INTERNATIONAL STANDARD

ISO/IEEE 11073-10427

First edition
2018-01

Health informatics — Personal health device communication —

Part 10427:
Device specialization — Power status monitor of personal health devices

Informatique de santé — Communication entre dispositifs de santé personnels —

Partie 10427: Spécialisation des dispositifs — Moniteur de l'état de puissance des dispositifs de santé personnels

Reference number
ISO/IEEE 11073-10427:2018(E)

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the 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.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is called to the possibility that implementation of this standard may require the use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. ISO/IEEE is not responsible for identifying essential patents or patent claims for which a license may be required, for conducting inquiries into the legal validity or scope of patents or patent claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance or a Patent Statement and Licensing Declaration Form, 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 ISO or the IEEE Standards Association.

ISO/IEEE 11073-10427 was prepared by the IEEE 11073 Standards Committee of the IEEE Engineering in Medicine and Biology Society (as IEEE Std 11073-10427-2016). It was adopted by Technical Committee ISO/TC 215, *Health informatics*, in parallel with its approval by the ISO member bodies, under the "fast-track procedure" defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE. IEEE is responsible for the maintenance of this document with participation and input from ISO member bodies.

Health informatics—Personal health device communication

Part 10427: Device specialization— Power Status Monitor of Personal Health Devices

Sponsor

IEEE 11073™ Standards Committee
of the
IEEE Engineering in Medicine and Biology Society

Approved 22 September 2016

IEEE-SA Standards Board

Abstract: Guidelines for establishing criteria for application, performance, interchangeability, tests, life cycle costs, and safety requirements of traction power rectifier transformers are established in this standard. Set forth are the electrical, mechanical and thermal design, manufacturing, and testing requirements for traction power rectifier transformers for dc electrification systems. Covered in this standard are liquid-immersed and dry-type transformers, including those with cast coil and epoxy resin encapsulated windings.

Keywords: basic lightning impulse insulation, BIL, commutating impedance, design optimization, electrical requirements, factory tests, ferroresonance, heavy rail, hot spot, IEEE 1653.1™, light rail, load cycle, overvoltage transient, partial discharge (PD) service conditions, tests, traction power duty cycle, traction power rectifier transformers, transit application

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

PDF: ISBN 978-1-5044-2268-0 STD21081
Print: ISBN 978-1-5044-2269-7 STDPD21081

IEEE prohibits discrimination, harassment, and bullying.
For more information, visit http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notices and Disclaimers Concerning IEEE Standards Documents." They can also be obtained on request from IEEE or viewed at http://standards.ieee.org/IPR/disclaimers.html.

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

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association ("IEEE-SA") Standards Board. IEEE ("the Institute") develops its standards through a consensus development process, approved by the American National Standards Institute ("ANSI"), which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed through scientific, academic, and industry-based technical working groups. Volunteers in IEEE working groups are not necessarily members of the Institute 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 Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. 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 his or her 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: PROCUREMENT OF 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 development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual 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. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide 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 Standards Coordinating Committees 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 revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

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 imply 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.

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, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, 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. 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 ten years. When a document is more than ten 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 the IEEE Xplore at http://ieeexplore.ieee.org/ or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE's standards development process, visit the IEEE-SA Website at http://standards.ieee.org.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: http://standards.ieee.org/findstds/errata/index.html. Users are encouraged to check this URL for errata periodically.

Patents

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 http://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.

Participants

At the time this standard was submitted to the IEEE-SA Standards Board for approval, the 11073 Personal Health Device Working Group had the following membership:

Daidi Zhong, *Co-Chair*
Michael J. Kirwan, *Co-Chair*
Miriam L. Makhlouf, *Vice Chair*

Karsten Aalders
Charles R. Abbruscato
Nabil Abujbara
Maher Abuzaid
James Agnew
Haidar Ahmad
Manfred Aigner
Jorge Alberola
Murtaza Ali
Rolf Ambuehl
David Aparisi
Paolo Ariano
Lawrence Arne
Diego B. Arquillo
Serafin Arroyo
Muhammad Asim
Merat Bagha
Doug Baird
David Baker
Anindya Bakshi
Ananth Balasubramanian
Sunlee Bang
M. Jonathan Barkley
Gilberto Barrón
David Bean
John Bell
Rudy Belliardi
Kathryn M. Bennett
Daniel Bernstein
George A. Bertos
Chris Biernacki
Ola Björsne
Thomas Blackadar
Marc Blanchet
Thomas Bluethner
Douglas P. Bogia
Xavier Boniface
Shannon Boucousis
Julius Broma
Lyle G. Bullock Jr.
Bernard Burg
Chris Burns
Anthony Butt
Jeremy Byford-Rew
Satya Calloji
Xiaoying Cao
Carole C. Carey
Craig Carlson
Santiago Carot-Nemesio
Randy W. Carroll
Simon Carter
Seungchul Chae
Rahul Chauhan
James Cheng
Peggy Chien
David Chiu
Chia-Chin Chong
Saeed A. Choudhary
Jinhan Chung
Malcolm Clarke
John A. Cogan
John T. Collins
Cory Condek
Todd H. Cooper
David Cornejo
Douglas Coup
Nigel Cox
Hans Crommenacker
Tomio Crosley
David Culp
Allen Curtis
Ndifor Cyril Fru
Jesús Daniel Trigo
Eyal Dassau
David Davenport
Russell Davis
Sushil K. Deka
Ciro de la Vega
Pedro de-las-Heras-Quiros
Jim DelloStritto
Matthew d'Entremont
Lane Desborough
Kent Dicks
Hyoungho Do
Xiaolian Duan
Brian Dubreuil
Sourav Dutta
Jakob Ehrensvard
Fredrik Einberg
Michihiro Enokida
Javier Escayola Calvo
Mark Estes
Leonardo Estevez
Roger Feeley
Hailing Feng
Bosco T. Fernandes
Christoph Fischer
Morten Flintrup
Joseph W. Forler
Russell Foster
Eric Freudenthal
Matthias Frohner
Ken Fuchs
Jing Gao
Xuemei Gao
Marcus Garbe
John Garguilo
Rick Geimer
Igor Gejdos
Ferenc Gerbovics
Nicolae Goga
Julian Goldman
Raul Gonzalez Gomez
Chris Gough
Channa Gowda
Charles M. Gropper
Amit Gupta
Jeff Guttmacher
Rasmus Haahr
Christian Habermann
Michael Hagerty
Jerry Hahn
Robert Hall
Nathaniel Hamming
Rickey L. Hampton
Sten Hanke
Aki Harma
Jordan Hartmann
Kai Hassing
Marc Daniel Haunschild
Wolfgang Heck
Nathaniel Heintzman
Charles Henderson
Jun-Ho Her
Takashi Hibino
Timothy L. Hirou
Allen Hobbs
Alex Holland
Arto Holopainen
Kris Holtzclaw
Robert Hoy
Frank Hsu
Anne Huang
Sen-Der Huang
Zhiqiang Huang

Ron Huby
David Hughes
Robert D. Hughes
Jiyoung Huh
Hugh Hunter
Hitoshi Ikeda
Yutaka Ikeda
Philip O. Isaacson
Atsushi Ito
Michael Jaffe
Praduman Jain
Wei Jin
Danny Jochelson
Phaneeth Junga
Akiyoshi Kabe
Steve Kahle
Tomio Kamioka
Kei Kariya
Andy Kaschl
Junzo Kashihara
Kohichi Kashiwagi
Ralph Kent
Laurie M. Kermes
Ikuo Keshi
Junhyung Kim
Minho Kim
Min-Joon Kim
Taekon Kim
Tetsuya Kimura
David Klonoff
Alfred Kloos
Jeongmee Koh
Jean-Marc Koller
John Koon
Patty Krantz
Raymond Krasinski
Alexander Kraus
Ramesh Krishna
Geoffrey Kruse
Falko Kuester
Rafael Lajara
Pierre Landau
Jaechul Lee
JongMuk Lee
Kyong Ho Lee
Rami Lee
Sungkee Lee
Woojae Lee
Yonghee Lee
Joe Lenart
Kathryn A. Lesh
Qiong Li
Ying Li
Patrick Lichter
Jisoon Lim
Joon-Ho Lim
John Lin
Wei-Jung Lo
Charles Lowe
Don Ludolph
Christian Luszick
Bob MacWilliams
Srikkanth Madhurbootheswaran
Miriam L. Makhlouf
Romain Marmot
Sandra Martinez
Miguel Martínez de Espronceda Cámara
Peter Mayhew
Jim McCain
László Meleg
Alexander Mense
Jinsei Miyazaki
Erik Moll
Darr Moore
Carsten Mueglitz
Piotr Murawski
Soundharya Nagasubramanian
Jae-Wook Nah
Alex Neefus
Trong-Nghia Nguyen-Dobinsky
Michael E. Nidd
Tetsu Nishimura
Jim Niswander
Hiroaki Niwamoto
Thomas Norgall
Anand Nonbade
Yoshiteru Nozoe
Abraham Ofek
Brett Olive
Begonya Otal
Marco Paleari
Charles Palmer
Bud Panjwani
Carl Pantiskas
Harry P. Pappas
Hanna Park
Jong-Tae Park
Myungeun Park
Soojun Park
Phillip E. Pash
TongBi Pei
Lucian Pestritu
Soren Petersen
James Petisce
Peter Piction
Michael Pliskin
Jeff Price
Harald Prinzhorn
Harry Qiu
Arif Rahman
Tanzilur Rahman
Steve Ray
Phillip Raymond
Tim Reilly
Barry Reinhold
Brian Reinhold
Melvin I. Reynolds
John G. Rhoads
Jeffrey S. Robbins
Moskowitz Robert
Chris Roberts
Timothy Robertson
David Rosales
Bill Saltzstein
Benedikt Salzbrunn
Giovanna Sannino
Jose A. Santos-Cadenas
Stefan Sauermann
John Sawyer
Guillaume Schatz
Alois Schloegl
Paul S. Schluter
Lars Schmitt
Mark G. Schnell
Richard A. Schrenker
Antonio Scorpiniti
Kwang Seok Seo
Riccardo Serafin
Sid Shaw
Frank Shen
Bozhi Shi
Min Shih
Mazen Shihabi
Redmond Shouldice
Sternly K. Simon
Marjorie Skubic
Robert Smith
Ivan Soh
Motoki Sone
Emily Sopensky
Rajagopalan Srinivasan
Andreas Staubert
Nicholas Steblay
Beth Stephen
Lars Steubesand
John (Ivo) Stivoric
Raymond A. Strickland
Chandrasekaran Subramaniam
Hermanni Suominen
Lee Surprenant
Ravi Swami
Ray Sweidan
Jin Tan
Yi Tang
Haruyuki Tatsumi
John W. Thomas
Jonas Tirén
Alexandra Todiruta
Janet Traub
Gary Tschautscher
Masato Tsuchid
Ken Tubman

Yoshihiro Uchida	Jerry P. Wang	Ariton Xhafa
Akib Uddin	Yao Wang	Yaxi Yan
Sunil Unadkat	Yi Wang	Ricky Yang
Fabio Urbani	Steve Warren	Melanie S. Yeung
Philipp Urbauer	Fujio Watanabe	Done-Sik Yoo
Laura Vanzago	Toru Watsuji	Jianchao Zeng
Alpo Värri	Mike Weng	Jason Zhang
Dalimar Velez	Kathleen Wible	Zhiqiang Zhang
Rudi Voon	Paul Williamson	Thomas Zhao
Isobel Walker	Jan Wittenber	Miha Zoubek
David Wang	Jia-Rong Wu	Szymon Zyskoter
	Will Wykeham	

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Lyle Bullock	Werner Hoelzl	Bartien Sayogo
Keith Chow	Noriyuki Ikeucki	Walter Struppler
Malcolm Clarke	Atsushi Ito	Jan Wittenber
Sourav Dutta	Piotr Karocki	Paul Work
Christoph Fischer	Erik Moll	Oren Yuen
Randall Groves	Michael Newman	Daidi Zhong
	Melvin Reynolds	

When the IEEE-SA Standards Board approved this standard on 22 September 2016, it had the following membership:

Jean-Philippe Faure, *Chair*
Ted Burse, *Vice Chair*
John D. Kulick, *Past Chair*
Konstantinos Karachalios, *Secretary*

Chuck Adams	Ronald W. Hotchkiss	Mehmet Ulema
Masayuki Ariyoshi	Michael Janezic	Yingli Wen
Stephen Dukes	Joseph L. Koepfinger*	Howard Wolfman
Jianbin Fan	Hung Ling	Don Wright
J. Travis Griffith	Kevin Lu	Yu Yuan
Gary Hoffman	Annette D. Reilly	Daidi Zhong
	Gary Robinson	

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 11073-10427-2016, Health Informatics—Personal health device communication—Part 10427: Device Specialization—Power Status Monitor of Personal Health Devices.

ISO/IEEE 11073 standards enable communication between medical devices and external computer systems. This document uses the optimized framework created in ISO/IEEE 11073-20601 and describes a specific, interoperable communication approach for power status monitor of personal health devices (PHDs). These standards align with and draw on the existing clinically focused standards to provide support for communication of data from clinical or PHDs.

Contents

Health informatics—Personal health device communication

Part 10427: Device specialization—Power Status Monitor of Personal Health Devices

1. Overview

1.1 Scope

This standard establishes a normative definition of communication between devices containing a power source (agents) and managers (e.g., cell phones, personal computers, personal health appliances, set-top boxes) in a manner that enables plug-and-play interoperability. Using existing terminology, information profiles, application profile standards, and transport standards as defined in other ISO/IEEE 11073 standards, this standard defines a common core of communication functionality of personal health devices (PHDs) containing a battery, including: 1) current device power status (e.g., on mains or on battery); 2) power charge status (e.g., percent of full charge); and 3) estimated time remaining (e.g., minutes).

