

IEEE Recommended Practice for Power Quality Data Interchange Format (PQDIF)

IEEE Power and Energy Society

Transmission and Distribution Committee

IEEE Std 1159.3™-2025
(Revision of IEEE Std 1159.3-2019)

STANDARDS

IEEE Recommended Practice for Power Quality Data Interchange Format (PQDIF)

Developed by the

**Transmission and Distribution Committee
of the
IEEE Power and Energy Society**

Approved 12 February 2025

IEEE SA Standards Board

Abstract: A file format suitable for exchanging power quality-related measurement and simulation data in a vendor-independent manner is defined in this recommended practice. The format is designed to represent all power quality phenomena identified in IEEE Std 1159™-2009 and other power-related measurement data and is also extensible to other data types. The recommended file format utilizes a highly compressed storage scheme to help reduce disk space and transmission times. Utilizing Globally Unique Identifiers (GUID) to represent each element in the file permits the format to be extensible without the need for a central registration authority.

Keywords: data interchange, file format, IEEE 1159.3™, measurement, monitoring, power quality, PQDI

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2025 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 22 July 2025. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 979-8-8557-2272-7 STD27992
Print: ISBN 979-8-8557-2273-4 STDPD27992

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all IEEE standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers involved in technical working groups are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning all standards, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. IEEE Standards documents do not guarantee safety, security, health, or environmental protection, or compliance with law, or guarantee against interference with or from other devices or networks. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE Standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document should rely upon their own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus balloting process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English language version published by IEEE is the approved IEEE standard.

Use by artificial intelligence systems

In no event shall material in any IEEE Standards documents be used for the purpose of creating, training, enhancing, developing, maintaining, or contributing to any artificial intelligence systems without the express, written consent of IEEE SA in advance. “Artificial intelligence” refers to any software, application, or other system that uses artificial intelligence, machine learning, or similar technologies, to analyze, train, process, or generate content. Requests for consent can be submitted using the Contact Us form.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual is not, and shall not be considered or inferred to be, the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE or IEEE SA. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter’s views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s). News releases about IEEE standards issued by entities other than IEEE SA should be considered the view of the entity issuing the release rather than the formal position of IEEE or IEEE SA.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of Governors are not able to provide an instant response to comments or questions, except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or revisions to an IEEE standard is welcome to join the relevant IEEE SA working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).¹ An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.²

¹ Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

² Available at: <https://standards.ieee.org/about/contact/>.

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#).³ For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).⁴ Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

Patents

IEEE standards are developed in compliance with the [IEEE SA Patent Policy](#).⁵

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

IMPORTANT NOTICE

Technologies, application of technologies, and recommended procedures in various industries evolve over time. The IEEE standards development process allows participants to review developments in industries, technologies, and practices, and to determine what, if any, updates should be made to the IEEE standard. During this evolution, the technologies and recommendations in IEEE standards may be implemented in ways not foreseen during the standard's development. IEEE standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, data privacy, and interference protection practices and all applicable laws and regulations.

³ Available at: <https://ieeexplore.ieee.org/browse/standards/collection/ieee>.

⁴ Available at: <https://standards.ieee.org/standard/index.html>.

⁵ Available at: <https://standards.ieee.org/about/sasb/patcom/materials.html>.

Participants

At the time this recommended practice was completed, the Power Quality Data Interchange Format (PQDIF) Working Group had the following membership:

Daniel Sabin, *Chair*
Kevin Kittredge, *Vice Chair*

Vong Chan
Tom Cooke
William Dabbs
Stéphane Do
Joseph Grappé

Steven Johnston
Christoph Lackner
Theo Laughner
Chris Mullins
Tony Murphy
Matt Norwalk

Gaurav Singh
Steve Tatum
Jeffrey Wischkaemper
Francisc Zavoda
Dave Zech

The following members of the individual Standards Association balloting group voted on this recommended practice. Balloters may have voted for approval, disapproval, or abstention.

Saleman Alibhay
Steven Bezner
Ivan Bilic
Wen-Kung Chang
Zhiman Chen
Sharmeela Chenniappan
William Dabbs
Neal Dowling
Donald Dunn
Carl Fredericks
Kamal Garg
Shubhanker Garg
Joseph Grappé

Haiping Guo
Randy Hamilton
Werner Hoelzl
Ronald Hotchkiss
Richard Jackson
John Kay
Kevin Kittredge
Christoph Lackner
Theo Laughner
Wang Lei
Greg Luri
Muthu Madhu
John McDaniel
Chris Mullins

Arthur Neubauer
Sivaraman P
Lorraine Padden
Bansi Patel
Pathik Patel
Howard Penrose
Daniel Sabin
Jhony Sembiring
Jerry Smith
David Tepen
Timothy Unruh
John Vergis
Jeffrey Wischkaemper

When the IEEE SA Standards Board approved this recommended practice on 12 February 2025, it had the following membership:

Lei Wang, *Chair*
Vacant Position, *Vice Chair*
David J. Law, *Past Chair*
Alpesh Shah, *Secretary*

Edward Au
Ted Burse
Xiaofeng (Alfred) Chen
Doug Edwards
Nehad El-Sherif
J. Travis Griffith
Deborah R. Hagar
Guido R. Hiertz

Ronald W. Hotchkiss
Tyler L. Jaynes
Thomas Koshy
Howard Li
Xiaohui Liu
Kevin W. Lu
Hiroshi Mano

Daleep C. Mohla
Annette D. Reilly
Robby Robson
Jon Walter Rosdahl
Daniel Sabin
F. Keith Waters
Sha Wei
Luyang (Eric) Zhang

Introduction

This introduction is not part of IEEE Std 1159.3-2025, IEEE Recommended Practice for Power Quality Data Interchange Format (PQDIF).

IEEE Std 1159.3-2003 provided the power quality industry with the specification for a Power Quality Data Interchange Format (PQDIF) for the transfer of power quality data between power quality monitoring instruments and computers. This transfer standard allows the processing and analysis of power quality measurements using multi-vendor and multi-device measurements. IEEE Std 1159.3 was reaffirmed in 2009 and first revised in 2019.

The IEEE P1159.3 Working Group completed the following changes in this latest revision of IEEE Std 1159.3™:

- Completed editorial changes and corrections to the 2019 edition of IEEE Std 1159.3.
- Replaced Annex A with the Bibliography. In the previous edition, Annex A was normative and defined physical format definitions. In its place, a normative requirement to utilize the PQDIF Physical Format Definitions (pqdif_ph.h) file from the IEEE SA Open Source platform was added in 4.3.3.
- Removed Annex B.1, which in the previous edition specified logical format constant definitions. In its place a normative requirement to utilize the PQDIF Logical Format Definitions (pqdif_lg.h) file from the IEEE SA Open Source platform was added in 5.2.2.
- Removed Annex B.2, which in the previous edition specified IDs for PQDIF tags. In its place a normative requirement to utilize the PQDIF Standard ID Definitions (pqdif_id.h) file from the IEEE SA Open Source platform was added in 5.2.2.
- Added new PQDIF Logical Format Definitions values and PQDIF Standard ID Definitions.
- Added 5.3, which explains how to use timestamps with the new tags tagUCTtoLST and tagLSTtoLocal to convert between Coordinated Universal Time (UTC), Local Time, and Local Standard Time.
- Changed Annex C from informative to normative.
- Added a new Annex G on the relationship between PQDIF and the Common Information Model.

The physical structure of PQDIF remains unchanged between IEEE Std 1159.3-2019 and this new edition, making PQDIF both backward and forward compatible. Changes to the logical structure to support new capability are in the form of additional tag and ID values. This allows software that reads PQDIF to maintain compatibility by simply ignoring unrecognized tag and ID values.

The primary users of this standard are engineers and software developers who need to create software to encode and decode PQDIF files utilized for power quality data interchange using the guidelines in this recommended practice. Although the document is thorough and complete, detailed examples and source code can be quite helpful when implementing the format described herein. Open source code for reading and writing files, as well as example files, can be found at <https://opensource.ieee.org/pqdif>.

More published information about PQDIF is available. An overview of IEEE Std 1159.3 is available in Gunther [B2].⁶ An example of how PQDIF was used as part of electric utility power quality monitoring program is described in Veloso, Lebre, and Melo [B23]. An overview of the previous IEEE 1159.3 revision project is available in Sabin and Dabbs [B22].

⁶ The numbers in brackets correspond to those of the bibliography in Annex A.

Contents

1. Overview	12
1.1 Scope	12
1.2 Purpose	12
1.3 Word usage	12
2. Normative references.....	13
3. Definitions, acronyms, and abbreviations	14
3.1 Definitions	14
3.2 Acronyms and abbreviations	14
4. Physical structure.....	15
4.1 General	15
4.2 Overall file structure.....	15
4.3 Internal record structure.....	17
5. Logical structure	19
5.1 Logical record structure.....	19
5.2 Logical element structure	20
5.3 Timestamps.....	20
5.4 Logical structure of container record.....	21
5.5 Logical structure of data sources and observations	22
5.6 How to assemble a series.....	24
6. Compression.....	26
6.1 Compression algorithm—zlib.....	26
6.2 Record-level compression	26
7. Logical device modeling	27
7.1 General	27
7.2 General procedures	28
7.3 Representing a waveform recording.....	30
7.4 Representing an RMS variation recording.....	34
7.5 Representing periodically recorded steady-state values	40
7.6 Representing magnitude-duration-time tables	45
Annex A (informative) Bibliography	48
Annex B (normative) Logical structure documentation	50
Annex C (normative) PQDIF file storage and file download	77
C.1 File storage and management.....	77
C.2 PQDIF file server interfaces	80
Annex D (informative) Representation of PQDIF in XML.....	86
D.1 Examples	87
Annex E (informative) Relationship of PQDIF to COMTRADE	94
E.1 PQDIF to COMTRADE.....	94
Annex F (informative) Relationship of PQDIF to IEC 61850.....	101

F.1 IEC TR 61850-90-17	101
Annex G (informative) Relationship of PQDIF to the Common Information Model (CIM)	102

IEEE Recommended Practice for Power Quality Data Interchange Format (PQDIF)

1. Overview

1.1 Scope

This recommended practice specifies the Power Quality Data Interchange Format (PQDIF) for the transfer of power quality data between monitoring instruments and computers. This includes raw data as well as processed, simulated, proposed, specified, and calculated data. The scope of the transfer file format includes the power quality measurements as well as appropriate characterization parameters, such as sampling rate, resolution, calibration status, instrument identification, location, and other related metadata or characteristics. The recommended practice also provides best practices for representing data and guidelines for transferring power quality data. This standard incorporates open-source material hosted on the IEEE SA Open Source Platform.

1.2 Purpose

IEEE Std 1159.3 provides the power quality industry with the specification for PQDIF, which is an open and accepted data format standard for the transfer of power quality data between instruments and computers. This transfer standard allows the processing and analysis of power quality measurements using multi-vendor and multi-device data. Wider acceptance of PQDIF as a power quality data transfer format will significantly add to the value of power quality monitoring and open new opportunities for the resolution, planning, and understanding of power quality activities, including analysis and analytics. Being able to exchange data between software systems will allow other functions needed in a power quality monitoring campaign, including validation, trending, comparison, overlay, transformation, and more.

1.3 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).^{7,8}

⁷ The use of the word *must* is deprecated and cannot be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.

⁸ The use of *will* is deprecated and cannot be used when stating mandatory requirements; *will* is only used in statements of fact.

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEC 61000-4-30, Electromagnetic compatibility (EMC)—Part 4-30: Testing and measurement techniques—Power quality measurement methods.⁹

IEEE Std 1159™, IEEE Recommended Practice on Monitoring Electric Power Quality.^{10, 11}

IETF RFC 1950, ZLIB Compressed Data Format Specification version 3.3.¹²

IETF RFC 1951, DEFLATE Compressed Data Format Specification version 1.3.

Sabin et al, 2024 “pqdif-normative v1.0.0,” IEEE Open Source code for ID, Logical Format, and Physical Format Definitions.¹³

⁹ IEC publications are available from the International Electrotechnical Commission (<http://www.iec.ch>) and the American National Standards Institute (<http://www.ansi.org/>).

¹⁰ The IEEE standards or products referred to in Clause 2 are trademarks owned by the Institute of Electrical and Electronics Engineers, Incorporated.

¹¹ IEEE publications are available from the Institute of Electrical and Electronics Engineers (<http://standards.ieee.org/>).

¹² IETF RFCs are available at <https://www.ietf.org/standards/rfcs/>.

¹³ <https://opensource.ieee.org/pqdif/pqdif-normative/-/archive/1.0.0/pqdif-normative-1.0.0.zip>

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.¹⁴

daylight saving time: Time as adjusted to achieve longer evening daylight, especially in summer, by setting the clocks an hour ahead of the standard time (IEEE Std C37.237™-2018).

globally unique identifier (GUID): A randomly generated 16-byte value used in PQDIF to uniquely identify an object, which is large enough to be considered globally unique for all practical purposes.

local standard time: The uniform time defined for a local time zone that is not adjusted for daylight saving time.

local time: The uniform time defined for a local time zone that is adjusted for daylight saving time.

PQDIF: An acronym for power quality data interchange format, which is a file- or stream-based mechanism of transferring power quality and related measurement or simulation data from a data source to a data sink in a compact, unambiguous, and deterministic way.

PQDIF file: A collection of PQDIF records consisting of a Container record, Data Source record, an optional Monitor Settings record, and one or more Observation records.

PQDIF stream: A collection of PQDIF records streamed to a client using communication media such as a network connection. A stream presents records such that they are downloaded in the same order as they would be if read from a PQDIF file.

3.2 Acronyms and abbreviations

GUID	globally unique identifier
LST	local standard time
PQDIF	power quality data interchange format
UTC	coordinated universal time

¹⁴*IEEE Standards Dictionary Online* is available at: <http://dictionary.ieee.org>. An IEEE account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

4. Physical structure

4.1 General

PQDIF achieves its flexibility by segregating the definition of its physical structure from that of its logical structure. The physical structure defines how basic and complex data types are constructed in a file; the logical structure defines how the physical data is interpreted as power quality data.

