Document number 2006015

VersaPHY Additions to IEEE-1394

December 5, 2007

Sponsored by:

1394 Trade Association

Accepted for publication by

1394 Trade Association Board of Directors

Abstract

VersaPHY™[1] is a compatible extension to IEEE-1394-1995, a-2000 and b-2002 designed to enable lower cost implementations by allowing communication between nodes without IEEE-1394 link or transaction layer overhead.

Keywords

IEEE 1394, Serial Bus, …

---

[1] VersaPHY is a trademark of Quantum Parametrics LLC

1394 Trade Association Specification

1394 Trade Association Specifications are developed within Working Groups of the 1394 Trade Association, a non-profit industry association devoted to the promotion of and growth of the market for IEEE 1394-compliant products. Participants in Working Groups serve voluntarily and without compensation from the Trade Association. Most participants represent member organizations of the 1394 Trade Association. The specifications developed within the working groups represent a consensus of the expertise represented by the participants.

Use of a 1394 Trade Association Specification is wholly voluntary. The existence of a 1394 Trade Association Specification is not meant to imply that there are not other ways to produce, test, measure, purchase, market or provide other goods and services related to the scope of the 1394 Trade Association Specification. Furthermore, the viewpoint expressed at the time a specification is accepted and issued is subject to change brought about through developments in the state of the art and comments received from users of the specification. Users are cautioned to check to determine that they have the latest revision of any 1394 Trade Association Specification.

Comments for revision of 1394 Trade Association Specifications are welcome from any interested party, regardless of membership affiliation with the 1394 Trade Association. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally, questions may arise about the meaning of specifications in relationship to specific applications. When the need for interpretations is brought to the attention of the 1394 Trade Association, the Association will initiate action to prepare appropriate responses.

Comments on specifications and requests for interpretations should be addressed to:

Editor, 1394 Trade Association
1560 East Southlake Blvd, Suite 226
Southlake, TX 76092
USA

Published by

1394 Trade Association
1560 East Southlake Blvd, Suite 226
Southlake, TX 76092 USA

Printed in the United States of America

Contents

Tables

Figures

Annexes

Foreword

(This foreword is not part of 1394 Trade Association Specification 2006015)

This specification defines communication between nodes without IEEE-1394 link or transaction layer overhead.

There are 5 annexes in this specification. Annexes A,B, and D, are normative and part of this specification. Annexes C and E, are informative and are not considered part of this specification.

This specification was accepted by the Board of Directors of the 1394 Trade Association. Board of Directors acceptance of this specification does not necessarily imply that all board members voted for acceptance. At the time it accepted this specification, the 1394 Trade Association. Board of Directors had the following members:

, Chair	Max Bassler
, Vice-Chair	Peter Helfet
, Secretary	Dave Thompson

....

The Industrial and Instrumentation Working Group, which developed and reviewed this specification, had the following members:

Michael Scholles, Chair

Roumen Botev	Brooks Automation
Matthew Coady	Brooks Automation
Duncan Beadnell	Oxford Semiconductor
Les Baxter	Baxter Enterprises
Max Bassler	Littelfuse
Michael Scholles	Fraunhofer IPMS
Richard Mourn	Quantum Parametrics
Barry Walker	Quantum Parametrics
Ricardo Wong	Nissan
Bill Rose	WJR Consulting Inc.
Branko Bukal	Atra Vision
Christopher Thomas	Symwave
David Thompson	LSI
Francesco Liburdi	Electronic Links International
Mike Gardner	Molex
Hans van der Ven	AV Connections
Will Harris	TI
Lawrence Thorne	Firecomms
Morgan Jair	Main Super Enterprises Co.
Sam Liu	Newnex
Sophia Liu	Electronics Testing Center, TaiwanRevision history

Revision 0.1 (May 16, 2006)

Original Version

Revision 0.2 (November 7, 2006)

Category	Description
Technical	Added Unique Device ID fields to register bank2 Added Label Management Section Added CRC-8 Details and Annex

Revision 0.3 (January 20, 2007)

Category	Description
Editorial	Multiple editorial clarifications Added VP-Function notation
Technical	Added concatenated responses Added multi-packet read requests Added VP-Controller and bridge profile annexes

Revision 0.4 (April 5, 2007)

Category	Description
Editorial	Table reformatting Changed Node-ID terminology to Physical-ID
Technical	Modified bus field in Physical-ID packets. Clarification permitting profiles to operate without reserved packet CRC bits.

Revision 0.5 (June 1, 2007)

Category	Description
Editorial	"Everything is broadcast" model explanation "Remote register" concept explanation "Multi-VP-Controller write" disclaimer Register space descriptions rewritten as separate sets with common fields. Re-organized labeling sections
Technical	Bridges should initialize to mode 33 Device_ID expanded to 64 bits to match 1394 OUI-64 (and moved in register block) Local-only and broadcast orphan label assignment recommendation Bridgeable bus address set to 0s consistently, local only set to 1s consistently Network broadcast bus enabled via bridging

	VP-Devices do not require GUIDs Orphans default to E (enabled) = 0 Power field description and operation added. Bridge profile removed. Controller profile modified with new register definitions Four mode config_ver added Watchdog bit added BusReset bit added Self enabled bit added Owner field added Unsolicited response tl changed to 111111_2 (makes room for Physical_ID tls) Many fields made optional in VP-Label space Blocks field changed to Instance_nOffset Label Mode tables added to appendix Removed bridge profile specification. Will be detailed in separate document.

Revision 0.6 (August 13, 2007)

Category	Description
Editorial	Updated to latest 1394 TA document template Changed Power bit field from 6 bits to 5 bit in figures 19 and 24

Revision 0.7 (September 25, 2007)

Category	Description
Editorial	Page 12 - 5.8.3 last word of 2nd paragraph should be 'poll' not 'pole' Page 15 - 6.3.1 First binary value on the first line is incorrect, should be "00000000000001" not "00000000000010" Page 16 - 6.3.2 First line of page. Should it read "... containing static local only labels" not "... containing static broadcast labels" Page 16 - 6.4 malformed reference "9.6Annex B" Page 22 - "bridge and VP-Controller" should be "VP-Bridge and VP-Controller"? Page 22 - regarding multiple controllers changing the label mode, would this benefit from some sort of policy? Page 22 - Last sentence on page. Should be "Where" instead of "Were" Page 24 - malformed reference "9.6Annex C" in CRC description in Table 5. Page 29 – Table 7 – Extra cell in Instance_nOffset description.

Revision 1.0 (December 5, 2007)

Category	Description
Editorial	Added conformance requirements annex

VersaPHY Additions to IEEE-1394

1 Scope and purpose

VersaPHY™[2] is a compatible extension to IEEE-1394-1995, a-2000 and b-2002 designed to enable lower cost implementations by allowing.

1.1 Scope

This document is limited to defining VersaPHY functionality. This document defines the VersaPHY extensions, describes the advantages they provide, outlines the VersaPHY node interaction model, and provides example implementations of VersaPHY systems. This document also defines the changes to IEEE-1394 needed to support VersaPHY functionality.

1.2 Purpose

The purpose of this document is to define, describe, and promote VersaPHY technology. The goal of this document is to prepare the VersaPHY extensions for inclusion in an appropriate IEEE standard.

[2] VersaPHY is a trademark of Quantum Parametrics LLC

2 Normative references

2.1 Reference scope

The specifications and standards named in this section contain provisions, which, through reference in this text, constitute provisions of this 1394 Trade Association Specification. At the time of publication, the editions indicated were valid. All specifications and standards are subject to revision; parties to agreements based on this 1394 Trade Association Specification are encouraged to investigate the possibility of applying the most recent editions of the specifications and standards indicated below.

2.2 Approved references

The following approved specifications and standards may be obtained from the organizations that control them.

IEEE Std 1394-1995, Standard for a High Performance Serial Bus

IEEE Std 1394a-2000, Standard for a High Performance Serial Bus—Amendment 1

IEEE Std 1394b-2002, Standard for a High Performance Serial Bus—Amendment 2

Throughout this document, the term “IEEE 1394" shall be understood to refer to IEEE Std 1394-1995 as amended by IEEE Std 1394a-2000 and IEEE Std 1394b-2002.

2.3 References under development

At the time of publication, the following referenced specifications and standards were under development.

2.4 Reference acquisition

The references cited may be obtained from the organizations that control them:

1394 Trade Association, 1560 East Southlake Blvd, Suite 220, Southlake, TX 76092 USA; (817) 416-2200 / (817) 416-2256 (FAX); http://www.1394ta.org/

American National Standards Institute (ANSI), 11 West 42nd Street, New York, NY 10036, USA; (212) 642-4900 / (212) 398-0023 (FAX); http://www.ansi.org/

Institute of Electrical and Electronic Engineers (IEEE), 445 Hoes Lane, PO Box 1331, Piscataway, NJ 08855-1331, USA; (732) 981-0060 / (732) 981-1721 (FAX); http://www.ieee.org/

In addition, many of the documents controlled by the above organizations may also be ordered through a third party:

Global Engineering Documents, 15 Inverness Way, Englewood, CO 80112-5776; (800) 624-3974 / (303) 792-2192; http://www.global.ihs.com/

3 Definitions and notation

3.1 Definitions

3.1.1 Conformance

Several keywords are used to differentiate levels of requirements and optionality, as follows:

3.1.1.1 expected: A keyword used to describe the behavior of the hardware or software in the design models assumed by this specification. Other hardware and software design models may also be implemented.

3.1.1.2 ignored: A keyword that describes bits, bytes, quadlets, octlets or fields whose values are not checked by the recipient.

3.1.1.3 may: A keyword that indicates flexibility of choice with no implied preference.

3.1.1.4 reserved: A keyword used to describe objects (bits, bytes, quadlets, octlets and fields) or the code values assigned to these objects in cases where either the object or the code value is set aside for future standardization. Usage and interpretation may be specified by future extensions to this or other specifications. A reserved object shall be zeroed or, upon development of a future specification, set to a value specified by such a specification. The recipient of a reserved object shall ignore its value. The recipient of an object defined by this specification as other than reserved shall inspect its value and reject reserved code values.

3.1.1.5 shall: A keyword that indicates a mandatory requirement. Designers are required to implement all such mandatory requirements to assure interoperability with other products conforming to this specification.

3.1.1.6 should: A keyword that denotes flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “is recommended.”

3.1.2 Glossary

The following terms are used in this specification:

3.1.2.1 byte: Eight bits of data, used as a synonym for octet.

3.1.2.2 CSR Architecture: A convenient abbreviation of the following reference (see clause 2): ISO/IEC 13213 : 1994 [ANSI/IEEE Std 1212, 1994 Edition], Information Technology—Microprocessor systems— Control and Status Register (CSR) Architecture for Microcomputer Buses.

3.1.2.3 quadlet: Four bytes of data.

3.1.3 Abbreviations

The following are abbreviations that are used in this specification:

IEEE - The Institute of Electrical and Electronics Engineers, Inc.

EUI-64 - A 64-bit extended unique identifier which consists of a concatenation of a 24-bit *company_id* value (which is assigned by the IEEE Registration Authority) and a 40-bit extension identifier (which is assigned by the organization which owns that *company_id*.) EUI-64 is a trademark of the IEEE.

VP-Controller - VersaPHY Controller -- A IEEE-1394 node, typically an OCHI device in a computer, responsible for management of one or more VP-Functions in remote VP-Devices.

VP-Device - VersaPHY device -- An IEEE-1394 enabled device containing a VP-Register set using VP-Packets to manage that register set. Note: A single VP-Device may contain one or more VP-Functions each described by its VP-Profile specification and addressed by its VP-Label.

VP-Function - VersaPHY function -- An autonomous function within a VP-Device that conforms to a specific VP-Profile specification.

VP-Label - VersaPHY label -- A static address assigned to each VP-Function within each VP-Device.

VP-Packet - VersaPHY packet -- One of the VersaPHY packets defined in this document.

VP-Profile - VersaPHY profile -- The specification defining the requirements for a particular type of VP-Function.

VP-Register - VersaPHY register -- A memory location within a VP-Device that is accessible using VP-Packets.

3.2 Notation

3.2.1 Numeric values

Decimal and hexadecimal are used within this specification. By editorial convention, decimal numbers are most frequently used to represent quantities or counts. Addresses are uniformly represented by hexadecimal numbers. Hexadecimal numbers are also used when the value represented has an underlying structure that is more apparent in a hexadecimal format than in a decimal format.

Decimal numbers are represented by Arabic numerals without subscripts or by their English names. Hexadecimal numbers are represented by digits from the character set 0 – 9 and A – F followed by the subscript 16. When the subscript is unnecessary to disambiguate the base of the number it may be omitted. For the sake of legibility hexadecimal numbers are separated into groups of four digits separated by spaces.

As an example, 42 and $2A_{16}$ both represent the same numeric value.

3.2.2 Bit, byte and quadlet ordering

This specification uses the facilities of Serial Bus, IEEE 1394, and therefore uses the ordering conventions of Serial Bus in the representation of data structures. In order to promote interoperability with memory buses that may have different ordering conventions, this specification defines the order and significance of bits within bytes, bytes within quadlets and quadlets within octlets in terms of their relative position and not their physically addressed position.

Within a byte, the most significant bit, *msb*, is that which is transmitted first and the least significant bit, *lsb*, is that which is transmitted last on Serial Bus, as illustrated below. The significance of the interior bits uniformly decreases in progression from *msb* to *lsb*.

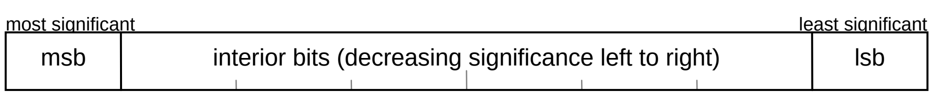

Figure 1 – Bit ordering within a byte

Within a quadlet, the most significant byte is that which is transmitted first and the least significant byte is that which is transmitted last on Serial Bus, as shown below.

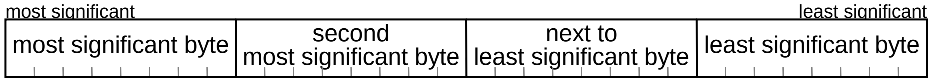

Figure 2 – Byte ordering within a quadlet

Within an octlet, which is frequently used to contain 64-bit Serial Bus addresses, the most significant quadlet is that which is transmitted first and the least significant quadlet is that which is transmitted last on Serial Bus, as the figure below indicates.

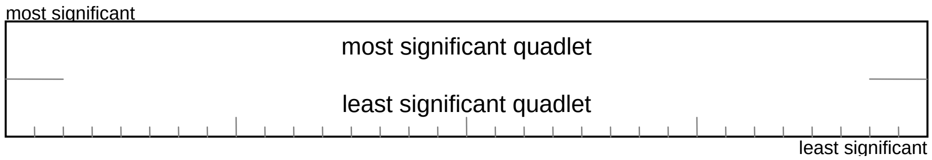

Figure 3 – Quadlet ordering within an octlet

When block transfers take place that are not quadlet aligned or not an integral number of quadlets, no assumptions can be made about the ordering (significance within a quadlet) of bytes at the unaligned beginning or fractional quadlet end of such a block transfer, unless an application has knowledge (outside of the scope of this specification) of the ordering conventions of the other bus.

4 Overview (informative)

VersaPHY extensions support opportunities for lower cost and simpler 1394 devices. VersaPHY technology accomplishes this by simplifying device discovery, bus reset handling, control data delivery, and bus management. At the same time the VersaPHY extensions leverage existing 1394 strengths like streaming, multiple physical implementation options, and existing 1394 infrastructure.

To enable VersaPHY three new 1394 facilities are defined in this document. The new facilities are:

- VersaPHY labels
- VersaPHY registers
- VersaPHY packets

These new facilities can be implemented using existing PHY silicon with external logic and software. No new PHY silicon is required. In the future, VersaPHY technology may be integrated into 1394 PHY devices or microcontrollers to further reduce cost.

4.1 VersaPHY labels

Perhaps the most important new feature of the VersaPHY extensions is static labels (VP-Labels). The VersaPHY definition makes available 16K labels for VersaPHY functions. In the simplest case labels can be used as permanent addresses for nodes. In more complex implementations labels can be used as permanent or semi-permanent addressable names for multiple VP-Functions within nodes. Nodes are still addressable using Physical-IDs but implementations may exclusively use VP-Label addressing to avoid the dynamic and complex nature of Physical-ID addressing.

4.2 VersaPHY registers

The VersaPHY extensions include a register set that may be read and written remotely using VersaPHY packets. These registers are addressed with existing Physical-IDs or the new VP-labels. A minimal set of registers is defined by this specification. The rest are defined by implementation specific VP-Profile specifications. This allows simple to complex implementations to utilize VersaPHY technology. Physical-ID addressing allows up to 4K byte-wide registers. VP-Label addressing allows each profile up to 1M byte-wide registers.