1.2 Purpose

This standard addresses a need for an openly defined, independent standard for power status information exchange from PHDs to manager(s) (e.g., cell phones, personal computers, personal health appliances, and set-top boxes). Interoperability is the key to growing the potential market for monitoring the power status of devices and enabling people to be better informed participants in the management of their health.

1.3 Context

See ISO/IEEE 11073-20601:2016[1] for an overview of the environment within which this standard is written.

This standard defines the specialization for monitoring the power status of PHDs, as well as being a specific agent type. It also provides a description of the concepts, its capabilities, and its implementation according to this standard.

[1]Information on references can be found in Clause 2.

This standard is based on ISO/IEEE 11073-20601:2016, ISO/IEEE 11073-20601a:2010, and IEEE Std 11073-20601-2008, which in turn draw information from both ISO/IEEE 11073-10201:2004 [B8] and ISO/IEEE 11073-20101:2004 [B9].[2] The medical device encoding rules (MDERs) used within this standard are fully described in Annex F of ISO/IEEE 11073-20601:2016.

This standard reproduces relevant portions of the nomenclature found in ISO/IEEE 11073-10101a:2015 [B6] and in ISO/IEEE 11073-10101:2004 [B7] and adds new nomenclature codes for the purposes of this standard. Together this standard, ISO/IEEE 11073-20601:2016, and IEEE Std 11073-20601a:2010 document all implementation required nomenclature codes.

NOTE 1—IEEE Std 11073-20601a:2010 is an amendment to IEEE Std 11073-20601:2008. It contains new material and corrections and does not copy the content of ISO/IEEE 11073-20601:2010. Any reference to ISO/IEEE 11073-20601:2016 refers to the document that is obtained after applying this new material and corrections to ISO/IEEE 11073-20601:2010.[3]

NOTE 2—In this standard, ISO/IEEE 11073-104zz is used to refer to the collection of device specialization standards that utilize ISO/IEEE 11073-20601:2016, where zz can be any number from 01 to 99, inclusive.

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so that 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.

IEEE Std 11073-20601™-2008, Health informatics—Personal health device communication—Part 20601: Application profile—Optimized exchange protocol. (Use with IEEE Std 11073-20601a-2010.)[4,5]

ISO/IEEE 11073-20601:2016, Health informatics—Personal health device communication—Application profile—Optimized Exchange Protocol.[6]

ISO/IEEE 11073-20601a:2010, Health informatics—Personal health device communication—Application Profile—Optimized Exchange Protocol—Amendment 1.

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.[7]

agent: A node that collects and transmits personal health data to an associated manager.

class: In object-oriented modeling, it describes the attributes, methods, and events that objects instantiated from the class utilize.

device: A term used to refer to a physical apparatus implementing either an agent or a manager role.

[2]The numbers in brackets correspond to those of the bibliography in Annex A.
[3]Notes in text, tables, and figures are given for information only and do not contain requirements needed to implement the standard.
[4]The IEEE standards or products referred to are trademarks of the Institute of Electrical and Electronics Engineers, Inc.
[5]IEEE publications are available from the Institute of Electrical and Electronics Engineers (http://standards.ieee.org/).
[6]ISO/IEEE publications are available from the International Organization for Standardization (http://www.iso.ch/) and the Institute of Electrical and Electronics Engineers (http://standards.ieee.org/).
[7]The *IEEE Standards Dictionary Online is* available at: http://ieeexplore.ieee.org/xpls/dictionary.jsp.

handle: An unsigned 16-bit number that is locally unique and identifies one of the object instances within an agent.

manager: A node receiving data from one or more agent systems. Some examples of managers include a cellular phone, health appliance, set-top box, or a computer system.

object: In object-oriented modeling, a particular instantiation of a class. The instantiation realizes attributes, methods, and events from the class.

obj-handle: *See:* **handle**.

personal health device (PHD): A device used in personal health applications.

power status monitor (PSM): An agent that monitors the power status of a device and communicates this to a manager.

3.2 Acronyms and abbreviations

APDU	application protocol data unit
ASN.1	Abstract Syntax Notation One
DIM	domain information model
EUI-64	extended unique identifier (64 bits)
ICS	implementation conformance statements
MDC	medical device communication
MDER	medical device encoding rules
MDS	medical device system
MOC	managed object class
OID	object identifier
PDU	protocol data unit
PHD	personal health device
PSM	power status monitor
VMO	virtual medical object
VMS	virtual medical system

4. Introduction to ISO/IEEE 11073 personal health devices (PHDs)

4.1 General

This standard and the remainder of the series of ISO/IEEE 11073 PHD standards fit in the larger context of the ISO/IEEE 11073 series of standards. The full suite of standards enables agents to interconnect and interoperate with managers and with computerized healthcare information systems. See IEEE Std 11073-20601 for a description of the guiding principles for this series of ISO/IEEE 11073 PHD standards.

IEEE Std 11073-20601 supports the modeling and implementation of an extensive set of PHDs. This standard defines aspects for monitoring power status of PHDs. It describes all aspects necessary to implement the application layer services and data exchange protocol between an ISO/IEEE 11073 PHD agent and a manager for power status monitoring. This standard defines a subset of the objects and functionality contained in IEEE Std 11073-20601 and it extends and adds definitions where appropriate. All new definitions are given in Annex B, titled Abstract Syntax Notation One (ASN.1). Nomenclature codes referenced in this standard, which are not defined in IEEE Std 11073-20601, are normatively defined in Annex C.

4.2 Introduction to IEEE 11073-20601 modeling constructs

4.2.1 General

The ISO/IEEE 11073 series of standards, and in particular IEEE Std 11073-20601, is based on an object-oriented systems management paradigm. The overall system model is divided into three principal components: the domain information model (DIM), the service model, and the communication model. See IEEE Std 11073-20601 for a detailed description of the modeling constructs.

4.2.2 Domain information model

The domain information model (DIM) is a hierarchical model that describes an agent as a set of objects. These objects and their attributes represent the elements that control behavior and report on the status of the agent and the data that an agent can communicate to a manager. Communication between the agent and the manager is defined by the application protocol in IEEE Std 11073-20601.

4.2.3 Service model

The service model defines the conceptual mechanisms for the data exchange services. Such services are mapped to messages that are exchanged between the agent and the manager. Protocol messages within the ISO/IEEE 11073 series of standards are defined in ASN.1. The messages defined in IEEE Std 11073-20601 can coexist with messages defined in other standard application profiles defined in the ISO/IEEE 11073 series of standards.

4.2.4 Communication model

In general, the communication model supports the topology of one or more agents communicating over logical point-to-point connections to a single manager. For each logical point-to-point connection, the dynamic system behavior is defined by a connection state machine as specified in IEEE Std 11073-20601.

4.2.5 Implementing the models

An agent implementing this standard shall implement all mandatory elements of the information, service, and communication models as well as all conditional elements when the condition is met. The agent should implement the recommended elements, and it may implement any combination of the optional elements. A manager implementing this standard shall utilize the mandatory conditional, recommended, or optional elements. In this context, "utilize" means to use the element as part of the primary function of the manager device. For example, a manager whose primary function is to display data would need to display a piece of data in the element in order to utilize it.

4.3 Compliance with other standards

Devices that comply with this standard may also be required to comply with other domain- and device-specific standards that supersede the requirements of this standard with respect to issues including safety, reliability, and risk management. A user of this standard is expected to be familiar with all other such standards that apply and to comply with any higher specifications thus imposed.

Typically, medical devices should comply with the IEC 60601-1:2005 [B1] base standards with respect to electrical and mechanical safety and any device-specific standard as might be defined in the IEC 60601-2 [B2] series of standards. Software aspects may apply through standards such as IEC 62304:2006/EN

62304:2006 [B3]. Devices that comply with this standard implement higher layers of network software and utilize lower layers as appropriate to the application. The requirements on performance of such applications and conformance are defined elsewhere and are outside the scope of this standard. Moreover, the use of any medical equipment is subject to risk assessment and risk management appropriate to the application. Some relevant examples are ISO 14971:2007 [B5] and IEC 80001-1:2010 [B4]. The requirements of such risk assessment, risk management, and conformance are outside the scope of this standard.

5. Power status monitor concepts and modalities

5.1 General

This clause presents the general concepts of power status monitors (PSMs). In the context of PHDs in the ISO/IEEE 11073 family of standards, a PSM provides the functionality that monitors the power status of a PHD. The following aspects should be kept in mind when reading this specification for power status monitoring.

PHDs are usually powered by main sources or batteries. The number and types of batteries connected to a PHD can vary. PHDs may obtain measures from their power sources that indicate the current power status of the device. Some PHDs can obtain basic information about the current status of their power sources. This basic information may include: whether power is being drawn from battery or main power lines, status of charging, percentage of battery capacity remaining, and predicted amount of operational time left on the batteries.

5.2 Use case

PHDs available today are allowing more and more critical care device dependent patients to live and function at home. This greatly reduces their healthcare costs and provides a better environment for the patient as well. However, issues arise when there is disruption to the power to these devices due to inclement weather. During such disruptions, patients leave their homes and go to a hospital or other location where emergency power is available. However, this can place an added burden on these facilities that may already be inundated with patients, thus overloading their emergency power supply. This PSM device specialization is developed in part to reduce the need for patients to leave their homes during a power outage.

An example use case for this device specialization has a homecare patient dependent on a critical care device. The device is equipped with a main battery for normal portable use and an internal backup battery for hot swapping the main battery, or for emergency use in the event of main battery failure. Through the use of this PSM device specialization, a manager is able to receive information from the device about the device's power status. In the case when there is a power outage in the patient's home, the PSM would communicate with a tracking service provider providing information such as the estimated operational time remaining, capacity of each of the batteries, and which batteries are active, etc. The tracking service provider can then, in turn, confirm that the adverse weather event has occurred and notify the patient's care team accordingly.

6. Power status monitor domain information model

6.1 Overview

This clause describes the DIM of the PSM.

6.2 Class extensions

In this standard, extensions are defined with respect to IEEE Std 11073-20601-2014.

The following two attributes are an extension to the Enumeration Class:

— The Capability-Mask-Basic attribute indicates whether the corresponding bit in the Enum-Observed-Value-Basic-Bit-Str attribute is supported by the Agent.

— The State-Flag-Basic attribute indicates whether the corresponding bit in the Enum-Observed-Value-Basic-Bit-Str attribute is a state or event.

If the Enum-Observed-Value-Basic-Bit-Str attribute is used, the Capability-Mask-Basic and State-Flag-Basic attributes are mandatory. The values of the Capability-Mask-Basic and State-Flag-Basic attributes are implementation specific. Please see Annex B for the ASN.1 structures.

The Capability-Mask-Basic attribute is dynamic. The State-Flag-Basic attribute is static.

6.3 Object instance diagram

The object instance diagram of the PSM DIM, which is defined for the purposes of this standard, is shown in Figure 1.

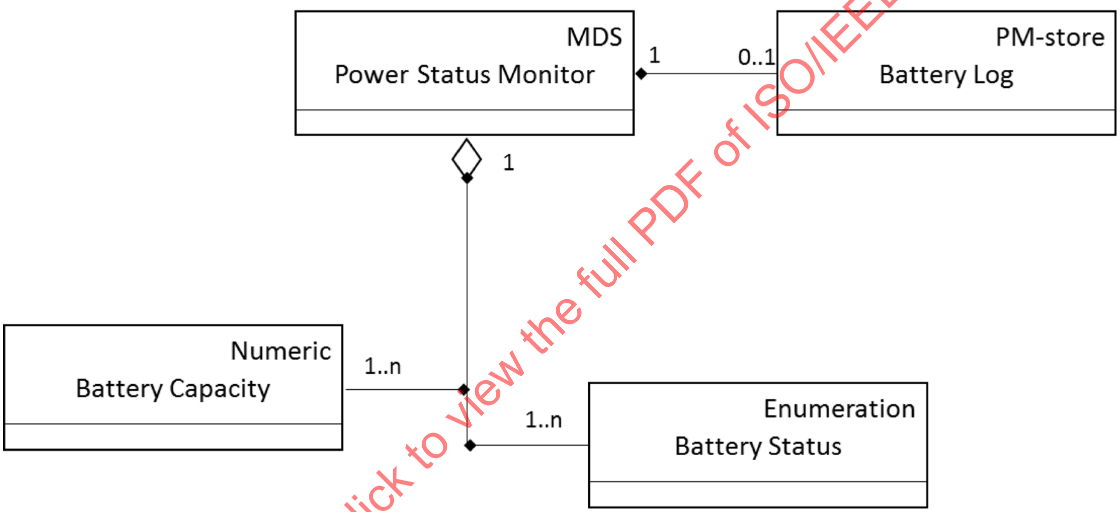

Figure 1—PSM DIM

The objects of the DIM, as shown in Figure 1, are described in 6.5 to 6.12. See 6.5 through 6.12 for descriptions of the different PSM objects. Each clause that describes an object of the PSM contains the following information:

— The nomenclature code used to identify the class of the object. One example where this code is used is the configuration event, wherein the object class is reported for each object. This allows the manager to determine whether the class of the object being specified is a numeric, real-time sample array, enumeration, scanner, or PM-store class.

— The attributes of the object. Each object has attributes that represent and convey information on the physical device and its data sources. Each object has a handle attribute that identifies the object instance within an agent. Attribute values are accessed and modified using methods such as GET and SET. Attribute types are defined using an ASN.1. The ASN.1 definitions for new attribute types specific to this standard are in Annex B, and the ASN.1 definitions for existing attribute types referenced in this standard are in IEEE Std 11073-20601.

— The methods available on the object.

— The potential events generated by the object. The data is sent to the manager using events.

— The available services, such as getting or setting attributes.

The attributes for each class are defined in tables that specify the name of the attribute, its value, and its qualifier. The qualifiers mean the following:

M Attribute is Mandatory

C Attribute is Conditional and depends on the condition stated in the Remark or Value column (if IEEE Std 11073-20601 is referenced, then it contains the conditions)

R Attribute is Recommended

NR Attribute is Not Recommended

O Attribute is Optional.