4.2 Overall file structure

4.2.1 Standard PQDIF flat file

The file is made up of a series of records that are arranged in a linked list. These links are provided by an *absolute link* (absolute file offset or position) in the header of the records. This allows new records to be inserted, and old records to be deleted. For the records shown in Figure 1, the order of the records is Record 1, Record 2, Record 3 ... Record n .

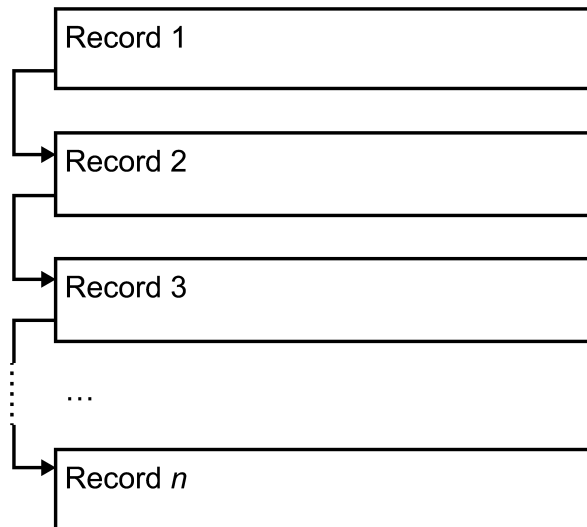


Figure 1—Standard record linkages

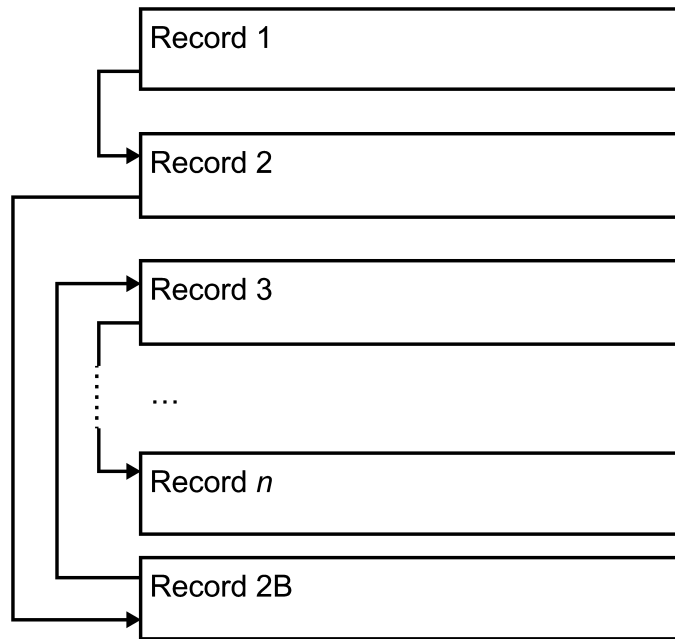


Figure 2—Inserting a new record

After the operation in Figure 2, the order of the records is now Record 1, Record 2, Record 2B, Record 3, ... Record n .

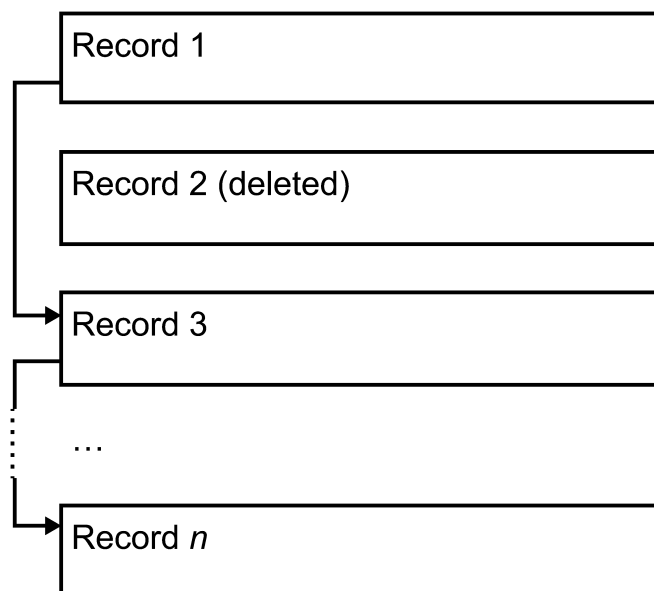


Figure 3—Deleting a record

After the operation in Figure 3, the order of the records is now Record 1, Record 3, ... Record n .

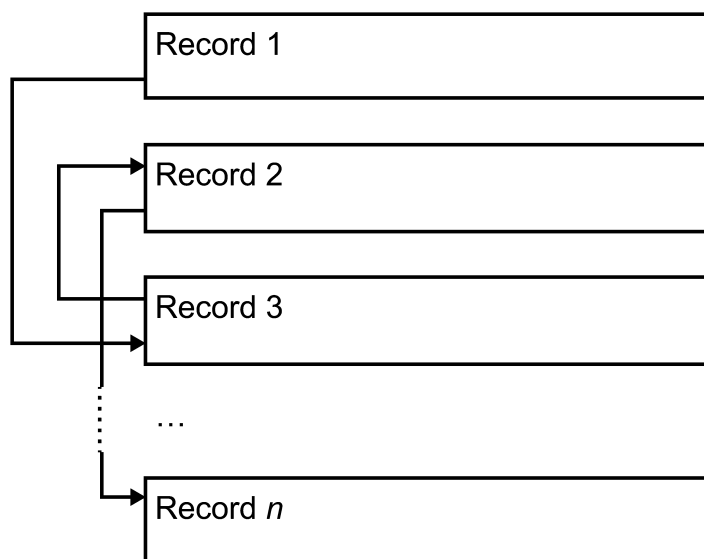


Figure 4—Reordering records

After the operation in Figure 4, the order of the records is now Record 1, Record 3, Record 2, ... Record n .

4.2.2 Alternate PQDIF structures

The PQDIF format is flexible enough that other structures besides the flat file can be used. For example, it would be possible to store individual PQDIF records in a database or a structured storage medium. In this case, some of the items in the header (the *absolute link*, for example) are not necessary and can be ignored. In some cases, the header can be discarded entirely. Using PQDIF in this fashion is out of the scope of this document, since the primary focus is the standard flat file that will be used for interchanging data.

See Annex D for an example of storing PQDIF records in XML format.

Some implementations of PQDIF involve use of a PQDIF Stream, in which a collection of PQDIF records are read by a client using a communications connection. The records in a stream are provided to a client in the same order that they would appear if read from a PQDIF file. The advantage to this approach is that the measurements could be used by a PQDIF reader before the complete PQDIF file is downloaded.

4.3 Internal record structure

Each record type in a PQDIF file has the same basic structure consisting of a header and then the record body. Figure 5 illustrates the structure of a record.