4.3 VersaPHY packets

One of the fundamental enhancements for the VersaPHY is the addition of writeable PHY registers. This allows PHY only devices to communicate in a way only transaction capable nodes could previously. There are two sets of new PHY packets defined for VP-Devices. The first uses Physical-ID addressing and the second uses VP-label addressing. Each set contains read request, read response, write request, and write response packets for a total of 8 new packet formats. The new packets are all fixed 2 quadlet packets like all existing IEEE-1394 defined PHY packets. They are formatted to be unique among the currently defined PHY packets. Most significantly, the second quadlet of a VersaPHY packet is not defined to be the inverse of the first.

5 VersaPHY Use (informative)

5.1 Introduction

This section is informative and describes examples of VersaPHY functionality and how it might be used.

5.2 Basic architecture

Fundamentally the VersaPHY interaction model consists of a VP-Controller and one or more VersaPHY devices (VP-Devices). Each VP-Device will contain a VersaPHY register set and profile specific logic to manage those local registers according to the VP-Profile specification for that device. A VP-Controller can be any 1394 node with software or firmware to send and receive VersaPHY packets and implement the VP-Controller profile.

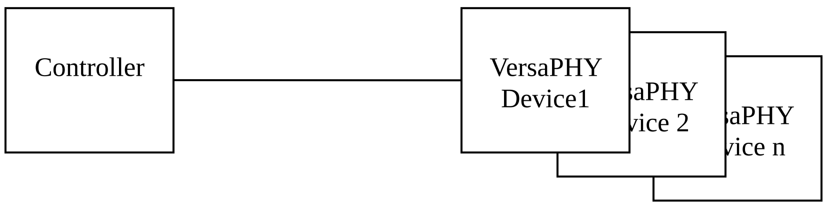

Figure 4 - Single VP-Controller VersaPHY Topology

5.3 Full System

A typical VP-Controller may be a PC with 1394 OHCI hardware and generic 1394 and VersaPHY software drivers. A typical VP-Device can be implemented without software. In cases were software is required for an application it need not know anything about 1394 or VersaPHY. A VP-Controller using a full VersaPHY driver stack, insulates applications from data transmission issues.

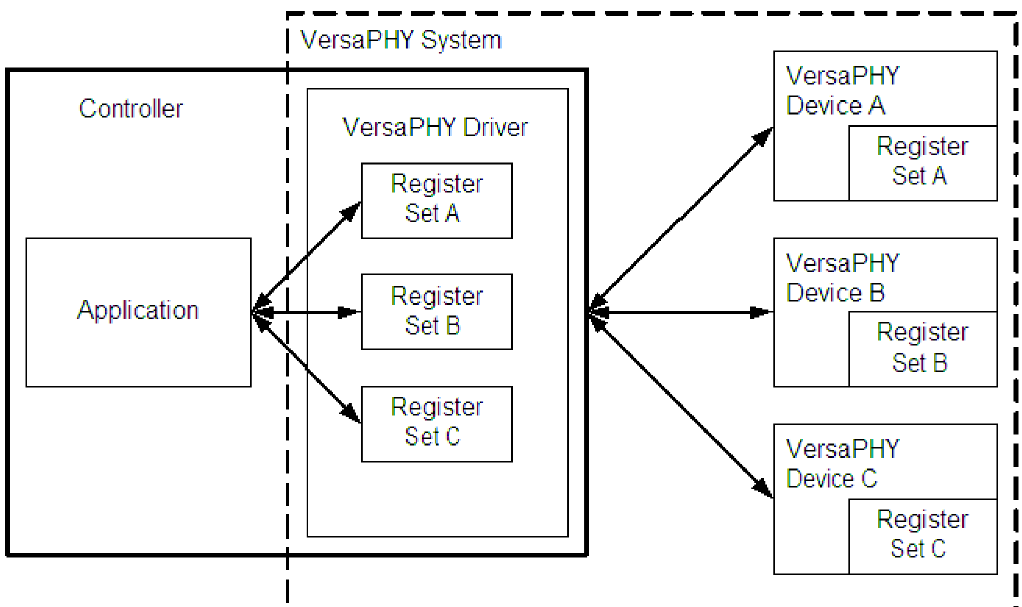

Figure 5 – Local Register Emulation

With VP-Device hardware and the VP-Controller software managing all aspects of remote data exchange, application development is not concerned with VersaPHY protocol or 1394 details. The application can manage registers on remote networked devices as if they are local registers. Through a VP-Profile defined protocol(s) local and remote registers are synchronized without any involvement from the application.

5.4 Multiple Controllers

In some cases multiple controllers may exist on one bus. Multiple controllers may even monitor the same VersaPHY device. (The links shown below are logical. All VP-Devices and controllers are physically connected to the same 1394 bus.)

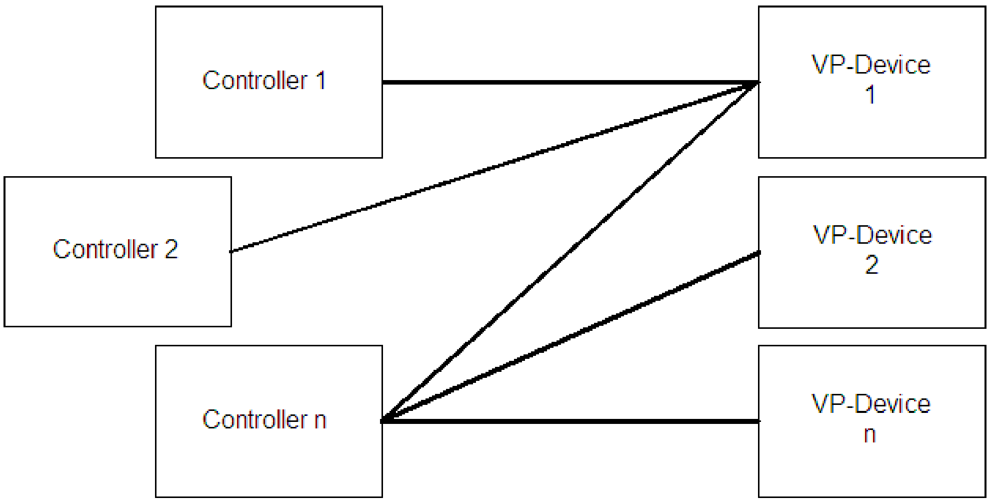

Figure 6 – Multiple Controller VersaPHY Topology

Of course care must be taken in development of applications allowing more than one VP- Controller to write to a single VP-Device. The approach to managing this multi-writer process is outside the scope of this document. This allows multiple solutions depending on the specific requirements of the implementation.

5.5 No VP-Controller Operation

Two VersaPHY devices may communicate between themselves using VP-Packets. A VP-Controller is not strictly required. In other implementations the VP-Devices may be configured by a VP-Controller then left to communicate with each other without VP-Controller involvement.

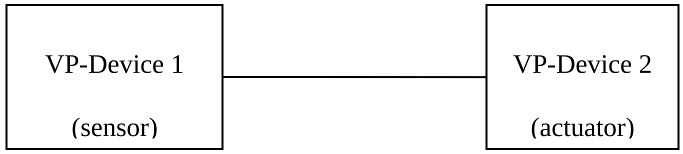

Figure 7 - No Controller VersaPHY Topology

5.6 Multi-function VP-Devices

Each physical VP-Device may provide multiple logical VP-Functions. Each of these VP-Functions may operate autonomously and only share the 1394 PHY with the other functions. Each VP-Function may be managed by a different VP-Controller. (Again, the connections shown are logical.)

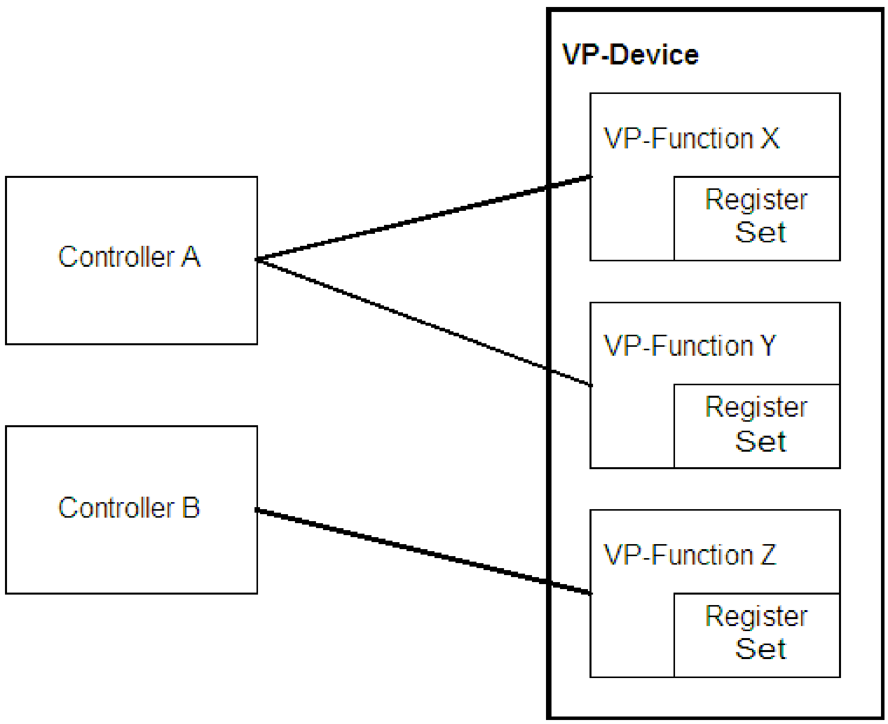

Figure 8 - Multi-Function VP-Device

5.7 Interoperability

Other non-VersaPHY 1394 devices may operate along side VersaPHY devices on the same 1394 network. VersaPHY systems can be added to existing 1394 systems to add value to existing infrastructure.

5.8 VersaPHY Interaction Model

As mentioned above, the base interaction is between a VersaPHY Controller (VP-Controller) and a VersaPHY Device (VP-Device). This specification makes no attempt to define how a VP-Controller and a VP-Device might interact. This section provides a few node interaction models to help the reader better understand the advantages of the VersaPHY architecture.

5.8.1 Polling

The VersaPHY architecture supports polling of the VP-Device by a VP-Controller. Using the polling model the VP-Controller simply reads VP-Device registers to get information from the VP-Device using Physical-IDs or VersaPHY labels (VP-Labels). This model has the advantage of being simple and each read is acknowledged, however it requires more VP-Controller CPU cycles than other models and loading of the VP-Controller may affect how often each VP-Device is polled.

5.8.2 Response

The VersaPHY architecture also supports a VP-Device initiated unsolicited response mechanism. Using this model, after the VP-Controller configures the VP-Device, the VP-Device autonomously broadcasts unsolicited response VP-Packets. The response packets contain the VP-Label of the VP-Function sending the response. This allows any VP-Controller to listen to the bus and collect data from all VP-Functions without expending any extra cycles. In this approach the VP-Device producing the data is responsible for the timing of the data transmission. It also generates less bus traffic.

This is the most efficient model, but the packets are not acknowledged. If guaranteed data delivery is required, the application will have to ensure it at a higher level.

In multiple VP-Controller environments with many VP-Functions, hardware filtering of VP-Packets using ranges of VP-labels maybe advantageous to reduce the number of packets controllers must parse in software.

5.8.3 Everything is Broadcast

"All data is broadcast" is a fundamental concept of VersaPHY communication. VP-Device response packets contain only the address of the source. The responses may include a transaction label to match a request transaction label, but they do not contain any information on the requestor or the intended data destination. Response data is broadcast to any device on the network that is listening for information from that VP-Device.

This feature can be used to simplify complex transactions like device discovery. During discovery only one VP-Controller needs to interrogate a new device, the rest of the controllers may simply listen to the responses. Obviously multiple controllers may listen to any VP-Function. If polling is used, only one VP-Controller need poll.

The broadcast feature may be managed in bridged bus applications. Through label configuration and bridge filtering, response packets may be restricted to the local bus and not broadcast to the entire network, but they are still broadcast to the local bus.

Accordingly, request packets are addressed to a single VP- Function, but do not contain the address of the sender. This means a VP- Function will respond appropriately to any request properly addressed to it. As previously mentioned, care must be taken when multiple controllers are allowed to write to a single VP- Function. For generic systems this is managed by controllers labeling all the VP-Functions they control. Typically other controllers would not write to VP-Functions labeled by another VP-Controller.

Some VP-Profiles may allow a function to accept and apply all properly addressed writes. To reduce hardware requirements, VP-Functions may only respond to the last write if more than one write is received before it can respond. Other VP-Profiles may define that only the first write should be accepted in specific instances. These VP-Profiles may specify a Physical_ID or Bus_ID/Cont_ID be used in the transaction label (tl) field so requestors may know with certainty which request was granted.

5.8.4 Streaming

The VersaPHY architecture supports legacy 1394 isochronous and asynchronous (non-transaction) streaming. All IEEE-1394 rules regarding resource allocation, packet format, and verification must be followed. However, by using the VersaPHY extensions the VP-Device requirements are reduced to packet format, CRC generation/verification, and cycle timer support. This permits very simple stream consumer and producer implementations. Typically a VP-Controller handles all resource reservations and configures the streaming VP-Device accordingly.

5.9 VersaPHY Benefits

5.9.1 Reduced Cable and Connector Costs

IEEE-1394 is being considered in a variety of applications where installation, maintenance support, cost and even weight of cable wiring is a concern. Many times in these applications the IEEE-1394 bus is adjacent to a device that could be controlled, monitored, and even powered via IEEE-1394. Often implementers elect to not use 1394 because of the complexity and expense of implementing a transaction capable node. The VersaPHY architecture eliminates these obstacles and allows the implementer to take advantage of reduced cable and connector costs, reduced weight, and easier wire harness installation and support.

5.9.2 Reduced Logic Requirements

Legacy 1394 devices are all relatively smart devices. Typically a logic intensive link device is connected to the 1394 PHY. Controlling this link device usually requires a microprocessor and software. This is appropriate for many complex devices, but is unnecessarily complicated and too expensive for many simple applications. VersaPHY technology enables 1394 devices with minimal logic and no microprocessor or software.

5.9.3 Leveraging Smart Nodes

The VP-Controller / VP-Device architecture enables a single “smart” VP-Controller device to manage multiple “dumb” devices. This leverages the hardware and software investment in a single VP-Controller by distributing its capabilities across multiple inexpensive devices. A single VP-Controller node can also service many 1394 network requirements for multiple devices instead of requiring all of those capabilities in each node.

5.9.4 VersaPHY and Streaming

Streaming as defined by IEEE-1394 is a datagram and is not a transaction. Using this model, the link layer connects directly to the application. The VersaPHY architecture takes advantage of this and removes the asynchronous transaction requirements from streaming devices. VersaPHY extensions replace these requirements with a much simpler set of VersaPHY requirements. This allows implementation of simple streaming devices such a digital cameras, speakers, and microphones with lighter hardware and without a link controller, 1394 transaction software, or microprocessor.

6 VersaPHY Labels

6.1 Overview

VP-Labels occupy a 14 bit field in register offset 0 of each VP-Function. VP-Label based request packets contain the VP-Label of the VP-Function(s) they are address to. VP-Label based response packets contain the VP-Label of the VP-Function issuing the response.

VP-Label usage is standardized to allow:

1. Simplified VersaPHY devices.
2. De-centralized label management.
3. Standardized device discovery process.
4. Reduced opportunity for inadvertent duplicate labels.
5. Minimized overhead for device connection / disconnection.
6. Simplified filtering of VersaPHY packets by controllers.
7. Simplified bridging of VersaPHY packets between busses.
8. Minimized overhead for bus addition / removal.

There are two categories of VersaPHY labels:

1. Static labels – Static labels are set during manufacture or installation and do not implement the ability to change. These VersaPHY Labels support the simplest VersaPHY devices. There are two static labels reserved for universal functions (bridges and orphans), and one reserved for assignable functions before assignment (orphans).
2. Assigned labels – VP-Controllers assign labels to VP-Functions that offer assignable labels. Managing assigned labels is divided into three areas.
    a. VP-Controller label spaces – VersaPHY controllers are responsible for annexing and managing their own label spaces.
    b. VersaPHY Bus label spaces – VersaPHY bridges must manage packet addressing across networks containing multiple 1394 busses.
    c. VersaPHY Label Mode – Advanced VersaPHY networks may re-configure label space allocation for larger numbers of 1394 busses or larger numbers of VersaPHY controllers by re-allocating bits in the VersaPHY Label.

6.2 Label Fields

The 14 bit VP-Label is sub-divided into 3 fields to facilitate label management.

0	1	2	3	4	5	6	7	8	9	10	11	12	13
Bus_ID/Cont_ID						Label							

Figure 9 - VP-Label fields

The 8 LSBs always contain the base label. The 6 MSBs are divided between Bus_ID and Controller_ID (Cont_ID) according to the current Label Mode as described in section 6.5. The Bus_ID bits identify a bus to facilitate bridging as described in section 6.6. The Cont_ID bits match VP-Controller address spaces to simplify multi-controller address management as described in section 6.4.

6.3 Static Label Space

The simplest VersaPHY devices may not have the ability to change their VP-Label. Their labels may be programmed in firmware, set by switches, or fixed by other methods. Static labels can eliminate the need for device discovery after power up, reducing complexity in both the VP-Controller and VP-Device. This method might be useful for applications like aircraft, automobiles or large pieces of equipment with multiple fixed location VP-Functions.