Mandatory attributes shall be implemented by an agent. Conditional attributes shall be implemented if the condition applies and may be implemented otherwise. Recommended attributes should be implemented by the agent. Not recommended attributes should not be implemented by the agent. Optional attributes may be implemented on an agent.

The attributes can be static, dynamic, or observational as specified in IEEE Std 11073-20601 or this standard.

In Figure 1, there may be from one to "n" instantiations of the BatteryCapacity and BatteryStatus objects. The number "n" represents the maximum number of batteries the device can support.

6.4 Types of configuration

6.4.1 General

As specified in IEEE Std 11073-20601, there are two styles of configurations available. Subclauses 6.4.2 and 6.4.3 briefly introduce standard and extended configurations.

6.4.2 Standard configuration

Standard configurations are defined in the ISO/IEEE 11073-104zz specializations (such as this standard) and are assigned a well-known identifier (Dev-Configuration-Id). The usage of a standard configuration is negotiated at association time between the agent and the manager. If the manager acknowledges that it understands and wants to operate using the configuration, then the agent can begin sending measurements immediately. If the manager does not understand the configuration, the agent provides the configuration prior to transmitting measurement information.

A total of eight standard configurations (2700-2707) are supported. The standard configuration that should be used depends on the maximum number of batteries, n, that can be configured for use by the device at any given time, whether present or not. For each standard configuration 2700 to 2707, the standard configuration defines the number of BatteryCapacity object(s) and BatteryStatus object(s) in that configuration as ("standard configuration" – 2700 + 1). Standard configurations do not contain a PM-Store. The standard configurations should be used to monitor the power status on a continuous basis. For intermittent monitoring of power status and to provide history of battery performance, an extended configuration with PM-store should be used.

In an agent where a battery is supported but not initially present, then the “BatteryStatus” object for that battery shall report this status in an initial event report. Thereafter, the agent should send no further event reports of status of that battery unless this status changes.

Agents that follow this standard and have an extended configuration with PM-Store are not required to support any standard configuration, as specified by IEEE Std 11073-20601.

6.4.3 Extended configuration

In extended configurations, the agent’s configuration is not predefined in a standard. The agent determines which objects, attributes, and values shall be used in a configuration, and assigns a configuration identifier. It is required that the agent supports a minimum of one battery for an extended configuration. When the agent associates with a manager, it negotiates an acceptable configuration. Typically, the manager does not recognize the agent’s configuration on the first connection, so the manager responds that the agent needs to send the configuration information as a configuration event report. If, however, the manager already understands the configuration, either because it was preloaded in some way or the agent had previously associated with the manager, then the manager responds that the configuration is known and no further configuration information needs to be sent.

6.5 Profiles

6.5.1 General

A profile further constrains the objects, services, and communication model of a specialization. By profiling the device specialization, the standard provides more guidance on the specific mandatory objects that shall be implemented for that profile. This standard defines two profiles for devices that support the following: Eight or fewer batteries (see Clause 11), and greater than eight batteries (see Clause 12). A PSM device shall implement one of these profiles.

6.6 Medical device system (MDS) object

6.6.1 MDS object attributes

Table 1 summarizes the attributes of the PSM MDS object. The nomenclature code to identify the MDS object class is MDC_MOC_VMS_MDS_SIMP.

Table 1—MDS object attributes

Attribute Name	Value	Qualifier
Handle	0	M
System-Type	See IEEE Std 11073-20601.	NR
System-Type-Spec-List	{MDC_DEV_SPEC_PROFILE_PSM, 1} And Profile value: {MDC_DEV_SUB_SPEC_PROFILE_EIGHT_OR_LESS_BATTERIES, 1} OR {MDC_DEV_SUB_SPEC_PROFILE_MORE_THAN_EIGHT_BATTERIES, 1}	M
System-Model	{“Manufacturer”,“Model”}	M
System-Id	Extended unique identifier (64 bits) (EUI-64)	M

Table continues

Table 1—MDS object attributes *(continued)*

Attribute Name	Value	Qualifier
Dev-Configuration-Id	Standard config: 0x0A8C (2700) Standard config: 0x0A8D (2701) Standard config: 0x0A8E (2702) Standard config: 0x0A8F (2703) Standard config: 0x0A90 (2704) Standard config: 0x0A91 (2705) Standard config: 0x0A92 (2706) Standard config: 0x0A93 (2707) Extended configs: 0x4000-0x7FFF	M
Attribute-Value-Map	See IEEE Std 11073-20601.	C
Production-Specification	See IEEE Std 11073-20601.	O
Mds-Time-Info	See IEEE Std 11073-20601.	C
Date-and-Time	See IEEE Std 11073-20601.	C
Base-Offset-Time	See IEEE Std 11073-20601.	M
Relative-Time	See IEEE Std 11073-20601.	C
HiRes-Relative-Time	See IEEE Std 11073-20601.	C
Date-and-Time-Adjustment	See IEEE Std 11073-20601.	C
Power-Status	onBattery or onMains	M
Battery-Level	See IEEE Std 11073-20601.	M
Remaining-Battery-Time	See IEEE Std 11073-20601.	M
Reg-Cert-Data-List	See IEEE Std 11073-20601.	O
Confirm-Timeout	See IEEE Std 11073-20601.	O
NOTE—See IEEE Std 11073-20601 for information on whether an attribute is observational, static, or dynamic.		

In the response to a GET MDS object command, only implemented attributes and their corresponding values are returned.

See IEEE Std 11073-20601 for descriptive explanations of the individual attributes as well as for information on attribute ID and attribute type.

The Dev-Configuration-Id attribute holds a locally unique 16-bit identifier that identifies the device configuration instance. The Device Configuration Identifiers for standard configurations are listed in Table 1. For a PSM agent with extended configuration, this identifier is chosen in the range of extended-config-start to extended-config-end (see IEEE Std 11073-20601) as shown in Table 1.

The agent sends the Dev-Configuration-Id during the Associating state (see 8.3) to identify its configuration for the duration of the association. If the manager already holds the configuration information relating to the Dev-Configuration-Id, it recognizes the Dev-Configuration-Id. Then the Configuring state (see 8.4) is skipped, and the agent and manager enter the Operating state. If the manager does not recognize the Dev-Configuration-Id, the agent and manager enter the Configuring state.

If an agent implements multiple IEEE 11073-104zz specializations, System-Type-Spec-List is a list of type/version pairs, each referencing the respective device specialization or profile and version of that specialization or profile.

6.6.2 MDS object methods

Table 2 defines the methods (actions) of the PSM agent's MDS object. These methods are invoked using the Action service. In Table 2, the Subservice type name column defines the name of the method; the Mode column defines whether the method is invoked as an unconfirmed action (i.e., roiv-cmip-action from IEEE Std 11073-20601) or a confirmed action (i.e., roiv-cmip-confirmed-action); the Subservice type (action-type) column defines the nomenclature code to use in the action-type field of an action request and response (see IEEE Std 11073-20601); the Parameters (action-info-args) column defines the associated ASN.1 data structure (see IEEE Std 11073-20601 for ASN.1 definitions) to use in the action message for the action-info-args field of the request; and the Results (action-info-args) column defines the structure to use in the action-info-args of the response.

Table 2—MDS object methods

Service	Subservice type name	Mode	Subservice type (action-type)	Parameters (action-info-args)	Results (action-info-args)
ACTION	MDS-Data-Request	Confirmed	MDC_ACT_DATA_REQUEST	DataRequest	DataResponse
ACTION	Set-Base-Offset-Time	Confirmed	MDC_ACT_SET_BO_TIME	SetBOTimeInvoke	—

— **MDS-Data-Request:**

This method allows the manager system to enable or disable measurement data transmission from the agent (see IEEE Std 11073-20601).

— **Set-Base-Offset-Time:**

This method allows the manager to set a real-time clock in the agent with the base time and offset. The agent indicates whether the Set-Base-Offset-Time command is valid using the mds-time-capab-set-clock bit in the Mds-Time-Info attribute (see IEEE Std 11073-20601.).

Since the agent supports the Base-Offset-Time-Stamp attribute, this method shall be implemented.

Agents following only this device specialization—and no others—may send event reports either by using agent-initiated or manager-initiated measurement data transmission. During the association procedure (see 8.3), data-req-mode-capab shall be set to the appropriate value for the event report style. As a result, the manager shall assume that if the PSM agent supports any of the MDS-Data-Request features, then it may use them to access the object value only if the object's Metric-Spec-Small attribute has its mss-acc-manager-initiated bit set (see IEEE Std 11073-20601 for additional information).

6.6.3 MDS object events

Table 3 defines the events that can be sent by the PSM MDS object.

Table 3—Power status MDS object events

Service	Subservice type name	Mode	Subservice type (event-type)	Parameters (event- info)	Results (event-reply-info)
EVENT REPORT	MDS-Configuration-Event	Confirmed	MDC_NOTI_CONFIG	ConfigReport	ConfigReportRsp
	MDS-Dynamic-Data-Update-Fixed	Confirmed	MDC_NOTI_SCAN_REPORT_FIXED	ScanReportInfoFixed	—
	MDS-Dynamic-Data-Update-Var	Confirmed	MDC_NOTI_SCAN_REPORT_VAR	ScanReportInfoVar	—

— **MDS-Configuration-Event:**

This event is sent by the PSM agent during the configuring procedure if the manager does not already know the configuration of the PSM agent from past associations or because the manager has not been implemented to recognize the configuration according to the PSM device specialization. The event provides static information about the supported measurement capabilities of the PSM agent.

— **MDS-Dynamic-Data-Update-Fixed:**

This event provides dynamic measurement data from the PSM agent for the PSM numeric and enumeration objects. These data are reported in the fixed format defined by the Attribute-Value-Map attribute of the object(s). The event is sent as an unsolicited message by the agent (i.e., an agent-initiated measurement data transmission). More information on unsolicited event reporting can be found in 8.5.3.

- This event should be used when "Fixed Format Event Reports" are used.

— **MDS-Dynamic-Data-Update-Var:**

This event provides dynamic measurement data from the PSM agent for the PSM numeric and enumeration objects. These data are reported using a generic attribute list variable format. The event is sent as an unsolicited message by the agent (i.e., an agent-initiated measurement data transmission). More information on unsolicited event reporting can be found in 8.5.3.

- This event should be used when "Variable Format Event Reports" are used.

NOTE—IEEE Std 11073-20601 requires that managers support all of the MDS object events listed above.

6.6.4 Other MDS services

6.6.4.1 GET service

A PSM agent shall support the GET service, which is provided by the MDS object to retrieve the values of all implemented MDS object attributes. The GET service can be invoked as soon as the PSM agent receives the Association Response and moves to the Associated state, including the Operating and Configuring sub-states.

The manager may request the MDS object attributes of the PSM agent; in which case, the manager shall send the "Remote Operation Invoke | Get" message (see roiv-cmip-get in IEEE Std 11073-20601) with the reserved MDS handle value of 0. The PSM agent shall report its MDS object attributes to the manager using the "Remote Operation Response | Get" message (see rors-cmip-get in IEEE Std 11073-20601). See Table 4 for a summary of the GET service including some message fields.

Table 4—PSM MDS object GET service

Service	Subservice type name	Mode	Subservice type	Parameters	Results
GET	<na>	<implied confirmed>	<na>	GetArgumentSimple = (obj-handle = 0), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = 0), attribute-list

See 8.5.2 for details on the procedure for getting the MDS object attributes.

6.6.4.2 SET service

The PSM specialization does not require an implementation to support the MDS object SET service.

6.7 Numeric objects

6.7.1 General

The PSM DIM for metric objects contains a numeric object that represents the battery's power; specifically the percentage of capacity and estimated operational time remaining for the batteries connected to the MDS.

6.7.2 Battery Capacity

Table 5 summarizes the attributes of the Battery Capacity numeric object. The Battery Capacity object shall be supported by a PSM agent.

Table 5—Battery capacity numeric object attributes

Attribute name	Extended configuration		Standard configurations (Dev-Configuration-Id = 0x0A8C) (Dev-Configuration-Id = 0x0A8D) (Dev-Configuration-Id = 0x0A8E) (Dev-Configuration-Id = 0x0A8F) (Dev-Configuration-Id = 0x0A90) (Dev-Configuration-Id = 0x0A91) (Dev-Configuration-Id = 0x0A92) (Dev-Configuration-Id = 0x0A93)	
	Value	Qual.	Value	Qual.
Handle	See IEEE Std 11073-20601. (See NOTE)	M	1 for 0x0A8C 1 | 3 for 0x0A8D 1 | 3 | 5 for 0x0A8E 1 | 3| 5 | 7 for 0x0A8F 1 | 3 | 5 | 7 | 9 for 0x0A90 1 | 3 | 5 | 7 | 9 | 11 | for 0x0A91 1 | 3 | 5 | 7 | 9 | 11 | 13 for 0x0A92 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 for 0x0A93	M
Type	{MDC_PART_PHD_DM | MDC_BATTERY_CAPACITY}	M	{MDC_PART_PHD_DM | MDC_BATTERY_CAPACITY}	M
Supplemental-Type	MDC_BATTERY_1 |.... | MDC_BATTERY_n	M **Static**	MDC_BATTERY_1 |.... | MDC_BATTERY_n	M **Static**
Metric-Spec-Small	See IEEE Std 11073-20601.	M	mss-avail-intermittent | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-agent-initiated	M
Metric-Structure-Small	—	NR	—	NR
Measurement-Status	—	NR	—	NR
Metric-Id	—	NR	—	NR
Metric-Id-List	—	NR	—	NR
Metric-Id-Partition	—	NR	—	NR
Unit-Code	—	NR	—	NR
Attribute-Value-Map	—	NR	—	NR
Source-Handle-Reference	—	NR	—	NR
Source-Handle-Reference-List	—	NR	—	NR
Label-String	See footnote a.	O	See footnote a.	O

Table continues

Table 5—Battery capacity numeric object attributes *(continued)*

Attribute name	Extended configuration		Standard configurations (Dev-Configuration-Id = 0x0A8C) (Dev-Configuration-Id = 0x0A8D) (Dev-Configuration-Id = 0x0A8E) (Dev-Configuration-Id = 0x0A8F) (Dev-Configuration-Id = 0x0A90) (Dev-Configuration-Id = 0x0A91) (Dev-Configuration-Id = 0x0A92) (Dev-Configuration-Id = 0x0A93)	
	Value	**Qual.**	**Value**	**Qual.**
Unit-LabelString	—	NR	—	NR
Absolute-Time-Stamp	—	NR	—	NR
Base-Offset-Time-Stamp	—	M	—	M
Relative-Time-Stamp	—	NR	—	NR
HiRes-Time-Stamp	—	NR	—	NR
Measure-Active-Period	—	NR	—	NR
Simple-Nu-Observed-Value	—	NR	—	NR
Compound-Simple-Nu-Observed-Value	—	NR	—	NR
Basic-Nu-Observed-Value	—	NR	—	NR
Compound-Basic-Nu-Observed-Value	—	NR	—	NR
Nu-Observed-Value	—	NR	—	NR
Compound-Nu-Observed-Value	See footnote b.	M	See footnote b.	M
Accuracy	—	NR	—	NR
NOTE—It is recommended that the battery capacity numeric objects should be allocated handle values (2i−1) for each battery i, where i can take on value 1…n where n is the maximum number of batteries that can be present in the agent.				

