

Recognized as an
American National Standard (ANSI)

IEEE Std 758™-1979(R2006)

IEEE Standard Subroutines for Computer Automated Measurement and Control (CAMAC)

Sponsor

**Instrumentation and Detectors Committee
of the
IEEE Nuclear and Plasma Sciences Society**

Reaffirmed March 30, 2006
Approved December 14, 1978

IEEE-SA Standards Board

Reaffirmed January 13, 2000
Approved December 21, 1990

American National Standards Institute

Copyright © 1979 by
The Institute of Electrical and Electronics Engineers, Inc.

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

SS7609-NAU
SH7609-NAU

Foreword

(This Foreword is not a part of IEEE Std 758-1979, Standard Subroutines for CAMAC.)

This standard presents a recommended set of software subroutines for use with the CAMAC modular instrumentation and interface system of IEEE Std 583-1975. These subroutines provide a general capability for communicating with CAMAC systems. They will be of primary interest to those who wish to write their own data-processing programs in a high-level programming language, such as Fortran. The achievable data transfer rate is, of course, dependent on a number of factors, including the language used, the operating system, the compiler, and the method and level of subroutines implementation and the computer.

The report on which this standard is based, and with which it is technically identical (DOE/EV-0016), was prepared by the NIM (National Instrumentation Methods) Committee of the US Department of Energy and the ESONE (European Standards on Nuclear Electronics) Committee of European Laboratories.

This standard was reviewed and balloted by the Nuclear Instruments and Detectors Committee of the IEEE Nuclear and Plasma Sciences Society.

Approved December 14, 1978

IEEE Standards Board

Joseph L. Koepfinger, *Chair*
Irvin N. Howell, Jr., *Vice Chair*
Ivan G. Easton, *Secretary*

William E. Andrus
C. N. Berglund
Edward J. Cohen
Warren H. Cook
David B. Dobson
R. O. Duncan
Charles W. Flint

Jay Forster
Ralph I. Hauser
Loering M. Johnson
Irving Kolodny
William R. Kruesi
Thomas J. Martin
John E. May

Donald T. Michael
Voss A. Moore
William S. Morgan
Robert L. Pritchard
Blair A. Rowley
Ralph M. Showers
B. W. Whittington

At the time of approval of this standard, the membership of the Nuclear Instruments and Detectors Committee of the IEEE Nuclear and Plasma Sciences Society was as follows:

D. E. Persyk, *Chair*
J. A. Coleman, *Vice Chair*
Louis Costrell, *Secretary*

R. L. Butenhoff
D. C. Cook
J. F. Detko
F. S. Goulding
F. A. Kirsten

T. R. Kohler
H. W. Kraner
W. W. Managan
G. L. Miller
J. H. Trainor

S. Wagner
F. J. Walter
H. R. Wasson

NIM Committee

L. Costrell, *Chair*

E. J. Barsotti
E. Davey
W. K. Dawson
S. Dhawan
J. Gallagher
C. E. L. Gingell
D. R. Heywood
N. W. Hill
G. A. Holt
D. Horelick

C. Kerns
F. A. Kirsten
R. S. Larsen
A. E. Larsh, Jr
N. Latner
F. R. Lenkszus
D. R. Machen
D. A. Mack
R. G. Martin
V. C. Negro

I. Pizer
S. Rankowitz
S. J. Rudnick
W. P. Sims
D. R. Stillwell
G. L. Strahl
R. F. Thomas, Jr
J. H. Trainor
H. R. Wasson
J. W. Woody, Jr

NIM Software Working Group

R. F. Thomas, Jr, *Chair*

W. K. Dawson, *Secretary*

L. Costrell
S. Dhawan
R. W. Goin

V. P. Elischer
R. A. LaSalle
F. R. Lenkszus

L. B. Mortara
D. G. Perry
J. P. Steffani

ESONE Committee

P. Christensen, Denmark, Chairman 1977/78

W. Attwenger, Austria
L. Binard, Belgium
J. Biri, Hungary
B. Bjarland, Finland
D. A. Boyce, England
B. A. Brandt, Germany
M. J. Cawthraw, England
F. Cesaroni, Italy
P. Clout, EMBL
M. Coli, Italy
W. K. Dawson, Canada
E. DeAgostino, Italy
B. V. Fefilov, JINR
P. Gallice, France
C. Gauffin, Sweden
H. R. Hidber, Switzerland