There are 509 labels reserved for static label VP-Functions on each bus. They are further divided between local bus static labels and network broadcast static labels. This theoretically permits (62*509) 31,558 unique statically labeled VP-Functions across the maximum 62 busses.

Bus/VP-Controller (VP_Label [0:5])		**Label** (VP_Label [6:13])	**Count**
Static Local Broadcast	000000_2	*00_{16} (orphans)*	*1*
		01_{16} –FE_{16} (available)	254
		FF_{16} (bridges)	*1*
Assignable Labels	*000001_2-111110_2*	*all*	--
Static Local Only	111111_2	*00_{16} (orphans)*	*1*
		01_{16} –FF_{16} (available)	255

Table 1 - Static Address Map

6.3.1 Static Local Broadcast

Bus 0, Controller 0, Labels 1-254 (00000000000001_2 –00000011111110_2) are reserved for static label VersaPHY devices mapped to their local bus.

Bus 0, Controller 0, VP_Label 0 (00000000000000_2) is not a static label. It is reserved as the initial (orphan) default value for VP-Functions with assignable labels that are likely to broadcast their data across bridges. The orphan label is not required for preconfigured controller-less networks (label mode 60).

Bus 0, Controller 0, Label 1 (00000011111111_2) is not a static label. It is reserved as the fixed label for bridge devices that manage data flow between busses. The bridge label is not reserved on preconfigured single bus networks (label mode 06).

VP-bridges will forward packets containing static local broadcast label to other busses. The bus value in the label of these packets will be altered as described later in the bridging section. Bridging allows duplicate static local labels on multiple busses of one network.

6.3.2 Static Local Only

Bus max., Controller max., Labels 1-255 (11111100000001_2 – 11111111111111_2) are reserved for static label VP-Functions that do not need their responses broadcast across bridges to the rest of the network.

VP-Bridges will not forward packets containing static local only labels to other buses. Static Local Only labels may be re-used on other busses without risk of interference. Care must be taken to ensure static broadcast labels are not duplicated on the same bus. Bridging functions are more completely discussed later.

Bus max., Controller max, Label 0 (11111100000000_2) is not a static label. It is reserved as the initial (orphan) default value for VP-Functions with assignable labels that are not likely to broadcast their data across bridges. The orphan label is not reserved on preconfigured controller-less networks (label mode 60). VP-Functions with a local only orphan address should be adopted by VP-Controllers managing a local only label space.

6.4 VP-Controller Label Spaces

VP-Functions with assignable VP-Labels are adopted by a VP-Controller and assigned an appropriate 14-bit label by that VP-Controller. Other controllers may monitor responses or issue requests to any VersaPHY device but each VP-Device has only one parent at a time. The bus and VP-Controller segments of the device's label identify the parent VP-Controller.

Note: Assigning the same label to multiple VP-Functions is permitted so labels may be used in multi-cast applications.

VersaPHY controllers may identify orphan VP-Functions by issuing reads to the default orphan addresses described above.

Each VP-Controller is responsible for managing its own label space(s). A VP-Controller has no responsibility for other VP-Controllers label spaces.

A VP-Controller that assigns labels shall respond with appropriate VP-Controller profile register set (Annex B) to all transactions addressed to label 0 of its address space. A VP-Controller that hasn't adopted a VP-Device may remain anonymous on the bus. Such a controller may only read VP-Functions or only interact with statically labeled VP-Functions.

VP-Controllers should not issue writes to VP-Functions they have not adopted unless the ramifications of nearly simultaneous writes from multiple controllers are addressed elsewhere in the system.

All controllers shall use the following process to annex a VP-Controller label space.

1. The VP-Controller shall issue a read to label 0 of the VP-Controller space it wishes to annex.
2. If no response is received, the VP-Controller shall begin responding to label 0 transactions addressed to that VP-Controller space.
3. The VP-Controller shall then re-read label 0 of the annexed label space to ensure no other VP-Controller has simultaneously annexed that space. If a response is received, the VP-Controller shall discontinue responding on label 0 and attempt the process on a different VP-Controller space.

The range of the controller identifier is configurable from 0 to 6 bits (0 to 62 address spaces). Section 6.6, Label Space Modes, describes how the bits are allocated. The table below describes label mode 33 where 3 bits are allocated to controller identification. Other modes are mapped similarly except label mode 60 where there are no controllers and all devices are statically labeled.

There are separate VP-Controller address spaces for each bus: local only, broadcast, and remote. Bus addressing is discussed in the next section.

Description	Bus	Controller	Base Label	Count
Reserved for static labels	000_2 (broadcast)	*000_2 (Static)*	*00_{16} to FF_{16}*	*256*
Broadcast Controller 1		001_2	00_{16} (controller label)	1
			01_{16}-FF_{16} (assignable)	255
Other Broadcast controllers		010_2-110_2	00_{16}-FF_{16}	1280
Broadcast Controller 7		111_2	00_{16} (controller label)	1
			01_{16}-FF_{16} (assignable)	255
Other busses	*001_2 - 110_2*	*000_2-111_2*	*00_{16}-FF_{16}*	*12,288*
Local Controller 0	111_2 (local only)	000_2	00_{16} (controller label)	1
			01_{16}-FF_{16} (assignable)	255
Other Local Controllers		001_2-101_2	00_{16}-FF_{16}	1280
Local Controller 6		110_2	00_{16} (controller label)	1
			01_{16}-FF_{16} (assignable)	255
Reserved for static labels		*111_2 (Static)*	*00_{16} to FF_{16}*	*256*

Table 2 - VP-Controller Address Space (3 bit)

A single VP-Controller may annex any unused label space; local only, local broadcast, remote broadcast, or network broadcast. The VP-Controller may annex multiple label spaces if required. The VP-Controller shall respond to transactions to label 0 of each label space it controls.

Individual VP-Controller label spaces eliminate the need to negotiate with other controllers for labels, and simplify packet filtering for low-level drivers and hardware.

6.5 Bus Label Spaces

6.5.1 Introduction

To support more than 63 IEEE-1394 nodes as VP-Devices, VersaPHY packets are bridgeable between busses. The Bus_ID bits in VP-Labels enable bridging VP-Packets. Bridging VP-Packets requires hardware bridges to collect VP-Packets from a given bus, selectively manipulate the bus ID bits in the address portion of the VP-Packets, and re-broadcast (forward) the packets on other busses.

Physically, VersaPHY networks can support up to 62 busses. Logically, on each physical bus, there are 4 unique bus addressing modes.

6.5.2 Local Only Bus

A Bus_ID of all 1_2s indicates the local only bus. Packets addressed to the Local Only Bus are not broadcast to the rest of the network and are known to have originated on the local bus.

6.5.3 Local Broadcast Bus

A Bus_ID of all 0_2s indicates the local broadcast bus. Packets addressed to the Local Broadcast Bus are manipulated by bridges and broadcast to the entire network. Packets addressed to the Local Broadcast Bus may have been generated anywhere on the network.

6.5.4 Network Broadcast Bus

If the Bus_ID in a VP-Packet matches the Bus_ID of the local bus, then the packet is addressed to the Network Broadcast Bus. These packets are manipulated by bridges so their Bus_IDs always match the local Bus_ID then re-broadcast to each bus on the network.

6.5.5 Remote Broadcast Busses

Packets with a Bus_ID not covered by the preceding three cases are addressed to the Local Broadcast Bus of a remote bus. If these packets originated on the remote bus with the matching Bus_ID then they originated as Local Broadcast Bus packets. If they originated elsewhere, they will be converted to Local Broadcast Bus packets when they reach the bus with a matching Bus_ID.

6.5.6 Bridges

Bridges are the only devices required to know the actual bus_ID of the busses on which they are operating. Typically, VersaPHY controllers will operate on the local broadcast bus (0_2s) or the local only bus (1_2s). VP-Functions operate using the 14 bit label assigned to them regardless of the physical bus to which they are connected.

The network broadcast bus, where the bus_ID in the packet equals the bus_ID of the bus carrying the packet, relies on the sender and receiver knowing their bus_IDs. This is most useful for bridge functions since they must know their bus_IDs. The network broadcast bus may be used by bridges to communicate with all bridges using a single packet.

Each bridge shall implement a VersaPHY Bridge Profile at Cont_ID 0, Label FF_{16} of each local broadcast bus to which it is connected. This allows multiple bridge devices at the same label, but since all bridges on each bus must operate in unison, this is desirable. A VP-Controller can learn on which bus it is connected by reading the bridge profile VP-Registers of the bridges on the local bus.

Detailed VP-Bridge profile functionality is outside the scope of this document. Bus_ID assignment, bus_ID re-assignment, bus_ID detection, network joining, loop avoidance, redundant bridging, and stream packet forwarding features will be specified by a VP-Bridge profile.

VP-Bridges have five bus addressing responsibilities when forwarding VP-Packets from one bus to another.

1. If the bus bits in the source_ID or destination_ID of an incoming VP-Packet match the local broadcast Bus_ID (0s), the bridge shall change those bus bits to match the bus_ID of the outgoing bus and forward the VP-Packet on the outgoing bus. (See Figure 10 - Bridging Example (label mode 33) - Examples: A, C, D, F)

2. If the bus bits in the source_ID or destination_ID of an incoming VP-Packet match the local only Bus_ID (1s), the bridge shall not forward the VP-Packet to the outgoing bus. (See Figure 10 - Bridging Example (label mode 33) - Examples: G, H)

3. If the bus bits in the source_ID or destination_ID of an incoming VP-Packet match the Bus_ID of the outgoing bus, the bridge shall change those bus bits to the local broadcast Bus_ID (0s) and forward the VP-Packet on the outgoing bus. (See Figure 10 - Bridging Example (label mode 33) - Example E).

4. If the bus bits in the source_ID or destination_ID of an incoming VP-Packet match the Bus_ID of the incoming bus, the bridge shall change those bus bits to match the Bus_ID of the outgoing bus and forward the VP-Packet on the outgoing bus. (See Figure 10 - Bridging Example (label mode 33) - Examples: I, J)

5. If the bus bits in the source_ID or destination_ID of an incoming VP-Packet do not match the Bus_ID of: the local broadcast bus (0s), the local only bus (1s), the incoming bus, or the outgoing bus, the bus shall forward the VP-Packet on the outgoing bus unchanged. (See Figure 10 - Bridging Example (label mode 33) - Example B)

The figure and table below demonstrate label mode 33 where the bus ID occupies 3 bits of the VP-Label.

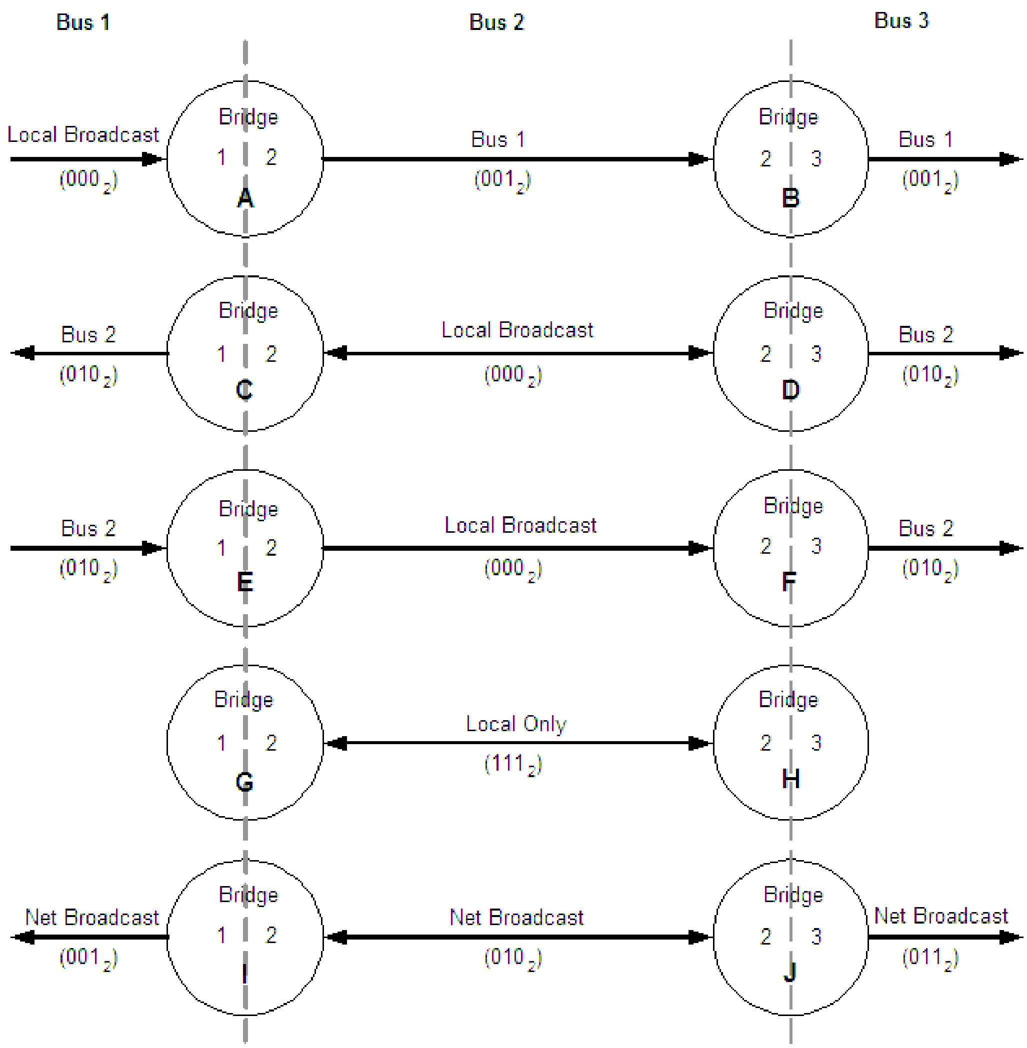

Figure 10 - Bridging Example (label mode 33)

Description	**Bus**	**Controller**	**Label**	**Count**
Static labels	000_2 (broadcast)	*000_2 (Static)*	*00_{16} to FF_{16}*	*256*
Local Broadcast Controller 1		001_2	00_{16}-FF_{16}	1792
Other Local Broadcast controllers		010_2-110_2		
Local Broadcast Controller 7		111_2		
Bus 1 Controller 1	001_2 (Bus 1)	001_2	00_{16}-FF_{16}	1792
Other Bus 1 Controllers		010_2-110_2		
Bus 1 Controller 7		111_2		
Other Busses	*010_2-101_2*	*001_2-111_2*	*00_{16} to FF_{16}*	
Bus 6 Controller 1	110_2 (Bus 6)	001_2	00_{16}-FF_{16}	1792
Other Bus 1 Controllers		010_2-110_2		
Bus 6 Controller 7		111_2		
Local Only Controller 0	111_2 (local only)	000_2	00_{16} (controller label)	1
			01_{16}-FF_{16} (assignable)	255
Other Local Only Controllers		001_2-101_2	00_{16}-FF_{16}	1280
Local Only Controller 6		110_2	00_{16} (controller label)	1
			01_{16}-FF_{16} (assignable)	255
Static labels		*111_2 (Static)*	*00_{16} to FF_{16}*	*256*

Table 3 - Bus ID Map (label mode 33)

6.6 Label Space Modes

To achieve maximum flexibility, the allocation of VP-Label bits to bus IDs and controller IDs is not fixed. Label Space Modes specify the range of bits in the VP_Label for these fields.

Label Space Modes can be changed to accommodate changing network requirements, but all bridges and controllers on a network must operate with the same Label Space Mode.

Six bits represent the bus ID and controller ID. Modes with a single controller ID bit are not useful due to reservations for static broadcast and static local only spaces. Modes with a single bus bit are not useful due reservations for local broadcast and local only bus reservations. These limitations yield the following Label Space Modes.

Label Mode	Bus Bits	VP-Controller Bits	Max Busses	Max Controllers / Bus
06	0	6	1	62
24	2	4	2	14
33	3	3	6	6
42	4	2	14	2
60	6	0	62	0 (all static)

Table 4 - Label Space Modes

General purpose controllers and bridges should initialize in mode 33.

If a VP-Controller is added to a bus and no VP-Controller space is available, it may attempt to change the Label Mode to a mode with more VP-Controller spaces, but first it shall verify and create new mode compatibility with all other controllers and bridges using the VP-Bridge and VP-Controller VP-Profiles to make appropriate changes.

If a bus is added to a network and a unique bus ID is not available, a VP-Controller may attempt to change the Label Mode to a mode permitting more buses, but first it shall verify and create new mode compatibility with all controllers and bridges. If multiple controllers attempt to increase the number of buses at they same time, assuming the number of controllers is sufficient, the larger value shall succeed. Only when VP-Controllers/Bridges disappear can the Label Mode be changed to a lower value if and when more controller ID spaces are required. If increasing the bus ID space is not possible due to the current number of annexed controller ID spaces, a controller shall reject the request to change Label Modes. Change requesting controllers shall not proceed with the Label Mode change if any controller on any bus rejects the request.