[a]Information about the location of the battery may be given in the attribute "Label-String" as a human-readable string.
[b]The Compound-Nu-Observed-Value attribute reports the estimated battery capacity in terms of remaining operating time (MDC_ATTR_TIME_BATT_REMAIN) and remaining charge (MDC_ATTR_VAL_BATT_CHARGE). Any unit can be used and is provided in the Compound-Nu-Observed-Value. Reporting on either (or both) the remaining operating time or remaining charge is required in this specialization.

The Supplemental-Type attribute specifies the battery that is being reported and shall be specified using one of the nomenclature codes MDC_BATTERY_i, where. "i" may take a value 1 to 8 for standard configurations, the value 1 to 16 for extended configuration, and is the ((value of the handle attribute)+1)/2. The value to use for the supplemental type of a battery greater than 16 is not defined in this specialization.

The MDS attribute "Remaining-Battery-Time" shall report the remaining battery time as the sum of the remaining battery time of all the batteries. The MDS attribute "Battery Capacity" shall report the remaining battery capacity as the summation of the remaining capacities of each battery divided by the summation of the current full charge capacities of each battery expressed as a percentage.

Updates to the value of battery capacity object shall be provided using the MDS-Dynamic-Data-Update-Var (see Table 3). When updates shall be sent is implementation and application-dependent; this may include scheduled transmission, transmission as a result of a given change in capacity, or on reaching specific conditions such as being fully charged or discharged. Changes of less than 5% would not normally be sent for critical care devices.

The value of the battery capacity shall be reported whenever a new association is established.

The most recently measured value of the battery capacity shall be returned whenever an MDS-Data-Request is received by an agent with the timestamp set to indicate the actual time of the measurement. Where possible, an agent should return the actual current value of the battery capacity in response to the MDS-Data-Request.

6.8 Real-time sample array objects

Real-time sample array objects are not required by this standard.

6.9 Enumeration objects

6.9.1 General

The enumeration objects listed in this clause report the status of each battery in a PSM agent. The nomenclature code to identify an enumeration object class is MDC_MOC_VMO_METRIC_ENUM. The following subclauses provide definitions of status that may be reported by the battery status enumeration object.

6.9.2 Battery status

Table 6 defines the attributes for the object that represents the detail of the battery status, including if the battery is present, active, charging, disposable, rechargeable, or faulty. This battery status object shall be supported by a PSM agent with standard configuration.

Table 6—Battery status enumeration object attributes

Attribute name	Extended configuration		Standard configurations (Dev-Configuration-Id = 0x0A8C) (Dev-Configuration-Id = 0x0A8D) (Dev-Configuration-Id = 0x0A8E) (Dev-Configuration-Id = 0x0A8F) (Dev-Configuration-Id = 0x0A90) (Dev-Configuration-Id = 0x0A91) (Dev-Configuration-Id = 0x0A92) (Dev-Configuration-Id = 0x0A93)	
	Value	**Qual.**	**Value**	**Qual.**
Handle	See IEEE Std 11073-20601. (See NOTE 1.)	M	2 for 0x0A8C 2 | 4 for 0x0A8D 2 | 4 | 6 for 0x0A8E 2 | 4 | 6 | 8 for 0x0A8F 2 | 4 | 6 | 8 | 10 for 0x0A90 2 | 4 | 6 | 8 | 10 | 12 | for 0x0A91 2 | 4 | 6 | 8 | 10 | 12 | 14 for 0x0A92	M
Type	{MDC_PART_PHD_DM | MDC_BATTERY_STATUS}	M	{MDC_PART_PHD_DM | MDC_BATTERY_STATUS}	M

Table continues

Table 6—Battery status enumeration object attributes *(continued)*

Attribute name	Extended configuration		Standard configurations (Dev-Configuration-Id = 0x0A8C) (Dev-Configuration-Id = 0x0A8D) (Dev-Configuration-Id = 0x0A8E) (Dev-Configuration-Id = 0x0A8F) (Dev-Configuration-Id = 0x0A90) (Dev-Configuration-Id = 0x0A91) (Dev-Configuration-Id = 0x0A92) (Dev-Configuration-Id = 0x0A93)	
	Value	Qual.	Value	Qual.
Supplemental-Type	MDC_BATTERY_1 |.... | MDC_BATTERY_n	M **Static**	MDC_BATTERY_1 |.... | MDC_BATTERY_n	M **Static**
Metric-Spec-Small	See IEEE Std 11073-20601.	M	mss-avail-intermittent | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-agent-initiated	M
Metric-Structure-Small	See IEEE Std 11073-20601.	O	See IEEE Std 11073-20601.	O
Measurement-Status	See IEEE Std 11073-20601.	C	See IEEE Std 11073-20601.	C
Metric-Id	—	NR	—	NR
Metric-Id-List	—	NR	—	NR
Metric-Id-Partition	—	NR	—	NR
Unit-Code	—	NR	—	NR
Attribute-Value-Map	See IEEE Std 11073-20601.	O	MDC_ATTR_ENUM_OBS_VAL_BASIC_BIT_STR, then MDC_ATTR_TIME_STAMP_BO.	M
Source-Handle-Reference	—	NR	—	NR
Source-Handle-Reference-List	—	NR	—	NR
Label-String	See footnote a.	O	See footnote a.	O
Unit-LabelString	See IEEE Std 11073-20601.	O	See IEEE Std 11073-20601.	O
Absolute-Time-Stamp	—	NR	—	NR
Base-Offset-Time-Stamp	—	M	—	M
Relative-Time-Stamp	—	NR	—	NR
HiRes-Time-Stamp	—	NR	—	NR
Measure-Active-Period	—	NR	—	NR
Enum-Observed-Value-Simple-OID	—	NR	—	NR
Enum-Observed-Value-Simple-Bit-Str	—	NR	—	NR

Table continues

Table 6—Battery status enumeration object attributes *(continued)*

Attribute name	Extended configuration		Standard configurations (Dev-Configuration-Id = 0x0A8C) (Dev-Configuration-Id = 0x0A8D) (Dev-Configuration-Id = 0x0A8E) (Dev-Configuration-Id = 0x0A8F) (Dev-Configuration-Id = 0x0A90) (Dev-Configuration-Id = 0x0A91) (Dev-Configuration-Id = 0x0A92) (Dev-Configuration-Id = 0x0A93)	
	Value	**Qual.**	**Value**	**Qual.**
Enum-Observed-Value-Basic-Bit-Str	See Table 7.	M	See Table 7.	M
Enum-Observed-Value-Simple-Str	—	NR	—	NR
Enum-Observed-Value	—	NR	—	NR
Enum-Observed-Value-Partition	—	NR	—	NR
Capability-Mask-Basic	CapabMaskBasic (see B.2)	M	CapabMaskBasic with value 0x6000 (see B.2)	M
State-Flag-Basic	StateFlagBasic (see B.3)	M	StateFlagBasic with value 0xFFC0 (see B.3)	M

NOTE 1—It is recommended that the battery status enumeration objects should be allocated handle values (2i) for each battery i, where i can take on value 1…n, where n is the maximum number of batteries that can be present in the agent, and where the status reported by battery status enumeration object (2i) corresponds to the capacity reported by battery status numeric object (2i−1).

NOTE 2—The initial value in the standard configuration assumes compliance to the mandatory minimum requirements, but Capability-Mask-Basic initial value may be updated after operating state is reached and before the first observation of the status object to indicate the actual functionality of the battery.

[a]Information about the location of the battery may be given in the attribute "Label-String" as a human-readable string.

The states that are reported by this object are defined in Table 7.

Table 7—Battery status mnemonics and states

Battery Status State	BatteryStatus Mnemonic
Agent reports battery status is undetermined	Battery-statusUndetermined
Agent reports that battery is present	Battery-present
Agent reports that battery is active	Battery-active
Agent reports that battery is charging	Battery-charging
Agent reports that battery is fully charged	Battery-fullyCharged
Agent reports that battery is disposable	Battery-disposable
Agent reports that battery is rechargeable	Battery-rechargeable
Agent reports that battery is over temperature	Battery-overTemperature

Table continues

Table 7—Battery status mnemonics and states *(continued)*

Battery Status State	BatteryStatus Mnemonic
Agent reports that battery is faulty	Battery-faulty
Agent reports that battery is incompatible	Battery-incompatible
NOTE—An agent is not required to implement all the states specified in Table 7. "Battery-present" and "Battery-active" are mandatory, whereas all others are optional.	

The Supplemental-Type attribute specifies the battery that is being reported and shall be specified using one of the nomenclature codes MDC_BATTERY_i, where "i" takes the value 1 to 8 for standard configurations, the value of 1 to 16 for extended configuration, and is the (value of the handle attribute)/2.

See IEEE Std 11073-20601 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

A battery may be in the "Battery-faulty" state as a result of any problem to or around the battery, including physical damage, sulphation, wear and tear, overcharging, short circuit, and/or internal break. If "Battery-incompatible" is true, then the other values for Battery Status and Battery Capacity objects may be unreliable. "Battery-statusUndetermined" may occur during the device or battery initialization phase or when status is pending.

Updates to the value of battery status object shall be provided using the MDS-Dynamic-Data-Update-Var or MDS-Dynamic-Data-Update-Fixed (see Table 3) when the Enum-Observed-Value-Basic-Bit-Str attribute value changes. Cases in which this would occur include when a battery is added, when a battery begins to charge, and when a battery enters a faulty state, etc.

The value of the battery status object shall be provided when a new association between agent and manager is established.

6.10 PM-store objects

6.10.1 General

PSM agents may collect observations of capacity and status on a continuous basis, whether associated or not, and store these for later retrieval. The long-term storage model for these observations is realized using PM-stores.

The data held in PM-store objects are deleted by user actions via the manager or user interface on the device and the capacity is limited only by the data storage capabilities of the agent.

6.10.2 Persistent store model

The PM-store model of this standard defines one optional PM-store object: the Battery Log for persistent storage of battery status. Note that PM-store objects are not part of the standard configurations defined in this standard.

It is recommended that a separate PM-Segment be used for the data of each type of data object (Figure 2). All objects of the same type shall store their data to a common PM-segment and identify their data by the Supplemental-Type. If the Battery Log PM-store is implemented, then the Battery Capacity and Battery Status PM-segments shall be implemented.

The attributes of the PM-segment instance are defined in Table 9.

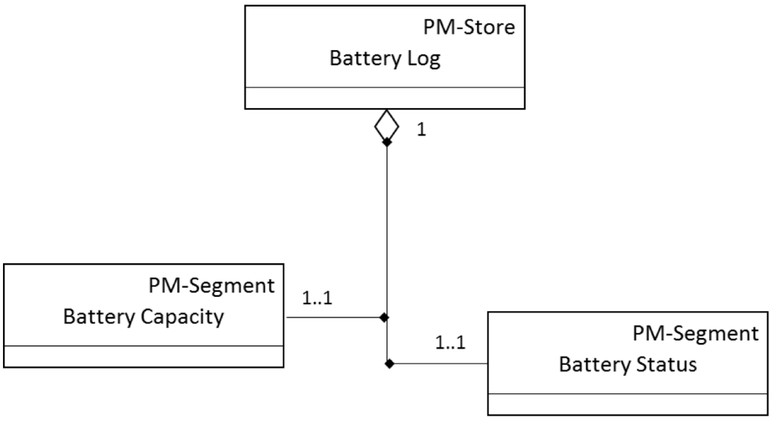

Figure 2—PSM-persistent store model

6.10.3 PM-store object attributes

Table 8 defines the attributes of the PM-store objects that shall be implemented by the agent. The nomenclature code to identify the PM-store objects is MDC_MOC_VMO_PMSTORE.

Table 8—PSM PM-store object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Handle	See IEEE Std 11073-20601.	M
PM-Store-Capab	See IEEE Std 11073-20601.	M
Store-Sample-Algorithm	See IEEE Std 11073-20601.	M
Store-Capacity-Count	See IEEE Std 11073-20601.	M
Store-Usage-Count	See IEEE Std 11073-20601.	M
Operational-State	See IEEE Std 11073-20601.	M
PM-Store-Label	See IEEE Std 11073-20601.	O
Sample-Period	See IEEE Std 11073-20601.	NR
Number-Of-Segments	See IEEE Std 11073-20601.	M
Clear-Timeout	See IEEE Std 11073-20601.	M

The PM-Store-Capab attribute shall set the following bits as indicated:

— pmsc-var-no-of-segm:
  - If the agent creates new segments either due to storing data of multiple sessions or due to time changes as described in 8.12.2.2, "Comparable time," of IEEE Std 11073-20601-2014, then pmsc-var-no-of-segm shall be set.

— pmsc-epi-seg-entries:
  - The pmsc-epi-seg-entries bit shall be set.

— pmsc-peri-seg-entries:
  - The pmsc-peri-seg-entries bit shall not be set.

The remaining bits of the PM-Store-Capab attribute are agent specific and shall be set appropriately.

6.10.4 PM-store object methods

Table 9 defines the methods of the PM-store objects.

Table 9—PM-store object methods

Service	Subservice type name	Mode	Subservice type (action-type)	Parameters (action-info-args)	Results (action-info-args)
ACTION	Clear-Segments	Confirmed	MDC_ACT_SEG_CLR	SegmSelection	
	Get-Segment-Info	Confirmed	MDC_ACT_SEG_GET_INFO	SegmSelection	SegmentInfoList
	Get-Segment-Id-List	Confirmed	MDC_ACT_SEG_GET_ID_LIST	(empty)	SegmIdList
	Trig-Segment-Data-Xfer	Confirmed	MDC_ACT_SEG_TRIG_XFER	TrigSegmDataXferReq	TrigSegmDataXferRsp

— **Clear-Segments:**

This method allows the manager to delete the data currently stored in one or more selected PM-segments. All entries in the selected PM-segments are deleted.

— **Get-Segment-Info:**

This method allows the manager to retrieve the PM-segment attributes.

— **Get-Segment-Id-List:**

This method allows the manager to retrieve a list of the instance numbers of all the PM-segments of a PM-store.

— **Trig-Segment-Data-Xfer:**

This method allows the manager to initiate the transfer of the data entries stored in the PM-segment object.

Refer to IEEE Std 11073-20601 for details.

6.10.5 PM-store object events

Table 10 defines the events sent by the PM-store objects.

Table 10—PM-store object events

Service	Subservice type name	Mode	Subservice type (event-type)	Parameters (event-info)	Results (event-reply-info)
EVENT REPORT	Segment-Data-Event	Confirmed	MDC_NOTI_SEGMENT_DATA	SegmentDataEvent	SegmentDataResult

— **Segment-Data-Event:**

This event allows the agent to send the data entries stored in the PM-segment object. This event is triggered by the manager using the Trig-Segment-Data-Xfer action. Refer to IEEE Std 11073-20601 for details.

6.10.6 PM-store object services

6.10.6.1 GET service

The GET service shall be provided by an agent implementing PM-store objects. This service shall be available only while the agent is in the Operating state. Refer to IEEE Std 11073-20601 for details.

6.10.6.2 SET service

There are currently no SET services defined for PM-store objects in this standard.

6.10.7 PM-segment objects

Table 11 defines the attributes of the PM-segment object contained in the PM-store object that represents the stored PSM observations. The nomenclature code to identify the PM-segment object class is MDC_MOC_PM_SEGMENT.

Table 11—Common PM-segment object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Instance-Number	See IEEE Std 11073-20601.	M
PM-Segment-Entry-Map	See IEEE Std 11073-20601.	M
PM-Seg-Person-Id	See IEEE Std 11073-20601.	C
Operational-State	See IEEE Std 11073-20601.	M
Sample-Period	See IEEE Std 11073-20601.	C
Segment-Label	See IEEE Std 11073-20601.	O
Segment-Start-Abs-Time	See IEEE Std 11073-20601.	C
Segment-End-Abs-Time	See IEEE Std 11073-20601.	C
Date-and-Time-Adjustment	See IEEE Std 11073-20601.	C
Segment-Start-BO-Time	See IEEE Std 11073-20601.	M
Segment-End-BO-Time	See IEEE Std 11073-20601.	M
Segment-Usage-Count	See IEEE Std 11073-20601.	M
Segment-Statistics	See IEEE Std 11073-20601.	O
Fixed-Segment-Data	Segment data transferred as an array of entries in a format as specified in the PM-Segment-Entry-Map attribute.	M
Confirm-Timeout	See IEEE Std 11073-20601.	O
Transfer-Timeout	See IEEE Std 11073-20601.	M
NOTE—See IEEE Std 11073-20601 for information on whether an attribute is observational, static, or dynamic.		

6.11 Scanner objects

Scanner objects are not required by this standard.

6.12 Class extension objects

In this standard, no class extension objects are defined with respect to IEEE Std 11073-20601.

6.13 PSM information model extensibility rules

The PSM DIM of this standard may be extended by including elements defined in IEEE Std 11073-20601 as well as vendor-specific elements. Any object or attribute extensions implemented should follow the guidelines of this standard as closely as possible.

A PSM agent having a configuration with extensions beyond the standard configuration, as specified in this standard, shall use a configuration ID in the range of IDs reserved for extended configurations (see IEEE Std 11073-20601).

7. PSM service model

7.1 General

The service model defines the conceptual mechanisms for data exchange services. These services are mapped to messages that are exchanged between the agent and the manager. Protocol messages within the ISO/IEEE 11073 series of standards are defined in ASN.1. See IEEE Std 11073-20601 for a detailed description of the PHD service model. Subclauses 7.2 and 7.3 define the specifics of object access and event reporting services for a PSM agent according to this standard.

7.2 Object access services

The object access services of IEEE Std 11073-20601 are used to access the objects defined in the DIM of the PSM device.

The following generic object access services are supported by a PSM agent according to this standard:

— GET service: used by the manager to retrieve the values of the agent MDS attributes or PM-store attributes. The list of PSM attributes is given in 6.6.1 for MDS object and 6.10.3 for the PM-store object.

— SET service: used by the manager to set the values of the agent object attributes. No settable attributes are defined for a PSM agent according to this standard.

— Event report service: used by the agent to send configuration reports and measurement data to the manager. The list of event reports for the PSM device specialization is given in 6.6.3 for the MDS object and in 6.10.5 for the PM-store object.

— Action service: used by the manager to invoke actions (or methods) supported by the agent. The list of MDS objects actions is given in 6.6.2. The list of PM-store object actions is given in 6.10.4.

Table 12 summarizes the object access services described in this standard.

7.3 Object access event report services

The event report service (see Table 12) is used by the agent to report its information (e.g., measurements). Event reports in this standard are a property of the MDS object only. The event reports used in this standard are defined in IEEE Std 11073-20601.

The following conditions apply for a PSM agent according to this standard:

— MDS event reports shall be used in confirmed mode.

— Agent initiated mode shall be supported for measurement data transmission.

— Persistently stored metric mode may be supported for measurement data transmission.

Table 12—PSM object access services

Service	Subservice type name	Mode	Subservice type	Parameters	Result	Remarks
GET	<na>	<implied Confirmed>	<na>	GetArgumentSimple = (obj-handle = 0), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = 0), attribute-list	Allows the manager to retrieve the value of attributes of the MDS object in the agent.
	<na>	<implied Confirmed>	<na>	GetArgumentSimple = (obj-handle = *handle of PM-store object*), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = *handle of PM-store object*), attribute-list	Allows the manager to retrieve the values of attributes of a PM-store object in the agent.
EVENT REPORT	MDS-Configuration-Event	Confirmed	MDC_NOTI_CONFIG	ConfigReport	ConfigReportRsp	Configuration Report to inform manager of the configuration of the agent.
	MDS-Dynamic-Data-Update-Var	Confirmed	MDC_NOTI_SCAN_REPORT_VAR	ScanReportInfoVar	—	Data Report to provide dynamic data to manager for some or all of the agent's objects in variable format.
	MDS-Dynamic-Data-Update-Fixed	Confirmed	MDC_NOTI_SCAN_REPORT_FIXED	ScanReportInfoFixed	—	Data Report to provide dynamic data to manager for some or all of the agent's objects in fixed format.

Table continues

Table 12—PSM object access services *(continued)*

Service	Subservice type name	Mode	Subservice type	Parameters	Result	Remarks
ACTION	Set-Base-Offset-Time	Confirmed	MDC_ACT_SET_BO_TIME	SetBOTimeInvoke	—	Manager method to invoke the agent to set time in base offset time format to requested value.
	Clear-Segments	Confirmed	MDC_ACT_SEG_CLR	SegmSelection	—	Allows the manager to delete data stored in selected PM-segments in the agent.
	Get-Segment-Info	Confirmed	MDC_ACT_SEG_GET_INFO	SegmSelection	SegmentInfoList	Allows the manager to retrieve the value of PM-segment attributes of one or more PM-segments in the agent.
	Get-Segment-Id-List	Confirmed	MDC_ACT_SEG_GET_ID_LIST	(empty)	SegmIdList	Allows the manager to retrieve a list of the instance numbers of all the PM-segments of a PM-store.
	Trig-Segment-Data-Xfer	Confirmed	MDC_ACT_SEG_TRIG_XFER	TrigSegmDataXferReq	TrigSegmDataXferRsp	Allows the manager to start the transfer of the Fixed-Segment-Data attribute of a PM-segment in the agent.
	DataRequest	Confirmed	MDC_ACT_DATA_REQUEST	DataRequest(start, single)	DataReqResult	Allows the manager to instruct the agent to send a single measurement or to start or stop sending measurement data.

— If an agent is to provide the current state of charge of a battery on demand, then it shall support manager initiated in an extended configuration.
  - Manager may use DataRequest to initiate transmission of metric data in single response mode.
  - Agent shall report the current values of the metric objects as indicated by the data-req-scope-* bits.

— An agent having a standard configuration is only intended to provide updates on the status and salient state of charge of the battery. It therefore shall not support manager initiated data.

— Event reports shall be used.
  - When an attribute value for "Battery status" object changes.
  - When an attribute value for "Battery capacity" object changes and when the attribute "Compound-Nu-Observed-Value" changes by a "given amount" where "given amount" is a value appropriate for the application.
  - When a new association between agent and manager is established.

8. PSM communication model

8.1 Overview

This clause describes the general communication model and procedures of the PSM agent as defined in IEEE Std 11073-20601. Therefore, the respective parts of IEEE Std 11073-20601 are not reproduced; rather, the specific choices and restrictions with respect to optional elements (e.g., objects, attributes, and actions) and specific extensions (e.g., nomenclature terms) are specified.

For an illustrative overview of the various message transactions during a typical measurement session, see the sequence diagram for the example use case in Annex D and the corresponding protocol data unit (PDU) examples in Annex E.

8.2 Communication characteristics

In this subclause, limits on the size of an application protocol data unit (APDU) transmitted or to be received by a PSM agent are defined. Small limits allow for simple implementations in terms of low cost and complexity.

A PSM agent implementing only this device specialization shall not transmit any APDU larger than N_{tx} and shall be capable of receiving any APDU up to a size of N_{rx}. For this standard, N_{tx} shall be calculated according to the below equation:

$$N_{tx}(i) = 28 + (102 + S) \times i \tag{1}$$

Where "i" is the number of batteries, "S" is the maximally supported OCTET STRING.length for the Label-String attribute of the battery capacity object, 28 is the number of octets in the fixed overhead of the APDU, and 102 is the number of octets to represent fields for one battery in the APDU minus "S." N_{tx} with S = 12 is 142 octets for implementations supporting one battery and is 940 octets for implementations supporting eight batteries. For this standard, N_{rx} is 46 octets.

For a PSM agent implementing functions from other device specializations, an upper bound estimation of the APDU sizes brings the following: An agent shall not transmit any APDU larger than the sum of N_{tx} of all the device specializations implemented and shall be capable of receiving any APDU up to the sum of N_{rx} of all the device specializations implemented. If these numbers are higher than the maximum size determined in IEEE Std 11073-20601, the latter shall be applied.

In case the APDU size limit does not allow for the inclusion of a certain amount of multiple pending measurements at the agent, they shall be sent using multiple event reports. See 8.5.3 for the maximum number of measurements allowed for inclusion in a single event report.

8.3 Association procedure

8.3.1 General

Unless otherwise stated, the association procedure for a PSM agent and manager according to this standard shall be pursued as specified in IEEE Std 11073-20601.

8.3.2 Agent procedure—Association request

In the association request sent by the agent to the manager:

a) The version of the association procedure used by the agent shall be set to assoc-version1 (i.e., *assoc-version* = 0x80000000).

b) The DataProtoList structure element of the data protocol identifier shall be set to data-proto-id-20601 (i.e., *data-proto-id* = 0x5079).

c) The *data-proto-info* field shall contain a PhdAssociationInformation structure that shall contain the following parameter values:

   1) The agent shall support protocol-version2 and protocol-version3. Support for other versions shall be indicated by setting additional bits. When protocols higher than protocol-version3 are used, the agent shall continue to use only features as specified in this standard. Protocol versions lower than 2 are not supported.

   2) At a minimum, the MDERs shall be supported (i.e., encoding-rules = 0x8000).

   3) The version of the nomenclature used shall be set to nom-version1 (i.e., nomenclature-version = 0x80000000).

   4) The field functional-units may have the test association bits set but shall not have any other bits set.

   5) The field system-type shall be set to sys-type-agent (i.e., system-type = 0x00800000).

   6) The system-id field shall be set to the value of the System-Id attribute of the MDS object of the agent. The manager may use this field to determine the identity of PSM with which it is associating and, optionally, to implement a simple access restriction policy.

   7) The dev-config-id field shall be set to the value of the Dev-Configuration-Id attribute of the MDS object of the agent.

   8) If the agent supports only the PSM specialization, then the field indicating the data request modes (data-req-mode-capab) supported by the PSM agent shall be set to data-req-supp-init-agent.

   9) If the agent supports only the PSM specialization, then data-req-init-manager-count shall be set to zero for standard configuration, and data-req-init-agent-count shall be set to 1.

8.3.3 Manager procedure—association response

In the association response message sent by the manager, the following shall apply:

a) The *result* field shall be set to an appropriate response from those defined in IEEE Std 11073-20601. For example, if all other conditions of the association protocol are satisfied, *accepted* is returned when the manager recognizes the *dev-config-id* of the agent and *accepted-unknown-config* otherwise.

1) In the DataProtoList structure element, the data protocol identifier shall be set to data-proto-id-20601 (i.e., *data-proto-id* = 0x5079).

b) The *data-proto-info* field shall be filled in with a PhdAssociationInformation structure that shall contain the following parameter values:

1) The manager following this specialization shall support protocol-version2 or protocol-version3.

2) The manager shall respond with a single selected encoding rule that is supported by both agent and manager. The manager shall support at least the MDERs.

3) The version of the nomenclature used shall be set to nom-version1 (i.e., *nomenclature-version* = 0x80000000).

4) The field *functional-units* shall have all bits reset except for those relating to a test association.

5) The field *system-type* shall be set to sys-type-manager (i.e., *system-type* = 0x80000000).

6) The *system-id* field shall contain the unique system ID of the manager, which shall be a valid EUI-64 type identifier.

7) The field *dev-config-id* shall be manager-config-response (0).

8) The field *data-req-mode-capab* shall be 0.

9) If the agent supports only the PSM specialization, data-req-init-manager-count shall be 0 for standard configuration and data-req-init-agent-count shall be 1.

8.4 Configuring procedure

8.4.1 General

The agent enters the Configuring state if it receives an association response of accepted-unknown-config. In this case, the configuration procedure as specified in IEEE Std 11073-20601 shall be followed. Subclause 8.4.2 specifies the configuration notification and response messages for PSM agent with standard configuration ID 2700 (0x0A8C). Normally, a manager would already know the standard configuration. However, for the purposes of this example, it does not.

8.4.2 PSM—Standard configuration

8.4.2.1 Agent procedure

The agent performs the configuration procedure using a “Remote Operation Invoke | Confirmed Event Report” message with an MDC_NOTI_CONFIG event to send its configuration to the manager (see IEEE Std 11073-20601). The ConfigReport structure is used for the *event-info* field (see Table 4). For a PSM agent with standard configuration ID 2700 (0x0A8C) the format and contents of the configuration notification message are as follows:

0xE7 0x00	APDU CHOICE Type (PrstApdu)
0x00 0x8A	CHOICE.length = 138
0x00 0x88	OCTET STRING.length = 136
0x00 0x01	invoke-id:1
0x01 0x01	CHOICE(Remote Operation Invoke | Confirmed Event Report)
0x00 0x82	CHOICE.length = 130
0x00 0x00	obj-handle = 0 (MDS object)
0xFF 0xFF 0xFF 0xFF	event-time = 0xFFFFFFFF
0x0D 0x1C	event-type = MDC_NOTI_CONFIG

0x00 0x78	event-info.length = 120
0x0A 0x8C	config-report-id (Dev-Configuration-Id value for standard configuration 2700)
0x00 0x02	config-obj-list.count = 2 metric objects will be “announced”
0x00 0x70	config-obj-list.length = 112
0x00 0x06	obj-class = MDC_MOC_VMO_METRIC_NU
0x00 0x01	obj-handle = 1 (→ 1st object is BatteryCapacity)
0x00 0x04	attributes.count = 4
0x00 0x2C	attributes.length = 44
0x09 0x2F	attribute-id = MDC_ATTR_ID_TYPE
0x00 0x04	attribute-value.length = 4
0x00 0x80 0x74 0xCC	PHD_DM partition | MDC_BATTERY_CAPACITY
0x0A 0x61	attribute-id = MDC_ATTR_SUPPLEMENTAL_TYPES
0x00 0x08	attribute-value.length = 8
0x00 0x01 0x00 0x04	
0x00 0x80 0x74 0xD8	1 element in list; 2 octets in list; new MDC code for Battery 1
0x0A 0x46	attribute-id = MDC_ATTR_METRIC_SPEC_SMALL
0x00 0x02	attribute-value.length = 2
0xB0 0x40	mss-acc-agent-initiated
0x09 0x27	attribute-id = MDC_ATTR_ID_LABEL_STRING
0x00 0x0E	attribute-value.length = 14
0x00 0x0C 0x74 0x68	
0x65 0x20 0x62 0x61	
0x74 0x74 0x65 0x72	
0x79 0x00	OCTET STRING.length = 12 “the battery”
0x00 0x05	obj-class = MDC_MOC_VMO_METRIC_ENUM
0x00 0x02	obj-handle = 2(→ 2nd object is BatteryStatus)
0x00 0x06	attributes.count = 6
0x00 0x2A	attributes.length = 42
0x09 0x2F	attribute-id = MDC_ATTR_ID_TYPE
0x00 0x04	attribute-value.length = 4
0x00 0x80 0x33 0x33	PHD_DM partition | MDC_BATTERY_STATUS
0x0A 0x46	attribute-id = MDC_ATTR_METRIC_SPEC_SMALL
0x00 0x02	attribute-value.length = 2
0xB0 0x40	mss-acc-agent-initiated
0x0A 0x61	attribute-id = MDC_ATTR_SUPPLEMENTAL_TYPES
0x00 0x08	attribute-value.length = 8
0x00 0x01 0x00 0x04	
0x00 0x80 0x74 0xD8	1 element in list; 2 octets in list; new MDC code for Battery 1
0x0A 0x55	attribute-id = MDC_ATTR_ATTRIBUTE_VAL_MAP
0x00 0x0C	attribute-value.length = 12
0x00 0x02	AttrValMap.count = 2
0x00 0x08	AttrValMap.length = 8
0x0A 0x66 0x00 0x02	MDC_ATTR_ENUM_OBS_VAL_BASIC_BIT_STR |value length = 2
0x0A 0x82 0x00 0x08	MDC_ATTR_TIME_STAMP_BO | value length = 8

0x0A 0x93	attribute-id = MDC_ATTR_ENUM_STATE_FLAG_BASIC
0x00 0x02	attribute-value.length = 2
0xFF 0xC0	State-Flag-Basic = 11 11 11 11 11 00 00 00
	(agent reports defined BatteryStatus bits are states)
0x0A 0x91	attribute-id = MDC_ATTR_ENUM_CAPABILITY_MASK_BASIC
0x00 0x02	attribute-value.length = 2
0x60 0x00	Capability-Mask-Basic = 01 10 00 00 00 00 00 00
	(agent reports Battery-present and Battery-active are supported)

8.4.2.2 Manager procedure

The manager shall respond to a configuration notification message using a "Remote Operation Response | Confirmed Event Report" data message with an MDC_NOTI_CONFIG event using the ConfigReportRsp structure for the *event-info* field (see Table 4). As a response to the standard configuration notification message in 8.4.2.1, the format and contents of the manager's configuration notification response message are as follows:

0xE7 0x00	APDU CHOICE Type (PrstApdu)
0x00 0x16	CHOICE.length = 22
0x00 0x14	OCTET STRING.length = 20
0x00 0x01	invoke-id (differentiates this message from any other outstanding)
0x02 0x01	CHOICE (Remote Operation Response | Confirmed Event Report)
0x00 0x0E	CHOICE.length = 14
0x00 0x00	obj-handle = 0 (MDS object)
0xFF 0xFF 0xFF 0xFF	currentTime
0x0D 0x1C	event-type = MDC_NOTI_CONFIG
0x00 0x04	event-reply-info.length = 4
0x0A 0x8C	ConfigReportRsp.config-report-id = 2700
0x00 0x00	ConfigReportRsp.config-result = accepted-config

8.5 Operating procedure

8.5.1 General

Measurement data and status information are communicated from the PSM agent during the Operating state. If not stated otherwise, the operating procedure for a PSM agent of this standard shall be as specified in IEEE Std 11073-20601.

8.5.2 GET PSM MDS attributes

See Table 4 for a summary of the GET service.

Refer to the respective standard for the details of getting the MDS attributes of a PSM for protocol-version2 and protocol-version3.

8.5.3 Measurement data transmission

See Table 3 and Table 8 for a summary of the event report services available for measurement data transfer.

After associating, a PSM agent shall report the current status of all of its batteries using all the mandatory metric objects defined in the DIM. Furthermore, the agent shall (immediately) generate an event report when the status of any of the batteries changes (as reported through the Battery status object) (agent-initiated or as part of manager-initiated sequence).

An event report shall be generated when the remaining capacity of the batteries drops by a “given amount,” where “given amount” is a sensible value appropriate for the application.

For PM-store transfers, the manager inspects the available segments and retrieves all applicable data.

8.6 Time synchronization

Time synchronization between the PSM agent and a manager may be used to coordinate the clocks used when reporting physiological events. Note that the mechanism for synchronizing an agent to a manager is outside the scope of this standard. If time synchronization is used, then this shall be reported in the Mds-Time-Info attribute of the MDS object.

9. Test associations

The Test association provides a manufacturer the mechanism to test or demonstrate features of a product in a comprehensive manner. This clause defines the behavior of the PSM agent during a test association. Support for test association is optional.

9.1 Behavior with standard configuration

To facilitate automated standardized test processes, a PSM that presents the standard configuration ID and enters into a test association should be able to simulate battery status and battery capacity events, without an operator needing to interact with the actual batteries.

After the agent enters the operating state, it simulates the reception of an event from the batteries representing a PSM measurement. To the extent possible, this event is observed only by those components of the agent that understand the test association. When the event is propagated into a numeric object, the testdata bit of the measurement-status attribute shall be set if the measurement-status attribute is supported. An agent is not required to use the measurement-status attribute if it would not normally do so outside of a test association.

The agent should send event reports for all simulated battery events within 30 s of entering the Operating state. The test association is terminated in a manner consistent with the agent’s normal behavior for terminating an association.

9.2 Behavior with extended configurations

This specification does not define a test association that uses an extended configuration.

10. Conformance

10.1 Applicability

This standard shall be used in conjunction with IEEE Std 11073-20601.

An implementation or a system can conform to the following elements of this standard:

— DIM class hierarchy and object definitions (object attributes, notifications, methods, and data type definitions)
— Nomenclature code values
— Protocol and service models
— Communication service model (association and configuration)

10.2 Conformance specification

This standard offers levels of conformance with respect to strict adherence to the standard device and the use of extensions for the following:

— Information model of a specific device
— Use of attributes, value ranges, and access methods

A vendor shall specify the level of conformance for an implementation based on this standard and provide details of the way in which the definitions of this standard and any extensions are applied.

Specifications shall be provided in the form of a set of implementation conformance statements (ICS) as detailed in 10.4.

This standard is used in conjunction with IEEE Std 11073-20601. It is recommended that the ICS for this standard be created first so that the ICS created for IEEE Std 11073-20601 may refer to the ICS for this standard where applicable.

10.3 Levels of conformance

10.3.1 General

This standard defines the following levels of conformance.

10.3.2 Conformance level 1: Base conformance

The application uses elements of the information, service, and communication models (object hierarchy, actions, event reports, and data type definitions) and the nomenclature scheme defined in IEEE Std 11073-20601 and ISO/IEEE 11073-104zz standards. All mandatory features defined in the object definition tables and in the ICS tables are implemented. Furthermore, any conditional, recommended, or optional features that are implemented shall follow the requirements in IEEE Std 11073-20601 and ISO/IEEE 11073-104zz documents.

10.3.3 Conformance level 2: Extended nomenclature (ASN.1 and/or ISO/IEEE 11073-10101a:2015 [B6])

Conformance level 2 meets conformance level 1 but also uses or adds extensions in at least one of the information, service, communication, or nomenclature models. Extensions to nomenclature codes shall conform to the ISO/IEEE 11073-10101a:2015 [B6] framework and lie within the private nomenclature extension range (0xF000 – 0xFFFF).

Extensions to the information or service models shall be fully defined using ASN.1 where appropriate and have their behavior fully described following the framework of the IEEE Std 11073-20601 and/or ISO/IEEE 11073-20101a:2015 [B9]. All extensions shall be specified and include reference to the definition for the extension, or where no publicly available reference is available, the definition of the extension should be appended to the conformance statement.

10.4 Implementation conformance statements (ICS)

10.4.1 General format

The ICSs are provided as an overall conformance statement document that comprises a set of tables in the form given by the templates in the following clauses.

Each ICS table has the following columns:

Index	Feature	Reference	Req/Status	Support	Comment

The table column headings have the following meaning:

— Index: An identifier (e.g., a tag) of a specific feature.

— Feature: Briefly describes the characteristic for which a conformance statement is being made.

— Reference: To the clause/paragraph within this document or to an external source for the definition of the feature (may be empty).

— Req/Status: Specifies the conformance requirement (e.g., mandatory or recommended); in some cases, this standard does not specify conformance requirements but requests the status of a particular feature be provided.

— Support: Specifies the presence or absence of a feature and any description of the characteristics of the feature in the implementation. This column is to be filled out by the implementer.

— Comment: Contains any additional information on the feature. This column is to be filled out by the implementer.

Subclauses 10.4.2 to 10.4.6 specify the format of the specific ICS tables.

10.4.2 General implementation conformance statement

The general ICS specifies the versions/revisions that are supported by the implementation and high-level system behavior.

Table 13 shows the general ICSs.

10.4.3 DIM MOC implementation conformance statement

The DIM MOC ICS defines which objects are implemented. Information on each object shall be provided as a separate row in the template of Table 14.

Table 14—Template for DIM MOC ICS table

Index	Feature	Reference	Req./Status	Support	Comment
MOC-*n*	Object description	Reference to the clause in the standard or other location where the object is defined.	Implemented	Specify restrictions (e.g., max. number of supported instances)	

The *n* in the Index column should be the object handle for implementations that have predefined objects. Otherwise the Index column shall simply be a unique number (1..*m*).

