* available from Endace Customer Support at support@endace.com.

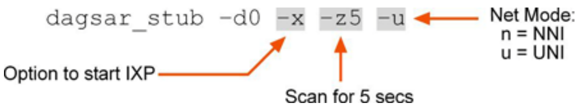
Using the AAL Reassembler

The AAL Reassembler allows you to reassemble AAL5 or AAL2 frames on the DAG 7.1S card without involving the host computer in the processing. The reassembler receives ATM traffic from the line and then sends it to the host 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.

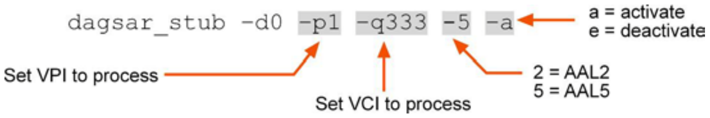
The SAR API is a library that provides the functionality to configure and read the status of the AAL Reassembler. The command line tool `dagsar_stub` provides an application layer around the SAR API and can be used to directly configure the AAL Reassembler. See dagsar_stub (page 59).

- To scan for available virtual connections on the line, use `dagsar_stub` with the "`-z`" option to specify the duration of the scan in seconds. Ensure you use the "`-x`" option to start the IXP on the **first** scan:



```
dagsar_stub -d0 -x -z5 -u
```

- To configure the ATM connection to reassemble either AAL2 or AAL5 SSSAR frames, use `dagsar_stub`. Ensure that if the IXP has not previously been started, you use the "-x" option to do so.



```
dagsar_stub -d0 -p1 -q333 -5 -a
```

- 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 -d0 -v -o output_filename
```

For more information on the specific configuration options available for the AAL5/AAL2 reassembler, please refer to the *EDM04-13 SAR API Programming Guide* available from Endace Customer Support at https://www.endace.com/support.

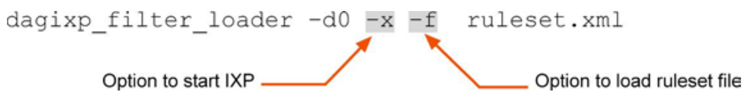
Using the PoS IP Filter

The PoS IP filter allows IPv4/IPv6 and Layer 4 filtering of PoS packets on the DAG 7.1S card without involving the host computer in processing. The filter receives PoS traffic from the line and then either sends it to the host unchanged or drops it, depending on the configuration used.

By default no filter rules are setup so all PoS traffic is sent to the host unchanged. You can setup filters using the DAGIXP Filtering API library or by using the `dagixp_filter_loader` command line tool.

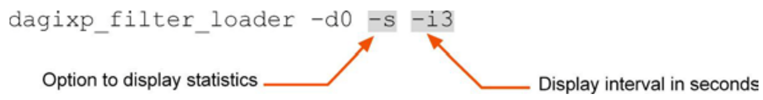
The `dagixp_filter_loader` tool accepts an XML based filter ruleset file which it then parses and loads directly into the DAG card, overwriting any existing rules.

- To load a filter ruleset into the DAG card use:



```
dagixp_filter_loader -d0 -x -f  ruleset.xml
```

- To display filtering statistics use:



```
dagixp_filter_loader -d0 -s -i3
```

- Start capturing the filtered data using:

```
dagsnap -d0 -v -o output_filename
```

For more information on the specific configuration options available for the IXP Filter, please refer to the *EDM04-11 IXP Filtering API* available from Endace Customer Support at https://www.endace.com/support.

dagsar_stub

`dagsar_sub` is a command line tool that provides an application layer around the SAR API and can be used to directly configure the AAL Reassembler.

The following is a list of available options in `dagsar_stub`:

Option	Description
`-0, --aal0`	Set SAR mode to AAL0.
`-2, --aal2`	Set SAR mode to AAL2.
`-5, --aal5`	Set SAR mode to AAL5.
`-7, --aal0tx`	Set SAR mode to AAL0 TX.
`-8, --aal2tx`	Set SAR mode to AAL2 TX.
`-9, --aal5tx`	Set SAR mode to AAL5 TX.
`-a, --activate`	Activate circuit.
`-b, --reset-filter`	Clear bitmask filter (accept all).
`-c, --channel <channel>`	Set channel to process.
`-d, --device <device>`	DAG device to use.
`-e, --deactivate`	Deactivate circuit.
`-f, --filter <bitmask.match.action>`	Set bitmask filter.
`-g, --getsar`	Get SAR mode.
`-h, --help, -?,--usage`	This page.
`-i, --cid-activate`	Activate CID.
`-j, --cid-deactivate`	Dectivate CID.
`-k, --dont-strip-trailer`	Disable AAL5 trailer stripping.
`-m, --no-erfmux`	Don't configure ERF-MUX.
`-n, --nni`	Set net mode to NNI.
`-o, --port <port>`	Set interface port to process.
`-p, --vpi <vpi>`	Set vpi to process.
`-q, --vci <vci>`	Set vci to process.
`-r, --reset-stats`	Reset statistics.
`-s, --get-stats`	Get statistics.
`-t, --strip-trailer`	Enable AAL5 trailer stripping (default).
`-u, --uni`	Set net mode to UNI.
`-v, --verbose`	Increase verbosity.
`-w, --cid <cid>`	Set cid to process.
`-x, --rst`	Reset and start the IXP.
`-z, --scan <seconds>`	Run scanning mode.

Synchronizing Clock Time

Overview

DAG cards have sophisticated time synchronization capabilities. This allows for high quality timestamps and optional synchronization to an external time standard.

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

A clock in each DAG card runs independently from the computer clock. The DAG card's clock is initialized using the computer clock, and then free-runs using a crystal oscillator.

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

DUCK Configuration

The DUCK (DAG Universal Clock Kit) 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 synchronization connector. Alternatively you can use the host computer's clock in software as a reference source without any additional hardware.

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

Common Synchronization

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

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

Endace also provides the TDS 2 Time Distribution Server modules and TDS 6 expansion 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 https://www.endace.com/support, or the *EDM05-01 Time Distribution Server User Guide*.

Network Time Protocol

NTP (Network Time Protocol) can be used to synchronize a computer clock to a network based reference. When the NTP daemon starts, it exchanges packets with network time servers to establish the correct time. If the computer clock is significantly different, the NTP can adjust the computer clock in a single large 'step'. Over time, NTP adjusts the rate of computer clock to minimize the offset from its reference. It can take several days for NTP to fully synchronize the computer clock.

The DAG card clock is initialized from the computer's clock rather than from the NTP. Using NTP to synchronize the computer's clock ensures the DAG card clock remains accurate.

DAG cards can also be synchronized to external references such as GPS or to the computer clock directly. In both cases the computer clock time is loaded onto the DAG clock when the DAG card is started (`dagload, dagreset, dagrom -p`).

When clock synchronization is enabled, the DAG card time is compared to the computer time once per second, regardless of the synchronization source. If the times differ by more than 1 second, the DAG card clock is reloaded from the computer clock and synchronization is restarted. For this reason, the computer clock should be maintained with better than 1 second accuracy.

If the DAG card clock is synchronized to the computer clock, then small 'step' adjustments of the computer clock by the NTP daemon can cause the DAG driver to emit warning messages to the console and system log files if the adjustment exceeds the warning threshold. These messages are intended to allow the user to monitor the quality of the clock synchronization over time.

The best synchronization is achieved when the DAG card is synchronized to an external GPS reference clock, and the computer clock is synchronized to a local NTP server.

Timestamps

ERF files contain a hardware generated timestamp of each packet's arrival.

The format of this timestamp is a single little-endian 64-bit fixed point number, representing the number of seconds since midnight on the 1st 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.

Different DAG cards have different actual resolutions. This is accommodated by the lower most 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. The DAG 7.1S implements the 26 most significant bits which provides a time resolution of 14.9 nanoseconds.

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.

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;
      }
```