R. Horvath, Czechoslovakia
R. Hunt, England
F. Iselin, CERN
W. Kessel, Germany
R. Klesse, MLPL
E. Kwakkel, Netherlands
J. Lecomte, France
J. Lingertat, Germany
P. F. Manfredi, Italy
Ch. Mantakas, Greece
D. Marino, Italy
G. Metzger, France
H. Meyer, EURATOM
K. D. Muller, Germany
J. G. Ottes, Germany
A. T. Overtoom, Netherlands

M. Patrutescu, Roumania
R. Patzelt, Austria
A. C. Peatfield, England
G. Perna, Italy
I. C. Pyle, England
B. Rispoli, Italy
M. Sarquiz, France
W. Schoeps, Switzerland
L. Stanchi, EURATOM
H. J. Stuckenberg, Germany
R. Trechcinski, Poland
M. Truong, France
A. J. Vickers, England
M. Vojinovic, Yugoslavia
K. Zander, Germany
D. Zimmermann, Germany

CAMAC Specifications and Reports				
Title	IEEE, ANSI Std No	IEC No	DOE No	EURATOM (EUR) or ESONE No
Modular Instrumentation and Digital Interface System (CAMAC)	ANSI/IEEE Std 583-1975*	516	†	EUR 4100e
Supplement to CAMAC IEEE Standards Documents	IEEE Std 583A-1979	—	DOE/EV-0009	—
Serial Highway Interface System (CAMAC)	ANSI/IEEE Std 595-1976*	640	†	EUR 6100e
Parallel Highway Interface System (CAMAC)	ANSI/IEEE Std 596-1976*	552	†	EUR 4600e
Multiple Controllers in a CAMAC Crate	ANSI/IEEE Std 675-1979	—	DOE/EV-0007	EUR 6500e
Block Transfers in CAMAC Systems	ANSI/IEEE Std 683-1976	‡	†	EUR 4100 suppl
CAMAC Instrumentation and Interface Standards‡§	SH06437‡ (Library of Congress No 76-39660)	—	—	—
Amplitude Analogue Signals within a 50Ω System	—	—	TID-26614	EUR 5100e
The Definition of IML A Language for use in CAMAC Systems	—	—	TID-26615	ESONE/IML/01
CAMAC Tutorial Articles	—	—	TID-26618	—
Real-Time BASIC for CAMAC	IEEE Std 726-1979**	—	TID-26619	ESONE/RTB/02
Recommendations for CAMAC Serial Highway Drivers and LAM Graders for the SCC-L2	—	—	DOE/EV-0006	ESONE/SD/02

* Includes supplementary information.

† Superseded by corresponding IEEE Standard listed.

‡ In preparation.

§ This is a hard cover book that contains IEEE Stds 583-1975, 595-1976, 596-1976, and 683-1976, plus introductory material.

** Contains some differences with respect to the two identical documents TID-26619 and ESONE/RTB/02.

NOTE — Availability of Documents

IEEE	IEEE Service Center, 445 Hoes Lane, Piscataway, New Jersey 08854, U.S.A.
IEC	International Electrotechnical Commission, 1, rue de Varembe, CH-1211 Geneva 20, Switzerland.
DOE and TID Reports	National Bureau of Standards, Washington, D.C. 20234, U.S.A., Attn: L. Costrell.
EURATOM	Office of Official Publications of the European Communities, P.O. Box 1003, Luxembourg.
ESONE	Commission of the European Communities, CGR-BCM, B-2440 GEEL, Belgium, Attn: ESONE Secretariat, H. Meyer.