Table 13—11073-10427 general ICS table

Index[a]	Feature	Reference	Req./Status	Support	Comment
GEN 11073-10427-1	Implementation Description	—	Identification of the device/ application. Description of functionality.		
GEN 11073-10427-2	Standards followed and their revisions	(standard documents)	(set of existing revisions)	(set of support-ed revision)	
GEN 11073-10427-3	Nomenclature document used and revision	(standard documents)	(set of existing revisions)	(set of support-ed revisions)	
GEN 11073-10427-4	Conformance Adherence—Level 1	See 10.3.3.	Base conformance declaration that device meets the following IEEE Std 11073-10427 conformance requirements: a) All mandatory requirements shall be implemented. b) If implemented, conditional, recommended, and op-tional requirements shall conform to standard.	Yes/No (No is not expected as No implies that the implementation is non-conformant)	
GEN 11073-10427-5	Conformance Adherence—Level 2	See 6.3.	In addition to GEN 11073-10427-4, if the device implements extensions and/ or additions, they shall conform to nomenclature codes from ASN.1 and/or 10101 framework. These extensions should also be defined in ICS tables pointing toward their reference.	Yes/No	
GEN 11073-10427-6	Object Containment Tree	See 6.3.	Provide Object Containment Diagram showing relations between object instances used by the application. A conforming imple-mentation uses only object relations as defined in the DIM.		
GEN 11073-10427-7	Nomenclature document used and revision	(standard documents)	(set of existing revisions)	(set of support-ed revision)	
GEN 11073-10427-8	Data Structure Encoding	—	—	Description of en-coding method(s) for ASN.1 data structures	

Table continues

Table 13—11073-10427 general ICS table *(continued)*

Index[a]	Feature	Reference	Req./Status	Support	Comment
GEN 11073-10427-9	Use of Private Objects	—	Does the implementation use objects that are not defined in the DIM?	Yes/No (If yes, explain in Table 12—Template for DIM MOC ICS table)	
GEN 11073-10427-10	Use of Private Nomenclature Extensions	—	Does the implementation use private extensions to the nomenclature (i.e., 0xF000-0xFFFF codes from ISO/IEEE 11073-10101a:2015 [B6])? Private Nomenclature extensions are *only* allowed if the standard nomenclature does not include the specific terms required by the application.	Yes/No (If yes: explain in Table 15—Template for MOC nomenclature ICS table)	
GEN 11073-10427-11	11073-20601 Conformance		Provide the conformance report required by IEEE Std 11073-20601.		

[a]The prefix GEN11073-10427- is used for the index in the general ICS table.

All private objects should be specified and include either a reference to the definition for the object, or where no publicly available reference is available, the definition of the object should be appended to the conformance statement.

The Support column should indicate any restrictions for the object implementation.

An object containment diagram (class instance diagram) should be provided as part of the DIM MOC ICS.

10.4.4 MOC attribute ICS

The MOC attribute ICS defines which attributes, including any inherited attributes, are used/supported in each object of an implementation. Information on each attribute of an object shall be provided as a separate row in the template of Table 15. A separate MOC attribute ICS shall be provided for each object.

Table 15—Template for MOC attribute ICS table

Index	Feature	Reference	Req./Status	Support	Comment
ATTR-*n*-*x*	Attribute Name. Extended attributes shall include the Attribute ID also.	Fill in the reference to the ASN.1 structure if the attribute is not defined in this standard.	M = Mandatory / C = Conditional / R = Recommended / O = Optional (as per definition in Attribute Definition Tables).	Implemented? Yes/No Static/Dynamic Specify restrictions (e.g., value ranges). Describe how attribute is accessed (e.g., Get, Set, sent in config event report, sent in a data event report). Describe any specific restrictions.	

The Support column shall specify whether the attribute is implemented; for extension attributes, whether the attribute value is static, dynamic or observational; any value ranges; restrictions on attribute access or availability; and any other information.

The *n* in the Index column refers to the ID of the managed object for which the table is supplied (i.e., the index of the managed object as specified in the MOC ICS). There is one separate table for each supported managed object.

The *x* in the Index column is a unique serial number (1..*m*).

10.4.5 MOC notification implementation conformance statement

The MOC notification ICS specifies all implemented notifications (typically in form of the event report service) that are emitted by the agent. Table 16 provides a template for use. One table has to be provided for each object that supports special object notifications. One row of the table shall be used for each notification.

Table 16—Template for MOC notification ICS table

Index	Feature	Reference	Req./Status	Support	Comment
NOTI-*n*-*x*	Notification Name and Notification ID	Reference to the clause in the standard or other location where the event is defined.		The Support column shall specify how the notification is sent and any restrictions.	

The *n* in the Index column refers to the ID of the managed object for which the table is supplied (i.e., the index of the managed object as specified in the MOC ICS). There is one separate table for each managed object that supports specific object notifications (i.e., events).

The *x* in the Index column is a unique serial number (1..*m*).

All private notifications should be specified and include reference to the definition for the notification. Where no publicly available reference is available, the definition of the notification should be appended to the conformance statement.

10.4.6 MOC nomenclature conformance statement

The MOC nomenclature ICS specifies all nonstandard nomenclature codes that are utilized by the agent. Table 17 provides a template for use. One row of the table is to be used for each nomenclature element.

Table 17—Template for MOC nomenclature ICS table

Index	Feature	Reference	Req./Status	Support	Comment
NOME-*n*	Nomenclature Name and Nomenclature value	Reference to the clause in the standard or other location where the nomenclature is defined or used.		Describe how the nomenclature is used. Describe any specific restrictions.	

The *n* in the Index column is a unique serial number (1..*m*).

11. Simple PSM Profile: Devices that can support one-to-eight batteries

11.1 General concepts

This clause presents the general concepts of a profile for a device that can support a maximum of one-to-eight batteries (referred to in this clause as a “one-to-eight battery device”).

11.2 One-to-eight batteries DIM

11.2.1 Object instance diagram

The object instance diagram of the simple one-to-eight battery device information model is shown in Figure 3, where “n” represents the maximum number of batteries the device can support. For this profile “n” can have a value from 1..8.

11.2.2 Types of configuration

The one-to-eight battery device supports the standard configurations 2700 (0x0A8C) to 2707 (0x0A93).

When the agent supports PM-store or other specializations within the same configuration then extended configurations shall be used.

11.2.3 MDS object

11.2.3.1 Specific MDS object attributes

Table 18 provides the System-Type-Spec-List and Dev-Configuration-Id of the one-to-eight battery device. All other MDS object attributes are provided in Table 1.

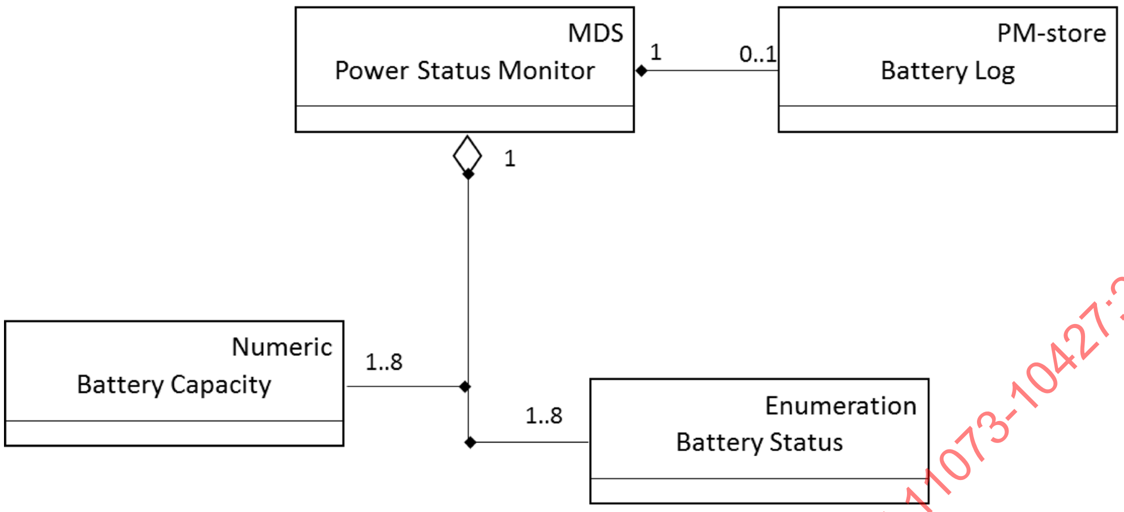

Figure 3—One-to-eight battery device—DIM

Table 18—Specific MDS object attributes

Attribute name	Value	Qual.
System-Type-Spec-List	{MDC_DEV_SPEC_PROFILE_PSM, 1} And Profile value {MDC_DEV_SUB_SPEC_PROFILE_EIGHT_OR_LESS_BATTERIES, 1}	M
Dev-Configuration-Id	Standard config: 0x0A8C (2700) | 0x0A8D (2701) | 0x0A8E (2702) | 0x0A8F (2703) | 0x0A90 (2704) | 0x0A91 (2705) | 0x0A92 (2706) | 0x0A93 (2707)	M
NOTE—See IEEE Std 11073-20601 for information on whether an attribute is observational, static or dynamic.		

The Dev-Configuration-Id attribute holds a locally unique 16-bit identifier that identifies the device configuration. For a one-to-eight battery agent, this identifier is 0x0A8C (2700) | .. | 0x0A93 (2707) as shown in Table 18 depending on how many batteries are supported.

11.2.3.2 MDS object methods

See 6.6.2.

11.2.3.3 MDS object events

See 6.6.3.

11.2.3.4 Other MDS service

See 6.6.4.

11.2.4 Numeric objects

The one-to-eight battery device DIM (see Figure 3) contains n mandatory numeric objects of type BatteryCapacity, each of which represents the capacity for the i-th battery, where “i” takes on a value from 1-to-n, where “n” is the number of batteries the device supports.

11.2.5 Real-time sample array objects

Real-time sample array objects are not supported by this profile.

11.2.6 Enumeration objects

The one-to-eight battery device DIM (see Figure 3) contains n mandatory enumeration objects of type BatteryStatus, each of which represents the status for the i-th battery, where “i” takes on a value from 1-to-n, where “n” is the number of batteries the device supports.

11.2.7 PM-store objects

The one-to-eight battery device DIM (Figure 3—One-to-eight battery device—DIM) contains zero-to-one PM-store objects.

11.2.8 Scanner objects

Scanner objects are not supported by this profile.

11.2.9 Communications characteristic

In this subclause, limits on the size of an APDU transmitted or to be received by a one-to-eight battery device are defined. Small limits allow for simple implementations in terms of low cost and complexity. A one-to-eight agent implementing only this device specialization shall not transmit any APDU larger than N_{tx}, and shall be capable of receiving any APDU up to a size of N_{rx}. For this standard, N_{tx} shall be 940 octets and N_{rx} shall be 46 octets.

For a one-to-eight battery device agent implementing functions from other device specializations, an upper bound estimation of the APDU sizes brings the following: An agent shall not transmit any APDU larger than the sum of N_{tx} of all the device specializations implemented and shall be capable of receiving any APDU up to the sum of N_{rx} of all the device specializations implemented.

If these numbers are higher than the maximum size determined in IEEE Std 11073-20601, then the latter shall be applied. In case the APDU size limit does not allow for the inclusion of a certain amount of multiple pending measurements at the agent, they shall be sent using multiple event reports. See 8.5.3 for the maximum number of measurements allowed for inclusion in a single event report.

12. Advanced PSM profile: device that can support more than eight batteries

12.1 General concepts

This clause presents the general concepts of a profile for a device that can support more than eight batteries (referred to in this clause as a “more-than-eight battery device”).

12.2 More-than-eight batteries DIM

12.2.1 Object instance diagram

The object instance diagram of the more-than-eight battery device information model is shown in Figure 4, where "n" represented the maximum number of batteries the device can support. For this profile "n" has a value of 9 or higher.

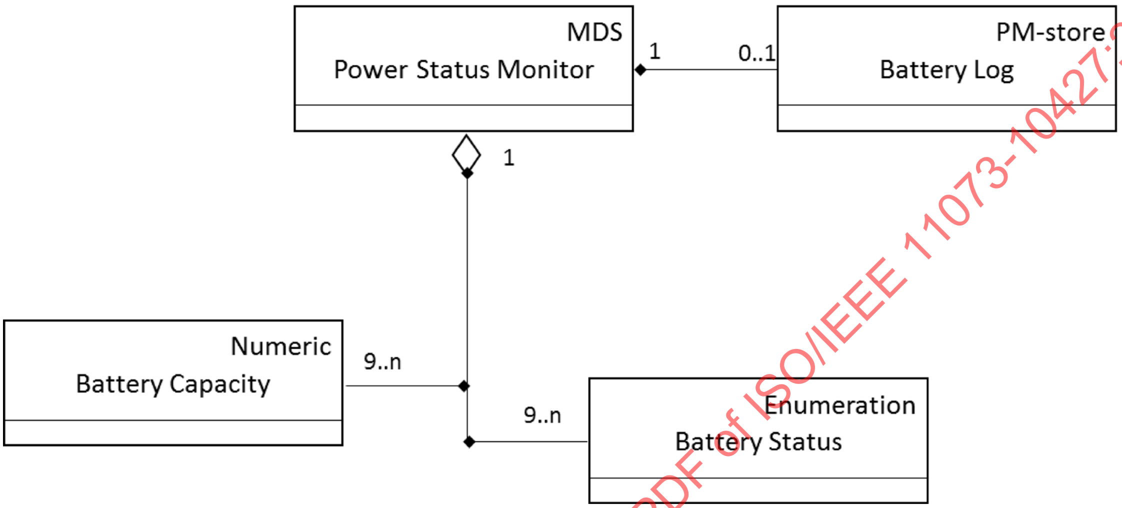

Figure 4—More-than-eight battery device—DIM

12.2.2 Types of configuration

The more-than-eight battery device does not support a standard configuration.

Extended configurations have two mandatory types of object: BatteryCapacity and BatteryStatus and shall contain n objects of each type, one to represent each battery. Extended configurations may support the BatteryLog object (Figure 4).

12.2.3 MDS object

12.2.3.1 Specific MDS object attributes

Table 19 provides the System-Type-Spec-List and Dev-Configuration-Id of the more-than-eight battery device. All other MDS object attributes are provided in Table 1.