7 VersaPHY Packet Formats

This specification defines two new sets of PHY packets. The first set is addressed using standard 1394 Physical-IDs and the second set is addressed using the new VersaPHY Labels.

7.1 VersaPHY Packets using Physical-ID Addressing

This section defines all VersaPHY packet formats using Physical-IDs. There are four packets defined in this section:

- Read Request VersaPHY Packet
- Read Response VersaPHY Packet
- Write Request VersaPHY Packet
- Write Response VersaPHY Packet

VP-Devices shall use immediate bus requests to respond to VersaPHY packets addressed using Physical-IDs. Where appropriate VP-Devices may concatenate multiple VP-Packets to increase throughput.

7.1.1 Read Request VersaPHY Packet with Physical-ID Addressing

1	1	Bus	0	0	Dest_Physical_ID	tl	0	0	0b1110	PReg_Off
Blk_Number			Reserved(0x00)			Num_Bytes			CRC	

Figure 11 - Read Request VersaPHY packet using Physical-ID

7.1.2 Read Response VersaPHY Packet with Physical-ID Addressing

1	1	Bus	0	0	Source_Physical_ID	tl	0	1	0b1110	PReg_Off
Blk_Number			Data						CRC	

Figure 12 - Read Response VersaPHY packet using Physical-ID

7.1.3 Write Request VersaPHY Packet with Physical-ID Addressing

1	1	Bus	0	0	Dest_Physical_ID	tl	1	0	0b1110	PReg_Off
Blk_Number			Data						CRC	

Figure 13 - Write Request VersaPHY packet using Physical-ID

7.1.4 Write Response VersaPHY Packet with Physical-ID Addressing

1	1	Bus	0	0	Source_Physical_ID	tl	1	1	0b1110	PReg_Off
Blk_Number			Data						CRC	

Figure 14 - Write Response VersaPHY packet using Physical-ID

7.1.5 VersaPHY Physical-ID Packet Fields

Field	Comment
Bus	Bus ID field for bridging Physical-ID packets. For responses VersaPHY devices

	set this field to 000000_2 and respond only to request packets containing 000000_2. 000000_2 designates the local broadcast bus (static VP-Controller). VersaPHY bridges will modify this field as discussed in the bridging section to allow Physical-ID packets to traverse busses across the network. VersaPHY controllers may generate and receive packets with other than 000000_2 to communication via Physical-ID packets with VersaPHY devices on other busses. The most significant bits of the Bus field correspond to the Bus_ID of the packet. The remainder of the field, depending on the Label Mode, is padded with 0s.
Dest_Physical_ID	Physical-ID of a request packet's intended destination.
Source_Physical_ID	Physical-ID of the node sending a response packet.
tl	Transaction label used to associate a request with a response. The requester must ensure no outstanding tls have the same value. Some profiles may require a response be received before the next request is issued. In this case tl could have a constant value. The tl value of 111111_2 is reserved for use in unsolicited response packets.
PReg_Off	Physical-ID Register Offset coupled with Blk_Number together address the targeted register. The PReg_Off are the lower 4 bits of the address with the Blk_Number representing the upper 8 bits.
Blk_Number	PReg_Off coupled with Blk_Number together address the targeted register. The PReg_Off are the lower 4 bits (LSBs) of the address with the Blk_Number representing the upper 8 bits (MSBs).
Data	For Write Requests this field contains the data to be written. For Responses this field contains the current register value.
Num_Bytes	Number of bytes requested for read request. VersaPHY device may respond with multiple packets for requests over 2 bytes.
CRC	The CRC covers the first 56-bits of the packet. It is an 8 bit CRC: poly = x^8 + x^2 + x^1 + x^0, initial value = FF_{16}, final result inverted. See 9.6Annex C for example code. Some profiles may not utilize this packet CRC. This field is reserved to permit hardware CRC validation for profiles that do use packet CRCs. VP-Controllers with hardware CRC validation shall allow transfer of packets with invalid CRCs to support profiles that don't use packet CRCs.

Table 5 – VersaPHY Packet using Physical-ID Field Definition

7.2 VersaPHY Packets using VersaPHY Label Addressing

This section defines all VersaPHY packet formats using VP-Labels. There are four packets defined in this section.

- Read Request VersaPHY Packet
- Read Response VersaPHY Packet

- Write Request VersaPHY Packet
- Write Response VersaPHY Packet

VersaPHY request packets addressed using VP-Labels when appropriate may respond using immediate bus requests or split and respond later. Timing requirements of split packets is profile dependant. Permission to use immediate requests is granted by the VP-Controller and stored in the VersaPHY I register bit. VP-Devices may concatenate multiple VP-Packets to increase throughput.

7.2.1 Read Request VersaPHY Packet with VP-Label

0	1	Dest_VP_Label	tl	0	0	0b1110	LReg_Off
Profile_Defined_Addr/Data						CRC	

Figure 15 - Read Request VersaPHY packet using VP-Label

7.2.2 Read Response VersaPHY Packet with VP-Label

0	1	Source_VP_Label	tl	0	1	0b1110	LReg_Off
Profile_Defined_Addr/Data						CRC	

Figure 16 - Read Response VersaPHY packet using VP-Label

7.2.3 Write Request VersaPHY Packet with VP-Label

0	1	Dest_VP_Label	tl	1	0	0b1110	LReg_Off
Profile_Defined_Addr/Data						CRC	

Figure 17 - Write Request VersaPHY Packet using VP-Label

7.2.4 Write Response VersaPHY Packet with VP-Label

0	1	Source_VP_Label	tl	1	1	0b1110	LReg_Off
Profile_Defined_Addr/Data						CRC	

Figure 18 - Write Response VersaPHY Packet using VP-Labels

7.2.5 VersaPHY VP-Label Packet Fields

Field	Comment
Dest_VP_Label	VersaPHY Label of the request packets destination.
Source_VP_Label	VersaPHY Label of the device sending the response packet.
tl	Transaction label is typically used to associate a request with a response. Responses should echo the tl provided by the request. The tl value of 111111_2 is reserved for use in unsolicited responses. VP-Profiles may offer further requirements.
LReg_Off	Register Offset Low – is the lowest four bits of the VersaPHY register being accessed (request) or having been accessed (response). This provides a minimum of 16 bytes of registers. For simple devices this maybe all that is required. For more complex devices this is enough to discover what profile is supported for the VP-label.
Profile Defined Addr/Data	A 24 bit field allocated between address and data information according to the VP-Profile specification in use. Unless required by profile specifics, it is recommended for new profiles to use the same format as the Physical-ID packets Blk_Number and data fields to minimize complexity.
CRC	The CRC covers the first 56-bits of the packet. It is an 8 bit CRC: poly = x^8 + x^2 + x^1 + x^0, initial value = FF_{16}, final result inverted. See 9.6Annex C for example code. Some profiles may not utilize this packet CRC. This field is reserved to permit hardware CRC validation for profiles that do use packet CRCs. VP-Controllers with hardware CRC validation shall allow transfer of packets with invalid CRCs to support profiles that don't use packet CRCs.

Table 6 – VersaPHY VP-Label Packet Field Definitions

8 VersaPHY Registers

VersaPHY Registers (VP-Registers) are accessed using the VP-Packets defined in the previous sections. VP-Registers are separate from the standard PHY registers defined in IEEE-1394a-2000 and IEEE-1394b-2002.

VP-Registers are structured to support simple to complex devices. A VP-Device may contain more than one VP-Function allowing a single VP-Device to perform several disparate functions. Each VP-Function within a VP-Device shall have a unique VP-Register space.

VersaPHY register addressing is divided into 16 byte 'blocks'. The VP-Device shall support the first block (block 0) of 'Base registers' for each VP-Device register space. Additional profile defined register blocks are optional for each VP-Profile.

8.1 VersaPHY Register Addressing

VP-Registers are accessed by two mechanisms: VP-Label packets, and Physical-ID packets. Physical-ID addressing is more strictly defined to provide for generic VP-Device discovery, VP-Function discovery, and VP-Label setup. Once VP-labels are properly configured, VP-Label addressing offers greater simplicity and flexibility.

VP-Label and Physical_ID accesses are performed on logically separate register sets. Many of the basic status fields are duplicated in the same relative position whether accessed via Physical_ID or VP-Label packets. Other fields required for Physical_ID access are not allocated as required spaces in the VP-Label register map.

A minimal VP-Function shall represent a 16 byte VP-Label register block and a 16 byte Physical_ID register block. VP-Profiles may duplicate registers between these blocks and set other registers to constants to minimize the memory requirements for the implementation.

Any fields not required for VP-Label access may be duplicated in the VP-Label register space to simplify address decoding logic in the VP-Device. All duplication or differences between VP-Label and Physical_ID access must be specified in the VP-Profile specification. This provides maximum flexibility for unique requirements of individual VP-Profiles.

Physical_ID access to specific register in a specific node is addressed using the Blk_Number and PReg_Off (12 bits total). This limits the number of Physical_ID addressable registers to 4K bytes per node.

The number of VP-Functions for each VP-Device is then limited to 256 since each VP-Function requires a minimum 16 byte register block. Fewer functions are possible when implementing VP-Profiles that require more than one block per VP-Function.

VP-Label access of a specific register within a VP-Function is addressed using the Block Address and LReg_Off. The 'Block Address' specified by a VP-Profile can range from 0 to 16 bits (assuming an 8 to 24 bit data field). This allows 16 to 1M byte wide registers per VP-Function label space.

8.2 Physical_ID Registers

8.2.1 VP-Function Ordering

As mentioned previously, each VP-Function must be represented by at least one 16 byte register block in the Physical_ID register space. Each 1394 node has a single Physical_ID, so there is only one Physical_ID register space for each VP-Device.

The Physical_ID register blocks for each VP-Function are contiguous and addressed by their block number. The Instance_nOffset field of each VP-Function provides the absolute block address of the next available function so VP-Controllers aren't required to understand all VP-Profile block counts in order to step through available VP-Functions in a VP-Device. An Instance_nOffset = 0 shall indicate the last VP-Function in the VP-Device.

8.2.2 Physical-ID Addressing Byte Alignment

Some VersaPHY implementations may support only single bytes access, or only evenly aligned doublet access. Odd aligned doublet access is not supported.

When an odd aligned register is accessed a VP-Device supporting single byte access shall respond with the requested address and data in the response packet.

When an odd aligned register is requested from a VP-Device supporting only doublet access, the VP-Device shall respond with the requested address – 1 and the data is considered unreliable. This allows a VP-Controller to quickly determine if the VP-Device supports single or double byte access.

When accessing the registers using single byte accesses bits 0:7 represent byte 0 of the doublet and therefore represents the lower numbered register in the doublet. Therefore when reading block 0 register 0 using Physical-IDs the read response for this register would contain field E, I and Identifier [0:5] and block 0 register 5 would contain Config_Ver [0:7].

Byte alignment for VP-Label packets is defined in the VP-Profile specification.

8.2.3 Physical_ID Base Register Format

The first 6 bytes of the Physical_ID register space are defined identically for all VP-Functions. The ID_Type and G fields define the next 2-14 bytes.

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	0	E	I	VP-Label													
n	2	W	B	reserved									Power				
n	4	V	S	r	r	O	ID_type		G	Instance_nOffset							

Figure 19 - PHYsical_ID Register Map 0-5 (required)

Field	Manager	Definition
E	Controller	Enabled -- When 1 the VP-Function is operational. When 0 the VP-Function is disabled. Shall be set to 1 if Enable functionality is not implemented.
I	Controller	Immediate -- Set to one by VP-Controller to enable immediate responses to VP-Label packets. If not supported, the VP-Function shall set I to 0 and not allow writes. Shall be set to 0 on power up, connection, or VP-Function reset. If more than one VP-Functions with identical VP-Labels have I=1 their responses may collide or otherwise not be delivered. VP-Controllers should attempt to clear I if responses are not reliable.
VP-Label	Controller	VP-Label (static or assigned) of VP-Function. Shall be a valid static address if not writeable. Should default to orphan address when writeable.

W	Controller	Watchdog – Used by the owner VP-Controller to indicate its presences after a bus reset. Set to 1 to prevent VP-Function from resetting itself a VP-Profile defined watchdog time after a bus reset. Shall be set to 1 if watchdog functionality is not implemented. If implemented, shall be set to 0 and watchdog timer started after bus reset. If watchdog timer expires before watchdog is written to 1, the VP-Function shall perform a VP-Profile specified reset.
B	Controller	Bus Reset Ack -- Set to enable VP-Device to send an unsolicited status response after a bus reset. Shall be set to 0 if bus reset ack functionality is not implemented. Timing and format of response is VP-Profile defined.
Power	Controller/ Device	Power Control – The Power control field is defined in section **9**.
V	Device	Valid -- Indicates VP-Function is operational. Shall be set to 1 for valid, operational VP-Functions.
S	Device	Self Enabled -- Set to indicate the E bit was set by the VP-Function, not a VP-Controller. Set to 1 if the E bit was set by the VP-Function. Set to 0 if the VP-Function defaults to E=0 and doesn't autonomously change it. This helps identify if a controller is listening to a statically labeled VP-Function.
O	Device	Owner Field -- Indicates implementation of an Owner field in byte 0 of the VP-Label register set
ID_type	Device	Identification Type -- Indicates one of 3 VP-Profile identification modes. 01_2: Short Configuration Version 10_2: Long Configuration Version 11_2: Custom Configuration Version 00_2: Reserved, not for use in the Physical_ID register block (See the profile identification modes in the next section.)
G	Device	GUID available – When set to 1, it Indicates a 64 bit Globally Unique Identifier follows the configuration ID. (See the configuration ID modes in the next section.) If set to 0, profile defined registers begin after the configuration ID and no GUID is available for the VP-Function. It is still possible to infer a GUID from another VP-Function in the same VP-Device.
Instance_nOffset	Device	Offset to next VP-Function -- The absolute offset in the Physical_ID register space of the next VP-Function. Because this is an absolute offset starting from the first registers (block 0 register 0) it is guaranteed to be a unique number and therefore could be combined with the GUID to provide an instance identifier by the VP-Controller. The last VP-Function in the VP-Device will report and offset of 0, pointing back to the first VP-Function, and indicating that it is the last VP-Function in the VP-Device.
r, reserved	Reserved	Reserved fields set to 0
All the fields above may be fixed (read only) values if their features are not required by the VP-Profile.		

Table 7 - Physical_ID Register Fields

8.2.4 Physical_ID Addressed Registers

There are three profile identification methods defined for the Physical_ID register set to conserve required registers for common standard functions and to provide VP-Profile identification for custom and proprietary VP-Profiles.

8.2.4.1 Short Configuration Format

When the VP-Function reports ID_type=01, register 6 is used to identify the VP-Profile of the VP-Device.

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	6	Short_Config_Ver								PD_ROM 7							

Figure 20 - Short Configuration Version

Field	Name	Description
Short_Config_Ver	Short Configuration Version	Short configuration versions are 8 bit identifiers assigned by the 1394 TA and optionally ratified by the IEEE to identify the most common or fundamental VP-Profiles. As there are only 256 available short configuration versions the list is exclusive. The main advantage of the Short_Config_Ver is it allows a base register block with a 64 bit GUID to fit in a single 16 byte register block.
PD_ROM 7	Profile Defined Read Only Register offset 7	Since the Short_Config_Ver is read only, it is recommended that VP-Profiles use this location as a read only field to simplify 16 bit accesses. It may be appropriate for the short configuration VP-Profile to define this field as a revision of the Short_Config_Ver.

Table 8 - Short Configuration Fields

8.2.4.2 Long Configuration Format

When the VP-Function reports ID_type=10, registers 6,7, and 8 are used to identify the VP-Profile of the VP-Device.

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	6	Long_Config_Ver [0:7]								Long_Config_Ver [8:15]							
n	8	Long_Config_Ver [16:23]								PD_ROM 9							

Figure 21 - Long Configuration Version

Field	Name	Description
Long_Config_Ver	Long Configuration Version	Long configuration versions are 24 bit identifiers assigned by the 1394 TA and optionally ratified by the IEEE to other standardized VP-Profiles.
PD_ROM 9	Profile Defined Read Only Register offset 9	Since the Long_Config_Ver[16:23] field is read only, it is recommended that VP-Profiles use this location as a read only field to simplify 16 bit accesses.

Table 9 - Long Configuration Fields

8.2.4.3 Custom Configuration Format

When the VP-Function reports ID_type=11, registers 6, 7, 8, 9, A, and B are used to identify the VP-Profile of the VP-Device.

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	6	Config_ID [0:7]								Config_ID [8:15]							
n	8	Config_ID [16:23]								Config_Ver [0:7]							
n	A	Config_Ver [8:15]								Config_Ver [16:23]							

Figure 22 - Custom Configuration Identification

Field	Name	Description
Config_ID	Configuration ID	Contains the 24 bit Organizationally Unique Identifier (OUI) of the organization controlling the VP-Profile specification.
Config_Ver	Configuration Version	Defined by the organization identified by Config_ID.

Table 10 - Custom Configuration Fields

8.2.5 VP-Device Identification

When the VP-Function reports G=1, the 64 bits following the configuration information shall report a Globally Unique Identifier. This identifier uses the IEEE EUI-64 format. If G=0, there is no GUID for that VP-Function and the registers following the configuration information are available for definition by the VP-Profile.

The location of the optional GUID is specified to enable VP-Controllers to collect available GUID information without specific VP-Profile knowledge.

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	8/A/C	GUID [0:7]								GUID [8:15]							
n	A/C/E	GUID [16:23]								GUID [24:31]							
n/n+1	C/E/0	GUID [31:39]								GUID [40:47]							
n/n+1	E/0/2	GUID [48:55]								GUID [56:63]							

Figure 23 - VP-Device Identification

The actual location of the GUID bits is determined by ID_type.

ID_type	GUID Start		GUID End	
	Block	Offset	Block	Offset
01_2	n	6_{16}	n	F_{16}
10_2	n	8_{16}	n+1	1_{16}
11_2	n	A_{16}	n+1	3_{16}

Table 11 - GUID Location

Field	Name	Description
GUID	Globally Unique IDentifier	64 bit unique identifier formatted as an EUI-64. The GUID identifies the VP-Device not the VP-Function. All VP-Functions in a given device that report a GUID shall report the same GUID value. The Instance_nOffset field may be used to uniquely identify individual VP-Functions sharing a common VP-Device GUID.

Table 12 - Device Identification Fields

8.2.6 Physical_ID Profile Defined Registers

Any remaining registers in block 1 following the declared configuration ID registers and the GUID (when indicated) are available for VP-Profile definition. Additional 16 byte blocks may also be specified by the VP-Profile and allocated with the Instance_nOffset field, to provide Physical_ID based access.

8.3 VP-Label Addressed Registers

8.3.1 VP-Label Base Register Format

The first 5 bytes of the VP-Label register space are defined identically for all VP-Functions.

Many fields in the VP-Label registers correspond exactly in location and value to the fields in the Physical_ID register set. At the option of the VP-Profile all registers after byte 1 may be identical to their Physical_ID register counterpart.

The VP-Label register set is not required to match the Physical_ID register set to allow for unique requirements of VP-Profiles. VP-Profiles developed for implementation in minimal logic may choose to duplicate many fields between the two register spaces to simplify address decoding.

Only byte 1 (the Physical_ID field) is required to be different from the Physical_ID register set.

Offset	Bit															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	E	I	Owner						0	0	Physical_ID					
2	W	B	reserved									Power				
4	V	S	r	r	O	ID_type		G	PDROM_5							

Figure 24 - PHYsical_ID Register Map 0-5 (required)

	Field	Definition
*	E	As defined Table 7 - Physical_ID Register Fields
*	I	As defined Table 7 - Physical_ID Register Fields
	Owner	Owner – If the O bit is set to 1, the owner field is available for a VP-Controller to claim a statically labeled VP-Function by writing the controller address (Bus_ID-Cont_ID) to the 6 bit Owner field. This enables other controllers to identify statically labeled VP-Functions that are in use by another VP-Controller. Note: The Owner field will not be readable or writeable by VP-Controllers on other busses when a local-only static label is assigned to the VP-Function. Owner shall be set to zeros if not operational.
	Physical_ID	Physical_ID -- 1394 node Physical_ID of the VP-Device
*	W	As defined Table 7 - Physical_ID Register Fields
*	B	As defined Table 7 - Physical_ID Register Fields
*	Power	As defined Table 7 - Physical_ID Register Fields
*	V	As defined Table 7 - Physical_ID Register Fields
*	S	As defined Table 7 - Physical_ID Register Fields
*	O	As defined Table 7 - Physical_ID Register Fields (Also see the owner field above.)
**	ID_type	As defined Table 7 - Physical_ID Register Fields with one additional type: 00_2: No Configuration Fields (See the profile identification modes in the Physical_ID register section.)
**	G	GUID available -- Indicates a 64 bit Globally Unique Identifier follows the VP-Profile identifier. (See the identification modes in the next section.)
**	PDROM_5	Profile Defined Read Only register 5 -- The Instance_nOffset field has little value in the VP-Label domain so it is not required to be repeated. It may be repeated at the option of the VP-Profile. This value is recommended for read only access to enable 16 bit management of registers 4 and 5.
	r, reserved	Reserved fields set to 0
* Indicates field shall be a redundant representation of the Physical_ID register set field ** Indicates field may be a redundant representation of the Physical_ID register set field.		

8.3.2 Configuration ID and GUID Fields

If ID_type or G is not set to 0, the configuration identification and GUID fields are repeated in the VP-Label registers as described in the Physical_ID register section.

8.3.3 Additional VP-Label Registers

All additional VP-Label registers are defined by the VP-Profile. They may also correspond to Physical_ID registers at the option of the VP-Profile.

Addressing additional registers past the 16 enabled by the 4 bit LReg_Off field in the label based VP-Packets is also defined by the VP-Profile. Reusing the 8 bit block addressing field specified for Physical_ID access is recommended to promote uniform access.

9 Power Control

The VersaPHY base register block allocated 5 bits from register 3 of block 0 in both the VP-Label addressed space and the Physical_ID space to optionally manage the power state of each VP-Function (or the entire VP-Device if that function represents the entire device).

The Power fields are defined in the base registers to allow a standard set of non-VP-Profile specific local power management capability. This enables otherwise non-VP-Devices to take full advantage PHY layer driven local power management with minimal complexity and cost.

VersaPHY power management allows several implementation options. A few options are listed below. These options are not mutually exclusive and may be combined to meet specific system requirements.

- Power control of individual VP-Functions on multi-function or single function VersaPHY devices may be managed independently.
- Power control for an entire VP-Device may be managed at once by a single VP-Function that operates a proxy for all the other VP-Functions in the VP-Device.
- Power control for multiple VP-Devices may be managed by a “broadcast power” VP-Label applied to power control VP-Functions in many devices. This permits a single broadcast packet from a power controller to manage the power state of many devices.
- VersaPHY power management only functions or devices may control the power state of ancillary circuits that have no other interaction with 1394.

9.1 Power Control Fields

The 5 bits reserved for power control are separated into two fields; R_Mode and Power_State.

The R_Mode bits are used for requests from the power controller to the function, the response from the function back to the power controller, reporting the current status of the function, and indicating the initialization status of the function after a status change.

Writes to Power_State contain the requested Power_State. The VP-Function shall always report the current Power_State in all write and read responses.

Bit	0	1	2	3	4
Field	R_Mode		Power_State		

Table 13 - Power control fields

Only two of the eight power states are defined by VersaPHY: 0b000 = off and 0b111 = on. The remaining states are defined by the profile specification of the VP-Function.

State	Power_State
Off	000_2
Profile Defined	001_2
Profile Defined	010_2

Profile Defined	011_2
Profile Defined	100_2
Profile Defined	101_2
Profile Defined	110_2
On	111_2

Table 14 - Power states

9.2 Requests

Three requests are supported: Announce, Set, and Force.

The power controller can "Announce" a power state change to collect an indication from the VP-Function whether or not it is ready and able to transition to the announced power state. No change in power state is performed by the VP-Function after an Announce request.

After receiving an affirmative indication from a VP-Function to an Announce request, the power controller may then issue a "Set" request for the VP-Function to enter the specified power state.

Whether or not the target VP-Function responds positively to an Announce request the power controller may issue a "Force" request to compel a VP-Function to transition to the specified power state.

Request	R_Mode	Power_State
reserved	00_2	Any state (no action)
Announce Power State	01_2	Requested State
Set Power State	10_2	Requested State
Force Power State	11_2	Requested State

Table 15 - Power write requests

9.3 Responses

Each power state write request from the power controller shall be honored with a write response from the VP-Function. There are two possible responses; Accepted and Rejected.

If the VP-Function is able to accept the request, it responds by echoing the R_Mode bits from the request in the R_Mode field of the write response.

If the VP-Function is unable to accept the request, it responds with 0's in the R_Mode field.

The response value of the Power_State field should indicate the current Power_State of the VP-Function. This value is not necessarily the same value contained the request. It may take longer to implement the request than to generate the response, or in the power off case a response should be sent before the VP-Function suspends operation.

Response	R_Mode	Power_State
Accepted	Echo Request R_Mode	Current Power_State

Rejected	00_2	Current Power_State

Table 16 - Power write responses

9.4 Reads

The power controller is also able to understand the current power status of any VP-Function by issuing a read request to the power register. The R_Mode field of the read response from the VP-Function should indicate if it is processing a request or idle.

Current Activity	R_Mode	Power_State
Idle	00_2	Current Power_State
Processing Announce	01_2	Current Power_State
Processing Set	10_2	Current Power_State
Processing Force	11_2	Current Power_State

Table 17 - Power read responses

9.5 Initial Value / Internal Change

After VP-Function power up, or if the VP-Function changes power states on its own, it may issue an unsolicited write response to indicate its power state to the power controller. If the VP-Function has restored internal states or variables to previous values, it should report and R_Mode of 0b11. If the VP-Function does not issue an unsolicited write response, it should hold R_Mode = 0b11 until the first read or write request to the register containing the Power field.

Field	R_Mode	Power_State
Restored	11_2	Current Power_State
Re-Initialized	00_2	Current Power_State

Table 18 - Value for unsolicited write responses

9.6 Default Value

VP-Functions are not required to support power management. If a function does not support any power management features it shall respond to all read and write requests with the following values.

Bit	**0**	**1**	**2**	**3**	**4**
Field	R_Mode		Power_State		
Value	0	0	1	1	1
Interpretation	Always Idle, All Requests Rejected		Power Always On		

Table 19 - Power unsupported ROM setting

Annex A
(normative)

Label Mode Network Device Address Maps

Label Mode Notes

Only small sections of the Label Mode 33 address map are shown in the specification. Below are complete representations of all 6 Label Mode address maps.

Note: The bus address of the network broadcast bus matches the bus_ID of each bus. One of the busses shown on each table will be the network broadcast bus. Which bus is the network broadcast bus depends on which bus the packet is traveling.

Note: Modes 06 and 60 are special cases for specific installations and may not work well for ad-hoc networks. Mode 06 is limited to a single bus and has no reserved bridge addresses. Mode 60 does not support controller address spaces. It assumes all VP-Devices are statically labeled so it doesn’t reserve any orphan addresses or controller addresses.

Ad-hoc networks may be enabled to move between modes 24, 33, and 42 if network requirements change.

Mode 06

Cont_ID (binary)	Label (hex)	Description	Cont_ID (binary)	Label (hex)	Description	Cont_ID (binary)	Label (hex)	Description
000000	00	Local Only Orphans Label	010101	01-FF	Local Only Controller 21 Devices	101011	00	Local Only Controller 43 Label
000000	01-FF	Local Only Static Devices	010110	00	Local Only Controller 22 Label	101011	01-FF	Local Only Controller 43 Devices
000001	00	Local Only Controller 1 Label	010110	01-FF	Local Only Controller 22 Devices	101100	00	Local Only Controller 44 Label
000001	01-FF	Local Only Controller 1 Devices	010111	00	Local Only Controller 23 Label	101100	01-FF	Local Only Controller 44 Devices
000010	00	Local Only Controller 2 Label	010111	01-FF	Local Only Controller 23 Devices	101101	00	Local Only Controller 45 Label
000010	01-FF	Local Only Controller 2 Devices	011000	00	Local Only Controller 24 Label	101101	01-FF	Local Only Controller 45 Devices
000011	00	Local Only Controller 3 Label	011000	01-FF	Local Only Controller 24 Devices	101110	00	Local Only Controller 46 Label
000011	01-FF	Local Only Controller 3 Devices	011001	00	Local Only Controller 25 Label	101110	01-FF	Local Only Controller 46 Devices
000100	00	Local Only Controller 4 Label	011001	01-FF	Local Only Controller 25 Devices	101111	00	Local Only Controller 47 Label
000100	01-FF	Local Only Controller 4 Devices	011010	00	Local Only Controller 26 Label	101111	01-FF	Local Only Controller 47 Devices
000101	00	Local Only Controller 5 Label	011010	01-FF	Local Only Controller 26 Devices	110000	00	Local Only Controller 48 Label
000101	01-FF	Local Only Controller 5 Devices	011011	00	Local Only Controller 27 Label	110000	01-FF	Local Only Controller 48 Devices
000110	00	Local Only Controller 6 Label	011011	01-FF	Local Only Controller 27 Devices	110001	00	Local Only Controller 49 Label
000110	01-FF	Local Only Controller 6 Devices	011100	00	Local Only Controller 28 Label	110001	01-FF	Local Only Controller 49 Devices
000111	00	Local Only Controller 7 Label	011100	01-FF	Local Only Controller 28 Devices	110010	00	Local Only Controller 50 Label
000111	01-FF	Local Only Controller 7 Devices	011101	00	Local Only Controller 29 Label	110010	01-FF	Local Only Controller 50 Devices
001000	00	Local Only Controller 8 Label	011101	01-FF	Local Only Controller 29 Devices	110011	00	Local Only Controller 51 Label
001000	01-FF	Local Only Controller 8 Devices	011110	00	Local Only Controller 30 Label	110011	01-FF	Local Only Controller 51 Devices
001001	00	Local Only Controller 9 Label	011110	01-FF	Local Only Controller 30 Devices	110100	00	Local Only Controller 52 Label
001001	01-FF	Local Only Controller 9 Devices	011111	00	Local Only Controller 31 Label	110100	01-FF	Local Only Controller 52 Devices
001010	00	Local Only Controller 10 Label	011111	01-FF	Local Only Controller 31 Devices	110101	00	Local Only Controller 53 Label
001010	01-FF	Local Only Controller 10 Devices	100000	00	Local Only Controller 32 Label	110101	01-FF	Local Only Controller 53 Devices
001011	00	Local Only Controller 11 Label	100000	01-FF	Local Only Controller 32 Devices	110110	00	Local Only Controller 54 Label
001011	01-FF	Local Only Controller 11 Devices	100001	00	Local Only Controller 33 Label	110110	01-FF	Local Only Controller 54 Devices
001100	00	Local Only Controller 12 Label	100001	01-FF	Local Only Controller 33 Devices	110111	00	Local Only Controller 55 Label
001100	01-FF	Local Only Controller 12 Devices	100010	00	Local Only Controller 34 Label	110111	01-FF	Local Only Controller 55 Devices
001101	00	Local Only Controller 13 Label	100010	01-FF	Local Only Controller 34 Devices	111000	00	Local Only Controller 56 Label
001101	01-FF	Local Only Controller 13 Devices	100011	00	Local Only Controller 35 Label	111000	01-FF	Local Only Controller 56 Devices
001110	00	Local Only Controller 14 Label	100011	01-FF	Local Only Controller 35 Devices	111001	00	Local Only Controller 57 Label
001110	01-FF	Local Only Controller 14 Devices	100100	00	Local Only Controller 36 Label	111001	01-FF	Local Only Controller 57 Devices
001111	00	Local Only Controller 15 Label	100100	01-FF	Local Only Controller 36 Devices	111010	00	Local Only Controller 58 Label
001111	01-FF	Local Only Controller 15 Devices	100101	00	Local Only Controller 37 Label	111010	01-FF	Local Only Controller 58 Devices
010000	00	Local Only Controller 16 Label	100101	01-FF	Local Only Controller 37 Devices	111011	00	Local Only Controller 59 Label
010000	01-FF	Local Only Controller 16 Devices	100110	00	Local Only Controller 38 Label	111011	01-FF	Local Only Controller 59 Devices
010001	00	Local Only Controller 17 Label	100110	01-FF	Local Only Controller 38 Devices	111100	00	Local Only Controller 60 Label
010001	01-FF	Local Only Controller 17 Devices	100111	00	Local Only Controller 39 Label	111100	01-FF	Local Only Controller 60 Devices
010010	00	Local Only Controller 18 Label	100111	01-FF	Local Only Controller 39 Devices	111101	00	Local Only Controller 61 Label
010010	01-FF	Local Only Controller 18 Devices	101000	00	Local Only Controller 40 Label	111101	01-FF	Local Only Controller 61 Devices
010011	00	Local Only Controller 19 Label	101000	01-FF	Local Only Controller 40 Devices	111110	00	Local Only Controller 62 Label
010011	01-FF	Local Only Controller 19 Devices	101001	00	Local Only Controller 41 Label	111110	01-FF	Local Only Controller 62 Devices

010100	00	Local Only Controller 20 Label	101001	01-FF	Local Only Controller 41 Devices	111111	00	Local Only Orphans Label
010100	01-FF	Local Only Controller 20 Devices	101010	00	Local Only Controller 42 Label	111111	01-FF	Local Only Static Devices
010101	00	Local Only Controller 21 Label	101010	01-FF	Local Only Controller 42 Devices			`

