

Working Group: Physical and Link Layer (PLL) Working Group

TITLE: Tunable Laser Implementation Agreement

SOURCE:**Eric Selvik**

Editor; Communications Work Group
lolon
1870 Lundy Ave
San Jose, CA 95131
Phone: +1 408 952 6957
Email:

Karl Gass

Working Group Vice Chair
Sandia National Laboratories
PO Box 5800, MS 0874
Albuquerque, NM 87185
USA
Phone: +1 505 844-8849
Email: kgass@sandia.gov

Jose Downes

Mechanical Work Group
Nortel Networks
299 Ballardvale St
Wilmington, MA 01887
Phone: +1 578 970 1242
Email: jdownes@nortelnetworks.com

Mike Lerer

Working Group Chair
Avici Systems Inc
101 Billerica Avenue
North Billerica, MA 01862-1256
Phone: +1 978 964 2058
Email:

DATE:**November 27, 2002**

Project Name: Tunable Laser Implementation Agreement

Abstract: Implementation Agreement for tunable lasers and other optical internetworking devices. It specifically addresses a common software protocol, control syntax, and physical (electrical) interfaces to be used with implementations for various tunable devices.

Notice: This Technical Document has been created by the Optical Internetworking Forum (OIF). This document is offered to the OIF Membership solely as a basis for agreement and is not a binding proposal on the companies listed as resources above. The OIF reserves the rights to at any time to add, amend, or withdraw statements contained herein. Nothing in this document is in any way binding on the OIF or any of its members.

The user's attention is called to the possibility that implementation of the OIF implementation agreement contained herein may require the use of inventions covered by the patent rights held by third parties. By publication of this OIF implementation agreement, the OIF makes no representation or warranty whatsoever, whether expressed or implied, that implementation of the specification will not infringe any third party rights, nor does the OIF make any representation or warranty whatsoever, whether expressed or implied, with respect to any claim that has been or may be asserted by any third party, the validity of any patent rights related to any such claim, or the extent to which a license to use any such rights may or may not be available or the terms hereof.

For additional information contact:

The Optical Internetworking Forum, 39355 California Street,
Suite 307, Fremont, CA 94538

510-608-5928 phone ♦ info@oiforum.com

Copyright (C) The Optical Internetworking Forum (OIF) (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction other than the following, (1) the above copyright notice and this paragraph must be included on all such copies and derivative works, and (2) this document itself may not be modified in any way, such as by removing the copyright notice or references to the OIF, except as needed for the purpose of developing OIF Implementation Agreements.

By downloading, copying, or using this document in any manner, the user consents to the terms and conditions of this notice. Unless the terms and conditions of this notice are breached by the user, the limited permissions granted above are perpetual and will not be revoked by the OIF or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE OIF DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY, TITLE OR FITNESS FOR A PARTICULAR PURPOSE.

Primary Authors:

Jeff Hutchins	iolon
Jose Downes	Nortel
Eric Selvik	iolon

List of Contributors:

Name:	Company:
John Marchionda	ADC
Mike Pepler	Agere Systems
Sean Hannam	Agere Systems
James Moffat	Agere Systems
Rang-Chen Yu	Agility Communications
Stephen Scott	Agility Communications
Jim Blair	AMCC
Charles Duvall	Bandwidth9
Yuan Li	Blue Sky Research
Tim Simmons	Bookham Technologies
Larry Davis	Ciena
David Young	Corvis
Jeff Hutchins	iolon
Eric Selvik	iolon
George Pontis	New Focus
Raj Batra	New Focus
Jose Downes	Nortel
Peter Dartnell	Nortel
Larry McAdams	Picarro
Steffen Koehler	Sparkolor
Jay Kubicky	Santur
Wes Stalcup	Texas Instruments
Sinthia Khan	Texas Instruments

1 TABLE OF CONTENTS

1	<u>TABLE OF CONTENTS</u>	3
2	<u>LIST OF FIGURES</u>	4
3	<u>LIST OF TABLES</u>	5
4	<u>DOCUMENT REVISION HISTORY</u>	6
5	<u>PROJECT SUMMARY</u>	7
6	<u>INTRODUCTION</u>	8
6.1	<u>Scope</u>	8
6.2	<u>Objectives</u>	8
6.3	<u>Document Organization</u>	8
6.4	<u>Hierarchy – Communication Protocol</u>	9
7	<u>OIF TUNABLE DEVICE IA COMMUNICATION MODEL</u>	10
7.1	<u>Level C – Packet Structure</u>	10
7.1.1	<u>Packet Structure</u>	10
7.1.2	<u>Generic Tunable Device Register Assignments</u>	12
7.2	<u>Level B – Congestion Control, & Error Handling</u>	21
7.3	<u>Level A – Physical interface</u>	21
8	<u>LEVEL A – PHYSICAL INTERFACE IMPLEMENTATIONS</u>	22
8.1	<u>LEVEL A – Generic Implementation</u>	22
8.1.1	<u>Physical Pins Required For All Level A Implementations</u>	22
8.2	<u>RS232 - Physical Interface Implementation</u>	24
8.3	<u>SPI - Physical Interface Implementation</u>	27
8.4	<u>I2C-BUS - Physical Interface Implementation</u>	31
8.5	<u>Physical Pins Required For Specific Level A Implementations</u>	38
8.5.1	<u>Physical Interface for CW Tunable Laser (DevTyp="CW Laser")</u>	38
9	<u>LEVEL B – CONGESTION CONTROL AND ERROR HANDLING IMPLEMENTATION</u>	41
9.1	<u>Checksum</u>	41
9.2	<u>In-Bound Packet Format</u>	41
9.3	<u>Out-Bound Packet Format</u>	41
9.4	<u>Congestion Control</u>	42
9.5	<u>Module Selection</u>	43
9.6	<u>Packet Framing</u>	43
10	<u>LEVEL C – REGISTERS FOR SPECIFIC IMPLEMENTATIONS</u>	44

10.1	Device Type Assignments	44
10.2	CW Tunable Laser (DevTyp="CW Laser")	44
10.2.1	Tunable CW Laser Source Register Assignments	44
11	OIF MECHANICAL FORM FACTOR IMPLEMENTATION	54
11.1	CW Tunable Laser (DevTyp = "CW Laser")	54
11.1.1	Mechanical Outline Dimensions	54
11.1.2	Mounting Hole Dimensions	55
11.1.3	Connector Specifications	56
12	REFERENCES	58
13	APPENDICES	59
13.1	Appendix A: General tunable Device Mechanical Considerations	59

2 LIST OF FIGURES

	FIGURE 8.2-1 RS232 TIMING	25
	FIGURE 8.3-1.3-2 SPI READ (OR WRITE) TIMING, 1 BYTE HOST	29
	FIGURE 8.4-1 CONNECTION OF STANDARD AND FAST MODE DEVICES TO THE I²C-BUS.	31
	FIGURE 8.4-2 BIT TRANSFER ON THE I²C-BUS	32
	FIGURE 8.4-3 START AND STOP CONDITIONS	32
	FIGURE 8.4-4 DATA TRANSFER ON THE I²C-BUS	33
	FIGURE 8.4-5 ACKNOWLEDGE ON THE I²C-BUS	34
	FIGURE 8.4-6 A COMPLETE DATA TRANSFER.	35
	FIGURE 11.1-1 MECHANICAL OUTLINE DIMENSIONS	54
	FIGURE 13.1-1: DETERMINATION OF A REFERENCE POINT ON THE CONNECTOR FOR A TUNABLE DEVICE	59
	FIGURE 13.1-2: ESTABLISHING THE VERTICAL REFERENCE PLANES FOR A TUNABLE DEVICE	59
	FIGURE 13.1-3: SUGGESTED DIMENSIONING OF A GENERIC TUNABLE DEVICE WITH RESPECT TO THE REFERENCE POINT.	60
	FIGURE 13.1-4: GRID CONCEPT FOR LOCATING MOUNTING HOLES ON TUNABLE DEVICES SO THAT THEY DO NOT CONFLICT WITH EXISTING HOLES FOR OTHER DEVICES.	61
	FIGURE 13.1-5: EXAMPLE PIN NUMBERING SCHEME FOR A DEVICE TO MAINTAIN COMPATIBILITY AND ALLOW FOR FUTURE EXPANSION.	63
	FIGURE 13.1-6: EXAMPLE OF Z-PLUGGABLE PIN OUT NUMBERING.	63

3 LIST OF TABLES

TABLE 6.3-1 DOCUMENT ORGANIZATION	8
TABLE 6.4-1 TUNABLE DEVICE COMMUNICATION MODEL	9
TABLE 7.1-1 REQUIRED REGISTER DESCRIPTIONS FOR TUNABLE DEVICES	12
TABLE 7.1-2 EXTENDED ADDRESS SPACE MODE SELECTION (EAM)	16
TABLE 8.0-1 TRANSFER TIMES FOR VARIOUS PHYSICAL INTERFACES	22
TABLE 8.1-1 PHYSICAL PINS FOR ALL TUNABLE DEVICE IMPLEMENTATIONS	22
TABLE 8.2-1 RS232 PHYSICAL INTERFACE PINS	24
TABLE 8.3-1 SPI PHYSICAL INTERFACE PIN TABLE	28
TABLE 8.4-1 I²C-BUS PHYSICAL INTERFACE PIN TABLE	35
TABLE 8.5-1 PIN ASSIGNMENT TABLE FOR “CW-LASER”	38
TABLE 10.1-1 DEVICE TYPE ASSIGNMENTS	44
TABLE 10.2-1 MINIMUM REQUIRED REGISTER DESCRIPTIONS FOR TUNABLE LASER SOURCES	44

4 DOCUMENT REVISION HISTORY

Version	Author	Date	Notes
00	Jose Downes Jeff Hutchins Eric Selvik	Apr 11, 2002	<p>Initial Revision DRAFT 1.0 (Merge of OIF2001.516 and OIF2002.118)</p> <ul style="list-style-type: none"> ▪ Note that changes were made with TRACKING CHANGES enabled! ▪ OIF changes as per OIF OFC2002 interim meeting motions made. ▪ Corrected register assignments – Section 4 ended at 1f whereas section 7 began at 0x10. The overlap was resolved by starting section 7 at 0x20. ▪ Merged documents OIF2001.516.12 and OIF2002.118.04 and made necessary changes to make content compatible. ▪ Added two pins in §8.4 (FATAL) and (WARN) to merge the functionality between the two documents. Note the conditions for the ALM* Mask were also changed to reflect the default LOCK behavior. ▪ Documentation for the laser first/last frequency was added since it was missing. ▪ The pin table from OIF2002.118 was added and shows reserved and manufacturer specific pins.
01	Jeff Hutchins	April 16,'02	<p>'Optimized' Revision (Enhancements of merged document)</p> <ul style="list-style-type: none"> ▪ §7.1 clarified; every inbound packet generates a response ▪ §7.1.1.2 Assigned pending operation meaning to XE=1, AEA=1 combination ▪ §7.1.3.2.1 (NOP) was modified to include a pending operation bit. ▪ §7.1.1.2 clarifies outbound packet values (module's response). ▪ §7.1.2.2 - clarified the multi-byte transfer options for vendor specific registers ▪ §7.1.2.1 – clarified that these strings are not null terminated. ▪ §7.1.2.3.10 – Extended addressing mode clarified. ▪ §8.0 Physical interface table clarified ▪ §8.1, 8.2, 8.3 clarifications ▪ §8.4.1 Physical pin numbers change to reflect new pin numbering scheme. ▪ Miscellaneous other changes (all recorded in OIF2002.235 with tracking changes on.
02	Jeff Hutchins	April 20,2002	<p>'Boston Meeting 4/02' - (Enhancements of merged document)</p> <ul style="list-style-type: none"> ▪ Numerous additional minor edits and clarifications received from OIF participants. ▪ Add a modified version of the download document submitted by Agility's Stephen Scott. ▪ Documented checksum C code for CRC16. ▪ Removed time from MFGDate format. ▪ Added section on packet framing §9.6. ▪ Added current temperature register (0x43) in §10.2.1. ▪ Cleared up ambiguity on status and trigger register's XE,CE flags as well as SRQ and ALM flags. ▪ Clarified a number of items dealing with status registers, triggers, errors with changing channels, first frequency, and grid spacing. Added definition around SRQ assertion with the various physical interfaces upon XE or CE errors.
03	Eric Selvik	May 7, 2002	<p>Changes from OIF Boston, 4/02:</p> <ul style="list-style-type: none"> • Changed pin numbering on connector as shown in Fig. 11.1-3 to be consistent with mfg numbering. • Changed pin assignments as shown in Table 8.5-1 • Added further definition to power supply in table 8.4-2 • Added detail on "Z-pluggable" connector as shown in 11.1.3.7. • Added Project Summary, Section 5 • Corrected typo in 11.2.7 • Corrected a number of typos and formatting errors
04	Jeff Hutchins Stephen Scott	May 15, 2002	<p>Final edits for straw ballot submission on 5/10/02:</p> <ul style="list-style-type: none"> • Incorporated register number reassignments • Updated DLConfig to describe response on read. • Clarified pending operation bit assignments.

			<ul style="list-style-type: none"> Changed language to indicate 5V tolerant I/Os instead of 5V compliant.
05&06	Eric Selvik	July 17, 2002	Edits made to resolve comments from Straw Ballot, see OIF2002.340 for details. <ul style="list-style-type: none"> Scope of document clarified to address CW tunable lasers only.
07	Eric Selvik	July 31, 2002	Edits made following OIF Copenhagen <ul style="list-style-type: none"> Add Section 8.4 on I2C Physical Interface Update pin numbering in 8.5
08	Eric Selvik	August 23, 2002	Corrections made to clerical errors <ul style="list-style-type: none"> 11.1.3.7.2: Changed P/N to CLP-120-02-G-D/DH-P to clarify inconsistency. 11.1.2.10 – updated tolerances 11.1.3.5 – updated tolerances Corrected mistyped AXC bit assignment in 10.2.1.8 and 10.2.1.14.
09	Eric Selvik	September 27, 2002	Corrections made to resolve comments from Straw Ballot #32, see OIF2002.454 for details. <ul style="list-style-type: none"> Modifications to Table 8.5-2 to remove 5V tolerance requirement – 3.45V was 5V; 4 mA was 8mA; -4mA was -8mA Added clarifying text to Section 8 re: support of physical interfaces Added clarifying text to Section 8.4 for slave addressing for I2C Modified description for Manufacturer Specific pins in Table 8.5-1 Corrected typo in Table 10.2-1 – “THz” was “GHz”
10	Eric Selvik	November 21, 2002	Corrections made as approved in OIF Orlando meeting, see OIF2002.514 and OIF2002.515 <ul style="list-style-type: none"> Removed all remaining references to 5V tolerance Section 8.5.1: RxD and IOCLK now I/O MRL was MR, CRL was CR in reg 0x20 bits 4 and 5 Corrected wording in 10.2.1.1 11.1.3.7.2 Mating z-connector now FLE-120-01-G-DV. Corrected pin #s in 8.4-1 to match 8.5-1

5 PROJECT SUMMARY

The tunable laser project was started in the PLL Working Group in January 2001 at OIF Tampa to specify tunable laser module form factor (size and exclusion zones), pin outs, and command sets. In November 2001 the project was split into two task groups: Mechanical Form Factor and Communications Protocol. In March 2002 the two task groups merged working documents into OIF2002.210 to be sent for ballot as the tunable laser implementation agreement. The combined document, OIF2002.210, was sent to Straw Ballot. In resolving the comments from the Straw Ballot, the document was edited, the most major of which was to add an I²C physical interface implementation. The document will be sent back to Straw Ballot, and then on to Principal Member Ballot. Document passed Principal Member Ballot in October 2002, and is now in standard maintenance mode.

6 INTRODUCTION

6.1 SCOPE

This document outlines a communication protocol, electrical interface, and mechanical form factor interoperability agreement for tunable CW lasers. The intention is to have this document form the basis of a highly extensible interoperability agreement within the OIF among tunable device manufacturers and can form the basis for other tunable device interoperability agreements.

6.2 OBJECTIVES

This document details an electrical and mechanical model for tunable CW laser communication which is readily extensible to tunable devices.

The objectives are:

- The model should support tunable lasers with a variety of capabilities (such as fast/slow communication) or with varying degrees of sophistication.
- The model should be flexible enough to cover a variety of physical interfaces and a variety of technology driven mechanical form factors.
- The model should be extensible to support a variety of future tunable devices.

This agreement can address a broad range of devices. For example, the communication model supports interfaces with variable time scales sufficient for both low-speed and high-speed communication devices.

6.3 DOCUMENT ORGANIZATION

The following table summarizes the document's organization.

Table 6.3-1 Document Organization

Section	Description
§7	OIF Tunable Device IA Communication Model Describes the generic tunable device communication model. This section provides the basis for all the specific electrical and communication specific implementations
§8	Level A – Physical Interface Implementations Describes the general and specific electrical and physical interface implementations.
§9	Level B – Congestion Control & Error Handling Implementation Describes the general implementation for this level.
§10	Level C – Registers for Tunable Lasers Describes the extensions required for §7.1 (Level C) for the tunable lasers.
§11	OIF Tunable Laser Mechanical Model Describes the mechanical form factor guidelines for tunable lasers.

6.4 HIERARCHY – COMMUNICATION PROTOCOL

The Tunable Device Communication Model describes three interoperability levels based loosely on the 7 level OSI network layer model (although similar, it does not map directly to the OSI network layers). The level descriptions below are used in a general way.

A brief description of the OSI model follows:

Level 7 – Application Layer

Provides appropriate semantics or meaning to the transferred data and setup and termination between application processes.

Level 6 – Presentation Layer

Manages and transfers end-user data-unit syntax, isolates application layer processes from differences in data representation and syntax, and handles errors.

Level 5 – Session Layer

Manages session to ensure orderly delivery of information of data and manages parameter negotiation.

Level 4 – Transport Layer

Utilizes network layer to ensure reliable, sequenced data exchange. Can be connection or connectionless. Connection oriented services that present packets in sequence.

Level 3 – Network Layer

Data units are called packets. Uses Data Link Layer to ensure error free delivery of data. Provides flow and congestion control.

Level 2 – Data Link Layer


Blocks of data are called frames. Ensures reliable and error-free data transfer across links between adjacent network nodes. Control information may be in the form of headers or trailers and allows the receiving node to perform synchronization and error detection at the bit level.

Level 1 – Physical Layer

Specifies data signal encoding, connectors/pin-outs.

The Tunable Device Model maps into the OSI levels as follows:

Table 6.4-1 Tunable Device Communication Model

OSI Level	Interoperability Level	Interoperability Description
HIGH  LOW	Level C	Semantics - register descriptions and data formatting
	Level B	Reliable data exchange (Checksums, pacing) Packet format and bit ordering Communication error handling
	Level A	Physical Layer: RS232, SPI, Parallel, etc.

7 OIF TUNABLE DEVICE IA COMMUNICATION MODEL

The communication model consists of three layers: Level A, Level B, and Level C. The model for these layers is defined in this section.

Complete implementation details are contained in later sections.

7.1 LEVEL C – PACKET STRUCTURE

The communication model is defined around a 28-bit command packet sent to the tunable module, and a 26-bit response packet returned from the tunable module. These packet lengths are sufficient to read and write 16 bit registers in the tunable laser module.

The packets are “binary”, compact, and designed for rapid parsing, and to be compatible with possible ASIC implementation.

The higher order bit number is the MSB.

The in-bound direction is defined as data sent by the host and which is received by the tunable module. The out-bound direction is defined as data being transmitted by the tunable module to the host. The in-bound and out-bound packets formats are nearly the same but not identical.

The command structure allows for a variety of tunable devices to be defined and controlled through the interface.

Every in-bound packet generates an out-bound packet whether the request is a read or write operation. Framing is handled in Level B (Congestion Control & Error Handling) and is accomplished through a fixed packet size and the IRDY* line which toggles high upon receipt of a packet.

Framing is controlled by the IRDY* line and the fixed packet size.

7.1.1 PACKET STRUCTURE

The in-bound and out-bound packets have different lengths. Level B (Congestion Control & Error Handling) will concatenate additional bits to form the actual packet transported by the Level A physical interface.

7.1.1.1 In-bound Packet Format

The in-bound packet consists of 28 bits.

Inbound Byte 0

7	6	5	4	3	2	1	0
Reserved for Level B				0x0 (reserved for future expansion)			Write=1

Bit 0 defines the operation (Read/Write). A request to write a value to a register is indicated by a Write=1. A request to read a register value from the module is indicated by Write=0.

Bits 3:1 are left for expansion.

Bits 7:4 are reserved for Level B, which may or may not utilize these bit positions.

Inbound Byte 1

7	6	5	4	3	2	1	0
Register Number (0x00 – 0xff)							

Byte 1 defines the register number, which can take on values from 0x00 to 0xff.

Inbound Byte 2

7	6	5	4	3	2	1	0
Data 15:8							

Inbound Byte 3

7	6	5	4	3	2	1	0
Data 7:0							

Bytes 2 and 3 contain the data for a write operation and are undefined for a read operation.

7.1.1.2 Outbound Packet Format

The out-bound packet consists of 26 bits.

Outbound Byte 0

7	6	5	4	3	2	1	0
Reserved for Level B						Status	

Bits 1:0 Value	Status Field
0x00	Normal return status
0x01	XE flag, (execution error)
0x02	AEA flag, (Automatic extended addressing result being returned)
0x03	Command not complete, pending

Bits 1:0=0x00, Normal. No execution errors and not using AEA mode for returning result.

Bits 1:0=0x01, XE flag (Execution Error) signifies that the previous command failed to execute properly. (Bits 1:0)<>0x01 signifies that the previous command completed successfully or is pending.

Bits 1:0=0x02, AEA¹ (automatic extended addressing) mode, indicates that the register (for which a read or write operation has been given) requires a multi-byte sequence². The unsigned value is returned in bytes 2 and 3 and represents the number of bytes in the multi-byte response.

Bits 1:0=0x03, Command not complete, indicates that the command will take longer than the maximum timeout specified for this device type.³ In this case the module returns a response within the timeout period and continues to execute the requested operation. The host can poll the module's status register (0x00) through the communication's interface to determine if the operation has completed.

Bits 7:2 are reserved for Level B which may or may not utilize these bit positions.

Outbound Byte 1

7	6	5	4	3	2	1	0
Register Number (0x00 – 0xff)							

Byte 1 returns the register number that a “read” requested or that a write targeted.

Outbound Byte 2

7	6	5	4	3	2	1	0
Data 15:8							

Outbound Byte 3

7	6	5	4	3	2	1	0
Data 7:0							

¹ See §7.1.2.2 and §7.1.2.3.10

² In the case where a write was done to a register that supports AEA, outbound bytes 2 and 3 are ignored. The write command will need to be repeated this time addressing the AEA-EAR register instead.

³ Device types/classes are specific implementations of tunable devices. See §10.1 for a table of device type assignments

Read operation: Bytes 2 and 3 contain the returned data (or byte count for an AEA pre-configured access) for the response to the current read operation.

Write operation: Bytes 2 and 3 are undefined, or the byte count for an AEA pre-configured access. In the case of an uncompleted operation, byte 2 is 0x00 and byte 3 contains the pending operation bit that has been assigned to this uncompleted operation (as shown in the following table).⁴

Outbound Byte 3 – Pending Operation Case				
7	6	5	4	3:0
Bit assigned to pending operation ⁵				0x0

7.1.2 GENERIC TUNABLE DEVICE REGISTER ASSIGNMENTS

Level C describes specific commands and responses for tunable modules. It describes the type of commands that can be issued and the format of any data values supplied or returned from the module. The commands are issued as requests to read and/or write 16 bit registers.

7.1.2.1 Register Assignments

The following table specifies the required registers for tunable devices. All register values are unsigned integers unless otherwise noted.

The DevTyp (Device Type) register describes the type of tunable device. Each device type has its own predefined registers map (defined later in §10).

The AEA (automatic extended addressing) column indicates if the register supports by default AEA mode. A vendor may decide to support extending addressing for a register even if the default configuration doesn't indicate AEA⁶.

Table 7.1-1 Required Register Descriptions for Tunable Devices

Register Number	Register Name	Read / Write	AEA	Description
0x00	NOP	R/W		Provide a way to read a pending response as from an interrupt, to determine if there is pending operation, and/or determine the specific error condition for a failed command.
0x01	DevTyp	R	AEA	Returns device type (tunable laser source, filter, modulator, etc) as a null terminated string. See §10.1 for allowed values.
0x02	MFGR	R	AEA	Returns manufacturer as a null terminated string in AEA mode (vendor specific format)
0x03	Model	R	AEA	Returns a model null terminated string in AEA mode (vendor specific format)
0x04	SerNo	R	AEA	Returns the serial null terminated string in AEA mode (vendor specific format)
0x05	MFGDate	R		Returns the mfg date as a null terminated string.
0x06	FW	R	AEA	Returns a manufacturer specific firmware release as a null terminated string in AEA mode (vendor specific format)
0x07	FWBack	R	AEA	Returns manufacturer specific firmware backwards compatibility as a null terminated string (vendor specific format)
0x08	General Configuration	RW		General module configuration

⁴ See §7.1.2.3.1 (NOP 0x00) for a description of the pending operation bits.

⁵ Valid values for this field are only either 0x1, 0x2, 0x4, 0x8 (Bit 4, bit 5, bit 6, or bit7).

⁶ For instance, suppose that a module monitors a number of temperature readings (case temperature and diode temperature). A non-vendor specific register may, such as temperature may not be specified as supported AEA mode. However, a vendor may implement AEA mode for this register so that "read" access returns 2 values.

Register Number	Register Name	Read / Write	AEA	Description
0x09	AEA – EAC	R/W		Automatic extended address configuration register - auto incr/decr flag on read and on write and additional address bits
0x0A	AEA - EA	R/W		Automatic extended address (16 bits)
0x0B	AEA - EAR	R/W		Location accessed “thru” AEA-EA and AEA-EAC
0x0C	Reserved			
0x0D	IOCap	R/W		Physical interface specific information in string form (such as data rate, etc.)
0x0E	EAC	R/W		Extended address configuration register - auto incr/decr flag on read and on write and additional address bits
0x0F	EA	R/W		Extended address (16 bits)
0x10	EAR	R/W		Location accessed “thru” EA and EAC
0x11	WChkSum	W		Asserts checksum for next command packet
0x12	RChkSum	R		Returns checksum for last response packet
0x13	LastResponse	R		Returns last response
0x14	DLConfig	R/W		Download configuration register
0x15	DLStatus	R		Download status register
0x16 – 0x1F	Reserved	--	--	Reserved for future use
0x20-0xFF	Reserved & Manufacturer specific			Reserved for use by specific device types (See §10.1). Each device type allocates these registers as either manufacturer specific or required registers.

7.1.2.2 Transferring Multi-Byte Values

Fields longer than 16 bits can be read or written in two ways. Vendors may employ either of the following techniques when implementing manufacturer specific registers.

1. Multiple Contiguous Registers

Multiple contiguous registers can be defined to hold a value. For instance, a command that returned a 32 bit value might utilize a register X and X+1 to hold the value. Reading the lower register (X) causes both to be defined. Reading an upper register first (such as X+1) returns an undefined result. Writing a lower register (X) causes no action to be taken. The changed register value is changed when the high register (X+1) is written. No general-purpose registers currently utilize this feature.

2. Extended Addressing

Some fields may have a manufacturer defined length. They can be transferred in one of two ways using extending addressing.

a) AEA (Automatic Extended Addressing) Register:

If a register, X, supports AEA (automatic extended addressing), a read (or write) of the register X utilizing AEA mode will return a unsigned integer representing the length in bytes of the field referred to by register X. The AEA indication will be set in out-bound byte 0 of the response. The extended addressing registers EAC (extended address configuration) and EA (extended address) will be pre-configured by the module such that multiple reads or writes can be performed on the EAR (extended address register) register. Data can be transferred up to the field length previously returned by reading register X. If the length is an odd number of bytes, the last byte transferred (out-bound byte 3) is undefined.

b) EA (Extending Addressing) Register

The vendor may specify a configuration of the 3 extended addressing registers (0x0E – 0x10) (See §7.1.2.3.11). Once the host has configured these registers, the values can be transferred the appropriate quantity of bytes by reading from or writing to the extended address register 0x10. This register is also specifically used and pre-configured by the download/upload procedures.⁷

⁷ The 2nd extended address register is provided so that large data transfers can be easily interrupted. For instance, servicing an exceptional condition may require the host to read a register which supports AEA addressing. If this condition occurs during a time intensive in-service firmware upgrade, the primary extended address registers (AEA) are still available for use. Accessing a register through the AEA mechanism will not

7.1.2.3 Detailed Register Definitions⁸

7.1.2.3.1 NOP (0x00)

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CE	0x00 (Reserved for expansion)							Pending Ops ⁹				Error Condition			

Writing to register NOP is a non-operation¹⁰. Reading from register NOP can also be used as a non-operation. This command is provided such that a command can be given to any tunable device without affecting the state of the module. In the event the module requests attention due to a communications error, reading the NOP register can be used to return error and status flags such as the XE (execution error) bit defined in Level C and any communication flags defined by level B.¹¹

Bits 3:0 – Error condition for last command

A read of the NOP register will return the error condition from the last command before setting it to 0x00 to reflect the status of the current command (which is reading the NOP register).

Value (Bits 3:0)	Symbol	Meaning
0x00	OK	Ok, no errors
0x01	RNI	The addressed register is not implemented
0x02	RNW	Register not write-able; register cannot be written
0x03	RVE	Register value range error; writing register contents causes value range error; contents unchanged
0x04	CIP	Command ignored due to pending operation
0x05	CII	Command ignored while module is initializing
0x06	ERE	Extended address range error (address invalid)
0x07	ERO	Extended address is read only
0x08	EXF	Execution general failure
0x09-0x0E	--	Reserved for future expansion
0x0F	VSE	Vendor specific error (see vendor specific documentation for more information)

affect the state of the 2nd extended address registers (0xE-0x10). The upgrade can continue after the condition is serviced.

⁸ In-bound bytes 2 and 3 contain the value to be written to the specified register. Out-bound bytes 2 and 3 contain the contents of the register that is being read.

⁹ The pending operation field provides space for 4 concurrent pending operations. This is expected to be sufficient for all modules. In the event that more than 4 concurrent pending operations are needed, additional bits from the reserved field (bits 14:8) may be allocated in the future.

¹⁰ Reading or writing to the NOP register is a non-operation. Physical interfaces such as SPI implement a delayed response. The response to the previous command is returned when issuing the current command. It might be useful to use a write NOP command to read the last response when using the SPI interface. The write command will not return the NOP register's contents while a read command will.

¹¹ A level B error flag, CE (communication error), is defined in section 9.3. Together the flag bits XE (execution error) and CE reflect the status of the previously transferred command.

Bits 7:4 –Pending operation status¹²

This value ranges between 0x00 and 0x0F and indicates which, if any, pending operations are currently executing. Each pending operation is assigned one of the 4 bit positions. A value of 0x00 indicates that there are no pending operations currently executing. The command used to read this register is not considered a pending operation and is not reflected in this status¹³. Note that the format of outbound byte 3 corresponds to the same bit positions in the NOP command.

Bits 14:8 – Reserved (default 0x00)

Bit 15 – Reflects the status of the CE bit in the last response packet.

7.1.2.3.2 DevTyp (0x01) (read-only)

The DevTyp register returns the device type as a null terminated ASCII string through AEA mode.

The string associated with a particular DevTyp is defined in specific Level C implementation. The maximum string length is 80 bytes. The default value is specified in section §10.1.

7.1.2.3.3 MFGR (0x02) (read-only)

The MFGR register returns the manufacturer as a null terminated ASCII string through AEA mode. The maximum string size is 80 bytes. The default value is vendor specific.

7.1.2.3.4 Model (0x03) (read-only)

The Model register returns the manufacturer's model number as a null terminated ASCII string through AEA mode. The maximum string size is 80 bytes. The default value is vendor specific.

7.1.2.3.5 SerNo (0x04) (read-only)

The SerNo register returns the manufacturer's assigned serial number as a null terminated ASCII string through the AEA mode. The maximum size is 80 bytes. The default value is vendor specific.

7.1.2.3.6 MFGDate (0x05) (read-only)

The MFGDate register returns the date of manufacture as a null terminated ASCII string formatted as "DD-MON-YYYY", DD is a 2 character field with leading zeros indicating the day of the month, MON is 3 character representation of the month (JAN,FEB,MAR,APR,MAY,JUN,JUL,AUG,SEP,OCT,NOV,DEC), and YYYY is the 4 digit year.

Example: "04-APR-2001"

7.1.2.3.7 FWRel (0x06) (read-only)

The FWRel register returns the firmware release information as a null terminated ASCII string through AEA mode. The maximum size is 80 bytes. The format of this field is manufacturer defined. Note that a module may have one or more firmware and/or hardware revisions to track. The default value is vendor specific.

7.1.2.3.8 FWBack (0x07) (read-only)

The FWBack register returns the firmware backwards compatibility information as a null terminated ASCII string through AEA mode. The maximum size is 80 bytes. The string is in the same format as FWRel and specifies the earliest revision the current firmware is compatible with. The default value is vendor specific.

7.1.2.3.9 General Module Configuration (0x08) (RW)

The general module configuration is provided to cover items that are generic across all tunable devices.

¹² When a response packet is returned which indicates an incomplete operation, outbound byte 3 indicates which pending operation bit is assigned to this operation. While the operation is still pending, the corresponding bit will be set in the NOP register. The module may assign any unset bit (bits 7:4) to the pending operation. When the operation completes, the corresponding pending operation bit is cleared.

¹³ An on-going module initialization (from reset or power up) would count as a pending operation.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0x0000															RCS

Bit 0: RCS - Require CRC-16 to execute for all commands.

“1”: If this bit is set to 1 by the host, all commands, except for WChkSum, require a valid CRC-16 to be written to the WChkSum register (0x11) prior to the command being sent by the host. See register 0x11.

“0”: Default = 0. CRC-16 is not supported.

7.1.2.3.10 IOCap (0x0D) (R/W)

The IOCap register returns or sets the I/O interface capabilities. Each physical interface (Level A) implementation must define the format of this register. Typically this register would specify a maximum data rate that the module could support. Other parameters may be necessary for the various physical interface implementations. The register returns to its default when hardware reset is asserted or then module is powered on. The physical interface can only be selected during power on or during hardware reset. (For default, see §8 for each of the physical interfaces).

7.1.2.3.11 Extended Addressing Mode Registers (0x09-0x0B, 0x0E – 0x10)

The predefined register set provides two sets of 3 registers each which are utilized for extended addressing. The two sets can be used identically. However, the first set can be automatically pre-configured (by the module) when the host reads from or writes to a register that supports AEA (automatic extended addressing) mode. The second set is an auxiliary set (which is automatically pre-configured only for downloads or uploads). The read or write returns the number of bytes in the field. After the AEA-EAC and AEA-EA are preset, reads or writes on AEA-EAR are used to read or write the field location. Default values are not defined. Note that although a write returns the number of bytes, no data is written. The write command must be re-issued in order to complete the write.¹⁴

The first register, AEA-EAC (0x09) or EAC (0x0E), configures the extended addressing mode.

EAC:

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
RAI		WAI		EAM		INCR		TBD	High order 6 address bits						

RAI: Read Auto Increment (Bits 15:14)
 0x0 No address change on read
 0x1 Address auto post increment by INCR on read
 0x2 Address auto post decrement by INCR on read
 0x3 Action not defined

WAI: Write Auto Increment (Bits 13:12)
 0x0 No address change on write
 0x1 Address auto post increment by INCR on write
 0x2 Address auto post decrement by INCR on write
 0x3 Action not defined

EAM: Extended Address Mode (Bits 11:9)
 These three bits provide 8 possible address spaces. The default register space is defined with EAM=0x0. A firmware upgrade procedure would select the appropriate “code address space”.

Table 7.1-2 Extended Address Space Mode Selection (EAM)

EAM	Address Space
0x0	Default register space (including 0x00 – 0xff)
0x1	Physical data space 1
0x2	Physical data space 2
0x3	Physical code space 1
0x4	Physical code space 2
0x5-0x7	Manufacturer specific

¹⁴ It is up to the vendor to determine whether writing or reading beyond the length of a field will generate an execution error.

INCR: Increment register (Bits 8:7)

The auto increment and auto decrement operations modify the address by this unsigned value. For register space, this would typically be 1. If the physical space addressed by bytes, the best increment might more naturally be 2. If the configuration transfers 1 byte per read or write, only the low order byte is transferred and the high order byte is ignored.

TBD: Reserved (Bit 6)

High order address bits (Bits 5:0)

The high order address bits are concatenated with the EA register forming a 22 bit physical or logical address or register number.

In order to access a location through the extended addressing interface, the EAC must be configured, the EA must be filled with an initial address, and then the extended address register can be read or written to transfer the data.

The second register, EA, contains the lower 16 address bits.

EA:

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Extended Address (low order 16 bits)															

This register is set to the address value. Note with EAM=0x0, this register accesses the default register space. With EAM=0x0, extended addresses from 0x00 to 0xFF are equivalent to registers 0x00 to 0xFF.

EAR:

A read on EAR causes the value referred to by EAC:EA to be returned. A write to EAR causes the location referred to by EAC:EA to be written, assuming the register is write-able.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Extended Address															

7.1.2.3.12 WChkSum (0x11) (W)

In the event that the system requires more complete error detection, the WChksum and RChkSum commands can be used to achieve this goal. These commands are only active when RCS=1 in the module configuration register.

The WChkSum allows the host to assert a checksum value before issuing a command. If this register is written, the next read or write in-bound packet will have its full CRC-16 checksum computed and compared to this stored value. If the checksum does not match that stored value, a CE=1 (communication error) will be asserted by Level B in the outbound response packet. Note that a WChkSum command can precede a WChkSum command although this is not very useful.

In-bound packets received without an immediately prior WChkSum assertion will not have the packet's CRC-16 checksum computed (unless the module configuration bit 0=1 in register 0x08).

The CRC-16 checksum is defined by the polynomial $x^{16}+x^{15}+x^2+1$. The following function shows one such implementation for this CRC-16 which is called for each of the 4 bytes. The initial value for crc is 0x0000.

```
#define CRC16    0xA001    /* X16 + X15 + X2 + 1 */
unsigned int calcCRC( unsigned int crc, unsigned int data )    {
    int i;
    for ( i = 8; i; i-- )    {
        if (( data ^ crc ) & 0x0001) crc = ( crc >> 1 ) ^ CRC16;
        else    crc >>= 1;
        data >>= 1;
    }
    return crc;
}

#include <stdio.h>
int main(int argc, char** argv)    {
    int crc = 0;
    int i;
    int data;
    if (argc!=5)    {
        fprintf(stderr,"Usage: crc16 hexdata0 hexdata1 hexdata2 hexdata3\n");
        fprintf(stderr,"    Example:Usage: crc16 0x0d 0x0d 0x0d 0x0d\n");
        exit(1);
    }
    crc=0;
    for (i=0; i<4; i++)    {
        sscanf(argv[ i ],"%x",&data);
        crc=calcCRC(crc,data);
    }
    printf("CRC-16 is %x\n",crc);
}
```

7.1.2.3.13 RChkSum (0x12) (R)

In the event that the system requires more complete error detection, the WChksum and RChkSum commands can be used to achieve this goal. These commands are only active when RCS=1 in the module configuration register.

The RChkSum command is used to read the checksum computed for the last response packet. The checksum is computed using the same CRC-16 defined for WChkSum operation. This checksum is computed for all out-bound packets excluding the response from a read RChkSum operation.

The host is expected to compare the result returned from the RChkSum command and compares this to the CRC-16 the host calculated on the received response packet. On detecting an error, the host can force the last response to be returned again via the LastResponse command.

7.1.2.3.14 LastResponse (0x13) (R)

Reading last response register forces the module to return all 4 bytes of the last non-RChkSum response. This is useful if a checksum error was found and the host wants to re-read the last response. Upon completion, the RChkSum register contains the same checksum value it had before the register was read. This register is active whether or not RCS=1 in the module configuration register.

7.1.2.3.15 DLConfig (0x14) (R/W)

The DLConfig register configures a *host to slave* download of code or data for reconfiguration purposes or configures a *slave to host* upload of code or data to the host.

A file transfer may occur at several locations such as vendor factory, customer site (on the bench), customer system (circuit down), or customer system (live circuit).

The following describes the actions required to transfer a file from the host to the module and then have the module run that file.

Step	Host Sends	Module Responds
1	Write the DLConfig register indicating the type of transfer coming and asserting INIT_WRITE. Code might be boot code, run time code, FPGA code, CPLD code.	Module responds by initializing the DLStatus register with the appropriate polling requirements for this type of transfer. Polling can be required for each packet, after INIT_WRITE, and/or after the DONE assertion. Module prepares for file transfer. Module also responds by pre-configuring the extended address registers (0x0E-0x0F) for this type of transfer.
2	Read DLStatus (poll module status) to determine if the module is ready for the file transfer to begin.	Module responds to polls and when ready, informs the host that it can accept the file by asserting SRDY (SRDY=1)
3	Host writes (or reads) to the extended address register (0x10) with the file data, 2 bytes at a time. ¹⁵	Module receives (or sends) file. Remember each out-bound packet response indicates if any errors have occurred.
4	If last DLStatus indicated PPOL=1 (packet polling is required), read DLStatus until PRDY=1 is asserted. If more data, go to previous step. If no more data, proceed to next step.	Respond with DLStatus. Assert PRDY=1 when ready for next packet. If PPOL=0, always assert PRDY=1.
5	Host writes DLConfig and asserts DONE=1.	Module completes transfer and checks for code consistency. If not, asserts ABRT=1.
6	Host reads DLStatus and waits until ERDY=1 is asserted (or 20 seconds expires)	When Module has completed transfer, ERDY=1 is asserted.

¹⁵ Note, data can be transferred one byte at a time (W 0x10 0x00XX, W 0x10 0x00YY) or two bytes at time with one write (W 0x10 0xXXYY). The transfer mode is pre-determined by the vendor setup of the EA register and the file to be transferred. The host transfers all bytes as they appear in the file. The EAC (0x0E) is configured such that the auto-increment on write and read is set to the appropriate number of bytes.

7	Host reads DLStatus and checks for VALID=1 and ABRT=0, If so, go to step 8	Module checks for code consistency at this location and asserts VALID.
8	Host writes DLConfig and asserts RUNV with the same value as was set in step 1 above.	Module begins running newly transferred code.

When the DLConfig register is read, the RUNV value returns the value for the firmware currently running in the module. This value is unchanged with a power down or reset. The other fields return the default values.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE				RUNV				Reserved (0x0)				INIT_READ	DONE	ABRT	INIT_WRITE

INIT_WRITE Bit 0 – This bit informs the module to prepare for download. The module should perform its necessary housekeeping to be ready for download. Pre-configures the extended address registers (0x0E-0x0F).

- 0 – Do not start download. (default)
- 1 – Prepare for download.

ABRT Bit 1 – This bit informs the module to abort the transfer. Clears the ABRT bit in DLStatus.

- 0 – Do not abort transfer. (default)
- 1 – Abort the transfer.

DONE Bit 2 – This bit informs the module that the transfer is complete.

- 0 – Transfer is not done. (default)
- 1 – Transfer is done.

INIT_READ Bit 3 – This bit informs the module to prepare for upload. Like INIT_WRITE, is pre-configures the extended address register. Pre-configures the extended address registers (0x0E-0x0F).

- 0 – Do not start upload. (default)
- 1 – Prepare for upload

TYPE Bit 12-15 – Type of code to Transfer. (0x00 – default)

TYPE Value	Code Type
0x00	No change to value
0x01	Boot Version 1
0x02	Boot Version 2
0x03	Main Version 1
0x04	Main Version 2
0x05	HW Version 1
0x06	HW Version 2
0x07	Test Version 1
0x08	Test Version 2
0x09 – 0xFE	Vendor specific
0xFF	Reserved

RUNV – Bit 8-11 - Host informs module to run this version when written, and returns the current version that is currently executing when read. The module will respond to the request and then take appropriate action to run the code, i.e. restart, reprogram hardware ... The default setting is vendor specific. Vendors may not support all RUNV values.

RUNV Value	Code Type
0x00	No change to value
0x01	Boot Version 1
0x02	Boot Version 2
0x03	Main Version 1
0x04	Main Version 2
0x05	HW Version 1
0x06	HW Version 2
0x07	Test Version 1
0x08	Test Version 2
0x09 - 0xFE	Vendor specific
0xFF	Run all latest version

7.1.2.3.16 DLStatus (0x15) (R)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRT	VALID								ERDY	PDRY	SRDY		EPOL	PPOL	SPOL

POL bits are vendor dependent and let the host know if polling is required between states. Note that after the INIT_WRITE configuration is received the module will initialize the POL bits to their appropriate vendor specific states. That is to say, if a vendor poll is not required at a specific state the bit RDY bit will always be 1 (ready). If polling is required, the RDY bit will be 0 when not ready and 1 when ready.

SPOL Bit 0 – Vendor indication if polling the SRDY bit is required after the INIT_WRITE (download configuration register) bit is set.

0 – Indicates no poll required.

1 – Indicates that polling the SRDY bit is require

PPOL Bit 1 – Vendor indication if polling the PRDY bit is required between data bytes. This is during the actual file transfer.

0 – Indicates no poll required.

1 – Indicates that polling the PRDY bit is required.

EPOL Bit 2 – Vendor indication if polling the EPOL bit is required after the DONE (download configuration register) bit is set.

0 – Indicates no poll required.

1 – Indicates that polling the ERDY bit is required.

SRDY Bit 4 – Indicates that the module is ready to start receiving the file.

0 – Indicates the transfer initialization process is not complete.

1 – Indicates ready for transfer.

PRDY Bit 5 – Indicates that the module is ready to get the next packet in the file transfer process.

0 – Indicates the module is not ready for another packet of data.

1 – Indicates ready for another packet.

ERDY Bit 6 – Indicates that the module has completed the file transfer process. The file has been deposited into its final location.

0 – Indicates the module is still done with the file transfer.

1 – Indicates the file transfer process is complete. Ready to run file.

VALID Bit 14 – Indicates that the module has a valid code type at this location. Asserted after the DLConfig (0x14) “TYPE” field is written with a non-zero type field.

0 – Indicates the module does not have a valid code type at this location.

1 – Indicates the module does have a valid code type at this location.

ABRT Bit 15 – Indicates that the module has determined that the download process has failed and needs to be aborted. The host should set ABRT in DLConfig to clear the ABRT bit in DLStatus and then start over.

0 – Indicates the module does not need to abort the file transfer process.

1 - Indicates the module needs to abort the file transfer process.

7.2 LEVEL B – CONGESTION CONTROL, & ERROR HANDLING

Independent of the command structure, this level defines the packet format that encapsulates the Level C command structure and response. It also addresses congestion control and reliable transfers.

There can be some dependencies upon the needs of the Level A physical interface.

7.3 LEVEL A – PHYSICAL INTERFACE

The physical layer supports the electrical or mechanical interface that transports the communication packets. The model supports a variety of physical interfaces.

The Level 1 interface description is designed to be taken directly from industry documents describing those interfaces and providing any additional supplemental information to completely specify the particular implementation. For instance, in the case of RS232, the Level 1 description would in addition to RS232 specify the number of stop bits, the number of wires used in the interface, the baud rate, etc.

8 LEVEL A – PHYSICAL INTERFACE IMPLEMENTATIONS

A variety of interfaces can be supported. A module must support at least one of the physical interfaces described in this section. The interface employed requires consideration of hardware availability, PCB layout constraints, and packet transfer speed. The following table summarizes the data transmission speeds achievable with the various supported physical interfaces. The estimated transfer time includes a transmit latency of 0.5 us/byte (total 7 inter-byte latencies for a 2 packet¹⁶, 8 byte transfer) for an 8 bit host and a 5 us/transfer for the tunable module.

Table 8.0-1 Transfer Times for Various Physical Interfaces

Interface	Typical Clock Rate	Inter-byte Latency (Total) (us)	Time To Transfer 2 Packets (us)		
			Min	Max	
RS232	9.6 k-baud - 115.2 k-baud	1*5 us	694.4	8333.3	
SPI ¹⁷	1 MHz - 50MHz	4 byte host	6*0.5 +1*5 us	6.3	69.0
		1 byte host	1*5 us	9.3	72.0
I ² C ¹⁸	Standard mode 100KHz		1752		
	Fast mode 400KHz		522		
	High speed mode 3.4 MHz		160.2		

8.1 LEVEL A – GENERIC IMPLEMENTATION

This section defines the generic requirements in order to implement all of the specific Level A implementations.

8.1.1 PHYSICAL PINS REQUIRED FOR ALL LEVEL A IMPLEMENTATIONS

The physical interfaces documented in the following sections require the following pin capabilities. All types of tunable modules have these 16 pins as specified below in Table 8.1-1 Physical Pins for All Tunable Device Implementations.

The responsible level column in the table indicates which level in this document (Level A, B, or C) is responsible for describing this pin's behavior.

Table 8.1-1 Physical Pins for All Tunable Device Implementations

Symbol	Type	Responsible Level	Name	Description
GND	Power	A	Ground	Ground
VCC	Power	A	Vcc	Vcc
RST*	LVTTL input, active low	C	Reset	Hard reset tunable module while LOW
IRDY*	LVTTL output, active low	B	Communication interface Ready	Communication interface ready to receive command packets data (at the packet level). ¹⁹

¹⁶ Two packets were chosen for this benchmark because some tunable laser technologies require 2 packets to change channels.

¹⁷ The table shows two entries for SPI. The host can be either 1-byte transfer capable or a host can be at least 4-byte transfer capable. Both are supported with the 4-byte slave implementation.

¹⁸ Assumes that a packet must be received for every packet sent. Therefore, an I2C read packet is required for the handshake increasing the total number of packets to 4.

¹⁹ The distinction is made "at the packet level". IRDY* is a packet pacing handshake signal used by all communication protocols.

Symbol	Type	Responsible Level	Name	Description															
SRQ*	LVTTTL output, active low	C	Programmable module service request	Service request line, primarily for "fault" conditions. Conditions which cause SRQ* to be asserted are programmable. ²⁰															
ALM*	LVTTTL output, active low	C	Programmable module alarm	Alarm line (non-latching) for module condition. ²¹ Conditions which cause ALM* to be asserted are fully programmable. ²²															
MS*	LVTTTL input, active low	A	Module IO Select	Module selection. When this line is low, the TxD line is driven in those implementations which tri-state the TxD line such as SPI.															
TxD	LVTTTL tri-state output	A	Module's Transmit Data	Transmit line.															
RxD	LVTTTL input, output (for I2C)	A	Module's Receive Data	Receive line.															
IOMODE	LVTTTL Input	A	Communication Interface Mode Select	Physical interface selection. <table border="1" style="margin: 5px 0;"> <thead> <tr> <th>IoMode</th> <th>IoMode1</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>SPI</td> </tr> <tr> <td>1</td> <td>0</td> <td>RS232</td> </tr> <tr> <td>0</td> <td>1</td> <td>I²C</td> </tr> <tr> <td>1</td> <td>1</td> <td>TBD</td> </tr> </tbody> </table> These pins are read only on hard reset or power up.	IoMode	IoMode1	Mode	0	0	SPI	1	0	RS232	0	1	I ² C	1	1	TBD
IoMode	IoMode1	Mode																	
0	0	SPI																	
1	0	RS232																	
0	1	I ² C																	
1	1	TBD																	
IOMODE 1	LVTTTL Input	A																	
A0	LVTTTL Input	A	Physical Interface Address Selection Lines	Address lines are only read on power up or hardware reset to set device address (A2, A1, A0); A0 is the LSB.															
A1	LVTTTL Input	A																	
A2	LVTTTL Input	A																	
IOCLK	LVTTTL input	A (SPI)	I/O Interface Clock	I/O Clock for physical interfaces requiring a clock signal.															
TMS	LVTTTL input	A	JTAG Mode Select	Optional – May be defined for use during PCB testing															
TDI	LVTTTL input	A	JTAG Data In																
TDO	LVTTTL output	A	JTAG Data Out																
TCK	LVTTTL input	A	JTAG Clock																
TRST	LVTTTL input	A	JTAG Reset																

²⁰ The SRQ* line is used to signal the attention of a controller. The line is a latching signal. In the event of a fatal condition, this line is asserted. The controller needs to interrogate the module when this error occurs in order to determine the error type, hence the name "service request". This line can be cleared by issuing a command to clear the status register.

²¹ If the module's condition (such as signal quality in a CW tunable source) is affected by a hard reset of the module, this line should be asserted during the reset.

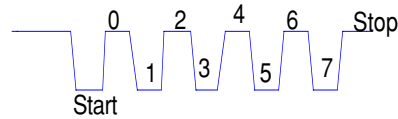
²² Alarm lines are considered non-latching and indicate an on-going alarm condition.

8.2 RS232 - PHYSICAL INTERFACE IMPLEMENTATION

The RS232 interface uses a 3-wire implementation (Tx, Rx + ground)²³.

The default baud rate (for initial communication) is 9600 baud which remains in effect otherwise changed. The maximum supported baud rate is 115.2 kbaud.

The interface is configured as 8 bit, no parity, 1 stop bit, no echo, no flow control, and is fully capable of transferring binary data. The following figure shows the timing of an RS232 signal transmitting 0xAA. The LSB is transmitted first²⁴.



The interface generates LVTTTL output signal levels.

The physical interface also provides an interface available line (IRDY*) which is used to pace communication on a packet level and is under the control of Level B.

The interface consists of the pins shown in the following table.

Table 8.2-1 RS232 Physical Interface Pins

PIN	I/O	FUNCTION															
RxD	input	LVTTTL serial input (break signal is 0v)															
TxD	output	LVTTTL serial output (break signal is 0v)															
Gnd	ground	Ground															
MS*	input	LVTTTL Module Select (RS232 Interface reset)															
IRDY*	output	LVTTTL (interface available/busy)															
IOMODE	Input	LVTTTL (Select physical interface) IOMODE and IOMODE1 determine the physical interface to be used. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>IOMODE</th> <th>IOMODE1</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>SPI</td> </tr> <tr> <td>1</td> <td>0</td> <td>RS232</td> </tr> <tr> <td>0</td> <td>1</td> <td>I²C</td> </tr> <tr> <td>1</td> <td>1</td> <td>TBD</td> </tr> </tbody> </table>	IOMODE	IOMODE1	Mode	0	0	SPI	1	0	RS232	0	1	I ² C	1	1	TBD
IOMODE	IOMODE1		Mode														
0	0		SPI														
1	0		RS232														
0	1		I ² C														
1	1	TBD															
IOMODE1	Input																

²³ This physical interface may be better described as an ASYNC interface but is usually referred to by the industry as an RS232 implementation.

²⁴ It is assumed that neither hardware or software flow control is needed to transfer a packet at the maximum data rate.

Figure 8.2-1 RS232 Timing shows the interface timing. The IRDY* indicates that a command is in process and that a packet will be returned. Once the packet is returned, the IRDY* line is asserted. The MS* line is not required to use the RS232 interface. It can be used to reset the serial interface and clear the I/O buffers. By default, de-selecting the module (MS*=high) will reset the interface baud rate to the default. This behavior can be modified by changing the IOCAP register.

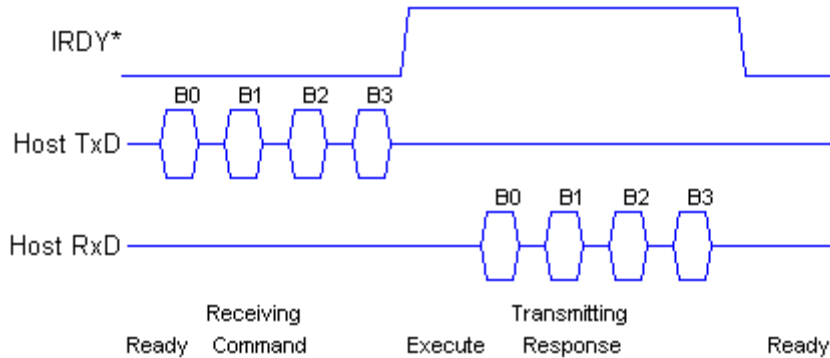


Figure 8.2-1 RS232 Timing

Note that the de-asserting the MS* line does not tri-state the Tx line.

The following figure (Figure 8.2-2) also shows a case in which a CE or XE (communications error or execution error) is asserted. In this case the SRQ* line is not asserted for the default RS232 configuration. The conditions reflected by the assertion of SRQ* line are configurable (SRQ* Trigger register (0x28) which can include XE (execution error) or CE (communication error)).

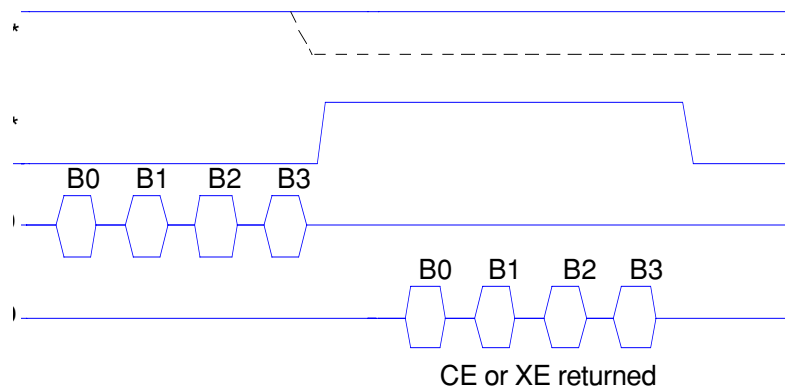


Figure 8.2-2 RS232 Communication or Execution Error Timing

If the command given was a tuning command, the ALM* line would be temporarily asserted. See Figure 8.2-3.

The alarm line is asserted only while the alarming condition exists. The conditions reflected by the assertion of the ALM* line are configurable. A typical use for the ALM* signal would be to assert when the module's output is not "locked" to a channel (ALM* = LOCK). This behavior is shown in Figure 8.2-3. The conditions for which this occurs are fully programmable.

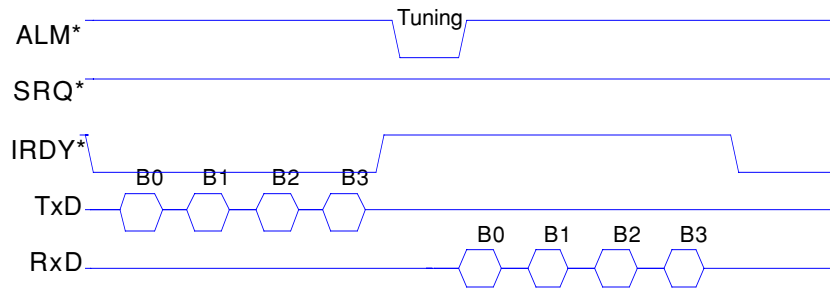


Figure 8.2-3 Alarm Timing

The IOCap register has the following format when the RS232 interface is selected and assumes default values upon power up or hardware reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0		RMS	0x0		ASCII		Current Baud Rate			Supported Baud Rates					

Bits 0-3 – Baud rates supported by the module. (Not writable)

- 0x00 – 9600
- 0x01 – 19200
- 0x02 – 38400
- 0x03 – 57600
- 0x04 – 115200
- 0x05 – 0x0F – Undefined

Bits 4-7 – The module’s currently configured baud rate (writable) (default 0x00)

- 0x00 – 9600
- 0x01 – 19200
- 0x02 – 38400
- 0x03 – 57600
- 0x04 – 115200
- 0x05 – 0x0F – Undefined

Bit 8 – ASCII - Enter ASCII interface debug mode when bit 8 is a 1, and exit when bit 8 is a 0.

Bits 9-11 Reserved

Bits 12 – RMS - Configurable action upon de-assertion of MS*

- 0x0 – Baud rate will be reset to default (0x00) and input buffer cleared upon de-assertion of MS* (**default**). This bit is for RS-232 mode only it has no affect in SPI mode
- 0x1 – No change

Bits 14-15 – Reserved (default 0x00)

Note: Bit 8 of the RS232 Implementation’s IOCap register, when set to one, sets the interface to operate in ASCII mode. This feature is optional, and if not implemented, generates an XE (execution error). The response from the write IOCap is sent formatted as the interface was set to before bit 8 was set. Then the interface switches into the desired binary or ASCII mode.

Commands must then be given in the following ASCII readable format:

[R,W] <REG_NO> <DATA>. (Example: W 0x03 0x0000)

Responses will be likewise readable:

CE=[0,1] Status=[0-3] <REG_NO> <DATA> (example: CE=0 Status=0 0x03 0x0000)

To enter this mode, the command "W IOCap 0x0100" (0x010d 0x0100) is given. From a terminal keyboard, this is the essentially the equivalent of pressing the <CR> key 4 times (0x0d0d, 0x0d0d). The interface can be reset to binary mode with the appropriate *write iocap* command or by sending any non ASCII character (other than <cr>). The binary character is not lost and becomes the first byte of the binary command. This allows the interface to be easily reset should the interface accidentally be placed into ASCII mode. No checksum is done with the ASCII interface enabled. Therefore a transmission error can put the interface into binary mode. Of course, this is easily recoverable by sending 4 <cr> characters again.

The 4 character "\r" (0x0d) sequence encodes the following: W 0x0D 0x0D0D (out-bound bytes 0-3)

7	6	5	4	3	2	1	0
BIP-4 = 0x0				Reserved = 0x6			1

7	6	5	4	3	2	1	0
Register = 0x0D							

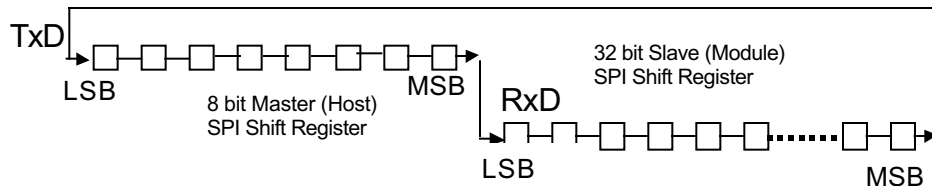
15	14	13	12	11	10	9	8
Reserved 0x0		BCMS=0	BRMS=0	Reserved 0x6			ASCII=1

7	6	5	4	3	2	1	0
Current Baud Rate=0x00				Supported Baud Rates (write ignored)=0xD			

8.3 SPI - PHYSICAL INTERFACE IMPLEMENTATION

The SPI implementation used a standard 4 wire SPI interface with an additional line for communication packet pacing and an additional line for requesting attention. Both lines can be either used to generate interrupts or can be polled.

The Serial Peripheral Interface (SPI) is essentially a shift register that serially transmits data bits to other SPI ports. During a data transfer, one SPI system acts as the "master" which controls the data flow, while the other system acts as the "slave" which has data shifted into and out of it by the master.



The master device initiates all data transactions and every

transaction is both a receive and a transmit operation. The master device transmits a new bit of data on the RxD pin and the slave device drives a new data bit on TxD pin on each active clock edge. However, only one slave may drive its output to write data back to the master at any given time.

When the master is addressing the device, the module select signal (MS*) is driven low. The SPI slave device will immediately begin sending the transmit shift register contents to the TxD pin. The slave device also simultaneously reads the receiver shift register by shifting data in from the RxD pin. Thus a read and write transaction can be carried out simultaneously.

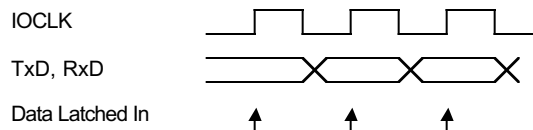


Figure 8.3-1 SPI IOCLK Timing

The SPI system consists of two data lines, a clock line, one control line, and a module select line.

Table 8.3-1 SPI Physical Interface Pin Table

PIN	SPI Name	I/O	FUNCTION
RxD	MOSI	input	LVTTTL serial input Master Out Slave In - Abbreviated MOSI, this data line supplies the output data from the master which is shifted into the input(s) of the slave(s).
TxD	MISO	output	LVTTTL serial output Master In Slave Out - Abbreviated MISO, this data line supplies the output data from a slave to the input of the master. There may be no more than one slave which is transmitting data during any particular transfer.
MS*	SS*	input	LVTTTL serial input active low Module Select* - Abbreviated MS* (SS*), this control line allows slave's TxD to be turned driven or tri-stated.
IRDY*	N/A	output	LVTTTL (interface available/busy)
IOCLK	SCLK	Input	LVTTTL serial input normally low I/O Clock - Abbreviated IOCLK (SCLK), this control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates; the SCLK line cycles once for each bit that is transmitted. This line is normally low. Inbound data is clocked on the rising edge and outbound data is clocked on the falling edge. The minimum clock speed ²⁵ is 1 MHz. All SPI devices will respond correctly at the minimum data rate. Faster data rates can be negotiated by reading the IOCAP register.
IOMODE	N/A	Input	LVTTTL serial input IOMODE and IOMODE1 determine the physical interface to be used..
IOMODE1	N/A	Input	
GND	GND	Ground	Ground

IOMODE	IOMODE1	Mode
0	0	SPI
1	0	RS232
0	1	I ² C
1	1	TBD

These pins are read only on hard reset or power up.

²⁵ A host may use any clock rate up to the maximum supported by the module. When a host first makes contact with a module, it can begin communications at a 1MHz clock rate or below to determine the maximum clock rate supported by the module. Further communication can be any clock rate up to the module's maximum supported clock rate.

The tunable device always acts as a slave device in this implementation.

The communication occurs in packets of 4 byte transfers. The MS* line is held low during the entire 4 byte transfer. Raising the line anytime during the transfer aborts the packet transmission. Upon MS* returning high, the tunable device will begin any operations that were specified in the packet. The MS* line can be raised between an in-bound read command and the corresponding out-bound response.

The following figure shows the timing for 1 byte capable host. A 4 byte capable host would have the same "Load-Read" timing as the slave.

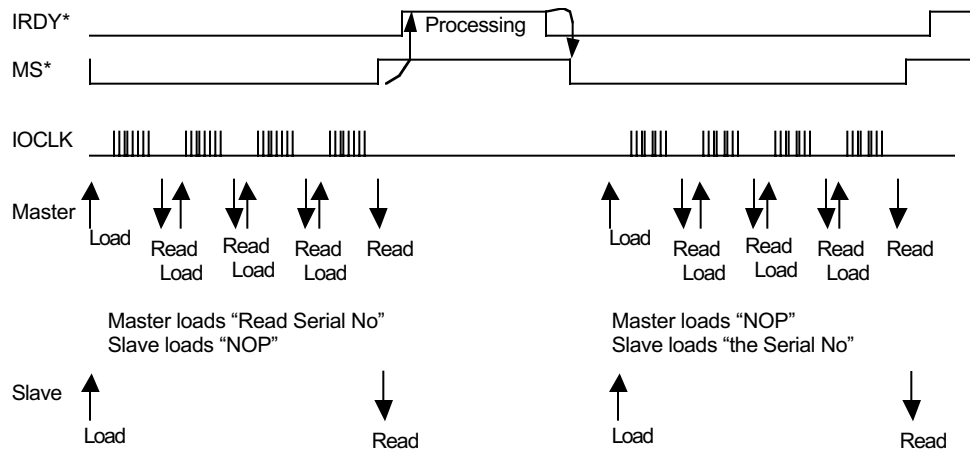


Figure 8.3-1.3-2 SPI Read (or Write) Timing, 1 Byte Host

The command is complete when the IRDY* line is low. If the command was a READ command, the response can be read. If the packet was a WRITE command, a NOP can be sent to read the status bits CE and XE.

In the event of an error (either communication error (CE) or execution error (XE)), the SRQ* (service request) line is asserted, if configured. See Figure 8.3-3.

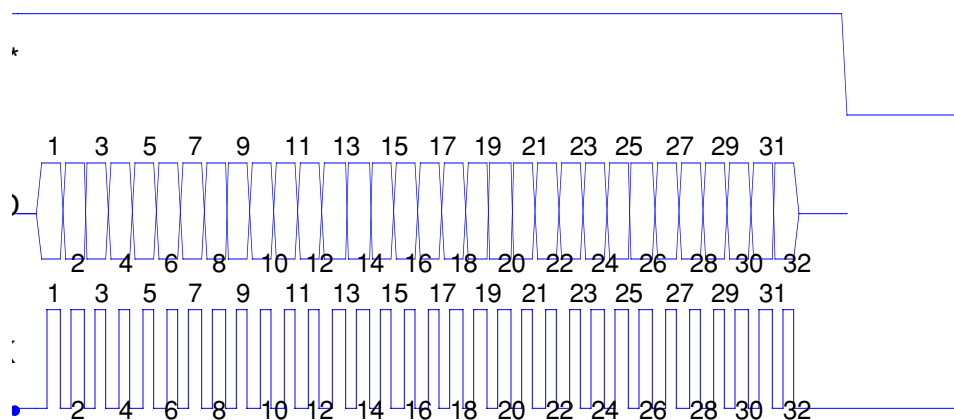


Figure 8.3-3 SPI Error Timing (Execution or Communication)

The SRQ* line stays low until the appropriate status register bits are reset by a command (the default behavior). If the condition persists after the reset, the SRQ* line will again be asserted.

The ALM* line is a non-latching version of SRQ and can be used to indicate loss of LOCK or any other programmable condition. When configured as a LOCK pin, it will be asserted during tuning.

The IOCap register has the following format when the SPI interface is selected and assumes default values upon power up or hardware reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000										SPI Data Rate					

Bits 0-5 – Maximum SPI data rate specified in MHz as an unsigned integer value. For example, the data rate maximum would be 20 MHz if Bits 5-0 are 0x14.

Bits 6-15 – Reserved (default: 0x00)

8.4 I²C-BUS - PHYSICAL INTERFACE IMPLEMENTATION

Acknowledgement: Some text and figures for this section have been obtained from the Philips I²C-Bus specification.

I²C-Bus is a simple bi-directional bus requiring only 2 bus lines – a serial data line (SDA) and a serial clock line (SCL) to communicate with multiple components attached to the bus. The I²C-Bus mechanical interface uses 5 connector pins – the clock, data line, and 3 address pins.

The data link physical layer shall comply with the Philips specifications as defined in the I²C-Bus Specification, Version 2.1, January 2000.

The 2 signals, Serial Data and Serial Clock, carry information between devices connected to the bus. Each device is recognized by a unique address and can operate as either a transmitter or receiver, depending on the function of the device. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers. A master is the device which initiates a data transfer on the bus, and generates the clock signals to permit that transfer, and terminates a transfer. At that time any other device connected to the bus is considered the slave, being addressed by the master. The I²C-Bus is a multi-master bus. More than one device capable of controlling the bus can be connected to it. The tunable module is connected as a slave only. The host is always the bus master.

Both SDA and SCL are bi-directional lines, connected to a positive supply voltage via a current-source or pull-up resistor (see Fig.8.4.1). When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function. Data on the I²C-bus can be transferred at rates of up to 100 kbit/s in the Standard-mode, up to 400 kbit/s in the Fast-mode, or up to 3.4 Mbit/s in the High-speed mode. The IOCAP register can be used to declare the maximum supported data rate. Note that the I²C specification calls for the fast and high-speed data rates to be compatible with the slow or standard mode. The number of interfaces connected to the bus is solely dependent on the bus capacitance limit of 400 pF.

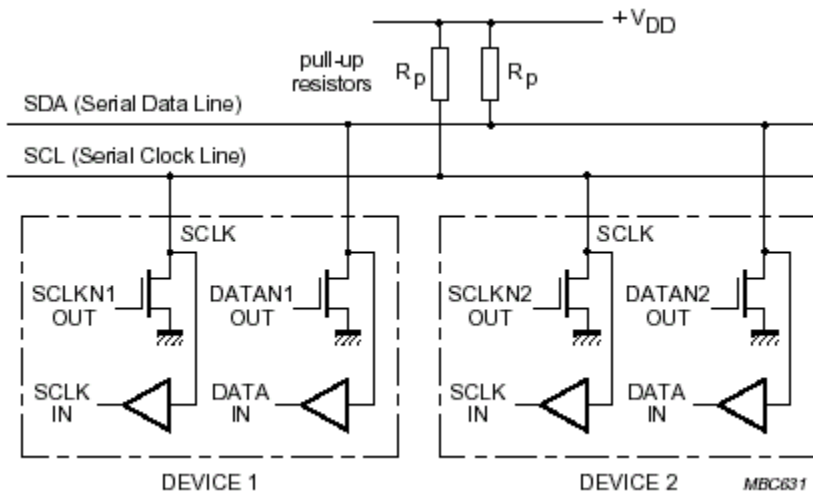


Figure 8.4-1 Connection of Standard and Fast mode devices to the I²C-Bus.

The data on the SDA line must be stable during the HIGH period of the clock. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW (see Fig.8.8.2).

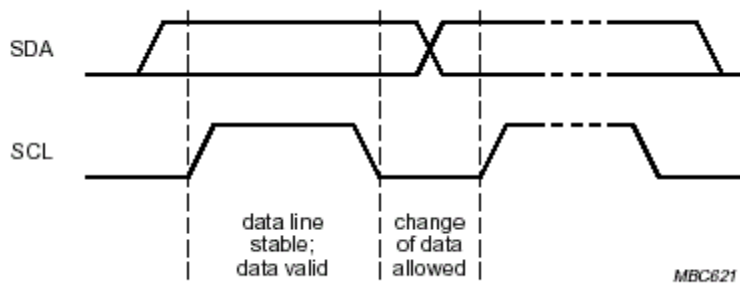


Figure 8.4-2 Bit transfer on the I²C-Bus

Within the procedure of the I²C-Bus, unique situations arise which are defined as START (S) and STOP (P) conditions (see Fig.8.4.3). A HIGH to LOW transition on the SDA line while SCL is HIGH is one such unique case. This situation indicates a START condition. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition. START and STOP conditions are always generated by the master. The bus is considered to be busy after the START condition. The bus is considered to be free again a certain time after the STOP condition. The bus stays busy if a repeated START (Sr) is generated instead of a STOP condition. In this respect, the START (S) and repeated START (Sr) conditions are functionally identical. Detection of START and STOP conditions by devices connected to the bus is easy if they incorporate the necessary interfacing hardware. However, microcontrollers with no such interface have to sample the SDA line at least twice per clock period to sense the transition.

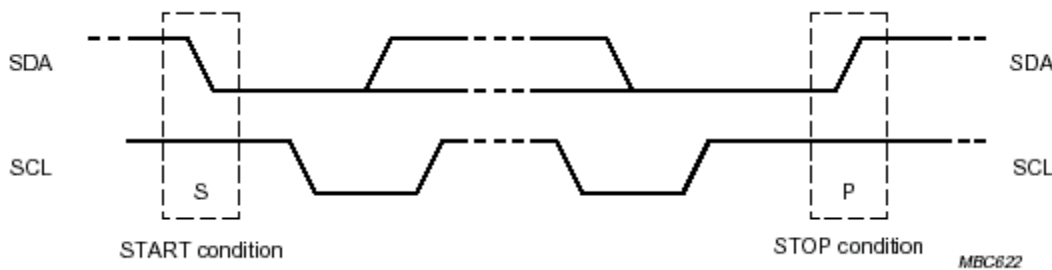


Figure 8.4-3 Start and Stop conditions

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unrestricted. However, for the tunable module source – packets are restricted to 4 bytes and are governed by layers B and C. Each byte has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first (see Fig.8.4.4). If a slave can't receive or transmit another complete byte of data until it has performed some other function, for example servicing an internal interrupt, it can hold the clock line SCL LOW to force the master into a wait state. Data transfer then continues when the slave is ready for another byte of data and releases clock line SCL. As a slave is capable of seizing the bus, a maximum wait state is imposed. A byte wait state shall not exceed the time of 1 minimum rate byte (9 bits including the acknowledge or 100 microseconds with a minimum rate of 90 kHz). The host shall not pause more than 1 minimum rate byte (9 bits including the acknowledge) between bytes of an ongoing message. The use of wait states is discouraged. A tunable module cannot use the clock line to indicate that a command is processing, before issuing a response. The module generates the IRDY* line to indicate that a command is executing. The host can either monitor the IRDY* line or poll the module to determine when the command is complete.

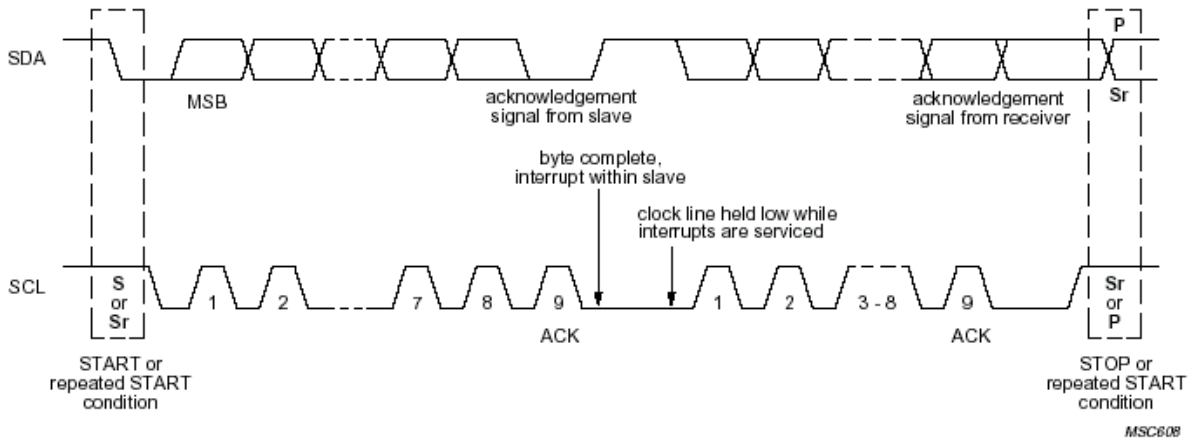


Figure 8.4-4 Data Transfer on the I²C-Bus

Data transfer with acknowledge is obligatory. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (HIGH) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse (see Fig.8.4.5). Of course, set-up and hold times must also be taken into account. Usually, a receiver which has been addressed is obliged to generate an acknowledge after each byte has been received. When a slave doesn't acknowledge the slave address (for example, it's unable to receive or transmit because it's performing some real-time function), the data line must be left HIGH by the slave. The master can then generate either a STOP condition to abort the transfer, or a repeated START condition to start a new transfer. If a slave-receiver does acknowledge the slave address but, some time later in the transfer cannot receive any more data bytes, the master must again abort the transfer. This is indicated by the slave generating the not-acknowledge on the first byte to follow. The slave leaves the data line HIGH and the master generates a STOP or a repeated START condition. If a master-receiver is involved in a transfer, it must signal the end of data to the slave-transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave-transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

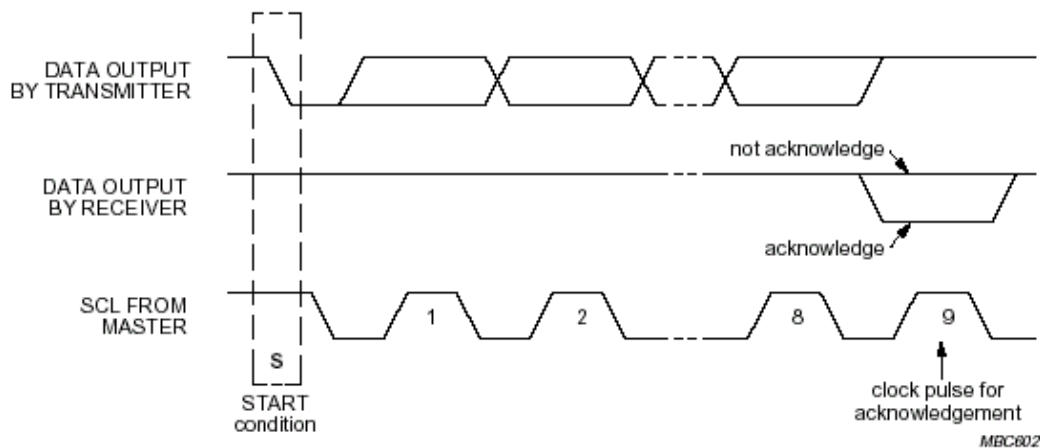


Figure 8.4-5 Acknowledge on the I²C-Bus

Bus arbitration and clock synchronization are considered beyond the scope of this document, and are described in the Philips I²C-Bus Specification. The tunable module may act only as a slave.

Data transfers follow the format shown in Figure 8.4.6. After the START condition (S), a slave address is sent. A device only responds to an I2C message when its own address and the address in the I2C frame match. The definition of the 7-bit address shall be xxxxyyy. The four most significant bits, xxxx, are specific to tunable sources and are defined as 1100 (allocated by Philips). The least significant bits, yyy, are specific to an individual module and are defined by the address pins A2, A1, A0. The address is then defined as “1 1 0 0 A2 A1 A0.” This 7-bit address is followed by an eighth bit that is a data direction bit (R/W) - a ‘zero’ indicates a transmission (WRITE), a ‘one’ indicates a request for data (READ). A data transfer is always terminated by a STOP condition (P) generated by the master. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition (Sr) and address another slave without first generating a STOP condition. Various combinations of read/write formats are then possible within such a transfer, which are described in the Philips I²C-Bus specification.

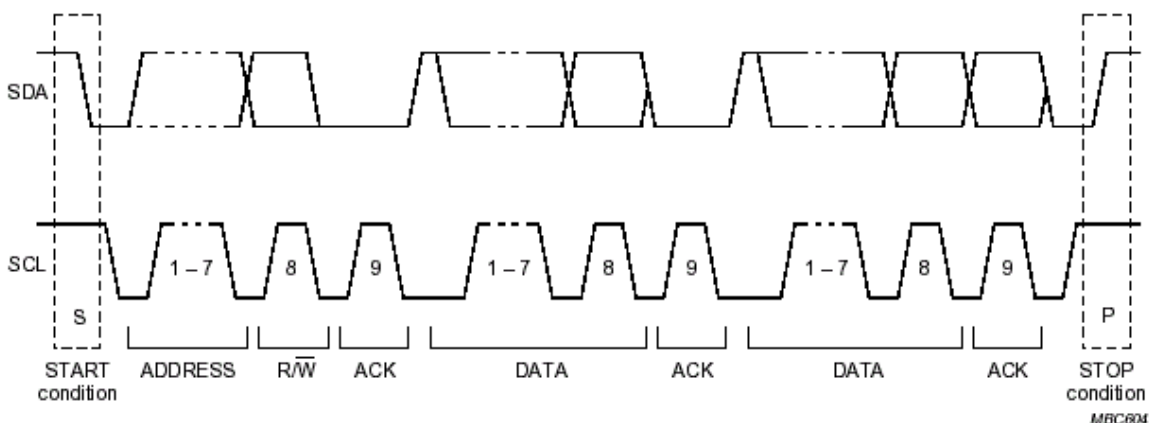


Figure 8.4-6 A complete data transfer

The I²C-Bus system requires 1 data line, 1 clock line. Each Tunable Laser module requires 3 address pins, which are hard wired to a fixed address code. I²C-Bus pins are defined in table 8.4.1 below.

Table 8.4-1 I²C-Bus Physical Interface Pin Table

PIN NO.	PIN	I ² C Name	I/O	FUNCTION															
16	RxD	SDA	Input/output	LVTTTL serial input/output Serial Data line															
20	IOCLK	SCL	Input/output	LVTTTL serial input/output Serial Clock line															
26	N/A	A0	Input	LVTTTL input Address lines are only read on power up or hardware reset to set device address (A2, A1, A0); A0 is the LSB.															
28	N/A	A1	Input																
30	N/A	A2	Input																
22	IOMODE	N/A	Input	IOMODE and IOMODE1 determine the physical interface to be used. <table border="1"> <thead> <tr> <th>IOMODE</th> <th>IOMODE1</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>SPI</td> </tr> <tr> <td>1</td> <td>0</td> <td>RS232</td> </tr> <tr> <td>0</td> <td>1</td> <td>I²C</td> </tr> <tr> <td>1</td> <td>1</td> <td>TBD</td> </tr> </tbody> </table> These pins are read only on hard reset or power up.	IOMODE	IOMODE1	Mode	0	0	SPI	1	0	RS232	0	1	I ² C	1	1	TBD
IOMODE	IOMODE1	Mode																	
0	0	SPI																	
1	0	RS232																	
0	1	I ² C																	
1	1	TBD																	
24	IOMODE1	N/A																	
4	MS*	N/A	Input	LVTTTL input Not required. Can be used to reset communications and clear level B communication buffers.															
18	IRDY*	N/A	Output	LVTTTL output, active low Controlled by level B. Indicates when the unit is still executing a previous command and not ready for the response to be read.															

An I²C-Bus interface does not require use the IRDY* (interface available/busy) pin although it is supported²⁶. The MS* (module select) pin is used only to clear in the input buffer and reset the communications interface²⁷. The bus is considered busy after generating a Start command, and is free after a Stop command. The module can therefore control the bus using the SDA and SCL pins. The packet length is a fixed size and is framed by the Start and Stop symbols. A module is selected using the I²C-Bus slave address bits, and does not require a separate select line.

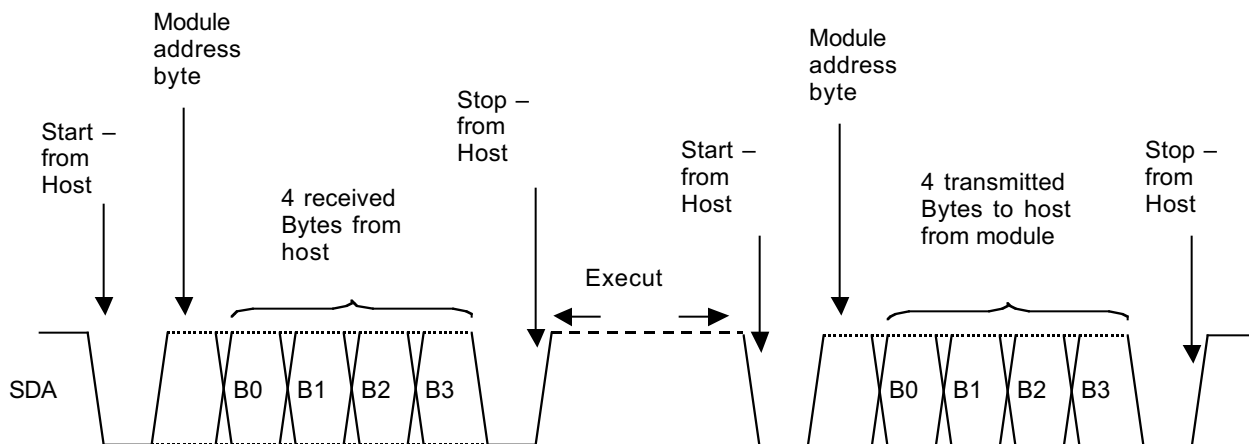


Figure 8.4-7 Receipt and Transmit of 4 Byte packets

The use of the Start and Stop commands is demonstrated in figure 8.4.7 above, which shows how the 4byte packets are received by the module and transmitted from the module.

The first transfer of the 4 bytes is an I²C “W” command independent of whether the level C command is a read or write. When the module is finished processing, the host issues an I²C “R” command to read the response. Level C defines a maximum timeout for commands. If the module will exceed this time during the execution, the module will return a response which includes indication of a pending operation.

In either case, the host must either check the IRDY* line or poll the module through the I²C interface to determine if the module is ready with a response. The use of the IRDY* by the host is purely optional but is provided for compatibility with Level B as well as allowing for faster transfer rates by eliminating the bus transfers during polling²⁸.

The IOMODE register is defined as follows for the I²C bus.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000													Max Supported I ² C Data Rate		

²⁶ The IRDY* line can be used by the host to determine that the module is currently executing a command and not available for communication. This can be useful if the host is restarted while the module is executing a (long) command. If the IRDY* line not used, a polling scheme is required.

²⁷ The behavior is preserved to keep the Level B and C layers identical although it is not required for normal I²C operation.

²⁸ This also increases the availability of the I²C bus for use with other devices.

Bits 2-0 – Maximum supported I²C data rate specified in the table below. Default manufacturer specific.

I ² C Data Rate Value (Bits 2,1,0)	I ² C Data Rate	
0x0	Standard	100kbits/s
0x1	Fast	400 kbits/s
0x2	High Speed	3.4Mbits/s
0x3-0x7	Reserved	

Note that a given interface speed is compatible with all lower valued data rates.

Bits 15-3 – Reserved (default: 0x00)

8.5 PHYSICAL PINS REQUIRED FOR SPECIFIC LEVEL A IMPLEMENTATIONS

This section outlines the specific pin requirements for the tunable lasers covered by this document

8.5.1 PHYSICAL INTERFACE FOR CW TUNABLE LASER (DEVTYPE="CW LASER")

The connector numbering is shown in Figure 11.1-3 "Y"-Pluggable Connector Dimensions and Figure 11.1-4 "Z"-Pluggable Connector Dimensions. The following table describes the pin-out and pin functions.

Table 8.5-1 Pin Assignment Table for "CW-Laser"

Pin Number	Symbol	Type	Name	Description
1	Vcc	Power	Vcc	Power supply for tunable laser 3.3 V
3				
5				
7				
9	Gnd	Ground	Ground	Ground for tunable laser
11				
13				
15				
17	TRST	LVTTTL input	TRST	JTAG Reset (for use by manufacturer) (optional)
19	TDO	LVTTTL output	TDO	JTAG Data Out (for use by manufacturer) (optional)
21	TDI	LVTTTL input	TDI	JTAG Data In (for use by manufacturer) (optional)
23	TMS	LVTTTL input	TMS	JTAG Mode select (for use by manufacturer) (optional)
25	TCK	LVTTTL input	TCK	JTAG Clock (for use by manufacturer) (optional)
27	--	--	Manufacturer Specific	Manufacturer Specific. Although Manufacturer Specific pins can provide additional flexibility to meet system needs, they can also be a source of interoperability issues. When utilizing manufacturing specific pins, care must be taken by the integrator to watch for potential conflicts
29				
31				
33				
35				
37				
39				
2	DIS*	LVTTTL input, active low	Disable module's optical output	<p>Purpose: Provide hardware control to kill laser output.</p> <p>Initial State: Any – user application specific</p> <p>Action: High = laser output controlled by protocol Low = laser output OFF</p> <p>Resultant State: When DIS* asserted, communication protocol is ON, software enable (SENA) reset.</p> <p>Attributes: Bypasses communication protocol to turn laser OFF. Re-enabling of the laser requires setting SENA²⁹. Otherwise does not interfere with module settings. Module "remembers" settings. See section 10.2.1.1 for Lasers</p>
4	MS*	LVTTTL input, active low	Module I/O interface selection	<p>Purpose: Provide hardware control to select IO interface.</p> <p>Initial State: Any – user application specific</p> <p>Action: High = Interface not active, TxD may be tri-state Low = IO interface selected</p> <p>Resultant State: Communication can be commenced. Upon de-assertion, interface may be reset (soft reset) and packet terminated.</p> <p>Attributes: Provides ability to "bus" tunable and reset packet framing.</p>

²⁹ When the laser disable pin is low, the optical output is attenuated. This could be done with any technique such as by blocking the beam or by turning off the laser diode current(s). Neither the amount of attenuation nor the speed at which the beam is sufficiently attenuated is specified in this document since these are considered performance specifications.

Pin Number	Symbol	Type	Name	Description
6	RST*	LVTTTL input, active low	Hard reset while low	<p>Purpose: Disables laser output and holds the module in RESET</p> <p>Initial State: Low</p> <p>Action: Laser OFF, TEC OFF, Module CPU held in RESET, Communication protocol is OFF</p> <p>Resultant State: High. Must remain high for laser to operate</p> <p>Attributes: When active, lowest current draw from the module. On low to high transition, module state is latched in, such as with the IOMODE pin.</p>
8	SRQ*	LVTTTL output, active low	Programmable module service request	<p>Purpose: General purpose service request.</p> <p>Initial State: High (No service requested)</p> <p>Action: Generates request for service as required to report a variety of conditions by setting line low. SRQ* is asserted when the result of the status (0x21) OR'd with srq trigger (0x28) is non-zero. See register 0x28 (§10.2.1.18) for default.</p> <p>Resultant State:</p> <ul style="list-style-type: none"> · Communication protocol is ON · SRQ* conditions can be read and cleared through interface <p>Attributes: SRQ conditions (and limits) are software configurable and can be re-configured by the user through the interface. Status bits must be cleared to de-assert SRQ*. Line is latching.</p>
10	FATAL*	LVTTTL output, active low	Fatal condition	<p>Purpose: To indicate FATAL condition has been triggered</p> <p>Initial State: High</p> <p>Action: Reports a variety of FATAL conditions by setting line low. FATAL* is asserted when the result of the fatal status (0x20) OR'd with FATAL trigger (0x29) is non-zero. See register 0x29 (§10.2.1.19) for default.</p> <p>Resultant State:</p> <ul style="list-style-type: none"> · Laser output is OFF, if configured (SDF bit in 0x33) · Communication protocol is ON <p>Attributes: Alarm sources and limits are software configurable and can be re-configured by the user through the interface. Line is latching.</p> <p>Default fatal alarm conditions:</p> <ul style="list-style-type: none"> · Optical output power alarm condition · Frequency error condition · Thermal alarm condition · Technology specific alarm conditions
12	ALM*	LVTTTL output, active low	Programmable module non-latching alarm (a.k.a. LOCKED)	<p>Purpose: Indicates laser is not in specified state</p> <p>Initial State: Dependent upon ALM* mask and laser's initial state.</p> <p>Action: Asserted (low) for whenever a variety of conditions occur by setting line low. ALM* is asserted when the result of the alarm (0x21) OR'd with alarm mask (0x2A) is non-zero. See register 0x28 (§10.2.1.20) for default. Line is cleared when condition goes away. Bit ADT in Register 0x33 (Module config) indicates that ALM* should be asserted when the appropriate bit in the alarm register (0x21 or'd with 0x2A) is set during tuning.</p> <p>Default Action: Asserts that laser is LOCKED on channel when high and low during tuning or other alarm conditions.</p> <p>Resultant State: High – when laser is in desired state</p> <p>Attributes: Positive indication to user that laser is in desired (programmable) state. This line is non-latching and may “chatter”. Delays in assertion and de-assertion are manufacturer specific.</p>
14	TxD	LVTTTL tri-state output	Module's transmit data	<p>Purpose: Transmit outbound packets from module</p> <p>Polarity: Dependent upon physical interface selected.</p> <p>Attributes: May be tri-stated for appropriate interface.</p>
16	RxD	LVTTTL, I/O	Module's receive data	<p>Purpose: Receive inbound packets from host; SDA for I2C</p> <p>Polarity: Dependent upon physical interface selected.</p> <p>Attributes: Receive ignores data when module not selected</p>
18	IRDY*	LVTTTL output, active low	Communication interface ready	<p>Purpose: Signals that the module's I/O interface is ready to receive a packet.</p> <p>Polarity: Ready asserted when IRDY* is low.</p> <p>Action: Upon being low, data may be transmitted to the module (consistent with the conditions of specific interface in use).</p>
20	IOCLK	LVTTTL input	I/O interface clock	<p>Purpose: IOCLK pin for serial interfaces such as SPI.</p> <p>Polarity: Polarity defined in SPI §8.3 and I2C §8.4.</p> <p>Action: Specific behavior is specified in SPI §8.3 and I2C §8.4. No interaction with the RS232 interface.</p>

Pin Number	Symbol	Type	Name	Description															
22	IOMODE	LVTTTL input	Communication interface mode selection	Purpose: To select one of three physical communication interfaces. Polarity: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>IOMODE</th> <th>IOMODE1</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>SPI</td> </tr> <tr> <td>1</td> <td>0</td> <td>RS232</td> </tr> <tr> <td>0</td> <td>1</td> <td>I²C</td> </tr> <tr> <td>1</td> <td>1</td> <td>TBD</td> </tr> </tbody> </table>	IOMODE	IOMODE1	Mode	0	0	SPI	1	0	RS232	0	1	I ² C	1	1	TBD
IOMODE	IOMODE1	Mode																	
0	0	SPI																	
1	0	RS232																	
0	1	I ² C																	
1	1	TBD																	
24	IOMODE1	LVTTTL input		Action: Must not change during low to high transition of RST* when the MODE is latched in. No effect if changed later. These pins are read only on hard reset or power up. Attributes: Allows user hardware to select physical interface															
26	A0	LVTTTL input	Physical Interface Address Selection Lines	Address lines are only read on power up or hardware reset to set device address (A2, A1, A0); A0 is the LSB.															
28	A1	LVTTTL input																	
30	A2	LVTTTL input																	
32	--	Reserved		Reserved pins are unavailable for vendor definition. These are reserved for future OIF implementation agreements.															
34																			
36																			
38																			
40																			

The following table describes the electrical supply and logic levels for the physical interface.

Table 8.5-2 Electrical Interface Levels

Parameter	Symbol	Min	Typ	Max	Unit
Supply voltage	V _{cc}	3.15	3.30	3.45	V
Supply current				4000	mA (Peak ³⁰)
Input voltage, low	V _{IL}	0		0.8	V
Input voltage, high	V _{IH}	2		3.45	V
Output voltage, low (I _{OL} = 4 mA)	V _{OL}	0		0.6	V
Output voltage, high (I _{OH} = -4 mA)	V _{OH}	2.4		V _{cc}	V
Power supply noise (for power supplied to the module) (100Hz to 20MHz)				1	%rms

³⁰ The instantaneous current cannot exceed 4 amps.

9 LEVEL B – CONGESTION CONTROL AND ERROR HANDLING IMPLEMENTATION

This layer may require some physical interface dependencies.

9.1 CHECKSUM

Each in-bound and out-bound packet contains a 4 bit checksum. The checksum is computed over all the bits being encapsulated using a BIP-4 checksum.³¹ A CRC-16 checksum is also available using registers 0x11-0x13.

9.2 IN-BOUND PACKET FORMAT

The in-bound packets are 4 bytes long. The 28 bit command packet is preceded with 4 zero bits (to make a 32 bit value) before the 4 bit checksum computed over the command packet.

31	30	29	28	Bits 27:0			
Checksum				Command Packet			

9.3 OUT-BOUND PACKET FORMAT

The out-bound packets are 4 bytes long. The 26 bit response packet is preceded with 6 zero bits (to make a 32 bit value) before the 4 bit checksum computed over the response packet.

31	30	29	28	27	26	Bits 25:0	
Checksum				CE	Response	Response Packet	

CE: The CE, Communication Error, bit is set to 1 if the previous in-bound packet contained a checksum or other physical layer transmission error. If no error occurred in the previous packet, the CE bit is set to zero.

As an example, an in-bound RS232 transfer may have a garbled stop bit which would cause CE to be set to 1.

A response packet with CE=1 can be returned by reading a command response. When CE=1, a SRQ* will be generated. To check the CE bit, any command can be sent. A likely action would be to "Read the status register".

The CE bit is cleared for subsequent packets immediately after the out-bound packet is sent.

Response:

The response bit is normally set to 1. It indicates that the out-bound packet is a response to a request to read.

Some physical interfaces may need to return a packet when there is no response. In this case, Level B will set the response bit to zero.³²

³¹ The BIP4 checksum is computed by xor'ing all the bytes in the packet together and then xor'ing the left nibble of the result with the right nibble of the result. This is similar to the BIP-8 checksum specified in SONET.

³² This condition occurs with the SPI interface. Note that before the SPI clock is started, the slave must load a value into the SPI register. In fact, this value should be loaded before the MS* is asserted. If there is no pending response, then the returned packet will contain no data. The Response bit would be set to "0" to indicate that the return packet contains no useful data. In the case of the RS232 interface, the module returns the 4 byte packet when the module is ready to respond with data. Therefore, the RS232 always as the response bit set to "1".

9.4 CONGESTION CONTROL

The interface uses the IRDY* (Interface Ready*) line to signal that the interface is available for the next packet.

The host watches the IRDY* line. When it is low, the host can begin transmission of a command to the module. The IRDY* line goes high after the last byte of the packet is received. At the end of the packet, the module can decode it and begin execution. The IRDY* line will remain high and return low after the execution completes or terminates in error.

The module is never allowed to transmit an out-bound packet without a request from the host.

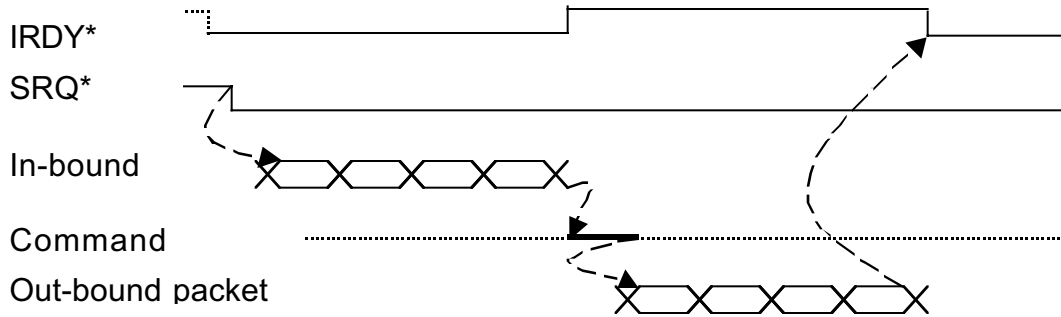


Figure 9.4-1 Ready Timing (Shown for the RS232 case)

The module can request attention from the host by asserting the SRQ* (Attention*) line. The host will respond by sending a command to read the module's status register³³. The return packet will contain the CE (communication error) bit as well as the status register's contents. The host can use this information to determine the appropriate course of action. The SRQ* line is reset when the host clears the status bits register with a write status command.

9.5 MODULE SELECTION

The module selection line, MS*, plays different roles depending on the interface type selected. In the RS-232 interface the MS* line is used to reset the module communication interface (i.e. baud rate, receive and transmit buffers).

The SPI interface allows the MS* line to be used to tri-state any output lines and also turn the receiver off. This will allow many SPI devices to be driven on the same lines.

9.6 PACKET FRAMING

The packet framing is accomplished by utilizing fixed size packets in conjunction with the IRDY* line. Four error scenarios can occur:

1) Host has delay between sending bytes to the module.

Module doesn't time out on reception from host. This is not a problem.

2) Host sends 2 bytes of a packet and then 4 bytes of a new packet.

Module will assume it has a complete packet after it receives the 2nd byte from the second packet. The module will detect problems with the packet so the packet response going back to the host will have the CE bit set. It is the responsibility of the host to resynchronize the packets.

3) Module responds with too few or too many bytes.

Too few bytes could occur because either the host times out or IRDY* goes low before 4th byte received. Too many bytes are detected after the host has received the 5th byte and IRDY* never went low. In either case, the host must timeout and monitor the byte count to determine the packet integrity and also if a synchronization problem exists. The host would need to use the MS* line to reset the communication channel on the module.

4) The module receives data when it isn't ready (IRDY* isn't low).

The module will ignore bytes received when the IRDY* line is high. Since these bytes are ignored no response is generated.

³³ The module's status register is defined in section §10.2.1.1 (register 0x20,0x21). It contains bits, which when asserted, indicate various error and warning conditions that had existed or are currently occurring.

10 LEVEL C – REGISTERS FOR SPECIFIC IMPLEMENTATIONS

At the current time, only the Tunable CW Laser command set is being defined.

10.1 DEVICE TYPE ASSIGNMENTS

The following table is a list of the DevTyp register (0x01) strings which are returned for the specific devices. The strings are case sensitive and null terminated.

Table 10.1-1 Device Type Assignments

Device	DevTyp Value (Byte Length)	AEA DevTyp String	Maximum Time Before Response Returned ³⁴
CW Tunable Laser	0x08	"CW Laser"	200 ms
--	--	--	

10.2 CW TUNABLE LASER (DEVTYPE="CW LASER")

The CW laser Level C implementation is identified by the DevTyp register (0x01) returning the string "CW Laser" through the AEA (automatic extended addressing) mode.

The commands detailed in the following section are the basic commands required by all tunable lasers. Manufacturers can extend the command set through the assignment of additional manufacturer specific registers.

10.2.1 TUNABLE CW LASER SOURCE REGISTER ASSIGNMENTS

The following table specifies the registers for tunable laser sources. Group 0x20 through 0x2F are alarm related registers, 0x30 through 0x3F are configuration registers, 0x40 through 0x4F are status registers and 0x50 through 0x5F are factory set registers.

Table 10.2-1 Minimum Required Register Descriptions for Tunable Laser Sources

Register Number	Register Name	Read / Write	AEA	Description
0x20	Status, Fatal	R/W		Contains reset status, optical faults and alarms, and enable status.
0x21	Status, Warning	R/W		Contains reset status, warning optical faults and alarms, and enable status.
0x22	Power Fatal Threshold	RW		Returns/Sets the threshold for the output power FATAL condition encoded as $\pm\text{dBm} \times 1000$
0x23	Power Warning Threshold	RW		Returns/Sets the threshold for the power warning encoded as $\pm\text{dBm} \times 1000$
0x24	Freq Fatal Threshold	RW		Returns/Sets the threshold for the frequency FATAL condition encoded as $\pm\text{GHz} \times 10$
0x25	Freq Warning Threshold	RW		Returns/Sets the threshold for the frequency error warning encoded as $\pm\text{GHz} \times 10$
0x26	Thermal Fatal threshold	RW		Returns/Sets the threshold for thermal deviations ($> \pm^\circ\text{C} \times 100$) at which FATAL is asserted.
0x27	Thermal warning threshold	RW		Returns/Sets the threshold for thermal deviations ($> \pm^\circ\text{C} \times 100$) at which a warning is asserted.

³⁴ Any command which requires longer than this time interval to complete will return a response and set the status bits (bits 1:0) in outbound (response) byte 0 to 0x03 (command not complete, command is pending).

Register Number	Register Name	Read / Write	AEA	Description
0x28	SRQ* Triggers	RW		Indicates which bits in the Fatal & Warning status registers, 0x20-0x21, cause a SRQ condition and asserts the SRQ* line.
0x29	FATAL* Triggers	RW		Indicates which bits in the Fatal & Warning status register, 0x20-0x21, cause a FATAL condition and asserts the FATAL* line.
0x2A	ALM* Triggers	R./W		Indicates which bits in the status registers, 0x20, 0x21, cause an alarm condition and asserts the alarm line (Default behavior asserted whether laser is LOCKED on frequency.
0x2B – 0x2F	Reserved			Reserved for OIF alarm registers
0x30	Channel	R/W		Setting valid channel causes a tuning operation to occur.
0x31	Optical Power setting ³⁵	RW		Sets the optical power set point as encoded as dBm*1000
0x32	Reset	RW		Writing a 1 to bit 0 causes a software reset
0x33	Module Configuration	RW		Various module configurations
0x34	GRID	RW		Allows the grid spacing to be set for channel numbering.
0x35	1 st channel- THz portion of frequency	RW		Allows the first channel's frequency to be defined for channel numbering.
0x36	1 st channel- GHZ portion of frequency	RW		Allows the first channel's frequency to be defined for channel numbering.
0x37 – 0x3F	Reserved			Reserved for OIF configuration registers
0x40	Laser Frequency- THz portion of frequency	R		Returns channel's frequency as THz
0x41	Laser Frequency- GHZ portion of frequency	R		Returns channel's frequency as GHZ
0x42	Optical Power ³⁶	R		Returns the optical power encoded as dBm*1000
0x43	Current Temperature	R		Returns the current temperature (monitored by the temperature alarm ³⁷) encoded as °C*100.
0x44 – 0x4F	Reserved			Reserved for OIF status registers
0x50	Optical Power Min	R		Returns the min possible optical power setting
0x51	Optical Power Max	R		Returns the max possible optical power setting
0x52	Laser's First Frequency- THz portion of frequency	R		Laser's first frequency
0x53	Laser's First frequency- GHZ portion of frequency	R		Laser's first frequency

³⁵ The optical power is measured however the manufacturer prefers. In most cases, the optical power measurement would be an internal measurement which can be directly correlated to the power measured at the pigtail connector. In some cases, a manufacturer may choose to use a splitter and a second pigtail so that the module can make a reasonably accurate determination of the fiber coupled output power without assumptions of pigtail coupling efficiency.

³⁶ It is conceivable to have a feature to set the power on a per channel basis. In this case, the laser would need to either present this as an array (using AEA), or provide a register to define the channel that this power value refers to without impacting the current channel.

³⁷ The module may need to report more than one temperature. Vendor specific registers can be used to report the additional temperature values and alarm settings.

Register Number	Register Name	Read / Write	AEA	Description
0x54	Laser's Last Frequency- THz portion of frequency	R		Laser's last frequency
0x55	Laser's Last frequency- GHz portion of frequency	R		Laser's last frequency
0x56	Laser's minimum grid spacing	R		Laser's minimum supported grid spacing
0x57 – 0x5F	Reserved			Reserved for OIF vendor constants registers
0x60 - 0x7F	Reserved			
0x80-0xFF	Manufacturer Specific			

10.2.1.1 Status (0x20, 0x21) (r/w)

0x20 Current Status (Fatal) – Read Only							
15	14	13	12	11	10	9	8
SRQ	ALM	FATAL	DIS	FVSF	FFREQ	FTHERM	FPWR

0x20 Latched Status (Fatal) – R/W							
7	6	5	4	3	2	1	0
XEL	CEL	MRL	CRL	FVSFL	FFREQL	FTHERML	FPWRL

0x21 Current Status (Warning) – Read Only							
15	14	13	12	11	10	9	8
SRQ	ALM	FATAL	DIS	WVSF	WFREQ	WTHERM	WPWR

0x21 Latched Status (Warning) – R/W							
7	6	5	4	3	2	1	0
XEL	CEL	MRL	CRL	WVSFL	WFREQL	WTHERML	WPWRL

There are two status registers, one that primarily indicates FATAL conditions (0x20) and the other that primarily indicates WARNING conditions (0x21).

The fatal and warning flags follow the follow pattern.

Condition	FATAL Conditions		Warning Conditions	
	Latching	Non-Latching	Latching	Non-Latching
Thermal	FTHERML	FTHERM	WTHERML	WTHERM
Output Power	FPWRL	FPWR	WPWRL	WPWR
Frequency	FFREQL	FFREQ	WFREQL	WFREQ
Vendor Specific Fault	FVSFL	FVSF	WVSFL	WVSF

The other status flags are defined as follows:

Condition	Latched Flag	Non-Latching Flag
SRQ* asserted	None	SRQ
ALM* asserted	None	ALM
FATAL* asserted	None	FATAL
DIS* asserted	None	DIS
Execution error asserted	XEL	XE flag in out-bound byte 0
Communications error asserted	CEL	CE flag in out-bound byte 0
Module Reset asserted	MRL	None
CR asserted	CRL	None

Bits 15:8 in the status registers are non-latching and indicate the current module condition. These bites cannot be cleared. Writing to these bits (0xFF00) does not cause an error.

Bits 7:0 in the status registers are latching and indicate whether any of these conditions that have occurred since the last time the status registers were cleared. These bits can be cleared by writing a 0x00FF to the status registers.

Bit 15: SRQ – Service Request Bit (read –only) (default 0)

The SRQ bit is read only. It reflects the state of the module's SRQ* line. When the SRQ* line is asserted (low or zero), this bit is set to 1. The SRQ* line is fully configurable through the SRQ* trigger register 0x28.

Bit 14: ALM – ALARM Flag bit (read-only) (default 0)

The ALM bit is read only. It reflects the state of the module's ALM* line. When the ALM* line is asserted (low or zero), this bit is set to 1. The conditions which set the ALM* line are fully configurable through the alarm trigger register (0x2A).

Bit 13: FATAL – FATAL alarm bit (read-only) (default 0)

The FATAL bit is read only. It reflects the state of the module's FATAL* line. When the FATAL* line is asserted (low or zero), this bit is set to 1. The conditions which set the FATAL* line are fully configurable through the fatal trigger register (0x29).

Bit 12: DIS – Module's output is hardware disabled (read-only)

The module's laser output disable bit is read only and represents the state of the hardware disable pin (DIS*). When set to one, the module is "hardware" disabled. When the DIS* pin is set to zero, the SENA bit is also cleared. Therefore when DIS* is set to one, the module does not re-enable the output until the SENA is also set. Any state change in DIS can cause SRQ* to be asserted if the appropriate SRQ* trigger is set.³⁸

1: Module disabled (DIS* line is low)
0: DIS* line is high

Bit 11: FVSF, WVSF – Vendor Specific Fault (read-only) (default 0)

The FVSF bit (0x20) is set to 1 whenever a fatal vendor specific condition is asserted. The WVSF bit (0x21) is set to 1 whenever a warning vendor specific condition is asserted. If either of these bits are set, the vendor will have a register defined which contains vendor specific fault conditions.

Bit 10: FFREQ & WFREQ – Frequency Fatal and Warning (read-only) (default 0)

The FFREQ bit (0x20) reports that the frequency deviation has exceeded the frequency fatal threshold (0x24) while WFREQ bit (0x21) reports that the frequency deviation has exceeded the frequency warning threshold (0x25).

When bit 10 is 1, it indicates that the frequency deviation threshold is being exceeded. When bit 10 is 0, the frequency deviation threshold is not being exceeded.

³⁸ The operation ensures that a tuning operation only occurs under s/w control. The primary purpose of the DIS* pin is to rapidly disable the laser output. In some implementations, the tuning command can make the module de-assert IRDY* during the tuning command. The behavior of clearing the SENA bit allows the IRDY* line to be solely activated by software transfers and simplifies possible sources of confusion.

Bit 9: FTHERM & WITHERM – Thermal Fatal and Warning (read-only) (default 0)

The FTHERM bit (0x20) reports that the thermal deviation has exceeded the thermal fatal threshold (0x26) while WITHERM bit (0x21) reports that the thermal deviation has exceeded the thermal warning threshold (0x27).

When bit 9 is 1, it indicates that the thermal deviation threshold is being exceeded. When bit 9 is 0, the thermal deviation threshold is not being exceeded.

Bit 8: FPWR & WPWR – Power Fatal and Warning (read-only) (default 0)

The FPWR bit (0x20) reports that the power deviation has exceeded the power fatal threshold (0x22) while WPWR bit (0x21) reports that the power deviation has exceeded the power warning threshold (0x23).

When bit 8 is 1, it indicates that the power deviation threshold is being exceeded. When bit 8 is 0, the power deviation threshold is not being exceeded.

Bit 7: XEL – Flags an execution error.

A “1” indicates an exceptional condition. Note that execution errors could be generated by a command just given which failed to execute as well as a command that was currently executing (a pending operation that just complete). The XE bit remains set until cleared.

Bit 6: CEL – Flags a communication error.

A “1” indicates a communication error. The CE bit remains set until cleared.

Bit 5: MRL – Module Restarted (latched) (read-only) (default 1 – by definition)

The MRL bit is read only. When it is set, it indicates that the module has been restarted either by power up, by hardware or software reset, or by a firmware mandated restart. Depending upon the implementation, this may indicate that the laser’s output signal may be invalid. Note that the module can be reset though the communication interface by writing to register 0x32. The bit remains set until cleared.

Bit 4: CRL – Communication Reset (latched) (read-only) (default 1 – by definition)

The CRL bit is read only. When it is set, it indicates that the module has undergone a communications reset. The input buffers were cleared. This can also occur after a manufacturer specific timeout period has elapsed in the middle of a packet transfer.³⁹ The bit remains set until cleared.

Bits 3,2,1,0: FVSFL, FFREQ, FTHERML, FPWRL, WVSFL, WFREQ, WITHERML, WPWRL – Latched fatal and warning indicators (RW) (default 0)

These flags are latched versions of bits 11-8 for the fatal and warning threshold deviations. These bit indicators can be cleared by writing a “1” to these bit positions.

When any of these bits is 1, it indicates that the corresponding deviation threshold has been exceeded at sometime in past (since the last clear) and may still be occurring.

When any of these bits are “0”, the corresponding deviation threshold has not occurred since the last clear.

³⁹ Added reference to the capability of a communications interface timeout which would occur if a packet transfer didn’t complete. The timeout would be manufacturer specific.

10.2.1.2 Power FATAL Threshold (0x22) (r/w)

This register contains the maximum power deviation \pm dB at which the fatal alarm is asserted. The value is stored in dB*1000 as an unsigned integer. Setting a value outside of the usable range causes the value to be set to the maximum allowed. The default is manufacturer specific.

10.2.1.3 Power Warning Threshold (0x23) (r/w)

This register contains the maximum power deviation \pm dB at which the warning condition is asserted. The value is stored in dB*1000 as an unsigned integer. It should typically be equal to or less than the value in register 0x22. Setting a value outside of the usable range cause the value to be set to the maximum allowed. The default is manufacturer specific.

10.2.1.4 Frequency FATAL Threshold (0x24) (r/w)

This register contains the maximum frequency deviation \pm GHz*10 that is allowed before asserting a FATAL condition. The value is stored as an unsigned integer. Setting a value outside of the usable range cause the value to be set to the maximum allowed. The default is manufacturer specific.

10.2.1.5 Frequency Warning Threshold (0x25) (r/w)

This register contains the maximum frequency deviation \pm GHz*10 that is allowed before asserting a warning condition. The value is stored as an unsigned integer. Setting a value outside of the usable range cause the value to be set to the maximum allowed. The default is manufacturer specific.

10.2.1.6 Thermal FATAL Threshold (0x26) (r/w)

This register contains the maximum thermal deviation \pm °C*100 that is allowed before asserting a FATAL condition. The value is stored as an unsigned integer. Setting a value outside of the usable range cause the value to be set to the maximum allowed. The default is manufacturer specific.

10.2.1.7 Thermal Warning Threshold (0x27) (r/w)

This register contains the maximum thermal deviation \pm °C*100 that is allowed before asserting a warning condition. The value is stored as an unsigned integer. Setting a value outside of the usable range cause the value to be set to the maximum allowed. The default is manufacturer specific.

10.2.1.8 SRQ* Triggers (0x28) (r/w)

This register sets the bits for which the SRQ* line is asserted. It follows the similar format as the status register (0x20, 0x21).

15	14	13	12	11	10	9	8
			DIS	WVSFL	WFREQ	WTHERM	WPWRL
0	0	0	1	1	1	1	1
7	6	5	4	3	2	1	0
XEL	CEL	MRL	CRL	FVSFL	FFREQ	FTHERM	FPWRL
1	1/0(see text)	1	1	1	1	1	1

The default setting is:

Interface	IOMODE State Latched At Reset/Power Up	IOMODE1 State Latched At Reset/Power Up	Default Value of AXC Bit
SPI	LOW	LOW	0x1
RS232	HIGH	LOW	0x0
I ² C	LOW	HIGH	0x1
TBD	HIGH	HIGH	0x0

When using RS232 communication, execution errors and communication errors for the immediate command are returned immediately in the module's response packet. However, pending operations can generate execution errors and should generate an SRQ*.

When using SPI and I²C communication, out-bound packets containing the error flags may be retrieved by the host module right away. In this case, the host would probably prefer seeing the SRQ* line asserted.

The SRQ* line can be de-asserted by either changing this register or by clearing the latched fault condition in the status registers (0x20, 0x21).

10.2.1.9 FATAL* Triggers (0x29) (r/w)

This register sets the bits for which the FATAL* line is asserted. It follows the similar format as the status register (0x20, 0x21).

15	14	13	12	11	10	9	8
				WVSFL	WFREQL	W THERML	WPWRL
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
		MRL		FVSFL	FFREQL	F THERML	FPWRL
0	0	0	0	1	1	1	1

The default setting is shown above and is 0x000F. The FATAL* line can be de-asserted by either changing this register or by clearing the latched fault condition in the status registers (0x20).

10.2.1.10 ALM* Triggers (0x2A) (r/w)

This register sets the bits for which the ALM* line is asserted. It follows the same format as the status registers. This line functions as the "locked" line.

15	14	13	12	11	10	9	8
				WVSF	WFREQ	W THERM	WPWR
0	0	0	0	0	1	0	0

7	6	5	4	3	2	1	0
				FVSF	FFREQ	F THERM	FPWR
0	0	0	0	0	1	0	0

The default setting is shown above and is 0x0404 which is useful (along with ADT in the Module configuration register (0x33) to cause the ALM* line to function as a LOCKED line which gets de-asserted during tuning or output disable. The ALM* line can be de-asserted by changing this register.

10.2.1.11 Channel (0x30) (r/w)

15-0
16 bit unsigned channel number

A 16 bit unsigned integer representing the desired channel number. (default: Lowest valid channel number). Channel 0 is an undefined channel number. Writing this register with an invalid channel number will not change the register value and will generate an execution error. Writing a value outside of the channel range will generate an execution error.