Table 19—Specific MDS object attributes

Attribute name	Value	Qual.
System-Type-Spec-List	{MDC_DEV_SPEC_PROFILE_PSM, 1} And Profile value: {MDC_DEV_SUB_SPEC_PRO-FILE_MORE_THAN_EIGHT_BATTERIES, 1}	M
Dev-Configuration-Id	Extended configs: 0x4000-0x7FFF	M
NOTE—See IEEE Std 11073-20601 for information on whether an attribute is observational, static, or dynamic.		

The Dev-Configuration-Id attribute holds a locally unique 16-bit identifier that identifies the device configuration. For a more-than-eight battery agent, this identifier is chosen in the range of extended config-start to extended-config-end (see IEEE Std 11073-20601).

12.2.3.2 MDS object methods

See 6.6.2.

12.2.3.3 MDS object events

See 6.6.3.

12.2.3.4 Other MDS service

See 6.6.4.

12.2.4 Numeric objects

The more-than-eight battery device DIM (see Figure 4) contains 9 or more BatteryCapacity numeric objects, each of which represents the capacity for the i-th battery, where “i” takes on a value from 9-to-n, where “n” is the number of batteries the device supports.

See 6.7.2 for details on these objects.

12.2.5 Real-time sample array objects

Real-time sample array objects are not supported by this profile.

12.2.6 Enumeration objects

The more-than-eight battery device DIM (see Figure 4) contains a mandatory enumeration object: BatteryStatus for each supported battery, representing the status for battery i, where “i” takes on a value from 9-to-n, where “n” is the number of batteries the device supports.

See 6.9.2 for details on these objects.

12.2.7 PM-store objects (6.10)

Persistent metric store objects may be supported by this profile.

12.2.8 Scanner objects (6.11)

Scanner objects are not supported by this profile.

12.2.9 Communications characteristic

In this subclause, limits on the size of an APDU transmitted or to be received by a more-than-eight battery device are defined. Small limits allow for simple implementations in terms of low cost and complexity. A more-than-eight agent implementing only this device specialization shall not transmit any APDU larger than N_{tx} and shall be capable of receiving any APDU up to a size of N_{rx}. For this standard, using Equation (1) from 8.2, N_{tx} with S=12 is 1660 octets for implementations supporting 16 batteries. For this standard, N_{rx} shall be 46 octets.

For a more-than-eight battery device agent implementing functions from other device specializations, an upper bound estimation of the APDU sizes brings the following: An agent shall not transmit any APDU larger than the sum of N_{tx} of all the device specializations implemented and shall be capable of receiving any APDU up to the sum of N_{rx} of all the device specializations implemented.

If these numbers are higher than the maximum size determined in IEEE Std 11073-20601, then the latter shall be applied. In case the APDU size limit does not allow for the inclusion of a certain amount of multiple pending measurements at the agent, they shall be sent using multiple event reports. See 8.5.3 for the maximum number of measurements allowed for inclusion in a single event report.

Annex A

(informative)

Bibliography

Bibliographical references are resources that provide additional or helpful material but do not need to be understood or used to implement this standard. Reference to these resources is made for informational use only.

[B1] IEC 60601-1:2005, Ed. 3, Medical electrical equipment—Part 1: General requirements for basic safety and essential performance.[8]

[B2] IEC 60601-2, Medical electrical equipment—Part 2: Particular requirements for the basic safety and essential performance for specific device. (See the entire series of standards, Part 2-1 through Part 2-51.)

[B3] IEC 62304:2006/EN 62304:2006, Medical device software—Software life-cycle processes.[9]

[B4] IEC 80001-1:2010, Application of risk management for IT-networks incorporating medical devices—Part 1: Roles, responsibilities, and activities.

[B5] ISO 14971:2007, Medical devices—Application of risk management to medical devices.[10]

[B6] ISO/IEEE 11073-10101a:2015, Health informatics—Point-of-care medical device communication—Part 10101: Nomenclature.[11]

[B7] ISO/IEEE 11073-10101:2004, Health informatics—Point-of-care medical device communication Part 10101: Nomenclature.

[B8] ISO/IEEE 11073-10201:2004, Health informatics—Point-of-care medical device communication—Part 10201: Domain information model.

[B9] ISO/IEEE 11073-20101:2004, Health informatics—Point-of-care medical device communication—Part 20101: Application profile—Base standard.

[8]IEC publications are available from the International Electrotechnical Commission (http://www.iec.ch/). IEC publications are also available from the American National Standards Institute (http://www.ansi.org/).
[9]EN publications are available from the European Committee for Standardization (CEN) (http://www.cenorm.be).
[10]ISO publications are available from the ISO Central Secretariat (http://www.iso.ch/). ISO publications are also available from the American National Standards Institute (http://www.ansi.org/).
[11]ISO/IEEE publications are available from the Institute of Electrical and Electronics Engineers (http://standards.ieee.org/). ISO/IEEE publications are also available from the ISO Central Secretariat (http://www.iso.ch/).

Annex B

(normative)

Any additional ASN.1 definitions

B.1 Battery Status bit mapping

The extension to the enumeration class for Battery Status requires the following ASN.1 structure definition:

```
BatteryStatus ::== BITS-16 {
        Battery-statusUndetermined(0),
        Battery-present(1),
        Battery-active(2),
        Battery-charging(3),
        Battery-fullyCharged(4),
        Battery-disposable(5),
        Battery-rechargeable(6),
        Battery-overTemperature(7),
        Battery-faulty(8),
        Battery-incompatible(9),
        -- reserved for future extension(10),
        -- reserved for future extension(11),
        -- reserved for future extension(12),
        -- reserved for future extension(13),
        -- reserved for future extension(14),
        -- reserved for future extension(15),
        }
```

B.2 Capability-mask

The capability mask defines if the corresponding bit in the bit string is supported.

```
CapabMaskBasic ::=BITS-16 {
        bit_0_supported(0),     -- bit is supported: 1 | bit is not supported: 0
        bit_1_supported(1),     -- bit is supported: 1 | bit is not supported: 0
        bit_2_supported(2),     -- bit is supported: 1 | bit is not supported: 0
        bit_3_supported(3),     -- bit is supported: 1 | bit is not supported: 0
        bit_4_supported(4),     -- bit is supported: 1 | bit is not supported: 0
        bit_5_supported(5),     -- bit is supported: 1 | bit is not supported: 0
        bit_6_supported(6),     -- bit is supported: 1 | bit is not supported: 0
        bit_7_supported(7),     -- bit is supported: 1 | bit is not supported: 0
        bit_8_supported(8),     -- bit is supported: 1 | bit is not supported: 0
        bit_9_supported(9),     -- bit is supported: 1 | bit is not supported: 0
        bit_10_supported(10),   -- bit is supported: 1 | bit is not supported: 0
        bit_11_supported(11),   -- bit is supported: 1 | bit is not supported: 0
```

```
        bit_12_supported(12),        -- bit is supported: 1 | bit is not supported: 0
        bit_13_supported(13),        -- bit is supported: 1 | bit is not supported: 0
        bit_14_supported(14),        -- bit is supported: 1 | bit is not supported: 0
        bit_15_supported(15)         -- bit is supported: 1 | bit is not supported: 0
}
```

B.3 State-flag

The state flag defines if the corresponding bit in the enumeration bit string is a state or an event. For a bit that is a state, then the value is set to 1.

In this specialization, the bits bit_0_state(0):bit_0_state(9) are states and as such shall be 1.

```
StateFlagBasic ::=BITS-16 {
        bit_0_state(0),        -- bit is state: 1 | bit is event: 0
        bit_1_state(1),        -- bit is state: 1 | bit is event: 0
        bit_2_state(2),        -- bit is state: 1 | bit is event: 0
        bit_3_state(3),        -- bit is state: 1 | bit is event: 0
        bit_4_state(4),        -- bit is state: 1 | bit is event: 0
        bit_5_state(5),        -- bit is state: 1 | bit is event: 0
        bit_6_state(6),        -- bit is state: 1 | bit is event: 0
        bit_7_state(7),        -- bit is state: 1 | bit is event: 0
        bit_8_state(8),        -- bit is state: 1 | bit is event: 0
        bit_9_state(9),        -- bit is state: 1 | bit is event: 0
        bit_10_state(10),      -- bit is state: 1 | bit is event: 0
        bit_11_state(11),      -- bit is state: 1 | bit is event: 0
        bit_12_state(12),      -- bit is state: 1 | bit is event: 0
        bit_13_state(13),      -- bit is state: 1 | bit is event: 0
        bit_14_state(14),      -- bit is state: 1 | bit is event: 0
        bit_15_state(15)       -- bit is state: 1 | bit is event: 0
}
```

Annex C

(normative)

Allocation of identifiers

C.1 General

This annex contains the nomenclature codes used in this document and not found in IEEE Std 11073-20601. For those not contained in this annex, the normative definition is found in IEEE Std 11073-20601.

C.2 Definitions of terms and codes

The format used here follows that of ISO/IEEE 11073-10101a:2015 [B6].

```
/*****************************************************************************************
* FROM Infrastructure (MDC_PART_INFRA)
*****************************************************************************************/
#define MDC_DEV_SPEC_PROFILE_PSM        4124    /*      */
#define MDC_DEV_SUB_SPEC_PROFILE_EIGHT_OR_LESS_BATTERIES 4249   /*      */
#define MDC_DEV_SUB_SPEC_PROFILE_MORE_THAN_EIGHT_BATTERIES      4250    /*      */
/*****************************************************************************************
* FROM Infrastructure (MDC_PART_OBJ)
*****************************************************************************************/
#define MDC_ATTR_ENUM_CAPABILITY_MASK_BASIC     2705 /*Capability-Mask-Basic attribute */
#define MDC_ATTR_ENUM_STATE_FLAG_BASIC  2707 /*  State-Flag-Basic attribute     */
/*****************************************************************************************
* FROM Infrastructure (MDC_PART_PHD_DM)
*****************************************************************************************/
#define MDC_BATTERY_CAPACITY 29900      /*      */
#define MDC_BATTERY_STATUS      29904   /*      */
#define MDC_BATTERY_1  29912    /* first battery        */
#define MDC_BATTERY_2  29920    /* second battery       */
#define MDC_BATTERY_3  29928    /* third battery        */
#define MDC_BATTERY_4  29936    /* fourth battery       */
#define MDC_BATTERY_5  29944    /* fifth battery        */
#define MDC_BATTERY_6  29952    /* sixth battery        */
#define MDC_BATTERY_7  29960    /* seventh battery      */
#define MDC_BATTERY_8  29968    /* eighth battery       */
#define MDC_BATTERY_9  29976    /* ninth battery        */
#define MDC_BATTERY_10 29984    /* tenth battery        */
#define MDC_BATTERY_11 29992    /* eleventh battery     */
#define MDC_BATTERY_12 30000    /* twelfth battery      */
#define MDC_BATTERY_13 30008    /* thirteenth battery */
```

```
#define MDC_BATTERY_14 30016    /* fourteenth battery          */
#define MDC_BATTERY_15 30024    /* fifteenth battery   */
#define MDC_BATTERY_16 30032    /* sixteenth battery  */
```

C.3 Object-oriented modeling elements: metric enumeration

Table C.1—Systematic derivations of terms and codes: Object-oriented modeling elements: metric enumeration

DIM name	Reference ID	Derived from	Code
Capability-Mask-Basic	MDC_ATTR_ENUM_CAPABILITY_MASK_BASIC	Metric enumeration and derived objects	2705
State-Flag-Basic	MDC_ATTR_ ENUM_STATE_FLAG_BASIC	Metric enumeration and derived objects	2707

Annex D

(informative)

Message sequence examples

Figure D.1 shows a sequence diagram of the messaging procedure corresponding to the following use case. The user of a power status monitor agent intends to connect it to a manager for the first time. In this use case, the device can support a maximum of one battery and that battery is currently connected to the device.

a) When the user connects the PSM, the manager does not recognize the agent's configuration and sends a response to the agent's association request with the result *acceptedunknown-config*. See E.2.2.2 and E.2.2.3 for the corresponding PDU examples.

b) As a consequence of this, the agent negotiates its configuration information to the manager. After getting confirmation from the manager accepting the agent's configuration, the agent device is ready to send metric data. Both devices enter the Operating state. See E.3.2.2 and E.3.2.3 for the corresponding PDU examples.

c) Subsequently, the manager may request the MDS object attributes of the agent by sending a data message with the "Remote Operation Invoke | GET" command. Note that the manager may request the MDS object attributes as soon as the agent enters the Associated state, including the Configuring and Operating substates. As a response, the agent reports its MDS object attributes to the manager using a data message with the "Remote Operation Response | GET" command. See E.4.1.2 and E.4.1.3 for the corresponding PDU examples.

d) The next step is for the agent to collect the status and capacity information from its battery. If a new association between agent and manager is established, the battery's status and capacity metric data shall be transmitted to the manager using a confirmed event report. After having successfully received this metric data, the manager sends a confirmation to the agent. See E.5.1.1 and E.5.1.2 for the corresponding PDU examples.

e) If an attribute value of the battery status object changes, the battery's status metric data will be transmitted to the manager using a confirmed event report. After having successfully received the status metric data, the manager sends a confirmation to the agent. See E.5.2.1 and E.5.2.2 for the corresponding PDU examples.

f) If an attribute value of the battery capacity object changes by a "given amount," where "given amount" is a sensible value appropriate for the application, the battery's capacity metric data shall be transmitted to the manager using a confirmed event report. After having successfully received the capacity metric data, the manager sends a confirmation to the agent. See E.5.3.1 and E.5.3.2 for the corresponding PDU examples.

g) At some point the agent decides to end the session (e.g., by disassociating to save power). As a consequence, the agent disassociates from the manager by sending an association release request. The manager responds with an association release response. See E.6.1 and E.6.2 for the corresponding PDU examples.

h) When the agent requests to associate to the manager for the next session (e.g., the next day), the result in the manager's response is accepted, as it already knows the agent's configuration from the previous session. Both devices transition directly to the Operating state.

i) Finally, the last steps shown are similar [as in item d) to item f)]. The agent transmits the metric data followed by releasing the association.