Mode 24

Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description
00-0000	00	Local Broadcast Orphans Label	01-0101	00	Bus 1 Broadcast Controller 5 Label	10-1010	01-FF	Bus 2 Broadcast Controller 10 Devices
00-0000	01-FE	Local Broadcast Static Devices	01-0101	01-FF	Bus 1 Broadcast Controller 5 Devices	10-1011	00	Bus 2 Broadcast Controller 11 Label
00-0000	FF	Local Broadcast Bridges Label	01-0110	00	Bus 1 Broadcast Controller 6 Label	10-1011	01-FF	Bus 2 Broadcast Controller 11 Devices
00-0001	00	Local Broadcast Controller 1 Label	01-0110	01-FF	Bus 1 Broadcast Controller 6 Devices	10-1100	00	Bus 2 Broadcast Controller 12 Label
00-0001	01-FF	Local Broadcast Controller 1 Devices	01-0111	00	Bus 1 Broadcast Controller 7 Label	10-1100	01-FF	Bus 2 Broadcast Controller 12 Devices
00-0010	00	Local Broadcast Controller 2 Label	01-0111	01-FF	Bus 1 Broadcast Controller 7 Devices	10-1101	00	Bus 2 Broadcast Controller 13 Label
00-0010	01-FF	Local Broadcast Controller 2 Devices	01-1000	00	Bus 1 Broadcast Controller 8 Label	10-1101	01-FF	Bus 2 Broadcast Controller 13 Devices
00-0011	00	Local Broadcast Controller 3 Label	01-1000	01-FF	Bus 1 Broadcast Controller 8 Devices	10-1110	00	Bus 2 Broadcast Controller 14 Label
00-0011	01-FF	Local Broadcast Controller 3 Devices	01-1001	00	Bus 1 Broadcast Controller 9 Label	10-1110	01-FF	Bus 2 Broadcast Controller 14 Devices
00-0100	00	Local Broadcast Controller 4 Label	01-1001	01-FF	Bus 1 Broadcast Controller 9 Devices	10-1111	00	Bus 2 Broadcast Controller 15 Label
00-0100	01-FF	Local Broadcast Controller 4 Devices	01-1010	00	Bus 1 Broadcast Controller 10 Label	10-1111	01-FF	Bus 2 Broadcast Controller 15 Devices
00-0101	00	Local Broadcast Controller 5 Label	01-1010	01-FF	Bus 1 Broadcast Controller 10 Devices	11-0000	00	Local Only Controller 0 Label
00-0101	01-FF	Local Broadcast Controller 5 Devices	01-1011	00	Bus 1 Broadcast Controller 11 Label	11-0000	01-FF	Local Only Controller 0 Devices
00-0110	00	Local Broadcast Controller 6 Label	01-1011	01-FF	Bus 1 Broadcast Controller 11 Devices	11-0001	00	Local Only Controller 1 Label
00-0110	01-FF	Local Broadcast Controller 6 Devices	01-1100	00	Bus 1 Broadcast Controller 12 Label	11-0001	01-FF	Local Only Controller 1 Devices
00-0111	00	Local Broadcast Controller 7 Label	01-1100	01-FF	Bus 1 Broadcast Controller 12 Devices	11-0010	00	Local Only Controller 2 Label
00-0111	01-FF	Local Broadcast Controller 7 Devices	01-1101	00	Bus 1 Broadcast Controller 13 Label	11-0010	01-FF	Local Only Controller 2 Devices
00-1000	00	Local Broadcast Controller 8 Label	01-1101	01-FF	Bus 1 Broadcast Controller 13 Devices	11-0011	00	Local Only Controller 3 Label
00-1000	01-FF	Local Broadcast Controller 8 Devices	01-1110	00	Bus 1 Broadcast Controller 14 Label	11-0011	01-FF	Local Only Controller 3 Devices
00-1001	00	Local Broadcast Controller 9 Label	01-1110	01-FF	Bus 1 Broadcast Controller 14 Devices	11-0100	00	Local Only Controller 4 Label
00-1001	01-FF	Local Broadcast Controller 9 Devices	01-1111	00	Bus 1 Broadcast Controller 15 Label	11-0100	01-FF	Local Only Controller 4 Devices
00-1010	00	Local Broadcast Controller 10 Label	01-1111	01-FF	Bus 1 Broadcast Controller 15 Devices	11-0101	00	Local Only Controller 5 Label
00-1010	01-FF	Local Broadcast Controller 10 Devices	10-0000	00	Bus 2 Broadcast Orphans Label	11-0101	01-FF	Local Only Controller 5 Devices
00-1011	00	Local Broadcast Controller 11 Label	10-0000	01-FE	Bus 2 Broadcast Static Devices	11-0110	00	Local Only Controller 6 Label
00-1011	01-FF	Local Broadcast Controller 11 Devices	10-0000	FF	Bus 2 Broadcast Bridges Label	11-0110	01-FF	Local Only Controller 6 Devices
00-1100	00	Local Broadcast Controller 12 Label	10-0001	00	Bus 2 Broadcast Controller 1 Label	11-0111	00	Local Only Controller 7 Label
00-1100	01-FF	Local Broadcast Controller 12 Devices	10-0001	01-FF	Bus 2 Broadcast Controller 1 Devices	11-0111	01-FF	Local Only Controller 7 Devices
00-1101	00	Local Broadcast Controller 13 Label	10-0010	00	Bus 2 Broadcast Controller 2 Label	11-1000	00	Local Only Controller 8 Label
00-1101	01-FF	Local Broadcast Controller 13 Devices	10-0010	01-FF	Bus 2 Broadcast Controller 2 Devices	11-1000	01-FF	Local Only Controller 8 Devices
00-1110	00	Local Broadcast Controller 14 Label	10-0011	00	Bus 2 Broadcast Controller 3 Label	11-1001	00	Local Only Controller 9 Label
00-1110	01-FF	Local Broadcast Controller 14 Devices	10-0011	01-FF	Bus 2 Broadcast Controller 3 Devices	11-1001	01-FF	Local Only Controller 9 Devices
00-1111	00	Local Broadcast Controller 15 Label	10-0100	00	Bus 2 Broadcast Controller 4 Label	11-1010	00	Local Only Controller 10 Label
00-1111	01-FF	Local Broadcast Controller 15 Devices	10-0100	01-FF	Bus 2 Broadcast Controller 4 Devices	11-1010	01-FF	Local Only Controller 10 Devices
01-0000	00	Bus 1 Broadcast Orphans Label	10-0101	00	Bus 2 Broadcast Controller 5 Label	11-1011	00	Local Only Controller 11 Label
01-0000	01-FE	Bus 1 Broadcast Static Devices	10-0101	01-FF	Bus 2 Broadcast Controller 5 Devices	11-1011	01-FF	Local Only Controller 11 Devices
01-0000	FF	Bus 1 Broadcast Bridges Label	10-0110	00	Bus 2 Broadcast Controller 6 Label	11-1100	00	Local Only Controller 12 Label
01-0001	00	Bus 1 Broadcast Controller 1 Label	10-0110	01-FF	Bus 2 Broadcast Controller 6 Devices	11-1100	01-FF	Local Only Controller 12 Devices
01-0001	01-FF	Bus 1 Broadcast Controller 1 Devices	10-0111	00	Bus 2 Broadcast Controller 7 Label	11-1101	00	Local Only Controller 13 Label
01-0010	00	Bus 1 Broadcast Controller 2 Label	10-0111	01-FF	Bus 2 Broadcast Controller 7 Devices	11-1101	01-FF	Local Only Controller 13 Devices
01-0010	01-FF	Bus 1 Broadcast Controller 2 Devices	10-1000	00	Bus 2 Broadcast Controller 8 Label	11-1110	00	Local Only Controller 14 Label
01-0011	00	Bus 1 Broadcast Controller 3 Label	10-1000	01-FF	Bus 2 Broadcast Controller 8 Devices	11-1110	01-FF	Local Only Controller 14 Devices
01-0011	01-FF	Bus 1 Broadcast Controller 3 Devices	10-1001	00	Bus 2 Broadcast Controller 9 Label	11-1111	00	Local Only Orphans Label
01-0100	00	Bus 1 Broadcast Controller 4 Label	10-1001	01-FF	Bus 2 Broadcast Controller 9 Devices	11-1111	01-FF	Local Only Static Devices
01-0100	01-FF	Bus 1 Broadcast Controller 4 Devices	10-1010	00	Bus 2 Broadcast Controller 10 Label			

Mode 33

Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description
000-000	00	Local Broadcast Orphans Label	010-101	00	Bus 2 Broadcast Controller 5 Label	101-010	00	Bus 5 Broadcast Controller 2 Label
000-000	01-FE	Local Broadcast Static Devices	010-101	01-FF	Bus 2 Broadcast Controller 5 Devices	101-010	01-FF	Bus 5 Broadcast Controller 2 Devices
000-000	FF	Local Broadcast Bridges Label	010-110	00	Bus 2 Broadcast Controller 6 Label	101-011	00	Bus 5 Broadcast Controller 3 Label
000-001	00	Local Broadcast Controller 1 Label	010-110	01-FF	Bus 2 Broadcast Controller 6 Devices	101-011	01-FF	Bus 5 Broadcast Controller 3 Devices
000-001	01-FF	Local Broadcast Controller 1 Devices	010-111	00	Bus 2 Broadcast Controller 7 Label	101-100	00	Bus 5 Broadcast Controller 4 Label
000-010	00	Local Broadcast Controller 2 Label	010-111	01-FF	Bus 2 Broadcast Controller 7 Devices	101-100	01-FF	Bus 5 Broadcast Controller 4 Devices
000-010	01-FF	Local Broadcast Controller 2 Devices	011-000	00	Bus 3 Broadcast Orphans Label	101-101	00	Bus 5 Broadcast Controller 5 Label
000-011	00	Local Broadcast Controller 3 Label	011-000	01-FE	Bus 3 Broadcast Static Devices	101-101	01-FF	Bus 5 Broadcast Controller 5 Devices
000-011	01-FF	Local Broadcast Controller 3 Devices	011-000	FF	Bus 3 Broadcast Bridges Label	101-110	00	Bus 5 Broadcast Controller 6 Label
000-100	00	Local Broadcast Controller 4 Label	011-001	00	Bus 3 Broadcast Controller 1 Label	101-110	01-FF	Bus 5 Broadcast Controller 6 Devices
000-100	01-FF	Local Broadcast Controller 4 Devices	011-001	01-FF	Bus 3 Broadcast Controller 1 Devices	101-111	00	Bus 5 Broadcast Controller 7 Label
000-101	00	Local Broadcast Controller 5 Label	011-010	00	Bus 3 Broadcast Controller 2 Label	101-111	01-FF	Bus 5 Broadcast Controller 7 Devices
000-101	01-FF	Local Broadcast Controller 5 Devices	011-010	01-FF	Bus 3 Broadcast Controller 2 Devices	110-000	00	Bus 6 Broadcast Orphans Label
000-110	00	Local Broadcast Controller 6 Label	011-011	00	Bus 3 Broadcast Controller 3 Label	110-000	01-FE	Bus 6 Broadcast Static Devices
000-110	01-FF	Local Broadcast Controller 6 Devices	011-011	01-FF	Bus 3 Broadcast Controller 3 Devices	110-000	FF	Bus 6 Broadcast Bridges Label
000-111	00	Local Broadcast Controller 7 Label	011-100	00	Bus 3 Broadcast Controller 4 Label	110-001	00	Bus 6 Broadcast Controller 1 Label
000-111	01-FF	Local Broadcast Controller 7 Devices	011-100	01-FF	Bus 3 Broadcast Controller 4 Devices	110-001	01-FF	Bus 6 Broadcast Controller 1 Devices
001-000	00	Bus 1 Broadcast Orphans Label	011-101	00	Bus 3 Broadcast Controller 5 Label	110-010	00	Bus 6 Broadcast Controller 2 Label
001-000	01-FE	Bus 1 Broadcast Static Devices	011-101	01-FF	Bus 3 Broadcast Controller 5 Devices	110-010	01-FF	Bus 6 Broadcast Controller 2 Devices
001-000	FF	Bus 1 Broadcast Bridges Label	011-110	00	Bus 3 Broadcast Controller 6 Label	110-011	00	Bus 6 Broadcast Controller 3 Label
001-001	00	Bus 1 Broadcast Controller 1 Label	011-110	01-FF	Bus 3 Broadcast Controller 6 Devices	110-011	01-FF	Bus 6 Broadcast Controller 3 Devices
001-001	01-FF	Bus 1 Broadcast Controller 1 Devices	011-111	00	Bus 3 Broadcast Controller 7 Label	110-100	00	Bus 6 Broadcast Controller 4 Label
001-010	00	Bus 1 Broadcast Controller 2 Label	011-111	01-FF	Bus 3 Broadcast Controller 7 Devices	110-100	01-FF	Bus 6 Broadcast Controller 4 Devices
001-010	01-FF	Bus 1 Broadcast Controller 2 Devices	100-000	00	Bus 4 Broadcast Orphans Label	110-101	00	Bus 6 Broadcast Controller 5 Label
001-011	00	Bus 1 Broadcast Controller 3 Label	100-000	01-FE	Bus 4 Broadcast Static Devices	110-101	01-FF	Bus 6 Broadcast Controller 5 Devices
001-011	01-FF	Bus 1 Broadcast Controller 3 Devices	100-000	FF	Bus 4 Broadcast Bridges Label	110-110	00	Bus 6 Broadcast Controller 6 Label
001-100	00	Bus 1 Broadcast Controller 4 Label	100-001	00	Bus 4 Broadcast Controller 1 Label	110-110	01-FF	Bus 6 Broadcast Controller 6 Devices
001-100	01-FF	Bus 1 Broadcast Controller 4 Devices	100-001	01-FF	Bus 4 Broadcast Controller 1 Devices	110-111	00	Bus 6 Broadcast Controller 7 Label
001-101	00	Bus 1 Broadcast Controller 5 Label	100-010	00	Bus 4 Broadcast Controller 2 Label	110-111	01-FF	Bus 6 Broadcast Controller 7 Devices
001-101	01-FF	Bus 1 Broadcast Controller 5 Devices	100-010	01-FF	Bus 4 Broadcast Controller 2 Devices	111-000	00	Local Only Controller 0 Label
001-110	00	Bus 1 Broadcast Controller 6 Label	100-011	00	Bus 4 Broadcast Controller 3 Label	111-000	01-FF	Local Only Controller 0 Devices
001-110	01-FF	Bus 1 Broadcast Controller 6 Devices	100-011	01-FF	Bus 4 Broadcast Controller 3 Devices	111-001	00	Local Only Controller 1 Label
001-111	00	Bus 1 Broadcast Controller 7 Label	100-100	00	Bus 4 Broadcast Controller 4 Label	111-001	01-FF	Local Only Controller 1 Devices
001-111	01-FF	Bus 1 Broadcast Controller 7 Devices	100-100	01-FF	Bus 4 Broadcast Controller 4 Devices	111-010	00	Local Only Controller 2 Label
010-000	00	Bus 2 Broadcast Orphans Label	100-101	00	Bus 4 Broadcast Controller 5 Label	111-010	01-FF	Local Only Controller 2 Devices
010-000	01-FE	Bus 2 Broadcast Static Devices	100-101	01-FF	Bus 4 Broadcast Controller 5 Devices	111-011	00	Local Only Controller 3 Label
010-000	FF	Bus 2 Broadcast Bridges Label	100-110	00	Bus 4 Broadcast Controller 6 Label	111-011	01-FF	Local Only Controller 3 Devices
010-001	00	Bus 2 Broadcast Controller 1 Label	100-110	01-FF	Bus 4 Broadcast Controller 6 Devices	111-100	00	Local Only Controller 4 Label
010-001	01-FF	Bus 2 Broadcast Controller 1 Devices	100-111	00	Bus 4 Broadcast Controller 7 Label	111-100	01-FF	Local Only Controller 4 Devices
010-010	00	Bus 2 Broadcast Controller 2 Label	100-111	01-FF	Bus 4 Broadcast Controller 7 Devices	111-101	00	Local Only Controller 5 Label
010-010	01-FF	Bus 2 Broadcast Controller 2 Devices	101-000	00	Bus 5 Broadcast Orphans Label	111-101	01-FF	Local Only Controller 5 Devices
010-011	00	Bus 2 Broadcast Controller 3 Label	101-000	01-FE	Bus 5 Broadcast Static Devices	111-110	00	Local Only Controller 6 Label
010-011	01-FF	Bus 2 Broadcast Controller 3 Devices	101-000	FF	Bus 5 Broadcast Bridges Label	111-110	01-FF	Local Only Controller 6 Devices
010-100	00	Bus 2 Broadcast Controller 4 Label	101-001	00	Bus 5 Broadcast Controller 1 Label	111-111	00	Local Only Orphans Label
010-100	01-FF	Bus 2 Broadcast Controller 4 Devices	101-001	01-FF	Bus 5 Broadcast Controller 1 Devices	111-111	01-FF	Local Only Static Devices

Mode 42

Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description
0000-00	00	Local Broadcast Orphans Label	0101-01	00	Bus 5 Broadcast Controller 1 Label	1010-10	01-FF	Bus 10 Broadcast Controller 2 Devices
0000-00	01-FE	Local Broadcast Static Devices	0101-01	01-FF	Bus 5 Broadcast Controller 1 Devices	1010-11	00	Bus 10 Broadcast Controller 3 Label
0000-00	FF	Local Broadcast Bridges Label	0101-10	00	Bus 5 Broadcast Controller 2 Label	1010-11	01-FF	Bus 10 Broadcast Controller 3 Devices
0000-01	00	Local Broadcast Controller 1 Label	0101-10	01-FF	Bus 5 Broadcast Controller 2 Devices	1011-00	00	Bus 11 Broadcast Orphans Label
0000-01	01-FF	Local Broadcast Controller 1 Devices	0101-11	00	Bus 5 Broadcast Controller 3 Label	1011-00	01-FE	Bus 11 Broadcast Static Devices
0000-10	00	Local Broadcast Controller 2 Label	0101-11	01-FF	Bus 5 Broadcast Controller 3 Devices	1011-00	FF	Bus 11 Broadcast Bridges Label
0000-10	01-FF	Local Broadcast Controller 2 Devices	0110-00	00	Bus 6 Broadcast Orphans Label	1011-01	00	Bus 11 Broadcast Controller 1 Label
0000-11	00	Local Broadcast Controller 3 Label	0110-00	01-FE	Bus 6 Broadcast Static Devices	1011-01	01-FF	Bus 11 Broadcast Controller 1 Devices
0000-11	01-FF	Local Broadcast Controller 3 Devices	0110-00	FF	Bus 6 Broadcast Bridges Label	1011-10	00	Bus 11 Broadcast Controller 2 Label
0001-00	00	Bus 1 Broadcast Orphans Label	0110-01	00	Bus 6 Broadcast Controller 1 Label	1011-10	01-FF	Bus 11 Broadcast Controller 2 Devices
0001-00	01-FE	Bus 1 Broadcast Static Devices	0110-01	01-FF	Bus 6 Broadcast Controller 1 Devices	1011-11	00	Bus 11 Broadcast Controller 3 Label
0001-00	FF	Bus 1 Broadcast Bridges Label	0110-10	00	Bus 6 Broadcast Controller 2 Label	1011-11	01-FF	Bus 11 Broadcast Controller 3 Devices
0001-01	00	Bus 1 Broadcast Controller 1 Label	0110-10	01-FF	Bus 6 Broadcast Controller 2 Devices	1100-00	00	Bus 12 Broadcast Orphans Label
0001-01	01-FF	Bus 1 Broadcast Controller 1 Devices	0110-11	00	Bus 6 Broadcast Controller 3 Label	1100-00	01-FE	Bus 12 Broadcast Static Devices
0001-10	00	Bus 1 Broadcast Controller 2 Label	0110-11	01-FF	Bus 6 Broadcast Controller 3 Devices	1100-00	FF	Bus 12 Broadcast Bridges Label
0001-10	01-FF	Bus 1 Broadcast Controller 2 Devices	0111-00	00	Bus 7 Broadcast Orphans Label	1100-01	00	Bus 12 Broadcast Controller 1 Label
0001-11	00	Bus 1 Broadcast Controller 3 Label	0111-00	01-FE	Bus 7 Broadcast Static Devices	1100-01	01-FF	Bus 12 Broadcast Controller 1 Devices
0001-11	01-FF	Bus 1 Broadcast Controller 3 Devices	0111-00	FF	Bus 7 Broadcast Bridges Label	1100-10	00	Bus 12 Broadcast Controller 2 Label
0010-00	00	Bus 2 Broadcast Orphans Label	0111-01	00	Bus 7 Broadcast Controller 1 Label	1100-10	01-FF	Bus 12 Broadcast Controller 2 Devices
0010-00	01-FE	Bus 2 Broadcast Static Devices	0111-01	01-FF	Bus 7 Broadcast Controller 1 Devices	1100-11	00	Bus 12 Broadcast Controller 3 Label
0010-00	FF	Bus 2 Broadcast Bridges Label	0111-10	00	Bus 7 Broadcast Controller 2 Label	1100-11	01-FF	Bus 12 Broadcast Controller 3 Devices
0010-01	00	Bus 2 Broadcast Controller 1 Label	0111-10	01-FF	Bus 7 Broadcast Controller 2 Devices	1101-00	00	Bus 13 Broadcast Orphans Label
0010-01	01-FF	Bus 2 Broadcast Controller 1 Devices	0111-11	00	Bus 7 Broadcast Controller 3 Label	1101-00	01-FE	Bus 13 Broadcast Static Devices
0010-10	00	Bus 2 Broadcast Controller 2 Label	0111-11	01-FF	Bus 7 Broadcast Controller 3 Devices	1101-00	FF	Bus 13 Broadcast Bridges Label
0010-10	01-FF	Bus 2 Broadcast Controller 2 Devices	1000-00	00	Bus 8 Broadcast Orphans Label	1101-01	00	Bus 13 Broadcast Controller 1 Label
0010-11	00	Bus 2 Broadcast Controller 3 Label	1000-00	01-FE	Bus 8 Broadcast Static Devices	1101-01	01-FF	Bus 13 Broadcast Controller 1 Devices
0010-11	01-FF	Bus 2 Broadcast Controller 3 Devices	1000-00	FF	Bus 8 Broadcast Bridges Label	1101-10	00	Bus 13 Broadcast Controller 2 Label
0011-00	00	Bus 3 Broadcast Orphans Label	1000-01	00	Bus 8 Broadcast Controller 1 Label	1101-10	01-FF	Bus 13 Broadcast Controller 2 Devices
0011-00	01-FE	Bus 3 Broadcast Static Devices	1000-01	01-FF	Bus 8 Broadcast Controller 1 Devices	1101-11	00	Bus 13 Broadcast Controller 3 Label
0011-00	FF	Bus 3 Broadcast Bridges Label	1000-10	00	Bus 8 Broadcast Controller 2 Label	1101-11	01-FF	Bus 13 Broadcast Controller 3 Devices
0011-01	00	Bus 3 Broadcast Controller 1 Label	1000-10	01-FF	Bus 8 Broadcast Controller 2 Devices	1110-00	00	Bus 14 Broadcast Orphans Label
0011-01	01-FF	Bus 3 Broadcast Controller 1 Devices	1000-11	00	Bus 8 Broadcast Controller 3 Label	1110-00	01-FE	Bus 14 Broadcast Static Devices
0011-10	00	Bus 3 Broadcast Controller 2 Label	1000-11	01-FF	Bus 8 Broadcast Controller 3 Devices	1110-00	FF	Bus 14 Broadcast Bridges Label
0011-10	01-FF	Bus 3 Broadcast Controller 2 Devices	1001-00	00	Bus 9 Broadcast Orphans Label	1110-01	00	Bus 14 Broadcast Controller 1 Label
0011-11	00	Bus 3 Broadcast Controller 3 Label	1001-00	01-FE	Bus 9 Broadcast Static Devices	1110-01	01-FF	Bus 14 Broadcast Controller 1 Devices
0011-11	01-FF	Bus 3 Broadcast Controller 3 Devices	1001-00	FF	Bus 9 Broadcast Bridges Label	1110-10	00	Bus 14 Broadcast Controller 2 Label
0100-00	00	Bus 4 Broadcast Orphans Label	1001-01	00	Bus 9 Broadcast Controller 1 Label	1110-10	01-FF	Bus 14 Broadcast Controller 2 Devices
0100-00	01-FE	Bus 4 Broadcast Static Devices	1001-01	01-FF	Bus 9 Broadcast Controller 1 Devices	1110-11	00	Bus 14 Broadcast Controller 3 Label
0100-00	FF	Bus 4 Broadcast Bridges Label	1001-10	00	Bus 9 Broadcast Controller 2 Label	1110-11	01-FF	Bus 14 Broadcast Controller 3 Devices
0100-01	00	Bus 4 Broadcast Controller 1 Label	1001-10	01-FF	Bus 9 Broadcast Controller 2 Devices	1111-00	00	Local Only Controller 0 Label
0100-01	01-FF	Bus 4 Broadcast Controller 1 Devices	1001-11	00	Bus 9 Broadcast Controller 3 Label	1111-00	01-FF	Local Only Controller 0 Devices
0100-10	00	Bus 4 Broadcast Controller 2 Label	1001-11	01-FF	Bus 9 Broadcast Controller 3 Devices	1111-01	00	Local Only Controller 1 Label
0100-10	01-FF	Bus 4 Broadcast Controller 2 Devices	1010-00	00	Bus 10 Broadcast Orphans Label	1111-01	01-FF	Local Only Controller 1 Devices
0100-11	00	Bus 4 Broadcast Controller 3 Label	1010-00	01-FE	Bus 10 Broadcast Static Devices	1111-10	00	Local Only Controller 2 Label
0100-11	01-FF	Bus 4 Broadcast Controller 3 Devices	1010-00	FF	Bus 10 Broadcast Bridges Label	1111-10	01-FF	Local Only Controller 2 Devices
0101-00	00	Bus 5 Broadcast Orphans Label	1010-01	00	Bus 10 Broadcast Controller 1 Label	1111-11	00	Local Only Orphans Label
0101-00	01-FE	Bus 5 Broadcast Static Devices	1010-01	01-FF	Bus 10 Broadcast Controller 1 Devices	1111-11	01-FF	Local Only Static Devices
0101-00	FF	Bus 5 Broadcast Bridges Label	1010-10	00	Bus 10 Broadcast Controller 2 Label			

Mode 60

Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description	Bus_ID Cont_ID (bin)	Label (hex)	Description
000000	00-FE	Local Broadcast Static Devices	010101	FF	Bus 21 Broadcast Bridges Label	101011	00-FE	Bus 43 Broadcast Static Devices
000000	FF	Local Broadcast Bridges Label	010110	00-FE	Bus 22 Broadcast Static Devices	101011	FF	Bus 43 Broadcast Bridges Label
000001	00-FE	Bus 1 Broadcast Static Devices	010110	FF	Bus 22 Broadcast Bridges Label	101100	00-FE	Bus 44 Broadcast Static Devices
000001	FF	Bus 1 Broadcast Bridges Label	010111	00-FE	Bus 23 Broadcast Static Devices	101100	FF	Bus 44 Broadcast Bridges Label
000010	00-FE	Bus 2 Broadcast Static Devices	010111	FF	Bus 23 Broadcast Bridges Label	101101	00-FE	Bus 45 Broadcast Static Devices
000010	FF	Bus 2 Broadcast Bridges Label	011000	00-FE	Bus 24 Broadcast Static Devices	101101	FF	Bus 45 Broadcast Bridges Label
000011	00-FE	Bus 3 Broadcast Static Devices	011000	FF	Bus 24 Broadcast Bridges Label	101110	00-FE	Bus 46 Broadcast Static Devices
000011	FF	Bus 3 Broadcast Bridges Label	011001	00-FE	Bus 25 Broadcast Static Devices	101110	FF	Bus 46 Broadcast Bridges Label
000100	00-FE	Bus 4 Broadcast Static Devices	011001	FF	Bus 25 Broadcast Bridges Label	101111	00-FE	Bus 47 Broadcast Static Devices
000100	FF	Bus 4 Broadcast Bridges Label	011010	00-FE	Bus 26 Broadcast Static Devices	101111	FF	Bus 47 Broadcast Bridges Label
000101	00-FE	Bus 5 Broadcast Static Devices	011010	FF	Bus 26 Broadcast Bridges Label	110000	00-FE	Bus 48 Broadcast Static Devices
000101	FF	Bus 5 Broadcast Bridges Label	011011	00-FE	Bus 27 Broadcast Static Devices	110000	FF	Bus 48 Broadcast Bridges Label
000110	00-FE	Bus 6 Broadcast Static Devices	011011	FF	Bus 27 Broadcast Bridges Label	110001	00-FE	Bus 49 Broadcast Static Devices
000110	FF	Bus 6 Broadcast Bridges Label	011100	00-FE	Bus 28 Broadcast Static Devices	110001	FF	Bus 49 Broadcast Bridges Label
000111	00-FE	Bus 7 Broadcast Static Devices	011100	FF	Bus 28 Broadcast Bridges Label	110010	00-FE	Bus 50 Broadcast Static Devices
000111	FF	Bus 7 Broadcast Bridges Label	011101	00-FE	Bus 29 Broadcast Static Devices	110010	FF	Bus 50 Broadcast Bridges Label
001000	00-FE	Bus 8 Broadcast Static Devices	011101	FF	Bus 29 Broadcast Bridges Label	110011	00-FE	Bus 51 Broadcast Static Devices
001000	FF	Bus 8 Broadcast Bridges Label	011110	00-FE	Bus 30 Broadcast Static Devices	110011	FF	Bus 51 Broadcast Bridges Label
001001	00-FE	Bus 9 Broadcast Static Devices	011110	FF	Bus 30 Broadcast Bridges Label	110100	00-FE	Bus 52 Broadcast Static Devices
001001	FF	Bus 9 Broadcast Bridges Label	011111	00-FE	Bus 31 Broadcast Static Devices	110100	FF	Bus 52 Broadcast Bridges Label
001010	00-FE	Bus 10 Broadcast Static Devices	011111	FF	Bus 31 Broadcast Bridges Label	110101	00-FE	Bus 53 Broadcast Static Devices
001010	FF	Bus 10 Broadcast Bridges Label	100000	00-FE	Bus 32 Broadcast Static Devices	110101	FF	Bus 53 Broadcast Bridges Label
001011	00-FE	Bus 11 Broadcast Static Devices	100000	FF	Bus 32 Broadcast Bridges Label	110110	00-FE	Bus 54 Broadcast Static Devices
001011	FF	Bus 11 Broadcast Bridges Label	100001	00-FE	Bus 33 Broadcast Static Devices	110110	FF	Bus 54 Broadcast Bridges Label
001100	00-FE	Bus 12 Broadcast Static Devices	100001	FF	Bus 33 Broadcast Bridges Label	110111	00-FE	Bus 55 Broadcast Static Devices
001100	FF	Bus 12 Broadcast Bridges Label	100010	00-FE	Bus 34 Broadcast Static Devices	110111	FF	Bus 55 Broadcast Bridges Label
001101	00-FE	Bus 13 Broadcast Static Devices	100010	FF	Bus 34 Broadcast Bridges Label	111000	00-FE	Bus 56 Broadcast Static Devices
001101	FF	Bus 13 Broadcast Bridges Label	100011	00-FE	Bus 35 Broadcast Static Devices	111000	FF	Bus 56 Broadcast Bridges Label
001110	00-FE	Bus 14 Broadcast Static Devices	100011	FF	Bus 35 Broadcast Bridges Label	111001	00-FE	Bus 57 Broadcast Static Devices
001110	FF	Bus 14 Broadcast Bridges Label	100100	00-FE	Bus 36 Broadcast Static Devices	111001	FF	Bus 57 Broadcast Bridges Label
001111	00-FE	Bus 15 Broadcast Static Devices	100100	FF	Bus 36 Broadcast Bridges Label	111010	00-FE	Bus 58 Broadcast Static Devices
001111	FF	Bus 15 Broadcast Bridges Label	100101	00-FE	Bus 37 Broadcast Static Devices	111010	FF	Bus 58 Broadcast Bridges Label
010000	00-FE	Bus 16 Broadcast Static Devices	100101	FF	Bus 37 Broadcast Bridges Label	111011	00-FE	Bus 59 Broadcast Static Devices
010000	FF	Bus 16 Broadcast Bridges Label	100110	00-FE	Bus 38 Broadcast Static Devices	111011	FF	Bus 59 Broadcast Bridges Label
010001	00-FE	Bus 17 Broadcast Static Devices	100110	FF	Bus 38 Broadcast Bridges Label	111100	00-FE	Bus 60 Broadcast Static Devices
010001	FF	Bus 17 Broadcast Bridges Label	100111	00-FE	Bus 39 Broadcast Static Devices	111100	FF	Bus 60 Broadcast Bridges Label
010010	00-FE	Bus 18 Broadcast Static Devices	100111	FF	Bus 39 Broadcast Bridges Label	111101	00-FE	Bus 61 Broadcast Static Devices
010010	FF	Bus 18 Broadcast Bridges Label	101000	00-FE	Bus 40 Broadcast Static Devices	111101	FF	Bus 61 Broadcast Bridges Label
010011	00-FE	Bus 19 Broadcast Static Devices	101000	FF	Bus 40 Broadcast Bridges Label	111110	00-FE	Bus 62 Broadcast Static Devices
010011	FF	Bus 19 Broadcast Bridges Label	101001	00-FE	Bus 41 Broadcast Static Devices	111110	FF	Bus 62 Broadcast Bridges Label
010100	00-FE	Bus 20 Broadcast Static Devices	101001	FF	Bus 41 Broadcast Bridges Label	111111	00-FE	Local Only Static Devices
010100	FF	Bus 20 Broadcast Bridges Label	101010	00-FE	Bus 42 Broadcast Static Devices	111111	FF	Local Only Bridges Label
010101	00-FE	Bus 21 Broadcast Static Devices	101010	FF	Bus 42 Broadcast Bridges Label			

Annex B
(normative)

VP-Controller Profile Specification

VP-Controller Profile VersaPHY Register Set

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	0	E	I	BusID-ControllerID						00_{16} (Controller Label)							
n	2	W	B	reserved								Power					
n	4	V	S	r	r	O	ID_type		G	Instance_nOffset							
n	6	00_{16} (VP-Controller Profile)								Controller_Rev							
n	8	GUID [0:7]								GUID [8:15]							
n	A	GUID [16:23]								GUID [24:31]							
n	C	GUID [31:39]								GUID [40:47]							
n	E	GUID [48:55]								GUID [56:63]							

Figure 25 - Controller Profile Physical_ID Register Set

Field	Value	Description
E	-	Standard Enabled bit functionality, except managed by local VP-Controller.
I	-	Immediate response capability not required. OHCI based VP-Controllers may not have immediate response ability.
BusID	-	Bus ID (usually 0s for local broadcast bus or 1s for local only bus bus)
ControllerID	-	VP-Controller Address Space. Shall default to 0s (static broadcast) until a specific space is annexed.
Label	00_{16}	Reserved Label for VP-Controllers. Orphan label before controller space is annexed.
W	1	Watchdog is not enabled for VP-Controllers
B	0	Bus reset notification is not enabled for VP-Controllers
Power	-	Standard Power field functionality allows VP-Controller to be powered down by a power controller if enabled.
V	-	Standard Valid bit functionality
S	1	VP-Controllers enable themselves.
O	0	VP-Controllers do not support the owner field, they are the owner.