The frequency for this channel is defined as:
 $\text{Freq} = (\text{Channel} - 1) * \text{grid_spacing} + \text{Frequency_of_first_channel}$.

Assuming the module is enabled (DIS=1 and SENA=1), the module will tune to a channel. Otherwise the tuning will be initiated by setting SENA=1. Note that changing the DIS* pin to high will not cause a tune. SENA must be set to "1" after the DIS* pin is set high.

The output is disabled under the following conditions:

$$\text{Disable} = (\text{Fatal_Status} \& \text{Fatal_Trigger} \& \text{SDF}) \mid \sim \text{SENA} \mid \sim \text{DIS}$$

Where:

Fatal_Status is register (0x20)

Fatal_Trigger is register (0x29)

SDF is a software enable for fatal alarm to control output [0x0002 & (register 0x33)]

SENA is a software control of the output [0x0008 & (register 0x33)]

DIS is hardware control of the output. [0x1000 & (register 0x20)]. This is latched and cleared by SENA. Increasing channel may be associated with increasing frequency or decreasing frequency depending upon the sign of the grid_spacing value.

An execution error (XE) will leave the output in an undefined state.

10.2.1.12 Optical Power Set Point (0x31) (r/w)

This register sets the optical output power set point in dBm*1000 as a signed integer. Changing it while locked on channel will cause the output power to change. If the requested power change cannot be accommodated, an execution error is generated. Note: this power is an approximate value since it will typically be measured internally and the correlation between the fiber coupled optical output power and internally measured power will vary. The default is manufacturer specific.

10.2.1.13 Reset (0x32) (r/w)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														SR	MR

Bit 0: Module Reset (write-able) (default 0x00)

When set to 1, the module undergoes a “hard” reset. The impact to the optical signal is undefined. This bit is self clearing.

Bit 1: Soft Reset⁴⁰ (write-able) (default 0x00)

The intention is that the module will undergo a soft reset without impacting the traffic carrying capacity of the optical signal. Not all manufacturers will be able to support this feature in which case it will generate an execution error (XE). The resulting state after a soft reset is vendor specific. This bit is self-clearing.

10.2.1.14 Module Configuration Behavior (0x33) (r/w)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											AXC	SENA	SDF	ADT	RTC

Bit 0: RTC Return To Channel After Reset (Default: disable) (R/W) (Default 0x0)

When this bit is set to 1, the laser will return to its previous state (frequency, power, etc) after a power up, and any other module resetting condition. Note that when this bit is a “1”, the module’s configuration is persistent across power failure and resets.

Bit 1: ADT – Alarm during tuning or disable (warning status flags)

The default (0x1) allows alarm conditions during tuning or disable. If set to 0x1, ALM* is asserted during tuning or when the output is disabled. This default causes the ALM* line to function as a LOCKED to channel indicator, even during tuning.

Bit 2: SDF – Shut down optical output on fatal condition.

A fatal condition occurs when the FATAL is asserted (FATAL* line is low)

The default (0x0) does not cause the optical output to shutdown on fatal alarm.

Fatal conditions are somewhat technology specific but would be signaled by any of the bits 10:8 in register 0x20 (status) being set.

Bit 3: SENA – Software enable of output (default 0)

A “1” indicates that the software is allowing the output to be enabled.

A “0” indicates that the software had disabled the output.

This pin is used in conjunction with the DIS* pin. In order for a signal to appear at the output, both the DIS*=high and the SENA=1.

The output is disabled under the following conditions:

$$\text{Disable} = (\text{Fatal_Status} \& \text{Fatal_Trigger} \& \text{SDF}) | \sim\text{SENA} | \sim\text{DIS}$$

⁴⁰ The soft reset will include the communication’s interface reset. The communication’s interface reset can also be accomplished through de-asserting the MS* pin.

Where:

Fatal_Status is register (0x20)

Fatal_Trigger is register (0x29)

SDF is a software enable for fatal alarm to control output [0x0004 & (register 0x33)]

SENA is a software control of the output [0x0008 & (register 0x33)]

DIS is hardware control of the output. [0x1000 & (register 0x20)]. This is latched and cleared by SENA.

Bit 4: AXC – Assert XEL and CEL flags in the status register for the current command.

The XEL flag is always asserted when pending commands generate an execution error. In SPI mode, it may be desirable to have the XEL and CEL flags such that an SRQ* can be asserted.

A “1” indicates that XEL and CEL will be asserted for the current command.

A “0” indicates that XEL will only be asserted for pending commands that fail. The CEL flag will never be asserted.

Interface	IOMODE State Latched At Reset/Power Up	IOMODE1 State Latched At Reset/Power Up	Default Value of AXC Bit
SPI	LOW	LOW	0x1
RS232	HIGH	LOW	0x0
I ² C	LOW	HIGH	0x1
TBD	HIGH	HIGH	0x0

10.2.1.15 GRID Spacing (0x34) (r/w)⁴¹

This value can only be changed when the output is disabled. Changing it while the output is enabled will generate an execution error. This register is only used during tuning to set the output frequency register. The register is a signed integer. This allows for grid spacings as high as ±3.28 THz.

The grid spacing in MHz is equal to $(0x34 * 10^2)$.

The default value for this register is manufacturer specific.

10.2.1.16 First Channel's Frequency (0x35 – 0x36) (r/w)⁴²

This value can only be changed when the output is disabled. Changing it while the output is enabled will generate an execution error. The registers contents are unsigned integers.

The frequency in GHz is equal to $(0x35 * 10^3 + 0x36 * 10^{-1})$.

For instance, 194.175 THz would be represented by

Register	Hex Value	Decimal Value
0x35	0x00C2	194
0x36	0x06D6	1750

Frequency (GHz) is then $194 * 10^3 + 1750 * 10^{-1}$

The default value for this register will be manufacturer specific.

⁴¹ Grid Spacing and first channel frequency are r/w. This is to provide flexibility for users who would like the channel numbering for the various lasers to be the same. By allowing the first channel to be set and the spacing to be set, any supported frequency map can be accommodated. In the event that a value is set that is not-supported, a XE, execution error is returned. Note that 4 registers 0x34-0x30 are available for reading the supported frequency range.

10.2.1.17 Laser Frequency (0x40 – 0x41) (r)

The frequency in GHz is equal to $(0x40 \cdot 10^3 + 0x41 \cdot 10^{-1})$. This is the current frequency set point which is consistent with the assigned channel number.

Default value consistent with Grid spacing register and channel number.

For instance, 194.175 THz would be represented by

Register	Hex Value	Decimal Value
0x40	0x00C2	194
0x41	0x06D6	1750

Frequency (GHz) is then $194 \cdot 10^3 + 1750 \cdot 10^{-1}$

10.2.1.18 Optical Output Power (0x42) (r)

The optical module's output power.⁴³ This value is in $\text{dBm} \cdot 1000$ and is a signed integer. In units with internal power monitors, this is of course, an approximate value.

10.2.1.19 Current Temperature (0x43) (r)⁴⁴

The current temperature reported as an integer encoded in 0.01°C . The temperature set point is vendor specific. This register is monitored in order to set the temperature alarm.

10.2.1.20 Optical Power Min/Max Set Points (0x50 – 0x51) (r)

The minimum power and the maximum power set points are returned in these registers once a channel has been specified (whether or not the output is enabled). The powers are stored as $\text{dBm} \cdot 1000$ as signed integers. Register 0x50 holds the minimum possible power set point and register 0x51 holds the maximum possible power set point. (Default depends upon manufacturer).

10.2.1.21 Laser's First/Last Frequency (0x52-0x53)

The laser's first frequency is stored in 0x52 and 0x53 in the same format as 0x40-0x41 (Laser Frequency Hi/Lo). The laser's last frequency is stored in 0x54 and 0x55 in the same format as well.

10.2.1.22 Laser's Minimum Grid Spacing (0x56)

The lasers minimum grid spacing is stored in this register in the same format as 0x34 (GRID)

10.2.1.23 Reserved (0x57-0x7F)

These registers are reserved for future expansion.

10.2.1.24 Manufacturer Specific (0x80-0xFF)

These registers are reserved for manufacturer specific needs.

⁴³ Given that the mechanical form factor is defined with one pigtail, the power reading must reflect the modules internal sense of pigtailed output power. If the value is measured, it would likely be measured before the optical pigtail and therefore not a true reading of the pigtailed output power.

⁴⁴ If a module technology monitors more than one temperature, other vendor specific temperature registers may exist and be assigned in the vendor specific area. A vendor may use the AEA property with this register to provide a way to read a series of module specific temperatures.

11 OIF MECHANICAL FORM FACTOR IMPLEMENTATION

Appendix A discusses the general considerations for a mechanical model for a generic tunable device and proposes a reference model to provide a framework to address future tunable devices so that they are compatible or non-interfering with other devices.

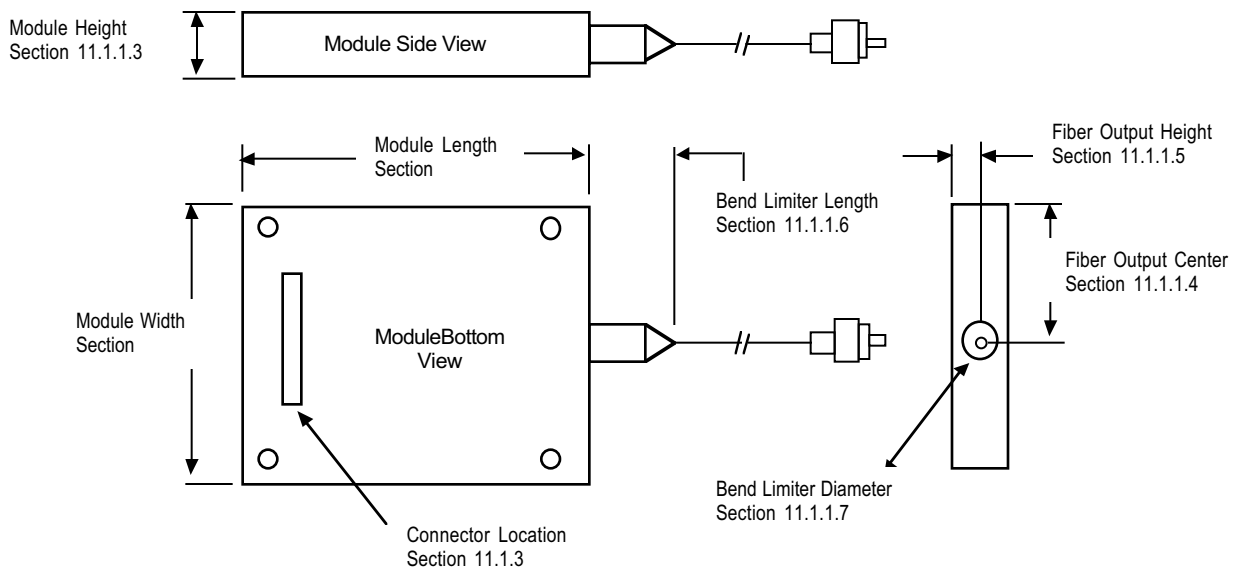
11.1 CW TUNABLE LASER (DEVTYPE = "CW LASER")

11.1.1 MECHANICAL OUTLINE DIMENSIONS

Section Number	Parameter	Minimum Dimension	Maximum Dimension	Tolerance	Units
11.1.1.1	Module Length	-	80	±0.1	mm
11.1.1.2	Module Width	-	50.8	±0.1	mm
11.1.1.3	Module Height	-	13	±0.1	mm
11.1.1.4	Fiber Output Center (centered on width, opposite connector side)	Width (Section 8.1.1.2) / 2		±6	mm
11.1.1.5	Fiber Output Height (from PCB mounting side of module)	3.5	9.5		mm
11.1.1.6	Bend Limiter/Fiber Boot Length	-	30	±1	mm
11.1.1.7	Bend Limiter/Fiber Boot Diameter	-	12	±1	mm

Note:

Figure 11.1-1 Mechanical Outline Dimensions



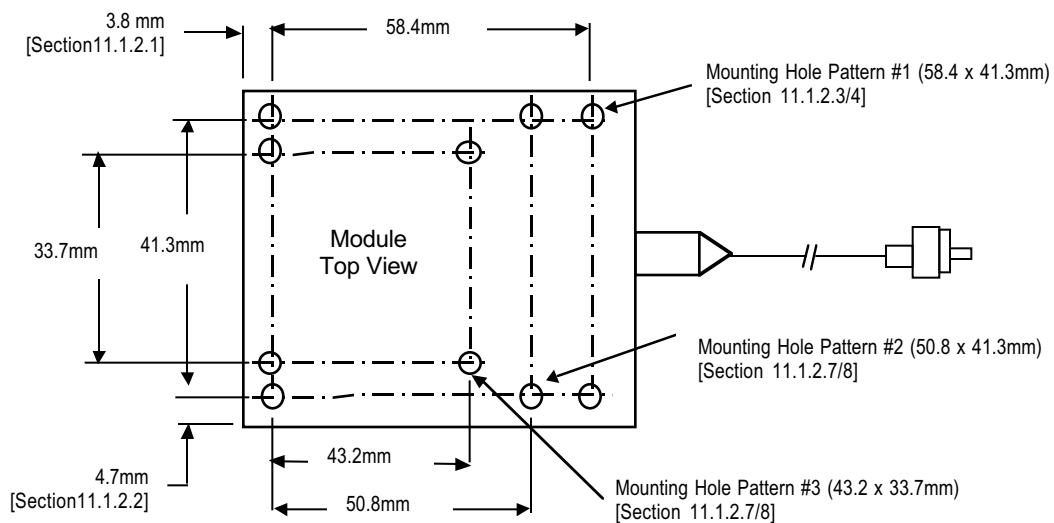
11.1.2 MOUNTING HOLE DIMENSIONS

Section Number	Parameter	Dimension	Tolerance	Units
11.1.2.1	Mounting hole X offset (from origin)	3.8	±0.1	mm
11.1.2.2	Mounting hole Y offset (from origin)	4.7	±0.1	mm
11.1.2.3	Mounting Hole Pattern #1 X spacing	58.4	±0.1	mm
11.1.2.4	Mounting Hole Pattern #1 Y spacing	41.3	±0.1	mm
11.1.2.5	Mounting Hole Pattern #2 X spacing	50.8	±0.1	mm
11.1.2.6	Mounting Hole Pattern #2 Y spacing	41.3	±0.1	mm
11.1.2.7	Mounting Hole Pattern #3 X spacing	43.2	±0.1	mm
11.1.2.8	Mounting Hole Pattern #3 Y spacing	33.7	±0.1	mm
11.1.2.9	Mounting Hole Threads	M2.5 X 0.45-6H through		
11.1.2.10	Grid spacing (for future hole patterns)	3.81	±0.1	mm
11.1.2.10.1	X dimension offset (from reference mounting holes, Section 11.1.2.)	3.81	±0.1	mm
11.1.2.10.2	Y dimension offset (from reference) (from reference mounting holes, Section 11.1.2.)	3.81	±0.1	mm

Notes:

- 1) Three mounting hole patterns are documented for potential variations in footprint
- 2) “X spacing” refers to the Module Length dimension axis (from Figure 11.1-1 Mechanical Outline Dimensions)
- 3) “Y spacing” refers to the Module Width dimension axis (from Figure 11.1-1 Mechanical Outline Dimensions)

Figure 11.1-2 Mounting Hole Dimensions



11.1.3 CONNECTOR SPECIFICATIONS

11.1.3.1 Connector Type: Pluggable

11.1.3.2 Connector Pin Count: 40

11.1.3.3 Connector Pin Spacing: 0.050"

11.1.3.4 "Y" Pluggable Connector

11.1.3.4.1 Module: Samtec FTSH-120-03-F-DV-ES (pin) or equivalent

11.1.3.4.2 Mating connector: Samtec CLP-120-02-G-D-P (socket) or equivalent

11.1.3.5 "Y" Pluggable Pin 1 Dimensions

11.1.3.5.1 Pin 1 X offset: 2.81mm (±0.2mm) (from reference)

11.1.3.5.2 Pin 1 Y offset: 8.64mm (±0.2mm) (from reference)

11.1.3.6 "Y" Pluggable Pin Numbering

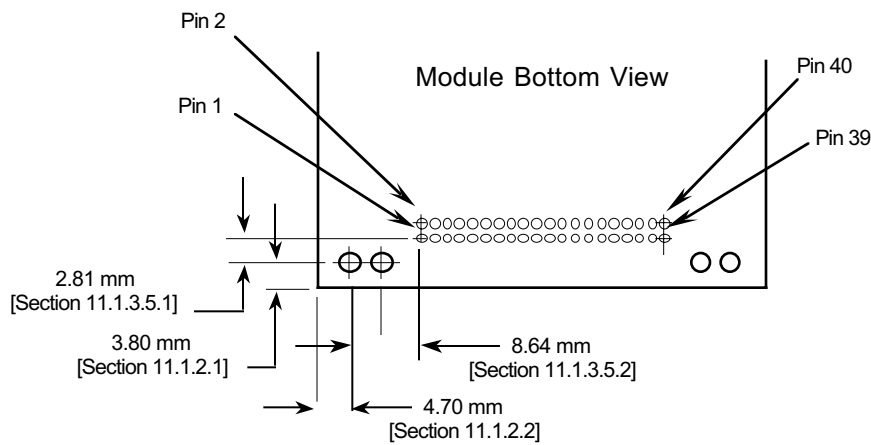


Figure 11.1-3 "Y"-Pluggable Connector Dimensions

Notes:

1. "X offset" refers to the Module Length dimension axis (from Figure 11.1-1 Mechanical Outline Dimensions)
2. "Y offset" refers to the Module Width dimension axis (from Figure 11.1-1 Mechanical Outline Dimensions)
3. Connector pins should be flush with the module housing surface

11.1.3.7 "Z" Pluggable Connector

11.1.3.7.1 Module connector: FTSH-120-02-L-DH or equivalent

11.1.3.7.2 Mating connector: FLE-120-01-G-DV or equivalent

11.1.3.8 "Z" Pluggable Pin 1 Dimensions

11.1.3.8.1 Pin 1 Y offset: 13.3 mm

11.1.3.8.2 Pin 1 Z offset: Not Specified

"Z" Pluggable Pin Numbering

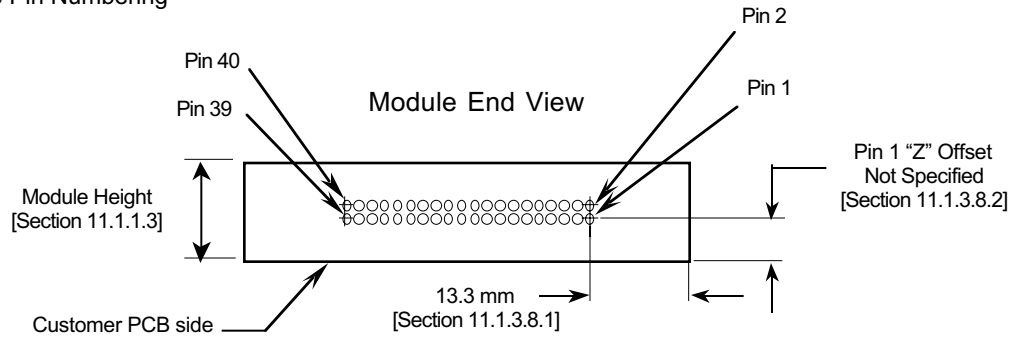


Figure 11.1-4 "Z"- Pluggable Connector Dimensions

Notes:

1. "Y offset" refers to the Module Width dimension axis (from Figure 11.1-1 Mechanical Outline Dimensions)
2. "Z offset" refers to the Module Height dimension axis (from Figure 11.1-1 Mechanical Outline Dimensions)
3. Connector pins should be flush with rear surface of module housing.

12 REFERENCES

OIF2001.516

OIF2002.001

OIF2002.002

OIF2002.118

13 APPENDICES

13.1 APPENDIX A: GENERAL TUNABLE DEVICE MECHANICAL CONSIDERATIONS

Each tunable device may have an interoperability agreement on form factor defined. This Appendix highlights the general considerations for a mechanical model for a generic tunable device and proposes a reference model to provide a framework to address future tunable devices so that they are compatible or non-interfering with other devices.

Methodology

Start by defining a reference point for all devices. For instance, establish reference point (RP) = Center of connector / pin out, with axes CL and CC extending perpendicular and parallel, respectively, to the long rows of pins. This is the first step in ensuring that pinouts and mounting holes are compatible among devices.

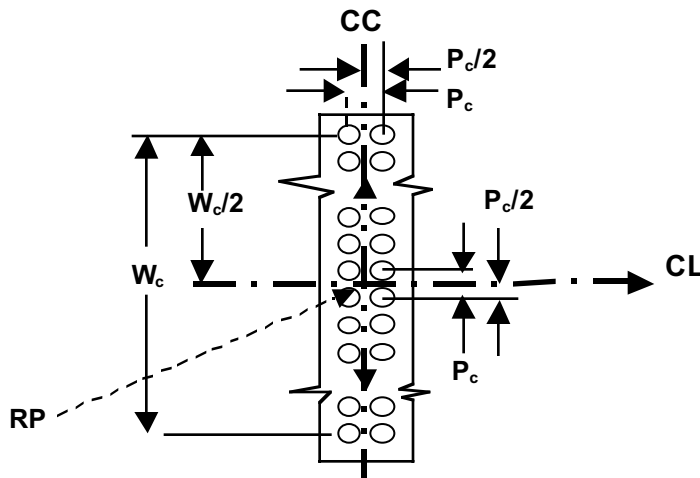


Figure 13.1-1: Determination of a reference point on the connector for a tunable device

Define the Y-reference as the mounting plane of the tunable device with the PCB

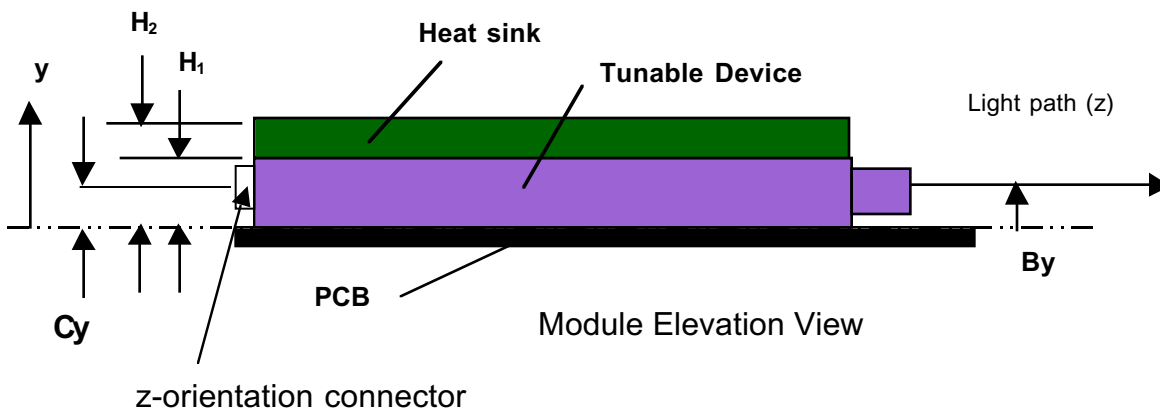


Figure 13.1-2: Establishing the vertical reference planes for a tunable device

With the X,Y, and Z references established, the tunable device can be dimensioned. Notice that there might be several light paths on a device, and that these might not be centered on the device centerline.

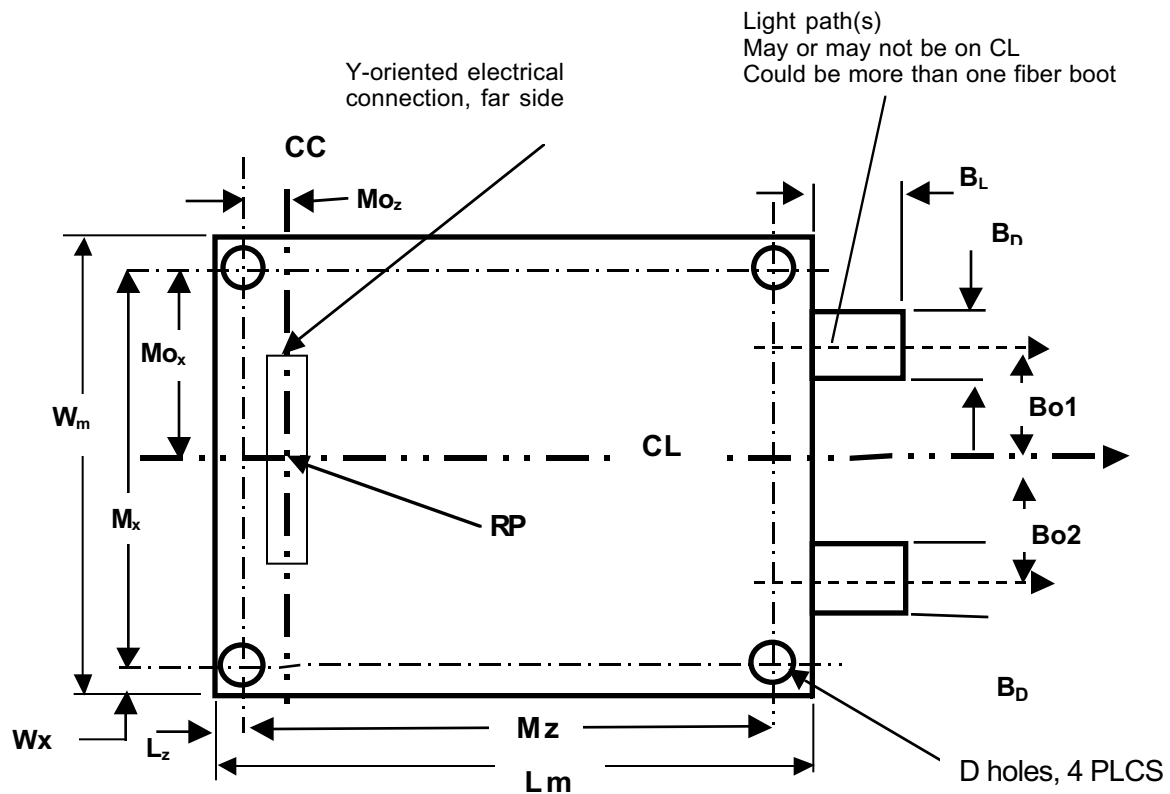


Figure 13.1-3: Suggested dimensioning of a generic tunable device with respect to the Reference Point.

Mounting Hole Location Grid Spacing:

It is important that mounting hole locations are chosen so that they are compatible, if not identical, to mounting holes for existing devices. To ensure that mounting holes of different devices do not conflict, the following convention is recommended:

- To allow for more flexibility and optimization of design, establish grid with a regular spacing on which all future mounting holes will be placed.
- Grid can be referenced off of CL and CC, or other existing features (like current mounting hole locations or module surfaces)
- Avoids future interference of mounting holes and “swiss cheese” effect on board
- **Each device implementation can have specific set(s) of mounting holes defined (however located on grid)**

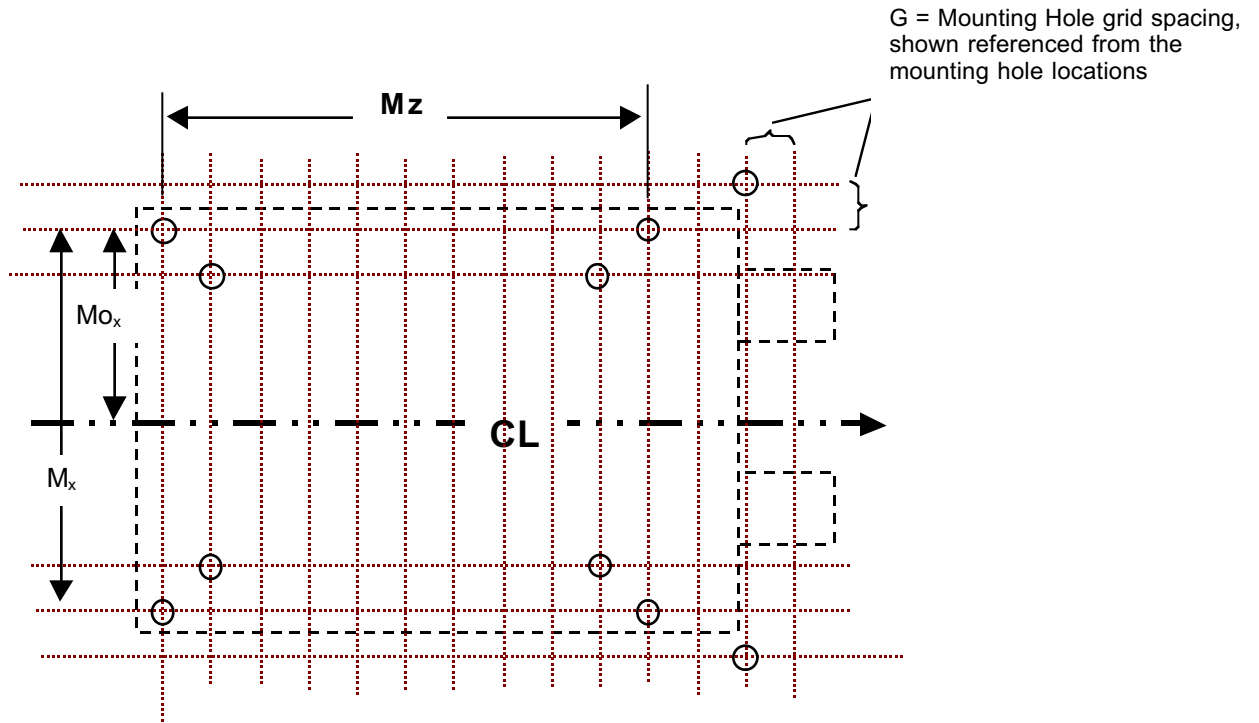


Figure 13.1-4: Grid concept for locating mounting holes on tunable devices so that they do not conflict with existing holes for other devices.

Generic Tunable Device Mechanical Parameters

The following table captures many of the important parameters necessary in defining a mechanical form factor for a tunable device, and can be used as a template for developing new tunable device types.

Spec	Parameter	Symbol	Unit	Note	
A	Module Max Length	Lm	mm		
B	Module Max Width	Wm	mm		
C	Module Max Height	Without Heatsink	H1	mm	From PCB mounting surface
		With heatsink	H2		
D E	Fiber Output location(s)	Center wrt CL	Bo	mm	more than one fiber I/O permissible
		Height	By		
F G	Fiber Boot / Bend Limiter Max Dims	Length	Bl	mm	
		Diameter	Bd		
H I J K	Mounting hole location	X offset	Mox	mm	Referenced to connector (Measured from CC and CL)
		Z offset	Moz		
		X offset	Wx	mm	Measured from module sides
		Z offset	Lz		
L M	Mounting Hole Pattern spacing	X	Mx	mm	Center to center. Can define more than one set threaded
		Z	Mz		
N	Mounting Hole size	D	mm		
O	Grid spacing	Gr	mm		
P	Electrical Connection Type 1 (Pluggable, thru hole, other)				More than one type can be called out
Q	Pin count	# pins	pins		
R	Pin spacing	Pc	mm/in		
S	Electrical Connection orientation	X, Y, Z			
T U V	Electrical connection location	X offset	Mox	mm	
		Y Offset	Moy		
		Z offset	Moz		
W	Pin Numbering				See Below
X Y	Power supply	Nom voltage	Vnom	V	
		Max current	Imax	A	
Z	Environmental	Min Tcase	Tcmin	C	Operating and storage
AA		Max Tcase	Tcmax		
AB		Max ambient air temp	Tamb	C	Must specify if heatsink provided by TL supplier
AC		Airflow speed	Sa		

Pin Numbering Method for Tunable Devices:

When choosing a connector for a tunable device standard form factor, a balance must be achieved between allowing for functionality expansion, while allowing designs to be as small as possible while accomplishing the desired functionality.

The following methodology can be used to achieve this balance, while maintaining compatibility between devices with different pin out requirements to achieve certain levels of functionality.

- Centered pin out on centerlines (CC, CL).
- First pins on each side of the centerline will be spaced $Pc/2$ away from center line (= half the selected pin spacing)
- Pin outs continue in +/- x direction on “A-side” and “B-side” until requisite number of pins are defined. For instance, a 40-pin connector will have pins A1 to A20 and B1 to B20
- Define pins so that the common set of required pins are in the lowest numbered pin locations, closest to the center of the connector (ie A1, B1, A2, B2, A3, B3....etc).
- Allow the connector to grow away from the centerline to address mfg specific pins, as well as to support additional functionality and higher levels of integration.
- A preferred pin numbering scheme (for instance a pin1 to pin40) can be adopted when a specific connector is chosen

The following diagram shows a pin numbering methodology that can be used for the Y-pluggable connector (viewing the module from the top).

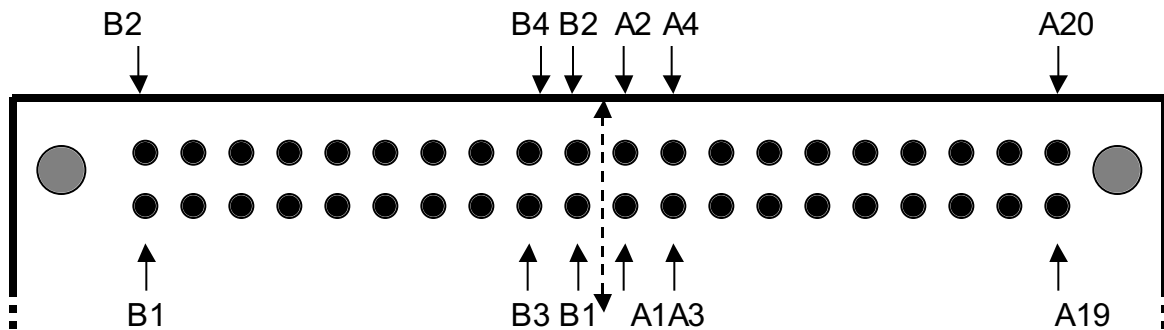


Figure 13.1-5: Example pin numbering scheme for a device to maintain compatibility and allow for future expansion.

A similar pin numbering scheme can be used to define a connector/pin out that emerges from the rear surface of the module (optical Z axis).

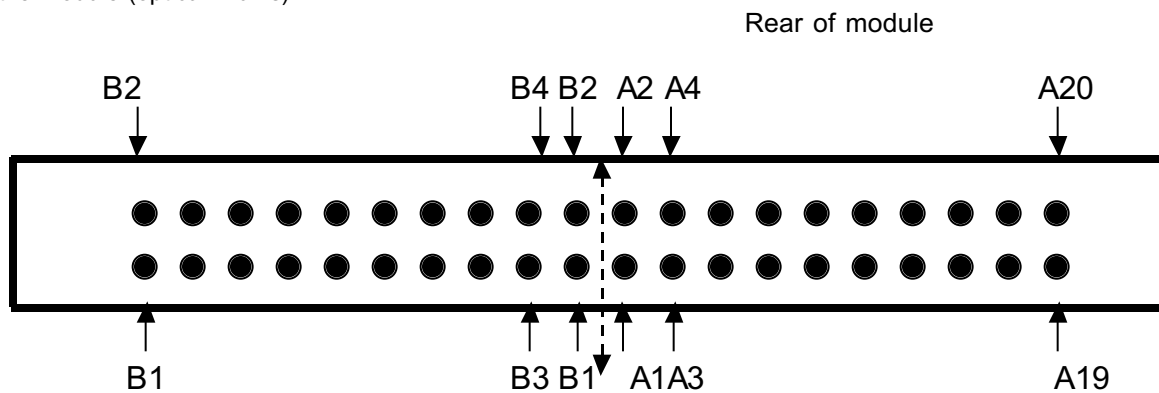


Figure 13.1-6: Example of Z-pluggable pin out numbering.

Appendix: List of companies belonging to OIF when document is approved:

Accelerant Networks
Accelight Networks
Actel
Acterna Eningen GmbH
ADC Telecommunications
Aeluros
Agere Systems
Agilent Technologies
Agility Communications
Alcatel
All Optical Networks, Inc.
Altamar Networks
Altera
Alvesta Corporation
AMCC
America Online
Ample Communications
Analog Devices
ANDO Corporation
Anritsu
Aralight
ASTRI
AT&T
Atrica Inc.
Avici Systems
Axiowave Networks
Bandwidth9
Bay Microsystems
Big Bear Networks
Bit Blitz Communications
Blaze Network Products
Blue Sky Research
Bookham Technology
Booz-Allen & Hamilton
Broadcom
Cable & Wireless
Cadence Design Systems
Calient Networks
Calix Networks
Caspian Networks
Celion Networks
Centellax
Centillum Communications
Ceyba
Chiaro Networks
Chunghwa Telecom Labs

Ciena Communications
Cisco Systems
Coherent Telecom
Conexant
CoreOptics
Coriolis Networks
Corrigent Systems
Cortina Systems
Corvis Corporation
Cypress Semiconductor
Data Connection
Department of Defense
DeriveIt
E2O Communications
ELEMATICS
Elisa Communications
Emcore
Equant Telecommunications SA
Equipe Communications
Ericsson
ETRI
Extreme Networks
EZChip Technologies
Fiberhome Telecommunications
Fiberspace
Finisar Corporation
Flextronics
Force 10 Networks
France Telecom
Free Electron Technology
Fujikura
Fujitsu
Furukawa Electric Technologies
Galazar Networks
General Dynamics
Glimmerglass Networks
Harris Corporation
Harting Electro-Optics GmbH
Helix AG
Hi/fn
Hitachi
Huawei Technologies
IBM Corporation
Ignis Optics
Industrial Technology Research Institute
Infineon Technologies
Infinera

Innovance Networks
Inphi
Integrated Device Technology
Intel
Internet Machines
Interoute
Intune Technologies, Ltd.
Iolon
Japan Telecom
JDS Uniphase
Jennic
Juniper Networks
KDDI R&D Laboratories
Kirana Networks
KT Corporation
Larscom
Lattice Semiconductor
LSI Logic
Lucent
Lumentis
LuxN
LYNX - Photonic Networks
Mahi Networks
Marconi Communications
MathStar
Maxim Integrated Products
MergeOptics GmbH
Meriton
Metro-OptiX
Mintera
Mitsubishi Electric Corporation
Multilink Technology Corporation
Multiplex
MultiWave Networks
Myrica Networks
Mysticom
National Semiconductor
Nayna Networks
NEC
NetTest
Network Elements
NIST
Nortel Networks
NTT Corporation
NurLogic Design
OpNext
Optical Datacom

Optillion
Optium
Optix Networks
Optobahn
OptronX
PacketLight Networks
Parama Networks
Paxonet Communications
Peta Switch Solutions
PhotonEx
Photuris, Inc.
Phyworks
Picarro
Pine Photonics Communications
PMC Sierra
Polaris Networks, Inc.
Princeton Optronics
Procket Networks
Quake Technologies
Qwest Communications
RedClover Networks
RF Micro Devices
RHK
Sandia National Laboratories
Santec Corporation
Santel Networks
Santur
SBC
Siemens
Sierra Monolithics
Silicon Access Networks
Silicon Labs
Silicon Logic Engineering
Sky Optix
Solidum
Southampton Photonics
Spirent Communications
StrataLight Communications
Stratos Lightwave
Sumitomo Electric Industries
Sun Microsystems
Sycamore Networks
TDK Semiconductor
Tektronix
Telcordia Technologies
Telecom Italia Lab
Tellabs

Tellium
Tenor Networks
TeraBurst Networks
TeraConnect
Teradant Networks, Inc.
Texas Instruments
T-Networks, Inc.
Toshiba Corporation
Transpectrum
Transpera Networks
TriQuint Semiconductor
Tropic Networks Inc.
Tsunami Photonics
T-Systems Nova
Turin Networks
Tyco Electronics
US Conec
Velio Communications
Velocium (TRW)
Verizon
Vitesse Semiconductor
VSK Photonics
W.L. Gore & Associates
Wavecrest Corporation
Wavium AB
West Bay Semiconductor
Xanoptix
Xelerated
Xignal Technologies
Xilinx
Xindium
Xlight Photonics
Zagros Networks
Zarlink Semiconductor