Dagclock

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

Synchronization settings are controlled by the `dagclock` utility.

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

Note: when you run dagconfig `-d0 default` the `dagclock` inputs and outputs are also reset to defaults.

A description of each argument is shown below:

Option	Description

`-d DEVICE`
`--device DEVICE`
	Use the DAG device referred to by DEVICE. Supported syntax includes 0, dag1, and /dev/dag3 to refer to DAG cards.

`-h, -?`
`--help, --usage`
	Display the information on this page

`-k` `--sync`	Wait for DUCK synchronization before exiting
`-K NUM`	Set the synchronization timeout in seconds (default is 60 seconds)
`-l NUM`	Set the Health threshold in nanoseconds. (default is 596ns)
`-v`	Increase output verbosity
`-V`	Display version information
`-x` `--clearstats`	Clear clock statistics

Command	Description
`default`	Set the `dagclock` input and output to RS422 in and none out.
`none`	Clears the input and output settings.
`rs422in`	Sets the `dagclock` input to RS422.
`hostin`	Sets the `dagclock` input to Host (unused)
`overin`	Sets the `dagclock` input to Internal input
`auxin`	Sets the `dagclock` input to Auxiliary input (unused)
`rs422out`	Sets the `dagclock` output to repeat the RS422 input signal
`loop`	Output the selected input
`hostout`	Sets the `dagclock` output to host (unused)
`overout`	Internal output (master card)
`set`	Sets the DAG card's clock to computer clock time and clears clock statistics. The DAG card takes approximately 20 to 30 seconds to re-synchronize.
`reset`	Full clock reset. Load time from computer, set RS422in, none out. Clears clock statistics. The DAG card takes approximately 20 to 30 seconds to re-synchronize.

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

Dagclock Statistics reset

Statistics are reset to zero when the following occur:

- Loading a DAG driver
- Loading firmware
- `dagclock` with a `-x` option
- `dagclock` with a `set` or `reset` command.

Example

To view the default `dagclock` configuration:

```
dagclock -d0
```

The following is the output from DAG card that has its clock reference connected. The clock statistics have been reset since the card was last synchronized. **Note:** Values will differ for each DAG card type.

```
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 2007
host    Thu Apr 28 14:35:35 2007
dag     Thu Apr 28 14:35:35 2007
```

Note: For a description of the `dagclock` output see Dagclock output explained (page 66).

Dagclock output explained

Muxin

Lists the `dagclock` time input source for this DAG card. The options are `RS422in, Hostin, Overin or Auxin.`

Example
```
   muxin   rs422
```

Muxout

Lists the `dagclock` time output source for this DAG card. The options are `RS422out, Hostout, Overout or Loop.`

Example
```
   muxout  none
```

Status

This line reports on the status of the DAG card.

Output	Description
Synchronised / Not synchronised	This indicates whether this DAG card is synchronized to the time source listed (`Muxin`) . The DAG card becomes **Not Synchronized** when the absolute *Phase error (page 66)* is above the *Threshold* value for 10 consecutive seconds.
Threshold	This is the value above which the DAG card port is considered **Not Synchronized**. The *Threshold* value changes depending on the type of input time synchronization. The defaults are: • 596 for RS422 synchronization • 12000 for host synchronization (Unix) • 50000 for host synchronization (Windows) This value can be adjusted using the `dagclock -l` option.
Failures	This is a count of the number of times the DAG card has become **Not Synchronized.**
Resyncs	This is a count of the number of times the DAG card Phase error has exceeded 1 second. See Error (Dagclock) (page 66). If the DAG card is **Not Synchronized** for more than 10 seconds the DAG card automatically runs the following command to update the time on the DAG card: `dagclock -d0 set` Where "0" is the device number.

Example
```
   status  Synchronised Threshold 596ns Failures 0 Resyncs 0
```

Error

This line reports on the synthesized frequency of the DAG card.

Output	Description
Freq	An estimate of the synthesized frequency error over the last second in parts per billion.
Phase	The difference between the DAG card's clock and the reference clock at the last time pulse.
Worst Freq	Highest absolute value of the *Frequency error* since statistic collection began. Reset to zero when statistics are reset, see Dagclock Statistics reset (page 65).
Worst Phase	Highest absolute value of the *Phase error* since statistic collection began. Reset to zero when statistics are reset, see Dagclock Statistics reset. (page 65)

Example
```
   error   Freq -30ppb Phase -60ns Worst Freq 75ppb Worst Phase 104ns
```

Crystal

This line reports on the DAG card crystal oscillator.

Output	Description
Actual	The DAG card's crystal frequency calculated based on the reference clock.
Synthesized	The target time stamping frequency. Different for each DAG card type.

Example

```
crystal Actual 100000028Hz Synthesized 67108864Hz
```

Input

This line reports on the time pulses received by the DAG card.

Output	Description
Total	The total number of time pulses received. Reset to zero when statistics are reset, see Dagclock Statistics reset (page 65).
Bad	The number of time pulses that were rejected (considered Bad) by the DAG card. Reset to zero when statistics are reset, see Dagclock Statistics reset (page 65). Time pulses are considered *Bad* if they were not received 1 second (approximately) after the last time pulse. These may be caused by noise.
Singles missed	The number of times a single time pulse failed to be received by the DAG card (i.e. a two second gap). Reset to zero when statistics are reset, see Dagclock Statistics reset (page 65).
Longest Sequence Missed	This displays the longest time gap (in seconds) between a pair of time pulses. Reset to zero when statistics are reset, see Dagclock Statistics reset (page 65).

Example

```
input   Total 3765 Bad 0 Singles Missed 5 Longest Sequence Missed 1
```

Start / Host / DAG

Output	Description
Start	This is the time statistics collection started. See Dagclock Statistics reset. (page 65)
Host	Current Host (computer) time.
DAG	The DAG card time at the last time pulse. If the DAG card has never been synchronized, the following displays: `No active input - free running.`

Example

```
start   Thu Apr 28 13:32:45 2007
host    Thu Apr 28 14:35:35 2007
dag     Thu Apr 28 14:35:35 2007
```

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 computer’s clock must be accurate to UTC to within one second. This is used to initialize the DUCK.

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

Pulse Signal from External Source

The DAG time synchronization 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 DAG 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.

Synchronize to an external source as follows:

```
dagclock -d0
```

Output:

```
muxin   rs422
muxout  none
status  Synchronised Threshold 596ns Failures 0 Resyncs 0
error   Freq 30ppb Phase -15ns Worst Freq 238ppb Worst Phase 326ns
crystal Actual 100000023Hz Synthesized 67108864Hz
input   Total 225 Bad 0 Singles Missed 1 Longest Sequence Missed 1
start   Thu Apr 28 14:55:20 2007
host    Thu Apr 28 14:59:06 2007
dag     Thu Apr 28 14:59:06 2007
```

Connecting the Time Distribution Server

You can connect the TDS 2 module to the DAG card using DUCK crossover cable (page 72) (**Note:** A 4-pin to RJ45 adapter may be required). The TDS may be located up to 600m (2000ft) from the DAG card depending upon the quality of the cable used, possible interference sources and other environmental factors. Please refer to the *EDM05-01 Time Distribution Server 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 synchronization 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 -d0
```

`dagpps` measures the input state many times over several seconds, displaying the polarity and length of input pulse.

Single Card No Reference

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

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

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

```
dagclock –d0 none overin
```

Output

```
muxin   overin
muxout  none
status  Synchronised Threshold 11921ns Failures 0 Resyncs 0
error   Freq 1836ppb Phase 605ns Worst Freq 147ppb Worst Phase 324ns
crystal Actual 49999347Hz Synthesized 16777216Hz
input   Total 87039 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Wed Apr 27 14:27:41 2007
host    Thu Apr 28 14:38:20 2007
dag     Thu Apr 28 14:38:20 2007
```

Two Cards No Reference

Overview

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

- One DAG card can be a clock master for the second. This is useful if you want both DAG cards to be accurately synchronized with each other, but not so for absolute time of packet time-stamps, or
- One DAG card can synchronize to the host and also act as a master for the second DAG card.

Synchronizing with Each Other

Although the master DAG card's clock drifts against UTC, the DAG cards will be locked together. This is achieved by connecting the time synchronization connectors of both DAG cards using a DUCK crossover cable (page 72) (**Note:** A 4-pin to RJ45 Adapter may be required).

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

```
dagclock –d0 none overout
```

Output:

```
muxin   none
muxout  over
status  Not Synchronised Threshold 596ns Failures 0 Resyncs 0
error   Freq 0ppb Phase 0ns Worst Freq 213ppb Worst Phase 251ns
crystal Actual 100000000Hz Synthesized 67108864Hz
input   Total 0 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Thu Apr 28 14:48:34 2007
host    Thu Apr 28 14:48:34 2007
dag     No active input - Free running
```

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

Synchronizing with Host

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

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

```
dagclock –d0 none overin overout
```

Output:

```
muxin   over
muxout  over
status  Synchronised Threshold 11921ns Failures 0 Resyncs 0
error   Freq -691ppb Phase -394ns Worst Freq 147ppb Worst Phase 424ns
crystal Actual 49999354Hz Synthesized 16777216Hz
input   Total 87464 Bad 0 Singles Missed 0 Longest Sequence Missed 0
start   Wed Apr 27 14:27:41 2007
host    Thu Apr 28 14:59:14 2007
dag     Thu Apr 28 14:59:14 2007
```

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

Connector Pin-outs

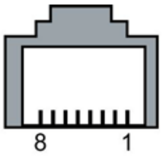
Overview

DAG cards have an 8-pin RJ45 connector for time synchronization. The connector has two bi-directional RS422 differential circuits, A and B. The Pulse Per Second (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 PPS+
2.	Out PPS-
3.	In PPS+
4.	In SERIAL+
5.	In SERIAL-
6.	In PPS-
7.	Out SERIAL+
8.	Out SERIAL-



RJ45 socket

Normally, you connect the GPS input to the PPS (A) channel input (pins 3 and 6).

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

To connect two DAG cards, use a DUCK crossover cable (page 72) to connect the two time synchronization sockets.

DUCK Crossover cable

To synchronize two DAG cards together use a cable with RJ45 plugs and the following wiring.

TX PPS+	**1**	**3**	RX PPS+
TX PPS-	**2**	**6**	RX PPS-
RX PPS+	**3**	**1**	TX PPS+
RX SERIAL+	**4**	**7**	TX SERIAL+
RX SERIAL-	**5**	**8**	TX SERIAL-
RX PPS-	**6**	**2**	TX PPS-
TX SERIAL+	**7**	**4**	RX SERIAL+
TX SERIAL-	**8**	**5**	RX SERIAL-

Note: This wiring is the same as an Ethernet crossover cable (Gigabit crossover, All four pairs crossed).

Data Formats

Overview

The DAG Card produces trace files in its 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 7.1S supports the following ERF Types:

ERF Type	Description
1	TYPE_HDLC_POS PoS with HDLC Frame Record
3	TYPE_ATM ATM Cell Record
4	TYPE_AAL5 Reassembled AAL5 Frame Record
5	TYPE_MC_HDLC: Multi-channel HDLC Frame Record
6	TYPE_MC_RAW: Multi-channel Raw time slot link data
7	TYPE_MC_ATM Multi-Channel ATM Cell Record
9	TYPE_MC-AAL5 Multi Channel Reassembled AAL5 Frame Record
12	TYPE_MC-AAL2 Multi Channel Reassembled AAL2 Frame Record
18	TYPE_AAL2 Reassembled AAL2 Frame Record

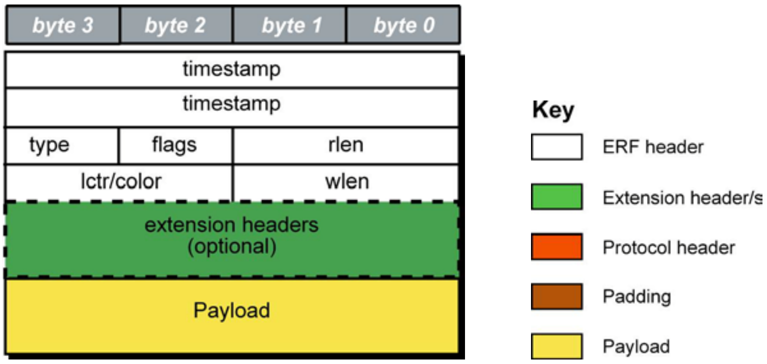
The ERF file contains a series of ERF records with each record describing one packet. ERF files consists only of ERF records, there is no file header or trailer. This allows for simple concatenation and splitting of files to be performed on ERF record boundaries.

For information on other ERF types, please refer to *EDM11-01 ERF types*.

Generic ERF 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 in network order, no byte or re-ordering is applied.

The generic ERF header is shown below:



The fields are described below:

<table>
<tr><td>timestamp</td><td></td><td>The time of arrival of the cell, an ERF 64-bit timestamp.</td></tr>
<tr><td rowspan="2">type</td><td>Bit 7</td><td>Extension header present.</td></tr>
<tr><td>Bit 6:0</td><td>Extension header type. See table below:</td></tr>
<tr><td>flags</td><td></td><td>This byte is divided into several fields as follows:
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>1-0:</td><td>Binary enumeration of capture interface:
11 Interface 3 or D
10 Interface 2 or C
01 Interface 1 or B
00 Interface 0 or A
Cards with more than four interfaces typically use Multichannel ERF types (type 5 to 9, 12 and 17) which provide a separate larger interface field.</td></tr>
<tr><td>2:</td><td>Varying length record. When set, packets shorter than the snap length are not padded and rlen resembles wlen.
When clear, longer packets are snapped off at snap length and shorter packets are padded up to the snap length. rlen resembles snap length. Setting novarlen and slen greater than 256 bytes is wasteful of bandwidth</td></tr>
<tr><td>3:</td><td>Truncated record - insufficient buffer space.
• wlen is still correct for the packet on the wire.
• rlen is still correct for the resulting record. But, rlen is shorter than expected from snap length or wlen values.
<b>Note:</b> truncation is depreciated and this bit is unlikely to be set in an ERF record.</td></tr>
<tr><td>4:</td><td>RX error. An error in the received data. Present on the wire</td></tr>
<tr><td>5:</td><td>DS error. An internal error generated inside the card annotator. Not present on the wire.</td></tr>
<tr><td>6:</td><td>Reserved</td></tr>
<tr><td>7:</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>rlen</td><td></td><td>Record length in bytes. Total length of the record transferred over the PCIe bus to storage.
The timestamp of the next ERF record starts exactly rlen bytes after the start of the timestamp of the current ERF record.</td></tr>
<tr><td>lctr</td><td></td><td>Depending upon the ERF type this 16 bit field is either a loss counter of color field. The loss counter records the number of packets lost between the DAG card and the stream buffer due to overloading on the PCIe bus. The loss is recorded between the current record and the previous record captured on the same stream/interface. The color field is explained under the appropriate type details.</td></tr>
</table>

wlen		Wire length. Packet length "on the wire" including some protocol overhead. The exact interpretation of this quantity depends on physical medium. This may contain padding.
extension headers		Extension headers in an ERF record allow extra data relating to each packet to be transported to the host. Extension header/s are present if bit 7 of the type field is '1'. If bit 7 is '0', no extension headers are present (ensures backwards compatibility). **Note:** There can be more than one Extension header attached to a ERF record.
Payload		Payload is the actual data in the record. It can be calculated by either : • Payload = rlen - ERF header - Extension headers (optional) - Protocol header - Padding

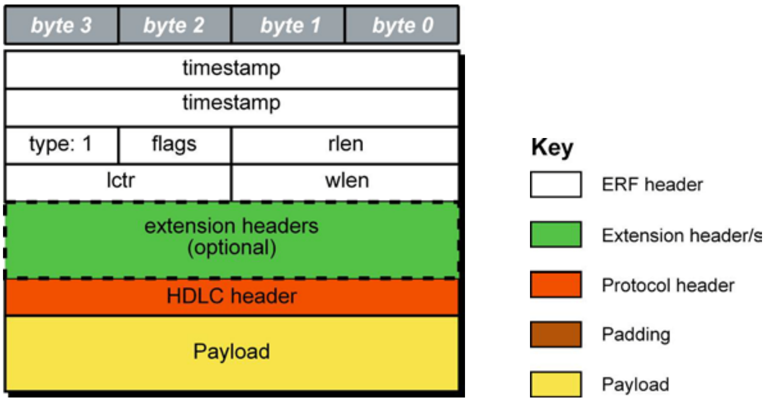
Extension header types

Number	Type	Description
0:	TYPE_LEGACY	Old style record
1:	TYPE_HDLC_POS	Packet over SONET / SDH frames, using either PPP or CISCO 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 time slot link data
7:	TYPE_MC_ATM	Multi-channel ATM Cell
8:	TYPE_MC_RAW_ CHANNEL	Multi-channel Raw link data
9:	TYPE_MC_AAL5	Multi-channel AAL5 frame
10:	TYPE_COLOR_HDLC_ POS	HDLC format like TYPE_HDLC_POS, but with the LCNTR field reassigned as COLOR
11:	TYPE_COLOR_ETH	Ethernet format like TYPE_ETH, but with the LCNTR field reassigned as COLOR
12:	TYPE_MC_AAL2	Multi-channel AAL2 frame
13:	TYPE_IP_COUNTER	IP Counter ERF Record
14:	TYPE_TCP_FLOW_ COUNTER	TCP Flow Counter ERF Record
15:	TYPE_DSM_COLOR_ HDLC_POS	HDLC format like TYPE_HDLC_POS, but with the LCNTR field reassigned as DSM COLOR
16:	TYPE_DSM_COLOR_ ETH	Ethernet format like TYPE_ETH, but with the LCNTR field reassigned as DSM COLOR
17:	TYPE_COLOR_MC_ HDLC_POS	Multi-channel HDLC like TYPE_MC_HDLC, but with the LCNTR field reassigned as COLOUR
18:	TYPE_AAL2	Reassembled AAL2 Frame Record
19:	TYPE_COLOR_HASH_ POS	Colored PoS HDLC record with Hash load balancing
20:	TYPE_COLOR_HASH_ ETH	Colored Ethernet variable length record with Hash load balancing
21:	TYPE_INFINIBAND	Infiniband Variable Length Record
22:	TYPE_IPV4	IPV4 Variable Length Record
23:	TYPE_IPV6	IPV6 Variable Length Record
24	TYPE_RAW_LINK	Raw link data, typically SONET or SDH Frame
32-47:	-	Reserved for CoProcess Development Kit (CDK) Users and Internal use
48:	TYPE_PAD	Pad Record type

ERF 1. TYPE_POS_HDLC

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 1
Short description	TYPE_POS_HDLC	
Long description	Type 1 PoS HDLC Record	
Use	This record format is for HDLC data links. For example: • Packet over SONET • Point-to-Point Protocol [PPP] over SONET • Frame Relay • MTP2 (SS7)	

The *TYPE_POS HDLC* record is shown below:



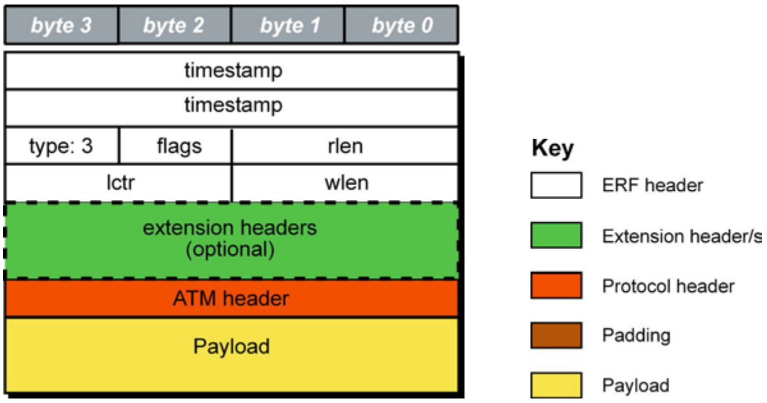
The following is a description of the *TYPE_POS_HDLC* record format:

Field	Description
HDLC Header (4 bytes)	Protocol Header. Length may vary depending on protocol, typically 4 bytes.
Payload (bytes of record)	Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (4 bytes)

ERF 3. TYPE_ATM

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 3
Short description	TYPE_ATM	
Long description	Type 3 ATM Cell Record	
Use	This record format is for ATM cell capture.	

The *TYPE_ATM* record is shown below:



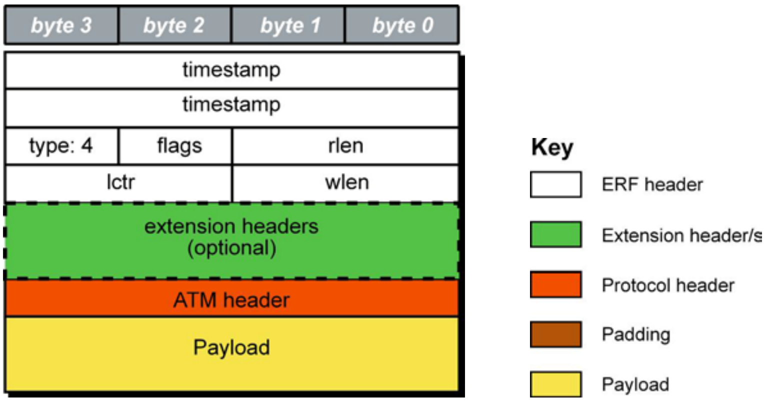
The following is a description of the *TYPE_ATM* record format:

Field	Description
ATM Header (4 bytes)	Protocol header. Does not include the 8-bit HEC.
Flags (1 byte)	ATM cells should not have the variable length flag set.
Payload (bytes of cell)	Payload = 48 bytes of cell + HEC (1 byte)

ERF 4. TYPE_AAL5

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type
Short description	TYPE_AAL5	
Long description	Type 4 Reassembled AAL5 Frame Record	
Use	This record format is for reassembled ATM AAL5 frames.	

The *TYPE_AAL5* record is shown below:



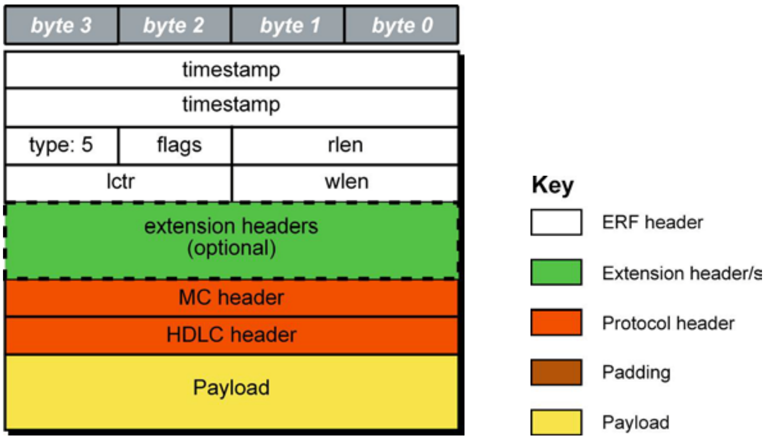
The following is a description of the *TYPE_AAL5* record format:

Field	Description
ATM header (4 bytes)	Protocol header of first cell in the frame not including the 8-bit HEC, all other cells in fame must have identical headers so are not included.
Payload (4 bytes)	Payload contains all cells in the frame: • trailing padding (0 - 47 bytes) • 1 byte `cpcs-un` field • 1 byte `cpi` field • 2 byte `length` filed, and • 4 byte `crc` field
Flags (1 byte)	The rx error flag in the ERF haders is set should the AAL5 crc fail.