Field	Value	Description
ID_type	01_2	VP-Controllers are a fundamental VersaPHY device. The VP-Controller Profile is standardized by the 1394TA
G	1	VP-Controllers shall support a GUID
Instance_nOffset	-	Standard Instance_nOffset field functionality
Config_Ver	00_{16}	VP-Controller profile is Config_Ver = 00_{16}
Controller_Rev	00_{16}	This is revision 0 of the VP-Controller profile.
GUID	-	Standard GUID field. VP-Controllers that share a node with other 1394 functions shall report the same GUID for the VersaPHY Controller and the other functions listed in 1394 config ROM.

Table 20 - VP-Controller Profile Physical_ID Register Definitions

Block	Offset	Bit															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	E	I	00_{16}								Physical_ID					
0	2	W	B	reserved								Power					
0	4	V	1	r	r	0	01		1	Instance_nOffset							
0	6	00_{16} (VP-Controller Profile)								00_{16} (Controller_Profile_Revision)							
0	8	GUID [0:7]								GUID [8:15]							
0	A	GUID [16:23]								GUID [24:31]							
0	C	GUID [31:39]								GUID [40:47]							
0	E	GUID [48:55]								GUID [56:63]							
1	0	Reserved								Reserved					Bus_Bits		
1	2	Reserved								R	M	Res			N_Bus_Bits		
1	4	Reserved								R	C	N_VP-Controller					
1	6	Reserved							DR	Disown Label							
1	8	R	VP-Controller 1-7							VP-Controller 8-15							
1	A	VP-Controller 16-23								VP-Controller 24-31							
1	C	VP-Controller 32-39								VP-Controller 40-47							
1	E	VP-Controller 48-55								VP-Controller 56-62							R

Figure 26 - VP-Controller Profile VersaPHY VP-Label Register Set

VP-Controller Profile VersaPHY Register Definitions

Field	Value	Description
Owner	000000_2	VP-Controllers do not support the owner field. Set to 0s.
Physical_ID	-	Standard Physical_ID field functionality
ID_Type	01_2	Config_Ver and G are offered in VP-Label space because VP-Controllers respond at a reserved label address.
G	1	
Bus_Bits	-	Number of VP-Label bits reserved for bus Address
M	-	Mode change requested – Set by another VP-Controller to request a new Label Space Mode. VP-Controller shall not set bit if it is unable to change mode. VP-Controller shall set bit while attempting the change, and clear when change is complete. Bus_Bits must also be updated after change.
N_Bus_Bits	-	New bus bits – Value of Bus_Bits for change requested with M bit.
C	-	VP-Controller move request – Set by another VP-Controller to request a move to a new VP-Controller label space. Local VP-Controller shall not set bit if it is unable to move. Bit shall remain set until move is complete. At that point, the VP-Controller shall stop responding a the old VP-Controller address.
N_VP-Controller	-	New VP-Controller address – New VP-Controller address requested by C bit.
DR	-	Disown Request – Set by a remote VP-Controller requesting this controller to disown one of its adopted VP-Devices. When set to 0 the action is requested and the request may be denied. When set to 1 the action is required and shall be performed.
Disown_Label	-	Disown Request Label – The label of the VP-Device for the disown request. When the disown process is complete, accepted or rejected, the local VP-Controller shall reset this field to 00_{16}.
VP-Controller 1-62	-	Reserved Controllers – VP-Controller shall set to 1 the bits representing all VP-Controller spaces reserved by this controller.
R,Res,Reserved	0	Reserved – Set to 0.

Table 21 - VP-Controller Profile Unique VP-Label Register Definitions

VP-Controller Profile Read Request VersaPHY Packet with VP-Label Addressing

0	1	Destination_VP_Label	tl	0	0	0b1110	LReg_Off
Reserved	B	Reserved	Bytes	CRC			

Figure 27 - VP-Controller Profile Read Request VersaPHY packet using VP-Label

VP-Controller Profile Read Response VersaPHY Packet with VP-Label addressing

0	1	Source_VP_Label	tl	0	1	0b1110	LReg_Off
Reserved	B	Data [0:15]	CRC				

Figure 28 - VP-Controller Profile Read Response VersaPHY packet using VP-Label

VP-Controller Profile Write Request VersaPHY Packet with VP-Label addressing

0	1	Destination_VP_Label	tl	1	0	0b1110	LReg_Off
Reserved		B	Data [0:15]			CRC	

Figure 29 - VP-Controller Profile Write Request VersaPHY Packet using VP-Label

VP-Controller Profile Write Response VersaPHY Packet with VP-Label addressing

0	1	Source_VP_Label	tl	1	1	0b1110	LReg_Off
Reserved		B	Data [0:15]			CRC	

Figure 30 - VP-Controller Profile Write Response VersaPHY Packet using VP-Labels

VP-Controller Profile VersaPHY Packet Fields using VP-labels

Field	Comment
Dest_VP_Label	VersaPHY Label of the packets destination.
Source_VP_Label	VersaPHY Label of the device sending the packet.
tl	Transaction label used to associate a request with a response. The requester must ensure no outstanding tls have the same value. Some profiles may require a response be received before the next request is issued. In this case tl may be a constant value. The tl value of 111111_2 is reserved for use by unsolicited responses.
LReg_Off	Register Offset Low – is the lowest four bits of the (first) VersaPHY register being accessed (request) or having been accessed (response). This provides a minimum of 16 bytes of registers. For simple devices this maybe all that is required. For more complex devices this is enough to discover what profile is supported for the VP-label.
B	Register block address – 0 or 1
Data	Current register data value (response) or to be written (write request)
Bytes	Length of read request in bytes.
CRC	The CRC covers the first 56-bits of the packet. It is an 8 bit CRC: poly = x^8 + x^2 + x^1 + x^0, initial value = FF_{16}, final result inverted. See annex for example code.

Table 22 – VP-Controller Profile VersaPHY Packet using VP-labels Field Definition

Annex C
(informative)

CRC 8 Example

There are multiple ways to calculate a CRC. Below is prototype C code using a 256 byte lookup table. It produces CRC values compatible with the prototype VP-Device. Different methods that produce compatible results are acceptable.

```
//Initialize CRC lookup table
crct[0]=0x00; crct[1]=0x07; crct[2]=0x0E; crct[3]=0x09; crct[4]=0x1C; crct[5]=0x1B;
crct[6]=0x12; crct[7]=0x15; crct[8]=0x38; crct[9]=0x3F; crct[10]=0x36; crct[11]=0x31;
crct[12]=0x24; crct[13]=0x23; crct[14]=0x2A; crct[15]=0x2D; crct[16]=0x70; crct[17]=0x77;
crct[18]=0x7E; crct[19]=0x79; crct[20]=0x6C; crct[21]=0x6B; crct[22]=0x62; crct[23]=0x65;
crct[24]=0x48; crct[25]=0x4F; crct[26]=0x46; crct[27]=0x41; crct[28]=0x54; crct[29]=0x53;
crct[30]=0x5A; crct[31]=0x5D; crct[32]=0xE0; crct[33]=0xE7; crct[34]=0xEE; crct[35]=0xE9;
crct[36]=0xFC; crct[37]=0xFB; crct[38]=0xF2; crct[39]=0xF5; crct[40]=0xD8; crct[41]=0xDF;
crct[42]=0xD6; crct[43]=0xD1; crct[44]=0xC4; crct[45]=0xC3; crct[46]=0xCA; crct[47]=0xCD;
crct[48]=0x90; crct[49]=0x97; crct[50]=0x9E; crct[51]=0x99; crct[52]=0x8C; crct[53]=0x8B;
crct[54]=0x82; crct[55]=0x85; crct[56]=0xA8; crct[57]=0xAF; crct[58]=0xA6; crct[59]=0xA1;
crct[60]=0xB4; crct[61]=0xB3; crct[62]=0xBA; crct[63]=0xBD; crct[64]=0xC7; crct[65]=0xC0;
crct[66]=0xC9; crct[67]=0xCE; crct[68]=0xDB; crct[69]=0xDC; crct[70]=0xD5; crct[71]=0xD2;
crct[72]=0xFF; crct[73]=0xF8; crct[74]=0xF1; crct[75]=0xF6; crct[76]=0xE3; crct[77]=0xE4;
crct[78]=0xED; crct[79]=0xEA; crct[80]=0xB7; crct[81]=0xB0; crct[82]=0xB9; crct[83]=0xBE;
crct[84]=0xAB; crct[85]=0xAC; crct[86]=0xA5; crct[87]=0xA2; crct[88]=0x8F; crct[89]=0x88;
crct[90]=0x81; crct[91]=0x86; crct[92]=0x93; crct[93]=0x94; crct[94]=0x9D; crct[95]=0x9A;
crct[96]=0x27; crct[97]=0x20; crct[98]=0x29; crct[99]=0x2E; crct[100]=0x3B;
crct[101]=0x3C; crct[102]=0x35; crct[103]=0x32; crct[104]=0x1F; crct[105]=0x18;
crct[106]=0x11; crct[107]=0x16; crct[108]=0x03; crct[109]=0x04; crct[110]=0x0D;
crct[111]=0x0A; crct[112]=0x57; crct[113]=0x50; crct[114]=0x59; crct[115]=0x5E;
crct[116]=0x4B; crct[117]=0x4C; crct[118]=0x45; crct[119]=0x42; crct[120]=0x6F;
crct[121]=0x68; crct[122]=0x61; crct[123]=0x66; crct[124]=0x73; crct[125]=0x74;
crct[126]=0x7D; crct[127]=0x7A; crct[128]=0x89; crct[129]=0x8E; crct[130]=0x87;
crct[131]=0x80; crct[132]=0x95; crct[133]=0x92; crct[134]=0x9B; crct[135]=0x9C;
crct[136]=0xB1; crct[137]=0xB6; crct[138]=0xBF; crct[139]=0xB8; crct[140]=0xAD;
crct[141]=0xAA; crct[142]=0xA3; crct[143]=0xA4; crct[144]=0xF9; crct[145]=0xFE;
crct[146]=0xF7; crct[147]=0xF0; crct[148]=0xE5; crct[149]=0xE2; crct[150]=0xEB;
crct[151]=0xEC; crct[152]=0xC1; crct[153]=0xC6; crct[154]=0xCF; crct[155]=0xC8;
crct[156]=0xDD; crct[157]=0xDA; crct[158]=0xD3; crct[159]=0xD4; crct[160]=0x69;
crct[161]=0x6E; crct[162]=0x67; crct[163]=0x60; crct[164]=0x75; crct[165]=0x72;
crct[166]=0x7B; crct[167]=0x7C; crct[168]=0x51; crct[169]=0x56; crct[170]=0x5F;
crct[171]=0x58; crct[172]=0x4D; crct[173]=0x4A; crct[174]=0x43; crct[175]=0x44;
crct[176]=0x19; crct[177]=0x1E; crct[178]=0x17; crct[179]=0x10; crct[180]=0x05;
crct[181]=0x02; crct[182]=0x0B; crct[183]=0x0C; crct[184]=0x21; crct[185]=0x26;
crct[186]=0x2F; crct[187]=0x28; crct[188]=0x3D; crct[189]=0x3A; crct[190]=0x33;
crct[191]=0x34; crct[192]=0x4E; crct[193]=0x49; crct[194]=0x40; crct[195]=0x47;
crct[196]=0x52; crct[197]=0x55; crct[198]=0x5C; crct[199]=0x5B; crct[200]=0x76;
crct[201]=0x71; crct[202]=0x78; crct[203]=0x7F; crct[204]=0x6A; crct[205]=0x6D;
crct[206]=0x64; crct[207]=0x63; crct[208]=0x3E; crct[209]=0x39; crct[210]=0x30;
crct[211]=0x37; crct[212]=0x22; crct[213]=0x25; crct[214]=0x2C; crct[215]=0x2B;
crct[216]=0x06; crct[217]=0x01; crct[218]=0x08; crct[219]=0x0F; crct[220]=0x1A;
crct[221]=0x1D; crct[222]=0x14; crct[223]=0x13; crct[224]=0xAE; crct[225]=0xA9;
crct[226]=0xA0; crct[227]=0xA7; crct[228]=0xB2; crct[229]=0xB5; crct[230]=0xBC;
crct[231]=0xBB; crct[232]=0x96; crct[233]=0x91; crct[234]=0x98; crct[235]=0x9F;
crct[236]=0x8A; crct[237]=0x8D; crct[238]=0x84; crct[239]=0x83; crct[240]=0xDE;
crct[241]=0xD9; crct[242]=0xD0; crct[243]=0xD7; crct[244]=0xC2; crct[245]=0xC5;
crct[246]=0xCC; crct[247]=0xCB; crct[248]=0xE6; crct[249]=0xE1; crct[250]=0xE8;
crct[251]=0xEF; crct[252]=0xFA; crct[253]=0xFD; crct[254]=0xF4; crct[255]=0xF3;
crc = 0xFF; // Initialized to FF
```

```
crc = crct[crc ^ byte0]; // Standard Lookup process, crc xor’d with byte and looked up
crc = crct[crc ^ byte1];
crc = crct[crc ^ byte2];
crc = crct[crc ^ byte3];
crc = crct[crc ^ byte4];
crc = crct[crc ^ byte5];
crc = crct[crc ^ byte6]; // Don't lookup last byte, it's the CRC
crc = crc ^ 0xFF; // Inverted
byte7 = crc;
```

Annex D
(normative)

Conformance requirements

This annex is intended to assist VersaPHY device designers and implementers as well as developers of conformance tests; it provides a concise summary of mandatory and optional features and, for each feature, reference to the governing normative clauses.

If optional features are implemented the associated sub-features may or may not Mandatory.

Feature	Sub-feature	Implementation	Reference
VP-Labels		Optional	6
	VP-Label field	Mandatory	6.2
	Implementing at least one of the following three is mandatory		
	Static Label Space	Optional	6.3
	VP-Controller Label Space	Optional	6.4
	Bus Label Space	Optional	6.5
	Label Space Modes	Mandatory	6.6
VersaPHY Packet Formats		Mandatory	7
	Implementing at least one of the two packet format groups is mandatory: Physical-ID or VP-Label addressed		
	Read Request VersaPHY Packet with Physical-ID Addressing	Optional	7.1
	Read Response VersaPHY Packet with Physical-ID Addressing	Optional	7.1
	Write Request VersaPHY Packet with Physical-ID Addressing	Optional	7.1
	Write Response VersaPHY Packet with Physical-ID Addressing	Optional	7.1
	Read Request VersaPHY	Optional	7.2

	Packet with VersaPHY Label Addressing		
	Read Response VersaPHY Packet with VersaPHY Label Addressing	Optional	7.2
	Write Request VersaPHY Packet with VersaPHY Label Addressing	Optional	7.2
	Write Response VersaPHY Packet with VersaPHY Label Addressing	Optional	7.2
VersaPHY Registers		Mandatory	8
	VersaPHY Register Addressing	Mandatory	8.1
	Implementing at least one of the two is mandatory: Physical-ID and VP-Label addressed registers		
	Physical_ID Addressed Registers	Optional	8.2
	VP-Label Addressed Registers	Optional	8.3
Power Control		Optional	9
	Power Control Field	Mandatory	9.1
	Requests	Mandatory	9.2
	Responses	Mandatory	9.3

Annex E
(informative)

Bibliography

[B1] IEEE Std 1212-2001, Standard for a Control and Status Registers (CSR) Architecture for microcomputer buses

[B2] IEEE Std 1394-1995, Standard for a High Performance Serial Bus

[B3] IEEE Std 1394a-2000, Standard for a High Performance Serial Bus—Amendment 1

[B4] IEEE Std 1394b-2002, Standard for a High Performance Serial Bus—Amendment 2

[B5] ISO/IEC 9899:1990, Programming Languages—C