CLAUSE	PAGE
1. Introduction	1
2. Functional Specifications	2
2.1 Primary Subroutines.....	2
2.2 Single-Action Subroutines	3
2.3 Block Transfers, Multiple Actions, and Inverse Declarations	6
3. Definitions of Parameters.....	10
4. Bibliography.....	12
Annex A System-Dependent Subroutines (Informative)	13
Annex B FORTRAN Implementation (Informative).....	18
Annex C Function-Code Mnemonic Symbols (Informative).....	25

IEEE Standard Subroutines for Computer Automated Measurement and Control (CAMAC)

1. Introduction

This standard describes a set of standard subroutines to provide access to CAMAC facilities in a variety of computer-programming languages. It is specifically intended that the subroutines be suitable for use with FORTRAN, although they are not restricted to that language. Appendix B describes a recommended implementation of the subroutines explicitly for FORTRAN.

The present approach is based largely on IML and IML-M1 [6].¹ A distinction is made between *declarations*, which are used to name and specify computer and CAMAC entities, and *actions*, which are used to implement the various data movements and condition tests which make up the CAMAC-related portion of a program. As far as possible, the nomenclature of IML-M1 has been followed in order to take advantage of existing familiarity with that system and to provide as uniform a terminology and style as possible among the various CAMAC software documents.

Because of the widespread use of CAMAC on computers with a word length of less than 24 bits and because of the great differences in computer and operating-system features, special system-dependent features are often required to provide greater efficiency or to make appropriate use of the features of particular systems. The main body of this standard describes subroutines which, at the user interface, depend only on the features of CAMAC and therefore, when implemented in any standard procedural language, should be computer-independent. Appendix A describes subroutines which depend not only on CAMAC, but to some extent on individual computers. They cannot be made independent of the system on which they are implemented, and the user should take special precautions when it is necessary to incorporate them into a program. Such a program may not be transportable from one computer to another without modification.

The subroutines have been grouped into three subsets in order to provide different standard levels of implementation. The lowest level requires only two subroutines, but, nevertheless, gives access to most of the facilities which can be found in CAMAC. In higher levels of implementation subroutines are added which permit procedures to be written in more mnemonic terminology, provide better handling of LAM's, permit procedures to be independent of the type of CAMAC highway used, and provide efficient block-transfer capability.

¹Numbers in brackets correspond to those in the Bibliography, Section 4. of this standard.

2. Functional Specifications

This section introduces and describes in detail all the recommended subroutines. Since many implementations will not require the complete set, the subroutines are grouped into three subsets corresponding to the recommended implementation levels. Level A, the simplest, requires only the subroutines from 2.1. Level B, an intermediate level, requires the subroutines from 2.1 and 2.2. Level C is the highest level and requires implementation of the subroutines from 2.1, 2.2, and 2.3.

Two CAMAC facilities are not available through the use of these subroutines: the X response from an action and the BZ command. These facilities are available through the use of system-dependent subroutines described in Appendix A.

Naming conventions, compatible with the requirements of many computer languages, have been adopted. In order to make it simple for a user to avoid name conflicts, the name of each recommended subroutine begins with the letter C. The second letter of the subroutine name is coded to indicate the general function of the subroutine. Six letters have been used for this purpose:

- C indicates that the subroutine performs a control function
- D indicates that the subroutine is a declaration of a CAMAC entity
- F indicates that the subroutine transfers full-length (24-bit) data words
- G indicates that the subroutine analyzes a named CAMAC entity into its address components
- S indicates that the subroutine transfers short (less than 24-bit) data words
- T indicates that the subroutine tests the state of a signal or status indication

The remaining letters of each subroutine name (to a maximum of six) are chosen for their mnemonic value in identifying which function the subroutine performs.

Since no particular language is assumed in the body of this standard, no syntax can be defined for a subroutine call. The subroutines are described in terms of a subroutine name and an ordered sequence of parameters. As far as possible every implementation should retain the designated name and the order of the parameters. For the same reason no specific form can be defined for the subroutine parameters. CAMAC data words are bit strings with a length of twenty-four. An implementation must have the capacity to represent such strings. Other subroutine parameters are represented either as integer values or as the logical values *true* or *false*. In this standard the parameters of subroutines are described as variables or arrays of types CAMAC word, integer, or logical. For each implementation appropriate storage units and data formats must be chosen for each of these generalized entities.