Payload (bytes of AAL5 frame)	Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (4 bytes)

ERF 5. TYPE_MC_HDLC

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 5
Short description	TYPE_MC_HDLC	
Long description	Type 5 Multi-channel HDLC Frame Record	
Use	This record format is for channelized HDLC data links. For example E1, T1 and J1.	

The *TYPE_MC_HDLC* record is shown below:



The following is a description of the *TYPE_MC_HDLC* record format:

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>flags
(1 byte)</td><td>This field is the same as normal ERF types but capture interface is always zero.
• Fixed length mode not supported.
• RX Error is set if any MC Header Error bit is set.</td></tr>
<tr><td>MC header
(4 bytes)</td><td>Protocol Header. This field is divided into the following:
<table>
<tr><th>Bits</th><th>Attribute</th></tr>
<tr><td>0-9</td><td>Connection Number [0-1023].</td></tr>
<tr><td>10-15</td><td>Reserved.</td></tr>
<tr><td>16-23</td><td>Reserved.</td></tr>
<tr><td>24</td><td>FCS Error.</td></tr>
<tr><td>25</td><td>Short Record Error [<5 Bytes].</td></tr>
<tr><td>26</td><td>Long Record Error [>2047 Bytes].</td></tr>
<tr><td>27</td><td>Aborted Frame Error.</td></tr>
<tr><td>28</td><td>Octet Error. The closing flag was not octet aligned after bit stuffing.</td></tr>
<tr><td>29</td><td>Lost Byte Error. The internal data path had an unrecoverable error.</td></tr>
<tr><td>30</td><td>1ST Rec. This is the first record received since this connection was configured.</td></tr>
<tr><td>31</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>HDLC header
(4 bytes)</td><td>Protocol header. Length may vary depending on protocol.</td></tr>
<tr><td>Payload
(bytes of packet)</td><td>Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (8 bytes)</td></tr>
</table>

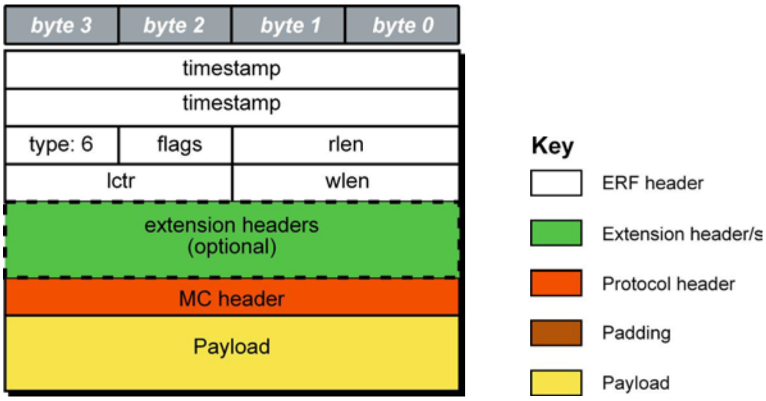
Note: When using this record type with the DAG 3.7T card the Interface number is 0, and the connection number is defined by the programmed context.

When using this record type with the DAG 7.1S card the interface number is used for the four ports, and the connection number is the VC identifier, as defined in the *EDM01-17 DAG 7.1S Card User Guide*.

ERF 6. TYPE_MC_RAW

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 6
Short description	TYPE_MC_RAW	
Long description	Type 6 Multi-Channel RAW Time Slot Link Data Record	
Use	This record format is for the RAW capture from data links. For example; E1, T1 and J1.	

The *TYPE_MC_RAW* record is below:



The following is a description of the *TYPE_MC_RAW* record format:

Field	Description
Flags (1 byte)	This field is the same as normal ERF types but capture interface is always zero. • Fixed length mode not supported. • RX Error is set if any MC Header Error bit is set.
MC header (4 bytes)	Protocol header. This field is divided into the following:

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 Byte. The internal datapath had an unrecoverable error.
30:	1st Rec. This is the first record received since this connection was configured.
31:	Reserved.

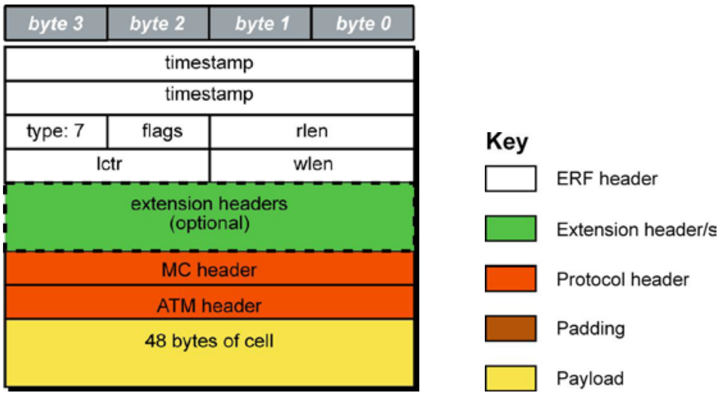
Field	Description
Payload (bytes of raw link data)	Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (4 bytes) This field is divided into the following:

Data type	Description
T1:	24 bytes for 24 time slots.
E1:	31 bytes for time slots 0-31. Slot 16 is signaling information.
Framed E1:	30 bytes of data for time slots 1-31, slot 0 used for framing is not captured. Slot 16 is signaling information.

ERF 7. TYPE_MC_ATM

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 7
Short description	TYPE_MC_ATM	
Long description	Type 7 Multi-channel ATM Cell Record	
Use	This record format is for ATM cells on channelized data links. For example; E1, T1 and J1.	

The *TYPE_MC_ATM* record is shown below:



The following is a description of the *TYPE_MC_ATM* record format:

Field	Description
flags (1 byte)	This field is the same as normal ERF types but capture interface is always zero. • Fixed length mode not supported. • RX Error is set if any MC Header Error bit is set.
MC header (4 bytes)	Protocol header. This field is divided into the following:

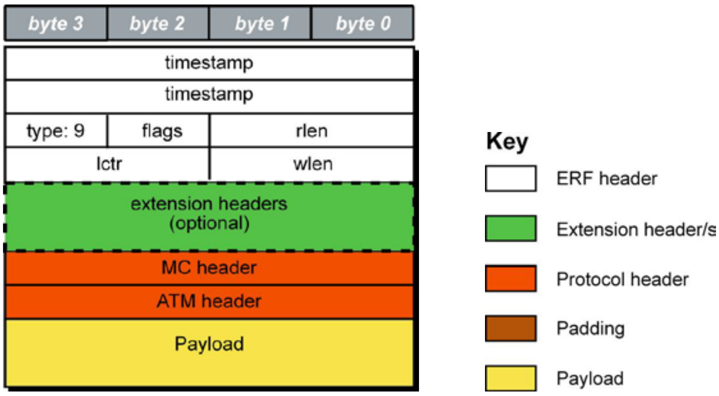
Bit	Description
0-9:	Connection number (0-1023). 512 connections are supported by DAG 3.7T card. For the DAG 7.1S card refer to *EDM01-17 DAG 7.1S Card User Guide* for details. Refer to the Channelized Configuration > Configuration File (page 36).
10-14:	Reserved.
15:	Multiplexed from IMA group into ATM stream. When bit 15 of the MC Header is set the bottom 9 bits (Connection Number/IMA ID) shall be treated as an IMA Group ID instead of a connection number.
16-19:	Physical port [0-15] cell was captured on. Physical ID is interpreted from the firmware perspective. For example, if a cable is plugged into port 0, examining the ERF MC Header field will give a Physical ID of 11. This is a little counter-intuitive and reflects the internal processing required. From the software/user perspective, this could be interpreted as the Logical ID, and as such, we can convert from the Logical to Physical ID using the provided `dagutil` function, `dagutil_37t_line_get_logical` which will return the Software Physical ID/Firmware Logical ID. In this case, assuming data is coming in on a cable plugged into port 0, we will convert 11 back to 0.
20-23:	Reserved.
24:	Lost Byte. The internal datapath had an unrecoverable error.
25:	HEC corrected.
26:	OAM Cell CRC-10 Error [not implemented].
27:	OAM Cell.
28:	1st Cell. This is the first cell received since this connection was configured.
29-31:	Reserved.

Field	Description
ATM header (4 bytes)	Protocol header. The ATM HEC channel is not captured. This record has a fixed length of 72 bytes. This does not include the 8-bit HEC.
Payload (bytes of cell)	Payload = 48 bytes of cell - HEC (1 byte)

ERF 9. TYPE_MC_AAL5

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 9
Short description	TYPE_MC_AAL5	
Long description	Type 9 Multi-channel AAL5: Multi-channel AAL5 Frame Record	
Use	This record format for reassembled ATM AAL5 frames from channelized data links. For example; E1, T1, J1.	

The *TYPE_MC_AAL5* record is shown below:



The following is a description of the *TYPE_MC_AAL5* record format:

Field	Description
flags (1 byte)	This field is the same as normal ERF types but capture interface is always zero. • Fixed length mode not supported. • RX Error is set if any MC. Header Error bit is set.
wlen (2 bytes)	This contains the length of the AAL5 frame including the ATM Header but not including the ERF Header. The ERF record will always be 64 bit aligned, if the AAL5 frame is not 64 bit aligned the record will be padded at the end of the record with the value 0x00. This padding will not be included in the `wlen` count.
MC header (4 bytes)	Protocol Header. This field is divided into the following:
ATM header (4 bytes)	Protocol Header. This does not include the 8-bit HEC.
Payload (bytes of AAL5 frame)	Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (8 bytes)

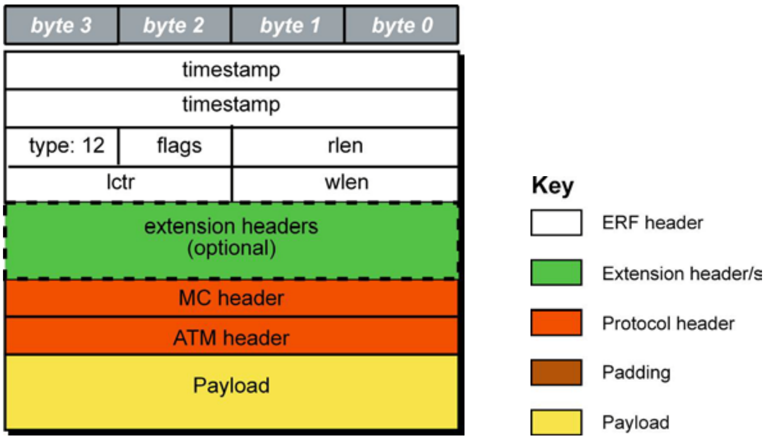
MC header (4 bytes) – Protocol Header. This field is divided into the following:

Bits	Attributes
0-10:	Connection number (0-2047). 512 connections are supported by DAG 3.7T card.
11-15:	Reserved.
16-19:	Physical port (0-15) cell was captured on. Physical ID is interpreted from the firmware perspective. For example, if a cable is plugged into port 0, examining the ERF MC Header field will give a Physical ID of 11. This is a little counter-intuitive and reflects the internal processing required. From the software/user perspective, this could be interpreted as the Logical ID, and as such, we can convert from the Logical to Physical ID using the provided `dagutil` function, `dagutil_37t_line_get_logical` which will return the Software Physical ID/Firmware Logical ID. In this case, assuming data is coming in on a cable plugged into port 0, we will convert 11 back to 0. For the 7.1S this field is always 0.
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.
29-31:	Reserved.

ERF 12. TYPE_MC_AAL2

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 12
Short description	TYPE_MC_AAL2	
Long description	Type 12 Multi-channel AAL25: Multi-channel AAL2 Frame Record	
Use	This record format is for channelized links is the same as the normal ERF Types but capture interface is always zero.	

The *TYPE_MC_AAL2* record is shown below:



The following is a description of the *TYPE_MC_AAL2* record format:

Field	Description
flags (1 byte1)	This field is the same as normal ERF types but capture interface is always zero. • Fixed length mode not supported. • RX Error is set if any MC Header Error bit is set.
MC header (4 bytes)	Protocol header. This field is divided into the following:

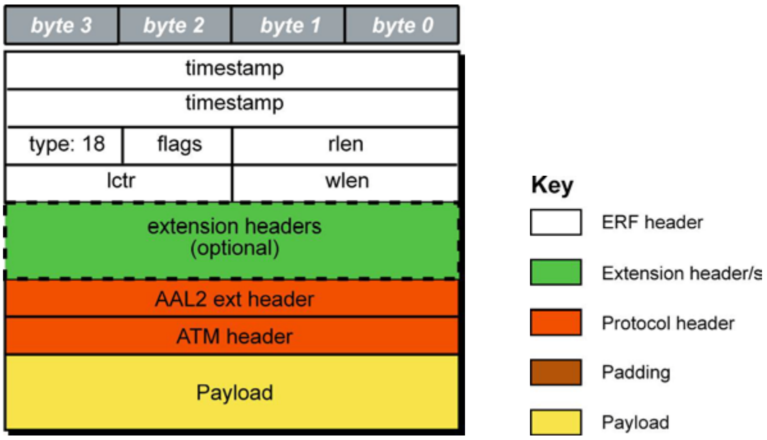
Bits	Attribute
0-9	Connection number (0-1023). 512 connections are supported by DAG 3.7T card.
10-12	Reserved for possible extra connection numbers
13-15	Reserved for indication of AAL2 type (a value of 0x0 indicates a SSSAR packet).
16-19	Physical port (0-15) cell was captured on. Physical ID is interpreted from the firmware perspective. For example, if a cable is plugged into port 0, examining the ERF MC Header field will give a Physical ID of 11. This is a little counter-intuitive and reflects the internal processing required. From the software/user perspective, this could be interpreted as the Logical ID, and as such, we can convert from the Logical to Physical ID using the provided `dagutil` function, `dagutil_37t_line_get_logical` which will return the Software Physical ID/Firmware Logical ID. In this case, assuming data is coming in on a cable plugged into port 0, we will convert 11 back to 0. For the 7.1S this field is always 0.
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)

Field	Description
ATM header (4 bytes)	Protocol header. This does not include the 8-bit HEC.
Payload (bytes of AAL5 frame)	Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (8 bytes)

ERF 18. TYPE_AAL2

Type	Bit 7	1 = Extension header present. See Extension Headers (page 85).
	Bits 6:0	Type 18
Short description	TYPE_AAL2	
Long description	Type 18 Reassembled AAL2 Frame Record	
Use	This record is for reassembled ATM AAL2 frames.	

The *TYPE_AAL2* record is shown below:



The following is a description of the *TYPE_AAL2* record format:

Field	Description
flags (1 byte)	This field is divided into the following: **Bit** — **Description** 0 — MAAL Error Indication, will be set if the frame has a MAAL error otherwise it is cleared. 1 — 1st Frame Indicator, will be set if this is the first frame reassembed on the Interface/Channel/VPI/VCI/CID. 2-7 — Reserved
AAL2 ext header (4 bytes)	Protocol Header. This field is divided into the following: **Field** — **Description** 0-7 — Channel Identification Number (cid) 8-15 — MAAL Error (errnum as specified in ITU I.363.2 is copied to the data part of this record) 16-23 — AAL2 flags, see above. 24-31 — Reserved
ATM header (4 bytes)	Protocol Header. This does not include the 8-bit HEC.
Payload (bytes of AAL2 frame)	Payload = rlen - ERF header (16 bytes) - Extension headers (optional) - Protocol header (8 bytes)

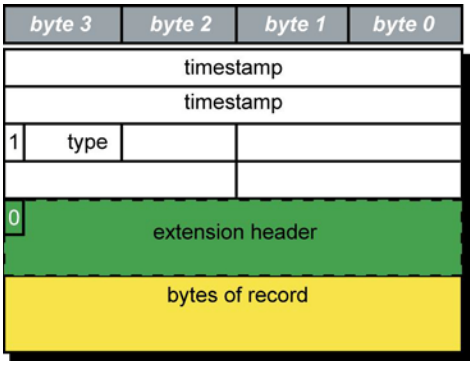
Extension Headers (EH)

Introduction

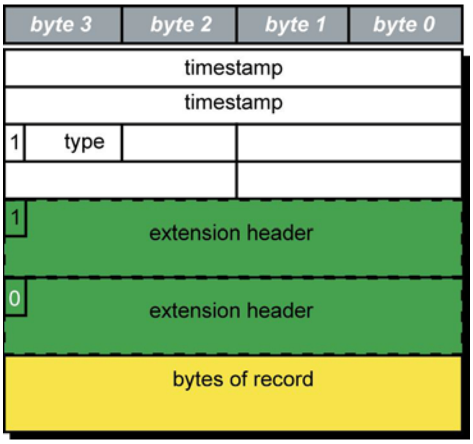
The addition of an Extension Header into the ERF record allows extra data relating to the packet to be transported to the host. The extension header allows certain features to be added independently of ERF types, for example, features shared by different ERF records do not have to be implemented separately. This results in automatic support across ERF types.

Bit 7 of the ERF type field is used to indicate that Extension Header's are present. If set to '1' Extension Headers are present. The Extension Header type field indicates the type and format of the Extension Header. It also indicates whether further Extension Headers are present. If bit 7 of the Extension Header is set to '1' further Extension Headers exist in the record. The Extension Headers are 8 bytes in length.

The following diagram shows presence of an Extension Header in addition to the ERF record.



The following diagram shows presence of two Extension Headers with Bit 7 of the first Extension Header set to '1'.



Reporting Problems

If you have problems with a DAG 7.1S 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.

The following items may assist in a quick resolution:

- DAG 7.1S card[s] model and serial number.
- Host computer type and configuration.
- Host computer 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: dagconfig
- Link statistics reported by: dagconfig `-si`
- Statistics either (depending on the DAG card):
  - Extended statistics reported by: dagconfig `-ei`
  - Universal statistics reported by: dagconfig `-ui`
- 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.
- If you have just rebooted and the system can not see any DAG cards, you need to load the DAG drivers. Run `dagload`.

Version History

The version history for this user guide is shown below.

Version	Date	Reason
1-2	-	Early Releases
3	August 2006	General review of content. Major layout and formatting changes. Addition of channelized configuration
4	November 2007	Changes to dagconfig section. General revision.
5	December 2007	Added External Power supply information.
6	January 2008	Added power supply cable graphics, dagld information.
7	June 2008	Updated dagconfig token list. Added card features section. Added ERF Extension header information.
8	November 2008	Updated Buffer_size and mem dagconfig tokens and associated cross references. Updated front matter. Update dagconfig options table. Added new dagrom options. Supported OS information now in release notes. Updated external power supply information. Added card description to the overview.

Status	Description
Preliminary	The products described in this technical document are in development and have yet to complete final production quality assurance.
Released	The products described in this technical document have completed development and final production quality assurance.