Record header • Tag: PQDIF Signature • Tag: Type of record • Size of record header • Size of record body	{ 4a111440-e49f-11cf-9900-505144494600 } tagContainer 64 bytes 512 bytes
Record body • Starts with a <i>Collection</i> • Links are relative file references, and point to elements within the body of the record. They are relative to the first byte of the record header.	<i>Collection</i> • Count: 12 Element 0 • Tag: tagFileName • Type: Vector • Physical type: CHAR1 • Link • Size (16 bytes – padded from 13) <i>Vector</i> Count: 13 (includes the NULL terminator) Data: "FILENAME.PQD"

Figure 5—Record structure

4.3.1 Record header

Each record has a standard header, which is marked with a unique “signature”—in this case, a Globally Unique Identifier (GUID). The other items in the header specify the *tag* of the record (container, data source, etc.), its *size*, and the *absolute link* to the next record. There may be others, such as *status* (active/deleted).

4.3.2 Record body

The record body is made up of a set of *elements*. There are three types of elements: *Collection*, *Scalar*, and *Vector*.

- A collection is essentially an array of *tags* and *relative links* to other elements. Because a collection can link to another collection, it is simple to create a hierarchical structure.
- A scalar is a single value of a specific physical type (INT4, REAL8, etc.).
- A vector is an arbitrarily-sized array of a specific physical type.

Each element is given a *tag* by the collection that contains it (except for the first collection). The record body always starts with a collection element. This element is tagged by its location in the record—i.e., the first element. Therefore, it is identified by the tag found in the record header.

A *relative link* is a file position that is relative to the beginning of the record header.

4.3.3 PQDIF Physical Format Definitions

The PQDIF Physical Format Definitions file (pqdif_ph.h) on the IEEE SA Open Source platform should be used as the definitions for the physical format of a PQDIF file.

5. Logical structure

5.1 Logical record structure

The basic structure is made up of a logical hierarchy of “records”—consisting of a single container, followed by one or more data sources, monitor settings, and observations. There can also be “extended” records by defining new tags.

The order of the records is defined by the absolute links in the header of each record. By following the list of records, a linear list of records is defined. This linear list has a logical hierarchy defined as shown in Figure 6.

Note that several monitor settings records can appear in the middle of the observations for a specific data source. This allows the monitor (instrument) setup to change. Consequently, some parameters in the observations may depend on the information in the appropriate monitor settings record. If multiple settings records were not allowed, a totally new data source would have to be created if any settings changed.

The Date Effective time stamps of data source and monitor settings records are used to determine which of these records applies to a given observation record. The data source and monitor settings records that have a Date Effective time stamp equal to or earlier than the subject observation record should be used.

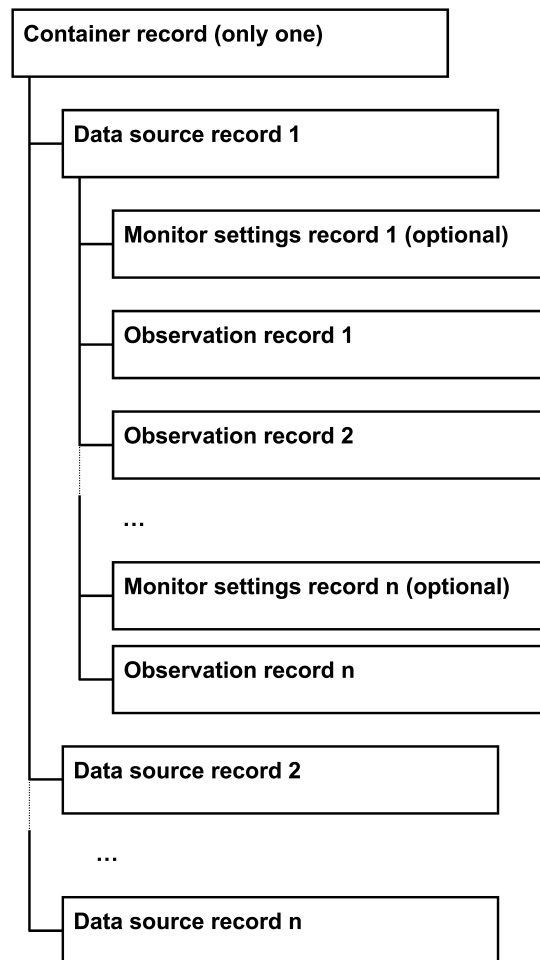


Figure 6—Logical record structure

5.2 Logical element structure

5.2.1 Introduction

The complete logical format is defined by a set of tags, and an expected structure associated with each tag. For example, a *tagName* element shall, by definition, be a vector of CHAR1—in other words, a string. It has no expected length (i.e., it can be as long or as short as necessary). A *tagVersionInfo* element is more complex: it shall be a vector of UINT4 and shall have four entries: major and minor version, and major and minor version “compatible.”

Even more complex is the way hierarchies are defined. A *tagChannelDefns* element shall be a collection, and shall contain another set of collections, which are all tagged with *tagOneChannelDefn*. (The number of elements in the *tagChannelDefns* element defines how many channel definitions are available.) Inside each *tagOneChannelDefn* collection is another set of required tags, and so forth.

See the normative references hosted on the IEEE SA Open Source platform for full documentation of all tags, the actual tag definitions, and the lists of IDs.

5.2.2 PQDIF Logical Format Definitions and Standard ID Definitions

The PQDIF Logical Format Definitions file (*pqdif_lg.h*) on the IEEE SA Open Source Platform should be used as the definitions for the tags of the logical format of a PQDIF file.

The PQDIF Standard ID Definitions File (*pqdif_id.h*) on the IEEE SA Open Source Platform should be used as the standards IDs for tags of a PQDIF file.

5.3 Timestamps

Time stamps shall be represented using the *TIMESTAMPPQDIF* structure defined in PQDIF Physical Format Definitions File (*pqdif_ph.h*). All time stamps shall be referenced to Coordinated Universal Time (UTC).

Time stamps may be converted from UTC to Local Standard Time (LST) or Local Time using optional *tagUTCtoLST* and *tagLSTtoLocal* tags.

tagUTCtoLST specifies the number of seconds to add to tag values and/or series values that store *TIMESTAMPPQDIF* types in the data source record or in settings records and/or observation records associated with this data source record from Coordinated Universal Time (UTC) to Local Standard Time (LST). For example, the offset value may be added to timestamps associated with *tagEffective*, *tagTimeInstalled*, *tagTimeRemoved*, *tagTimeCreate*, *tagTimeStart*, *tagTimeTriggered*, and *ID_QU_TIMESTAMP*.

tagLSTtoLocal specifies the number of seconds to add to tag values and/or series values that store *TIMESTAMP* types in a Data Source, Monitor Settings, or Observation record from Local Standard Time (LST) to Local Time. For example, the offset value may be added to timestamps associated with *tagEffective*, *tagTimeInstalled*, *tagTimeRemoved*, *tagTimeCreate*, *tagTimeStart*, *tagTimeTriggered*, and *ID_QU_TIMESTAMP*. If *tagLSTtoLocal* is used without *tagUTCtoLST*, then *tagUTCtoLST* shall be 0.

The specified *tagLSTtoLocal* value is inherited by all “child” records that reference a record containing the value. For example, a value specified in a Data Source record is applied to the Monitor Settings and Observation records that reference the Data Source. The value in a “parent” record can be overridden by the value specified in a child record. Time series data stored in Observation records may span Daylight Savings time transitions. When applied to Observation records local time adjustments using *tagLSTtoLocal* reflect the reference time stamp of the time series data in the observation. Separate observation records shall be used to specify local standard time and local time data. For example, to

represent value log data that spans a Standard Time to Daylight Saving Time transition, two observation records may be used: one with without tagLSTtoLocal or with tagLSTtoLocal set to 0 and the other with tagLSTtoLocal set to the appropriate offset.

5.4 Logical structure of container record

The container record contains overall—or summary—information about the PQDIF file. The following text shows an ASCII dump of the container record from a typical PQDIF file produced by a simple PQDIF utility program.

```
-Collection -- tag: tagContainer (level 0)
+-Vector -- tag: tagFileName (type: CHAR1) [ 32 ]
| value: 'D:\PQDIF\Native15\PQDIFr\wf.pqd'
+-Scalar -- tag: tagCreation (type: TIMESTAMPPQDIF)
| value: 7/7/1999 20:40:58.000000217
+-Vector -- tag: tagVersionInfo (type: UNS_INTEGER4) [ 4 ]
| values: 1, 5, 1, 5
+-Vector -- tag: tagLanguage (type: CHAR1) [ 11 ]
| value: 'US English'
+-Vector -- tag: tagTitle (type: CHAR1) [ 25 ]
| value: 'PQDIF Translator'
+-Vector -- tag: tagSubject (type: CHAR1) [ 15 ]
| value: 'PQDIF Export'
+-Vector -- tag: tagAuthor (type: CHAR1) [ 17 ]
| value: Thomas Edison'
+-Vector -- tag: tagKeywords (type: CHAR1) [ 37 ]
| value: 'PQDIF Translator for IEEE'
+-Vector -- tag: tagComments (type: CHAR1) [ 24 ]
| value: 'Comments field not used'
+-Vector -- tag: tagLastSavedBy (type: CHAR1) [ 27 ]
| value: 'lastSavedBy field not used'
+-Vector -- tag: tagApplication (type: CHAR1) [ 34 ]
| value: 'PQDIF translator for IEEE'
+-Vector -- tag: tagSecurity (type: CHAR1) [ 12 ]
| value: 'No security'
+-Vector -- tag: tagOwner (type: CHAR1) [ 26 ]
| value: ABCDE Corporation
+-Vector -- tag: tagCopyright (type: CHAR1) [ 67 ]
| value: 'Copyright (C) 2017 ABCDE Corporation, Inc. All Rights Reserved.'
+-Vector -- tag: tagTrademarks (type: CHAR1) [ 75 ]
| value: ABCDE Corporation is a registered trademark of ABCDE Corporation, Inc.'
+-Vector -- tag: tagNotes (type: CHAR1) [ 9 ]
| value: 'No notes'
+-Scalar -- tag: tagCompressionStyleID (type: UNS_INTEGER4)
| value: 2
+-Scalar -- tag: tagCompressionAlgorithmID (type: UNS_INTEGER4)
| value: 1
+-(End of collection)
+---- Record information
| Size on disk: 1028
+-(End of record)
```

The following two pieces of information are particularly important when interpreting the contents of the file:

- Version
- Compression

5.4.1 Version

Version information is used to control and track the evolution of the format. At some point, major changes may need to be made that affect the interpretation of some of the tags. If these changes are drastic, older readers may not be able to

interpret these standard tags properly. In this case, the “compatible version” (as described below) may have to be bumped up so that older readers will know not to attempt interpretation.

On the other hand, it may be possible to continue to update the standard without affecting backward compatibility. In this case, the compatible version will remain at 1.0, and all older readers will be able to interpret the file without any new tags.

Newer readers will be able to tell what tags to expect by looking at the “writer version.” Basically, all readers should do the following:

- If it can support the writer version, read the file using this version.
- Otherwise, if it can support the compatible version, read the file using this version.
- If it cannot support either, do not attempt to read the file.

This information is contained in a vector tagged with *tagVersionInfo*. This vector has four members, denoted as follows in Table 1.

Table 1—Version-related items

Index	Name	Meaning
0	Writer major version	The version of the logical structure being used by the writer of the file. This is for a reader to determine.
1	Writer minor version	“
2	Compatible major version	The file is also compatible with this version of the logical structure. Older readers can use this to determine if they will be able to read this file.
3	Compatible minor version	“

NOTE—The version information is intended to apply *only* to the logical structure; the physical structure will remain standard throughout the life of the format.¹⁵

The previous ASCII dump indicates that the subject PQDIF file was written by a program that uses version 1.5, and it is compatible with version 1.5 and later reader programs.

5.4.2 Compression

The container record contains an element indicating whether compression is used. The ASCII dump above specifies the *tagCompressionStyleID* and the *tagCompressionAlgorithmID*. The values shown indicate that the PQDIF file uses record-level compression, and the zlib compression algorithm is used. The integers used for these types are defined in Annex B. See Clause 6 for more information on compression.

5.5 Logical structure of data sources and observations

5.5.1 Structure

The internal structure of the *data source* and *observation* records parallel each other and in so doing, are implicitly linked together (Figure 7). In object-oriented design language, it can be described this way (note that “many” is a technical term here; it means 0 or more):

¹⁵ Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

A *data source* **has** many *channel definitions*; each *channel definition* **has** many *series definitions*.

An *observation* **has** many *channel instances*; each *channel instance* **has** many *series instances*.

This definition/instance structure parallels the object-oriented class/instance structure, and serves the same purpose. “Static” information common to all “instances” is kept at the definition (or class) level.

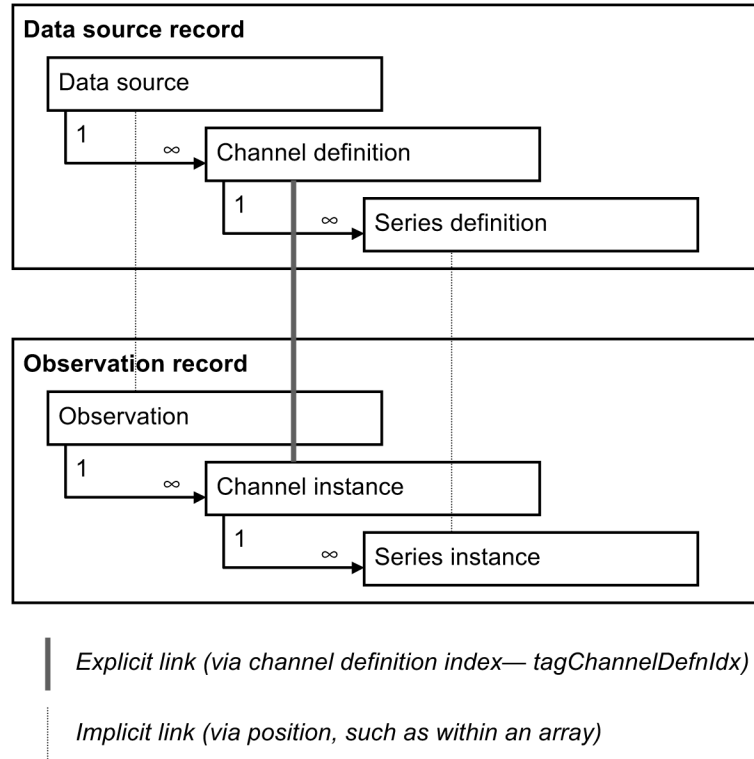


Figure 7—Data source and observation record relationship

5.5.2 Definitions

The channel definition (*tagChannelDefn*) holds the following information about the channels that would not need to be repeated for each observation:

- Channel name
- Phase or other name
- Name or phase not applicable
- High-level quantity type (waveform, trend, etc.)
- Group name (Bus 1, Bus 2, XMFR High Side, etc.)

The series definition (*tagSeriesDefn*), likewise, holds the following information that is common to all observations:

- Quantity measured (voltage, current, power, etc.)
- Quantity units (volts, amps, etc.)

- Quantity characteristic (instantaneous, peak, rms, etc.)
- The series value type (Min, Max, Average, etc.)
- Hints about how to display the data (expected Greek prefix, per unit versus engineering units, etc.)

5.5.3 Instances

The specific channel instance (*tagChannelInstance*) contains little data except for providing a link to the series instances. However, every channel definition *may or may not* be used by every single observation. Thus, there may be fewer channel instances than there are channel definitions. The channel instance then uses a definition index (*tagChannelDefnIdx*) to indicate that it is an instance of a specific channel definition.

In other words, the channel definitions provide a “library” of possible channels to be “instantiated.” The individual channel instance can decide which definition to use. It is not necessary for each observation to use every possible channel definition; the *tagChannelDefnIdx* tag is used instead to specify which channel definitions should be used.

Similarly, there may be more than one instance of a specific channel definition. This is most often encountered when reporting multiple attributes of a specific measured quantity characteristic. For example, if the third, fifth, and seventh harmonic group magnitudes of the voltage represented by a specific definition are to be recorded, three instances of this channel definition are stored in the file. Each instance contains an element that specifies which harmonic (more generally, the frequency) the instance represents. A similar technique is used to represent probability data. Multiple instances of the same channel are used to record data for a specified percentile.

The series instance (*tagSeriesInstance*) holds the array of data values that makes up its portion of the observation. The series instance is a little different from the channel instance in that each series instance shall match up directly with each series definition. If there is a need for different series topologies, additional channel definitions will be needed for each one.

5.6 How to assemble a series

Proper use of the series options described in this clause can result in a significant reduction in the size of a PQDIF file. Methods are provided to represent data series as scaled integers of varying size, referenced copies of another series, or series represented by a simple count and interval. The use of these options is recommended to help reduce PQDIF file size.

5.6.1 Explicit and scaled series

Most series will be a straight vector of data values, typically of the types INT1, INT2, INT4, REAL4, or REAL8. The vector’s count will be the total number of data points in the series. In this case, the storage method will be ID_SERIES_METHOD_VALUES.

This can be evaluated with a logical OR with ID_SERIES_METHOD_SCALED to scale each value by another constant. This would typically be used when the “raw” sample (the 2-byte integer return by an A/D converter, for example) is being stored. This way, the PQDIF file is smaller, and the final scaled value (which might otherwise have to be stored as a REAL4 or REAL8) can be calculated at read time.

5.6.2 Regular rate series

In some cases, there may be a series in which each number is separated by a constant value (this happens often in time series with a constant sampling rate). To represent this, an array of numbers can be put together that indicates how far

apart the values are supposed to be. The storage method will be `ID_SERIES_METHOD_INCREMENT`. This can even have multiple sampling rates as follows:

```
vector, count = ( N * 2 ) + 1
[ 0 ] = N (total number of rate changes)
[ 1 ] = number of points (0)
[ 2 ] = rate (0)
...
[ N * 2 - 1 ] = number of points (N-1)
[ N * 2 ] = rate (N-1)
```

Thus, if there is a single sampling rate (say, 0.01 seconds) for 1792 points, the series would be as follows:

```
vector, count = 3
[0] = 1
[1] = 1792
[2] = 0.01
```

In addition, the storage method can be evaluated with a logical OR with `ID_SERIES_METHOD_SCALED` to represent the sample rate. In this case, the series would be as follows:

```
vector, count = 3
[0] = 1
[1] = 1792'd
[2] = 1
```

and the *tagSeriesScale* element would contain 0.01.

Naturally, if there are multiple sampling rates, this scale factor would probably not be used, and instead, the scaling factor would be put directly into the series, as shown here. The following example shows a series that still contains 1792 points but has two different sampling rates:

```
vector, count = 5
[0] = 2
[1] = 896
[2] = 0.01
[3] = 896
[4] = 0.02
```

If the regular rate series does not start at 0, a *tagSeriesOffset* element can be added as a starting point. (Note that for time series, the first value should always be 0.)

tagSeriesScale, along with *tagSeriesOffset*, provides a method to translate the sample values stored in *tagSeriesValues* to real values as follows: $\text{real value} = (\text{tagSeriesValues} * \text{tagSeriesScale}) + \text{tagSeriesOffset}$.

5.6.3 Shared series

For a normal series, a *tagSeriesValues* vector is required. However, to save space, sharing data from another series is an option. In this case, the *tagSeriesValues* tag is replaced by two tags: *tagSeriesShareChannelIdx* and *tagSeriesShareSeriesIdx*. These tags indicate a “master” normal series (e.g., one that has a *tagSeriesValues* tag) that has the designated data to be shared.

These tags (*tagSeriesShareChannelIdx* and *tagSeriesShareSeriesIdx*) are indices into collections within the current observation.

5.6.4 Use of IEEE special floating-point numbers

Special floating-point numbers specified in IEEE Std 754™ [B9] should not be stored in a PQDIF series instance. Specific examples of special numbers that should not be used include positive infinity, negative infinity, not a number (NaN) values, quiet NaNs, and signaling NaNs.

Missing data should be handled by ending a series instance and beginning a new series instance. Intervals that need to be flagged should be use the flagging procedure described in 5.6.5.

5.6.5 IEC 61000-4-30 flagging

Some standards specify that values should be flagged to allow data readers to treat values in a special way. IEC 61000-4-30¹⁶ employs the concept of “flagging” certain aggregated measurements. Data is flagged if it was recorded during an interval that includes a voltage sag, a voltage swell, or a voltage interruption.

To store flagging information, the PQDIF file should contain a channel definition record with tagQuantityTypeID equal to ID_QT_VALUELOG and tagQuantityMeasuredID equal to ID_QM_STATUS.

The channel definition record should have two series definition records. The first should have tagValueTypeID equal to ID_SERIES_VALUE_TYPE_TIME. The second should have tagQuantityCharacteristicID set to ID_QC_FLAGGING_IEC_61000_4_30 and tagValueTypeID set to ID_PHYS_TYPE_BOOLEAN1, ID_PHYS_TYPE_BOOLEAN2, or ID_PHYS_TYPE_BOOLEAN4.

The intervals that need to be flagged can then be stored as individual observation records with the tagTimeStart equal to the conclusion time of the interval to be flagged, and with a series value set to 1 (for true).

Alternatively, a single observation record could be used with a series of values set to 1 if the ten-minute interval is flagged or 0 if the ten-minute interval is not flagged.

6. Compression

6.1 Compression algorithm—zlib

Since PQDIF is a binary format, it is compact even when uncompressed. Nevertheless, compression can still help in many cases. PQDIF uses the public domain *zlib* compression library when compression is desired. This algorithm is a LZ77 variant (like PKZIP) and is not covered by any patents (unlike the LZW algorithm). The *zlib* home page can be found at <http://zlib.net/>.

The data format used by the *zlib* library is described by IETF RFC 1950 and IETF RFC 1951.

6.2 Record-level compression

In record-level compression, all record headers are uncompressed (Figure 8). All record *bodies* (except the container record) are compressed as individual blocks. This compression method allows a PQDIF reader program to quickly discover the structure of the file without having to read and decode the entire contents of the file.

¹⁶ For information on references, see Clause 2.

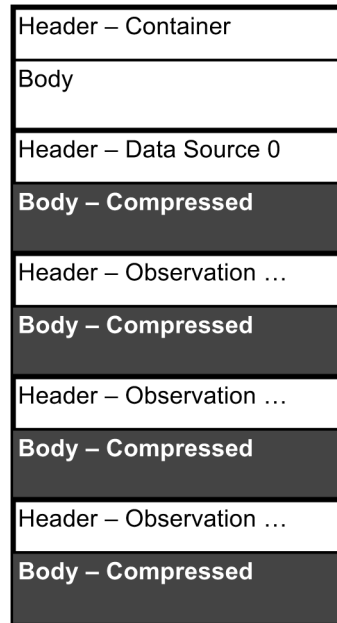


Figure 8—Compressed record layout

Since the headers are uncompressed, the absolute links work as before. The header (and the links, of course) shall reflect the *compressed* size of the body, not the uncompressed size. The relative links within the body work as before—they are all set before compression occurs. A checksum (a 32-bit CRC) for each record body shall be stored in the appropriate record header.

7. Logical device modeling

7.1 General

The previous clauses described the core format and introduced the concept of data source, monitor settings, observation records, and how to represent a basic data series. This clause describes the process of modeling power quality data in more detail.¹⁷ PQDIF is designed to support representing simulated as well as measured power quality data. For the purposes of this clause, the focus will be on power quality monitoring instrument modeling.

PQDIF is not intended to be an optimal format for use as a native information store for an actual instrument. For some instruments it could be, but not generally. More often, an instrument vendor will choose to store information recorded in their instrument in an efficient manner for the internal architecture of the device. A PQDIF file is only created when that information needs to be extracted in an instrument-independent manner for some undefined, external use. This translation may occur in the instrument itself upon demand or in an external conversion program.

The most common types of data produced by power quality monitoring devices include the following:

- Voltage and current waveform captures (e.g., transients, waveform distortion)
- Voltage and current phasor captures (e.g., short and long duration RMS variations)

¹⁷ Example PQDIF files and example source code for reading and writing PQDIF files are available in the IEEE SA Open Source platform (see Sabin et al. [B25]).

- Steady state value logs (e.g., true power factor at 15-minute intervals for some period of time)

PQDIF represents these data types by establishing a few basic concepts and categorization levels. The categorization levels are as follows:

- Quantity type
- Quantity measured
- Quantity characteristic
- Phase identification

These categories uniquely identify the primary data series of a logical channel that represents a specific measured or simulated quantity. As this document will show later, a logical channel can contain multiple data series that can be further qualified with attributes such as the following:

- Minimum value over an interval
- Maximum value over an interval
- Average value over an interval
- Instantaneous value
- Phase angle of a complex value

A waveform, or RMS trace, or value can be recorded due to a triggered event as defined by IEEE Std 1159, a periodic capture programmed into an instrument, a user-initiated manual trigger, or some external trigger based on a contact closure or algorithm evaluation. PQDIF provides the means to identify these possible states as well as the IEEE 1159 disturbance categorization of triggered events if appropriate.

The data associated with these data types are represented in an observation record. The format of that data is defined in the most recent data source record. Further information about the state of the instrument or simulation program that created the observation record is optionally contained within the most recent monitor settings record.

For triggered events, all the data associated with a disturbance as defined in IEEE Std 1159 is contained within a single observation. For example, an RMS variation is defined to begin when one or more phases of voltage goes outside of threshold and is defined to end when all phases are back within the threshold. All the channels used to represent the RMS trace(s) and/or waveform(s) captured during that disturbance would be stored in a single PQDIF observation record.

7.2 General procedures

Translation of power quality data of any format requires that the following information be gleaned from the native file format and translated to PQDIF:

- Measured data type—Description of what is represented by the actual quantitative measurement data. This data source information associated with a given numeric measurement data set is often referred to as the data channel definition.
- Monitor information—General monitor information such as threshold settings, installation date, calibration information, etc. The extent of monitor information that needs to be translated to PQDIF is dependent on the destination of the translated PQDIF files—the final application that will use the PQDIF files as an input.
- Measured data values—Actual quantitative measurement data associated with a defined data channel.

The following subclauses discuss the PQDIF element tags that should be populated to relate the three pieces of information listed. The next subclause discusses the required tags for specifying data channels (measurement data type) in the data source record. Subclause 7.2.2 discusses the required tags for specifying monitor information in the optional monitor settings record. Finally, 7.2.3 discusses the required tags for specifying measurement values recorded for a specified channel (measurement data values) in the observation record.

7.2.1 Channel specifications overview

Before the individual measurements can be translated to the PQDIF format, individual logical channels should be defined in the PQDIF file that relate what the measured data from the instrument's physical channels represents. A logical channel in PQDIF is a unique combination of quantity type, quantity measured, quantity characteristic, and phase. The data source record is used in PQDIF to define the attributes of all the logical channels that may be referenced by the instance data in a PQDIF file.

PQDIF requires several data fields to completely describe these elements that define each channel. Table 2 lists the required element tags that shall be assigned values for each channel. Table 2 shows two series definitions. Note that a channel may consist of only one series, but most channels consist of two (e.g., a series of time values and a parallel series of magnitude values representing the instantaneous value of a waveform at each time point in the time series). Also, it should be noted that the tags listed in Table 2 are only the required tags. There are many other optional tags that may be assigned values.

Each of the required element tags listed in Table 2 have an associated set of acceptable values. The “Example Value” column of Table 2 shows the values that should be assigned to define the channel that records instantaneous voltage waveforms on phase A-neutral. See Annex B for the list of acceptable values for each of the element tags defined in Table 2.

Table 2—Required data source record element tags

Level	Element tag	Example value
0	tagRecDataSource	N/A
1	tagNominalFrequency	60.0
1	tagChannelDfns	N/A
2	tagOneChannelDfn	N/A
3	tagPhaseID	ID Phase AN
3	tagQuantityTypeID	ID QT Waveform
3	tagQuantityMeasuredID	ID QM Voltage
3	tagSeriesDfns	N/A
4	tagOneSeriesDfn	N/A
5	tagValueTypeID	ID Series Value Type Time
5	tagQuantityUnitsID	ID QU Seconds
5	tagQuantityCharacteristicID	ID QC None
5	tagStorageMethodID	ID Series Method Values
4	tagOneSeriesDfn	N/A
5	tagValueTypeID	ID Series Value Type Val
5	tagQuantityUnitsID	ID QU Volts
5	tagQuantityCharacteristicID	ID QC Instantaneous
5	tagStorageMethodID	ID Series Method Values

7.2.2 Monitor settings specifications overview

The tags in the monitor settings record provide basic, vendor-independent instrument settings information. The monitor settings record is not required. If, however, information such as the trigger values for a given channel is critical to the data being translated, it is recommended that the monitor settings record be created. Most of the defined tags are optional and are to be populated only if the data exists and would be useful to other applications.

Table 3 lists the required monitor settings tags and example values for the example channel defined in Table 2. Note that only one channel is included in Table 3, but monitor setting tag values should be specified for each channel defined. Furthermore, there are many channel level (level 3) tags that are optional if the information is available.

Table 3—Required monitor settings record element tags

Level	Element tag	Example value
0	TagRecMonitorSettings	N/A
1	TagEffective	Date settings were implemented.
1	TagTimeInstalled	Date monitored installed
1	TagChannelSettingsArray	N/A
2	TagOneChannelSetting	N/A
3	TagChannelDefnIdx	57
3	TagTriggerTypeID	N/A

7.2.3 Observation specifications overview

Assignment of the observation element tags is primarily a matter of listing the measurement values in a way that coordinates with the way the associated channel has been specified. Note that a single PQDIF observation may consist of many different measurements on one or more physical channels that are associated with the same trigger. For example, a single voltage sag might trigger both waveform and rms phasor measurements. Although the data recorded due to this single sag is associated with many different channels, it is more efficient to store all the data associated with a single trigger in a single observation. This also allows for proper association of data channels in the final application that uses the PQDIF files.

Table 4 lists the required element tags that define a given observation. As with the channel specification, there are other optional tags that can be assigned as well. Permissible values for each element tag listed in Table 4 are provided in PQDIF logical structure document. The “Example Value” column of Table 4 lists possible values for a Phase A voltage waveform associated with the channel specified in Table 2.

Table 4—Required observation record element tags

Level	Element tag	Example value
0	TagRecObservation	N/A
1	TagObservationName	<i>Phase A Voltage Waveform</i>
1	TagTimeCreate	<i>12/9/1998 11:42:24.453813818</i>
1	TagTimeStart	<i>2/13/1998 23:17:44.086079597</i>
1	TagTriggerMethodID	<i>ID Trigger Meth Channel</i>
1	TagTimeTriggered	<i>2/13/1998 23:17:44.086079597</i>
1	TagChannelInstances	N/A
2	TagOneChannelInst	N/A
3	TagChannelDefnIdx	57
3	TagSeriesInstances	N/A
4	TagOneSeriesInstance	N/A
5	TagSeriesValues	0,0.00013,0.00026, . . .
4	TagOneSeriesInstance	N/A
5	TagSeriesValues	-4607.4621, -5030.9689, -5428.9193, . . .

Subclauses 7.3 through 7.6 describe how to represent these common power quality data types in PQDIF by example.

7.3 Representing a waveform recording

A waveform recording is represented in PQDIF using channel definitions of the QT_WAVEFORM quantity type. This is the simplest of the PQDIF quantity types and one of the most commonly used types for instruments that capture

waveforms of voltage and current. Figure 9 illustrates a transient event captured by a power monitor, exported into the PQDIF representation described here and visualized by a PQDIF viewer program.

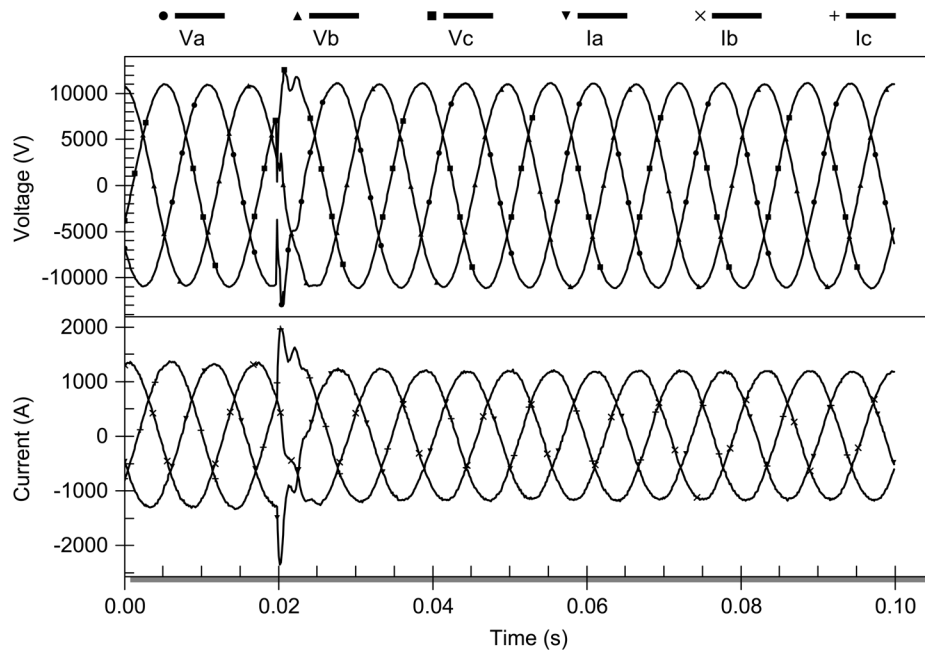


Figure 9—Low-frequency transient (capacitor switching)

The individual measurements logged by the instrument are stored in PQDIF “observation” records. If a single trigger causes a power quality monitor to save waveforms for many different quantities (for example, cross-triggered measurements), it is useful for all the measurements associated with the same trigger to be included in a single observation record. If an instrument can capture more than one waveform block during a single disturbance as defined by IEEE Std 1159 (e.g., five cycles of waveform at the beginning of a sag and five cycles of waveform at the end of the same sag), then more than one instance of a specific channel definition may be recorded in the PQDIF observation.

The following text is an ASCII dump from a PQDIF utility program that shows the top-level tags of a data source record for a real PQDIF file. These tags permit the specification of basic information about the data source.

```
-Collection -- tag: tagRecDataSource (level 0)
| The checksum for this record is correct.
+-Scalar -- tag: tagDataSourceTypeID (type: GUID)
| value: {e6b51730-f747-11cf-9d890080} - ID_DS_TYPE_MEASURE
+-Scalar -- tag: tagVendorID (type: GUID)
| value: {e6b5170a-f747-11cf-9d890080} - ID_VENDOR_ELECTROTEK
+-Scalar -- tag: tagEquipmentID (type: GUID)
| value: {e6b51721-f747-11cf-9d890080} - ID_EQUIP_ETK_TESTPROGRAM
+-Vector -- tag: tagSerialNumberDS (type: CHAR1) [ 4 ]
| value: '1.0'
+-Vector -- tag: tagVersionDS (type: CHAR1) [ 4 ]
| value: '1.0'
+-Vector -- tag: tagNameDS (type: CHAR1) [ 14 ]
| value: 'PQDIF Convert'
+-Vector -- tag: tagOwnerDS (type: CHAR1) [ 4 ]
| value: 'John Smith'
+-Vector -- tag: tagLocationDS (type: CHAR1) [ 1 ]
| value: ''
+-Vector -- tag: tagTimeZoneDS (type: CHAR1) [ 1 ]
| value: ''
```