2.1 Primary Subroutines

These two subroutines, which are required in all implementations, make up level A. The first provides the capability to define the address of a CAMAC entity and to access it. The second is used to perform CAMAC operations on the defined entities. In principle any CAMAC entity for which there is a defined standard mode of access can be accessed through the use of these two subroutines. In practice some systems may contain restrictions on the use of crate controllers or other system modules.

2.1.1 Declare CAMAC Register

Name: CDREG

Parameters:

ext (external address, see 3.1)

b (branch number, see 3.2)
c (crate number, see 3.3)
n (station number, see 3.4)
a (subaddress, see 3.5)

Function: CDREG combines the branch number *b*, the crate number *c*, the station number *n*, and the subaddress *a* into a convenient system-dependent form and stores the result in *ext*. Since the method of encoding depends on the implementation, the contents of *ext* should not be modified by the program. Some subroutines (see 2.2) require only a crate address; if the parameters *n* and *a* are both zero, CDREG encodes a crate address and stores it in *ext*.

2.1.2 Perform Single CAMAC Action

Name: CFSA

Parameters:

f (function code, see 3.6)
ext (external address, see 3.1)
int (CAMAC data word, see 3.7)
q (Q response, see 3.8)

Function: CFSA causes the CAMAC action specified by the function code *f* to be performed at the CAMAC address specified by *ext*. If *f* contains a read or write code, a twenty-four-bit data transfer occurs between the CAMAC register addressed by *ext* and the computer storage location *int*. Otherwise *int* is ignored. The state of Q resulting from the operation is stored in *q*, *true* if Q=1, *false* if Q=0.

2.2 Single-Action Subroutines

These subroutines, together with those described in 2.1, form the level B implementation, which provides a complete facility for specifying single CAMAC actions in a way which is mnemonic, compact, and independent of the type of highway or crate controller. Facilities are provided for declaring LAMs and performing LAM actions using constructions which are independent of the LAM access mode (that is, subaddress or register access).

2.2.1 Generate Dataway Initialize

Name: CCCZ

Parameter:

ext (external address, see 3.1)

Function: CCCZ causes Dataway Initialize (Z) to be generated in the crate specified by *ext*.

2.2.2 Generate Crate Clear

Name: CCCC

Parameter:

ext (external address, see 3.1)

Function: CCCC causes Dataway Clear (C) to be generated in the crate specified by *ext*.

2.2.3 Set or Clear Dataway Inhibit

Name: CCCI

Parameters:

ext (external address, see 3.1)
l (logical truth value, see 3.9)

Function: CCCI causes Dataway Inhibit (I) to be set in the crate specified by *ext* if the value of *l* is *true* and to be reset if the value of *l* is *false*.

2.2.4 Test Dataway Inhibit

Name: CTCI

Parameters:

ext (external address, see 3.1)
l (logical truth value, see 3.9)

Function: CTCI sets the value of *l* to *true* if Dataway Inhibit is set in the crate specified by *ext* and sets the value of *l* to *false* if Dataway Inhibit is not set.

2.2.5 Enable or Disable Crate Demand

Name: CCCD

Parameters:

ext (external address, see 3.1)
l (logical truth value, see 3.9)

Function: CCCD causes Crate Demand to be enabled in the crate specified by *ext* if the value of *l* is *true* and causes Crate Demand to be disabled if the value of *l* is *false*.

2.2.6 Test Crate Demand Enabled

Name: CTCD

Parameters:

ext (external address, see 3.1)
l (logical truth value, see 3.9)

Function: CTCD sets the value of *l* to *true* if Crate Demand is enabled in the crate specified by *ext* and sets the value of *l* to *false* if Crate Demand is disabled.

2.2.7 Test Crate Demand Present

Name: CTGL

Parameters: