

FINAL
DRAFT

INTERNATIONAL
STANDARD

ISO/IEC
FDIS
1539-1

ISO/IEC JTC 1/SC 22

Secretariat: ANSI

Voting begins on:
2023-08-04

Voting terminates on:
2023-09-29

Programming languages — Fortran —

**Part 1:
Base language**

Langages de programmation — Fortran —

Partie 1: Langage de base

iTeh Standards
(<https://standards.iteh.ai>)
Document Preview

[ISO/IEC 1539-1](#)

<https://standards.iteh.ai/catalog/standards/sist/b91563f9-a6f2-4ddc-bfd2-7fdedd8803b0/iso-iec-1539-1>

RECIPIENTS OF THIS DRAFT ARE INVITED TO SUBMIT, WITH THEIR COMMENTS, NOTIFICATION OF ANY RELEVANT PATENT RIGHTS OF WHICH THEY ARE AWARE AND TO PROVIDE SUPPORTING DOCUMENTATION.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.



Reference number
ISO/IEC FDIS 1539-1:2023(E)

© ISO/IEC 2023

iTeh Standards
(<https://standards.iteh.ai>)
Document Preview

[ISO/IEC 1539-1](#)

<https://standards.iteh.ai/catalog/standards/sist/b91563f9-a6f2-4ddc-bfd2-7fdedd8803b0/iso-iec-1539-1>



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2023

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

| | |
|--|------|
| Foreword | xii |
| Introduction | xiii |
| 1 Scope | 1 |
| 2 Normative references | 2 |
| 3 Terms and definitions | 3 |
| 4 Notation, conformance, and compatibility | 30 |
| 4.1 Notation, symbols and abbreviated terms | 30 |
| 4.1.1 Syntax rules | 30 |
| 4.1.2 Constraints | 31 |
| 4.1.3 Assumed syntax rules | 31 |
| 4.1.4 Syntax conventions and characteristics | 31 |
| 4.1.5 Text conventions | 32 |
| 4.2 Conformance | 32 |
| 4.3 Compatibility | 33 |
| 4.3.1 Previous Fortran standards | 33 |
| 4.3.2 New intrinsic procedures | 33 |
| 4.3.3 Fortran 2018 compatibility | 33 |
| 4.3.4 Fortran 2008 compatibility | 34 |
| 4.3.5 Fortran 2003 compatibility | 35 |
| 4.3.6 Fortran 95 compatibility | 36 |
| 4.3.7 Fortran 90 compatibility | 36 |
| 4.3.8 FORTRAN 77 compatibility | 37 |
| 4.4 Deleted and obsolescent features | 37 |
| 4.4.1 General | 37 |
| 4.4.2 Nature of deleted features | 37 |
| 4.4.3 Nature of obsolescent features | 38 |
| 5 Fortran concepts | 39 |
| 5.1 High level syntax | 39 |
| 5.2 Program unit concepts | 42 |
| 5.2.1 Program units and scoping units | 42 |
| 5.2.2 Program | 42 |
| 5.2.3 Procedure | 42 |
| 5.2.4 Module | 43 |
| 5.2.5 Submodule | 43 |
| 5.3 Execution concepts | 43 |
| 5.3.1 Statement classification | 43 |
| 5.3.2 Statement order | 43 |
| 5.3.3 The END statement | 44 |
| 5.3.4 Program execution | 44 |
| 5.3.5 Execution sequence | 45 |

| | | |
|--------|---|----|
| 5.3.6 | Image execution states | 45 |
| 5.3.7 | Termination of execution | 46 |
| 5.4 | Data concepts | 46 |
| 5.4.1 | Type | 46 |
| 5.4.2 | Data value | 47 |
| 5.4.3 | Data entity | 47 |
| 5.4.4 | Definition of objects and pointers | 48 |
| 5.4.5 | Reference | 49 |
| 5.4.6 | Array | 49 |
| 5.4.7 | Coarray | 49 |
| 5.4.8 | Established coarrays | 50 |
| 5.4.9 | Pointer | 50 |
| 5.4.10 | Allocatable variables | 50 |
| 5.4.11 | Storage | 51 |
| 5.5 | Fundamental concepts | 51 |
| 5.5.1 | Names and designators | 51 |
| 5.5.2 | Statement keyword | 51 |
| 5.5.3 | Other keywords | 51 |
| 5.5.4 | Association | 51 |
| 5.5.5 | Intrinsic | 51 |
| 5.5.6 | Operator | 52 |
| 5.5.7 | Companion processors | 52 |
| 6 | Lexical tokens and source form | 53 |
| 6.1 | Processor character set | 53 |
| 6.1.1 | Characters | 53 |
| 6.1.2 | Letters | 53 |
| 6.1.3 | Digits | 53 |
| 6.1.4 | Underscore | 53 |
| 6.1.5 | Special characters | 53 |
| 6.1.6 | Other characters | 54 |
| 6.2 | Low-level syntax | 54 |
| 6.2.1 | Tokens | 54 |
| 6.2.2 | Names | 54 |
| 6.2.3 | Constants | 55 |
| 6.2.4 | Operators | 55 |
| 6.2.5 | Statement labels | 56 |
| 6.2.6 | Delimiters | 56 |
| 6.3 | Source form | 57 |
| 6.3.1 | Program units, statements, and lines | 57 |
| 6.3.2 | Free source form | 57 |
| 6.3.3 | Fixed source form | 58 |
| 6.4 | Including source text | 59 |
| 7 | Types | 61 |
| 7.1 | Characteristics of types | 61 |
| 7.1.1 | The concept of type | 61 |
| 7.1.2 | Type classification | 61 |
| 7.1.3 | Set of values | 61 |
| 7.1.4 | Constants | 61 |
| 7.1.5 | Operations | 61 |
| 7.2 | Type parameters | 62 |
| 7.3 | Types, type specifiers, and values | 63 |
| 7.3.1 | Relationship of types and values to objects | 63 |
| 7.3.2 | Type specifiers | 63 |
| 7.3.3 | Type compatibility | 65 |

| | | |
|--------|--|-----|
| 7.4 | Intrinsic types | 66 |
| 7.4.1 | Classification and specification | 66 |
| 7.4.2 | Intrinsic operations on intrinsic types | 66 |
| 7.4.3 | Numeric intrinsic types | 66 |
| 7.4.4 | Character type | 70 |
| 7.4.5 | Logical type | 73 |
| 7.5 | Derived types | 73 |
| 7.5.1 | Derived type concepts | 73 |
| 7.5.2 | Derived-type definition | 74 |
| 7.5.3 | Derived-type parameters | 77 |
| 7.5.4 | Components | 79 |
| 7.5.5 | Type-bound procedures | 85 |
| 7.5.6 | Final subroutines | 88 |
| 7.5.7 | Type extension | 90 |
| 7.5.8 | Derived-type values | 92 |
| 7.5.9 | Derived-type specifier | 92 |
| 7.5.10 | Construction of derived-type values | 92 |
| 7.5.11 | Derived-type operations and assignment | 95 |
| 7.6 | Other nonintrinsic types | 95 |
| 7.6.1 | Interoperable enumerations and enum types | 95 |
| 7.6.2 | Enumeration types | 97 |
| 7.7 | Binary, octal, and hexadecimal literal constants | 99 |
| 7.8 | Construction of array values | 100 |
| 8 | Attribute declarations and specifications | 103 |
| 8.1 | Attributes of procedures and data objects | 103 |
| 8.2 | Type declaration statement | 103 |
| 8.3 | Automatic data objects | 105 |
| 8.4 | Initialization | 105 |
| 8.5 | Attributes | 105 |
| 8.5.1 | Attribute specification | 105 |
| 8.5.2 | Accessibility attribute | 106 |
| 8.5.3 | ALLOCATABLE attribute | 106 |
| 8.5.4 | ASYNCHRONOUS attribute | 106 |
| 8.5.5 | BIND attribute for data entities | 107 |
| 8.5.6 | CODIMENSION attribute | 107 |
| 8.5.7 | CONTIGUOUS attribute | 109 |
| 8.5.8 | DIMENSION attribute | 110 |
| 8.5.9 | EXTERNAL attribute | 114 |
| 8.5.10 | INTENT attribute | 114 |
| 8.5.11 | INTRINSIC attribute | 116 |
| 8.5.12 | OPTIONAL attribute | 116 |
| 8.5.13 | PARAMETER attribute | 116 |
| 8.5.14 | POINTER attribute | 117 |
| 8.5.15 | PROTECTED attribute | 117 |
| 8.5.16 | SAVE attribute | 118 |
| 8.5.17 | RANK clause | 118 |
| 8.5.18 | TARGET attribute | 119 |
| 8.5.19 | VALUE attribute | 119 |
| 8.5.20 | VOLATILE attribute | 119 |
| 8.6 | Attribute specification statements | 120 |
| 8.6.1 | Accessibility statement | 120 |
| 8.6.2 | ALLOCATABLE statement | 121 |
| 8.6.3 | ASYNCHRONOUS statement | 121 |
| 8.6.4 | BIND statement | 121 |
| 8.6.5 | CODIMENSION statement | 122 |

| | | |
|---------|---|-----|
| 8.6.6 | CONTIGUOUS statement | 122 |
| 8.6.7 | DATA statement | 122 |
| 8.6.8 | DIMENSION statement | 124 |
| 8.6.9 | INTENT statement | 125 |
| 8.6.10 | OPTIONAL statement | 125 |
| 8.6.11 | PARAMETER statement | 125 |
| 8.6.12 | POINTER statement | 125 |
| 8.6.13 | PROTECTED statement | 126 |
| 8.6.14 | SAVE statement | 126 |
| 8.6.15 | TARGET statement | 126 |
| 8.6.16 | VALUE statement | 126 |
| 8.6.17 | VOLATILE statement | 127 |
| 8.7 | IMPLICIT statement | 127 |
| 8.8 | IMPORT statement | 129 |
| 8.9 | NAMELIST statement | 131 |
| 8.10 | Storage association of data objects | 132 |
| 8.10.1 | EQUIVALENCE statement | 132 |
| 8.10.2 | COMMON statement | 134 |
| 8.10.3 | Restrictions on common and equivalence | 135 |
| 9 | Use of data objects | 136 |
| 9.1 | Designator | 136 |
| 9.2 | Variable | 136 |
| 9.3 | Constants | 137 |
| 9.4 | Scalars | 137 |
| 9.4.1 | Substrings | 137 |
| 9.4.2 | Structure components | 137 |
| 9.4.3 | Coindexed named objects | 139 |
| 9.4.4 | Complex parts | 139 |
| 9.4.5 | Type parameter inquiry | 139 |
| 9.5 | Arrays | 140 |
| 9.5.1 | Order of reference | 140 |
| 9.5.2 | Whole arrays | 140 |
| 9.5.3 | Array elements and array sections | 140 |
| 9.5.4 | Simply contiguous array designators | 144 |
| 9.6 | Image selectors | 144 |
| 9.7 | Dynamic association | 146 |
| 9.7.1 | ALLOCATE statement | 146 |
| 9.7.2 | NULLIFY statement | 150 |
| 9.7.3 | DEALLOCATE statement | 150 |
| 9.7.4 | STAT= specifier | 152 |
| 9.7.5 | ERRMSG= specifier | 153 |
| 10 | Expressions and assignment | 154 |
| 10.1 | Expressions | 154 |
| 10.1.1 | Expression semantics | 154 |
| 10.1.2 | Form of an expression | 154 |
| 10.1.3 | Precedence of operators | 158 |
| 10.1.4 | Evaluation of operations | 160 |
| 10.1.5 | Intrinsic operations | 161 |
| 10.1.6 | Defined operations | 167 |
| 10.1.7 | Evaluation of operands | 168 |
| 10.1.8 | Integrity of parentheses | 169 |
| 10.1.9 | Type, type parameters, and shape of an expression | 169 |
| 10.1.10 | Conformability rules for elemental operations | 171 |
| 10.1.11 | Specification expression | 171 |

| | | |
|---------|--|-----|
| 10.1.12 | Constant expression | 173 |
| 10.2 | Assignment | 174 |
| 10.2.1 | Assignment statement | 174 |
| 10.2.2 | Pointer assignment | 179 |
| 10.2.3 | Masked array assignment – WHERE | 183 |
| 10.2.4 | FORALL | 186 |
| 11 | Execution control | 189 |
| 11.1 | Executable constructs containing blocks | 189 |
| 11.1.1 | Blocks | 189 |
| 11.1.2 | Rules governing blocks | 189 |
| 11.1.3 | ASSOCIATE construct | 190 |
| 11.1.4 | BLOCK construct | 191 |
| 11.1.5 | CHANGE TEAM construct | 193 |
| 11.1.6 | CRITICAL construct | 195 |
| 11.1.7 | DO construct | 196 |
| 11.1.8 | IF construct and statement | 204 |
| 11.1.9 | SELECT CASE construct | 205 |
| 11.1.10 | SELECT RANK construct | 208 |
| 11.1.11 | SELECT TYPE construct | 210 |
| 11.1.12 | EXIT statement | 213 |
| 11.2 | Branching | 213 |
| 11.2.1 | Branch concepts | 213 |
| 11.2.2 | GO TO statement | 213 |
| 11.2.3 | Computed GO TO statement | 214 |
| 11.3 | CONTINUE statement | 214 |
| 11.4 | STOP and ERROR STOP statements | 214 |
| 11.5 | FAIL IMAGE statement | 215 |
| 11.6 | NOTIFY WAIT statement | 215 |
| 11.7 | Image execution control | 216 |
| 11.7.1 | Image control statements | 216 |
| 11.7.2 | Segments | 217 |
| 11.7.3 | SYNC ALL statement | 218 |
| 11.7.4 | SYNC IMAGES statement | 219 |
| 11.7.5 | SYNC MEMORY statement | 220 |
| 11.7.6 | SYNC TEAM statement | 221 |
| 11.7.7 | EVENT POST statement | 222 |
| 11.7.8 | EVENT WAIT statement | 222 |
| 11.7.9 | FORM TEAM statement | 222 |
| 11.7.10 | LOCK and UNLOCK statements | 223 |
| 11.7.11 | STAT= and ERRMSG= specifiers in image control statements | 225 |
| 12 | Input/output statements | 228 |
| 12.1 | Input/output concepts | 228 |
| 12.2 | Records | 228 |
| 12.2.1 | Definition of a record | 228 |
| 12.2.2 | Formatted record | 228 |
| 12.2.3 | Unformatted record | 228 |
| 12.2.4 | Endfile record | 229 |
| 12.3 | External files | 229 |
| 12.3.1 | External file concepts | 229 |
| 12.3.2 | File existence | 229 |
| 12.3.3 | File access | 230 |
| 12.3.4 | File position | 232 |
| 12.3.5 | File storage units | 233 |
| 12.4 | Internal files | 234 |

| | | |
|---------|---|-----|
| 12.5 | File connection | 234 |
| 12.5.1 | Referring to a file | 234 |
| 12.5.2 | Connection modes | 235 |
| 12.5.3 | Unit existence | 236 |
| 12.5.4 | Connection of a file to a unit | 236 |
| 12.5.5 | Preconnection | 237 |
| 12.5.6 | OPEN statement | 237 |
| 12.5.7 | CLOSE statement | 241 |
| 12.6 | Data transfer statements | 243 |
| 12.6.1 | Form of input and output statements | 243 |
| 12.6.2 | Control information list | 243 |
| 12.6.3 | Data transfer input/output list | 248 |
| 12.6.4 | Execution of a data transfer input/output statement | 250 |
| 12.6.5 | Termination of data transfer statements | 261 |
| 12.7 | Waiting on pending data transfer | 261 |
| 12.7.1 | Wait operation | 261 |
| 12.7.2 | WAIT statement | 261 |
| 12.8 | File positioning statements | 262 |
| 12.8.1 | Syntax | 262 |
| 12.8.2 | BACKSPACE statement | 263 |
| 12.8.3 | ENDFILE statement | 263 |
| 12.8.4 | REWIND statement | 263 |
| 12.9 | FLUSH statement | 264 |
| 12.10 | File inquiry statement | 264 |
| 12.10.1 | Forms of the INQUIRE statement | 264 |
| 12.10.2 | Inquiry specifiers | 265 |
| 12.10.3 | Inquire by output list | 271 |
| 12.11 | Error, end-of-record, and end-of-file conditions | 271 |
| 12.11.1 | Occurrence of input/output conditions | 271 |
| 12.11.2 | Error conditions and the ERR= specifier | 272 |
| 12.11.3 | End-of-file condition and the END= specifier | 272 |
| 12.11.4 | End-of-record condition and the EOR= specifier | 273 |
| 12.11.5 | IOSTAT= specifier | 273 |
| 12.11.6 | IOMSG= specifier | 274 |
| 12.12 | Restrictions on input/output statements | 274 |
| 13 | Input/output editing | 275 |
| 13.1 | Format specifications | 275 |
| 13.2 | Explicit format specification methods | 275 |
| 13.2.1 | FORMAT statement | 275 |
| 13.2.2 | Character format specification | 275 |
| 13.3 | Form of a format item list | 276 |
| 13.3.1 | Syntax | 276 |
| 13.3.2 | Edit descriptors | 276 |
| 13.3.3 | Fields | 278 |
| 13.4 | Interaction between input/output list and format | 278 |
| 13.5 | Positioning by format control | 280 |
| 13.6 | Decimal symbol | 280 |
| 13.7 | Data edit descriptors | 280 |
| 13.7.1 | Purpose of data edit descriptors | 280 |
| 13.7.2 | Numeric editing | 281 |
| 13.7.3 | Logical editing | 288 |
| 13.7.4 | Character editing | 288 |
| 13.7.5 | Generalized editing | 289 |
| 13.7.6 | User-defined derived-type editing | 290 |
| 13.8 | Control edit descriptors | 290 |

| | | |
|---------|---|-----|
| 13.8.1 | Position edit descriptors | 290 |
| 13.8.2 | Slash editing | 291 |
| 13.8.3 | Colon editing | 292 |
| 13.8.4 | SS, SP, and S editing | 292 |
| 13.8.5 | LZS, LZP and LZ editing | 292 |
| 13.8.6 | P editing | 292 |
| 13.8.7 | BN and BZ editing | 293 |
| 13.8.8 | RU, RD, RZ, RN, RC, and RP editing | 293 |
| 13.8.9 | DC and DP editing | 293 |
| 13.9 | Character string edit descriptors | 293 |
| 13.10 | List-directed formatting | 294 |
| 13.10.1 | Purpose of list-directed formatting | 294 |
| 13.10.2 | Values and value separators | 294 |
| 13.10.3 | List-directed input | 294 |
| 13.10.4 | List-directed output | 296 |
| 13.11 | Namelist formatting | 298 |
| 13.11.1 | Purpose of namelist formatting | 298 |
| 13.11.2 | Name-value subsequences | 298 |
| 13.11.3 | Namelist input | 298 |
| 13.11.4 | Namelist output | 301 |
| 14 | Program units | 303 |
| 14.1 | Main program | 303 |
| 14.2 | Modules | 303 |
| 14.2.1 | Module syntax and semantics | 303 |
| 14.2.2 | The USE statement and use association | 304 |
| 14.2.3 | Submodules | 307 |
| 14.3 | Block data program units | 307 |
| 15 | Procedures | 309 |
| 15.1 | Concepts | 309 |
| 15.2 | Procedure classifications | 309 |
| 15.2.1 | Procedure classification by reference | 309 |
| 15.2.2 | Procedure classification by means of definition | 309 |
| 15.3 | Characteristics | 310 |
| 15.3.1 | Characteristics of procedures | 310 |
| 15.3.2 | Characteristics of dummy arguments | 310 |
| 15.3.3 | Characteristics of function results | 310 |
| 15.4 | Procedure interface | 311 |
| 15.4.1 | Interface and abstract interface | 311 |
| 15.4.2 | Implicit and explicit interfaces | 311 |
| 15.4.3 | Specification of the procedure interface | 312 |
| 15.5 | Procedure reference | 321 |
| 15.5.1 | Syntax of a procedure reference | 321 |
| 15.5.2 | Actual arguments, dummy arguments, and argument association | 323 |
| 15.5.3 | Function reference | 335 |
| 15.5.4 | Subroutine reference | 335 |
| 15.5.5 | Resolving named procedure references | 336 |
| 15.5.6 | Resolving type-bound procedure references | 338 |
| 15.6 | Procedure definition | 338 |
| 15.6.1 | Intrinsic procedure definition | 338 |
| 15.6.2 | Procedures defined by subprograms | 338 |
| 15.6.3 | Definition and invocation of procedures by means other than Fortran | 344 |
| 15.6.4 | Statement function | 344 |
| 15.7 | Pure procedures | 345 |
| 15.8 | Simple procedures | 347 |

| | | |
|---------|--|-----|
| 15.9 | Elemental procedures | 348 |
| 15.9.1 | Elemental procedure declaration and interface | 348 |
| 15.9.2 | Elemental function actual arguments and results | 348 |
| 15.9.3 | Elemental subroutine actual arguments | 349 |
| 16 | Intrinsic procedures and modules | 350 |
| 16.1 | Classes of intrinsic procedures | 350 |
| 16.2 | Arguments to intrinsic procedures | 350 |
| 16.2.1 | General rules | 350 |
| 16.2.2 | The shape of array arguments | 351 |
| 16.2.3 | Mask arguments | 351 |
| 16.2.4 | DIM arguments and reduction functions | 351 |
| 16.3 | Bit model | 351 |
| 16.3.1 | General | 351 |
| 16.3.2 | Bit sequence comparisons | 352 |
| 16.3.3 | Bit sequences as arguments to INT and REAL | 352 |
| 16.4 | Numeric models | 352 |
| 16.5 | Atomic subroutines | 353 |
| 16.6 | Collective subroutines | 354 |
| 16.7 | Standard generic intrinsic procedures | 355 |
| 16.8 | Specific names for standard intrinsic functions | 360 |
| 16.9 | Specifications of the standard intrinsic procedures | 362 |
| 16.9.1 | General | 362 |
| 16.10 | Standard intrinsic modules | 457 |
| 16.10.1 | General | 457 |
| 16.10.2 | The ISO_FORTRAN_ENV intrinsic module | 458 |
| 17 | Exceptions and IEEE arithmetic | 465 |
| 17.1 | Overview of IEEE arithmetic support | 465 |
| 17.2 | Derived types, constants, and operators defined in the modules | 466 |
| 17.3 | The exceptions | 466 |
| 17.4 | The rounding modes | 469 |
| 17.5 | Underflow mode | 469 |
| 17.6 | Halting | 470 |
| 17.7 | The floating-point modes and status | 470 |
| 17.8 | Exceptional values | 470 |
| 17.9 | IEEE arithmetic | 470 |
| 17.10 | Summary of the procedures | 471 |
| 17.11 | Specifications of the procedures | 473 |
| 17.11.1 | General | 473 |
| 17.12 | Examples | 499 |
| 18 | Interoperability with C | 502 |
| 18.1 | General | 502 |
| 18.2 | The ISO_C_BINDING intrinsic module | 502 |
| 18.2.1 | Summary of contents | 502 |
| 18.2.2 | Named constants and derived types in the module | 502 |
| 18.2.3 | Procedures in the module | 503 |
| 18.3 | Interoperability between Fortran and C entities | 511 |
| 18.3.1 | Interoperability of intrinsic types | 511 |
| 18.3.2 | Interoperability with C pointer types | 512 |
| 18.3.3 | Interoperability of enum types | 512 |
| 18.3.4 | Interoperability of derived types and C structure types | 512 |
| 18.3.5 | Interoperability of scalar variables | 513 |
| 18.3.6 | Interoperability of array variables | 514 |
| 18.3.7 | Interoperability of procedures and procedure interfaces | 514 |

| | | |
|---------|---|-----|
| 18.4 | C descriptors | 517 |
| 18.5 | The source file ISO_Fortran_binding.h | 517 |
| 18.5.1 | Summary of contents | 517 |
| 18.5.2 | The CFI_dim_t structure type | 517 |
| 18.5.3 | The CFI_cdesc_t structure type | 518 |
| 18.5.4 | Macros and typedefs in ISO_Fortran_binding.h | 519 |
| 18.5.5 | Functions declared in ISO_Fortran_binding.h | 521 |
| 18.6 | Restrictions on C descriptors | 529 |
| 18.7 | Restrictions on formal parameters | 529 |
| 18.8 | Restrictions on lifetimes | 529 |
| 18.9 | Interoperation with C global variables | 530 |
| 18.9.1 | General | 530 |
| 18.9.2 | Binding labels for common blocks and variables | 531 |
| 18.10 | Interoperation with C functions | 531 |
| 18.10.1 | Definition and reference of interoperable procedures | 531 |
| 18.10.2 | Binding labels for procedures | 532 |
| 18.10.3 | Exceptions and IEEE arithmetic procedures | 532 |
| 18.10.4 | Asynchronous communication | 533 |
| 19 | Scope, association, and definition | 534 |
| 19.1 | Scopes, identifiers, and entities | 534 |
| 19.2 | Global identifiers | 534 |
| 19.3 | Local identifiers | 535 |
| 19.3.1 | Classes of local identifiers | 535 |
| 19.3.2 | Local identifiers that are the same as common block names | 536 |
| 19.3.3 | Function results | 536 |
| 19.3.4 | Components, type parameters, and bindings | 536 |
| 19.3.5 | Argument keywords | 536 |
| 19.4 | Statement and construct entities | 537 |
| 19.5 | Association | 538 |
| 19.5.1 | Name association | 538 |
| 19.5.2 | Pointer association | 542 |
| 19.5.3 | Storage association | 545 |
| 19.5.4 | Inheritance association | 547 |
| 19.5.5 | Establishing associations | 547 |
| 19.6 | Definition and undefinition of variables | 548 |
| 19.6.1 | Definition of objects and subobjects | 548 |
| 19.6.2 | Variables that are always defined | 548 |
| 19.6.3 | Variables that are initially defined | 548 |
| 19.6.4 | Variables that are initially undefined | 549 |
| 19.6.5 | Events that cause variables to become defined | 549 |
| 19.6.6 | Events that cause variables to become undefined | 551 |
| 19.6.7 | Variable definition context | 553 |
| 19.6.8 | Pointer association context | 554 |
| Annex A | (informative) Processor dependencies | 555 |
| Annex B | (informative) Deleted and obsolescent features | 562 |
| Annex C | (informative) Extended notes | 566 |
| Index | | 643 |

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see <https://patents.iec.ch>).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

This fifth edition cancels and replaces the fourth edition (ISO/IEC 1539-1:2018), which has been technically revised.

The main changes are as follows:

- an array can have a **coarray** component;
- additional forms of declaration;
- additional edit descriptors;
- additional intrinsic procedures;
- conformance with ISO/IEC 60559:2020;
- other changes listed in the Introduction.

A list of all parts in the ISO/IEC 1539 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

This document comprises the specification of the base Fortran language, informally known as Fortran 2023. With the limitations noted in 4.3.3, the syntax and semantics of Fortran 2018 are contained entirely within Fortran 2023. Therefore, any standard-conforming Fortran 2018 program not affected by such limitations is a standard-conforming Fortran 2023 program. New features of Fortran 2023 can be compatibly incorporated into such Fortran 2018 programs, with any exceptions indicated in the text of this document.

Fortran 2023 contains several extensions to Fortran 2018; these are listed below.

- Source form:

The maximum length of a line in free form source has been increased. The maximum length of a statement has been increased. The limit on the number of continuation lines has been removed.
- Data declaration:

A data object with a coarray component can be an array or allocatable. BIND(C) ENUM are now referred to as interoperable enumerations, and noninteroperable enumeration types are available. An interoperable enumeration can be given a type name. TYPEOF and CLASSOF type specifiers can be used to declare one or more entities to have the same type and type parameters as another entity. A PUBLIC namelist group can have a PRIVATE namelist group object. The DIMENSION attribute can be declared with a syntax that does not depend on the rank (8.5.8, 8.5.17).
- Data usage and computation:

Binary, octal, and hexadecimal literal constants can be used in additional contexts. A deferred-length allocatable *errmsg-variable* is allocated by the processor to the length of the explanatory message. An ALLOCATE statement can specify the bounds of an array allocation with array expressions. A pointer assignment statement can specify lower bounds or rank remapping with array expressions. Arrays can be used to specify multiple subscripts or subscript triplets (9.5.3.2). Conditional expressions provide selective evaluation of subexpressions.
- Input/output:

The AT edit descriptor provides output of character values with trailing blanks trimmed. The LEADING_ZERO= specifier in the OPEN and WRITE statements, and the LZP, LZS and LZ control edit descriptors, provide control of optional leading zeros during formatted output. A deferred-length allocatable *iomsg-variable* is allocated by the processor to the length of the explanatory message. A deferred-length allocatable scalar *io-unit* in a WRITE statement is allocated by the processor to the length of the record to be written.
- Execution control:

The REDUCE locality specifier for the DO CONCURRENT construct specifies reduction variables for the loop. The NOTIFY WAIT statement, NOTIFY= specifier on an image selector, and the NOTIFY_TYPE from the intrinsic module ISO_FORTRAN_ENV provide one-sided data-oriented synchronization between images.
- Intrinsic procedures:

The intrinsic functions ACOSD, ASIND, ATAND, ATAN2D, COSD, SIND, and TAND are trigonometric functions in which angles are specified in degrees. The intrinsic functions ACOSPI, ASINPI, ATANPI, ATAN2PI, COSPI, SINPI, and TANPI are trigonometric functions in which angles are specified in half-revolutions (that is, as multiples of π). The intrinsic function SELECTED_LOGICAL_KIND returns kind type parameter values for type logical. The intrinsic subroutine SPLIT parses a string into tokens, one at a time. The intrinsic subroutine SYSTEM_CLOCK supports more than one system clock for an image. The intrinsic subroutine TOKENIZE parses a string into tokens. When a deferred-length allocatable actual argument of an intrinsic procedure is assigned character data, it is allocated by the processor to the length of the data. Execution of a collective subroutine can be successful on an image even when an error condition occurs for the corresponding execution on another image.
- Intrinsic modules:

Additional named constants LOGICAL8, LOGICAL16, LOGICAL32, LOGICAL64, and REAL16 have been added to the intrinsic module ISO_FORTRAN_ENV. The subroutines IEEE_GET_ROUNDING_MODE, IEEE_GET_UNDERFLOW_MODE, IEEE_SET_ROUNDING_MODE, and IEEE_SET_UNDERFLOW_MODE, from the intrinsic module IEEE_ARITHMETIC, are now considered to be pure and simple. The subroutines IEEE_GET_MODES, IEEE_GET_STATUS, IEEE_SET_MODES, and

ISO/IEC FDIS 1539-1:2023 (E)

`IEEE_SET_STATUS`, from the intrinsic module `IEEE_EXCEPTIONS`, are now considered to be pure and simple. The procedures `C_F_STRPOINTER` and `F_C_STRING` have been added to the intrinsic module `ISO_C_BINDING` to assist in the use of null-terminated strings. The subroutine `C_F_POINTER` in the intrinsic module `ISO_C_BINDING` has an extra optional dummy argument, `LOWER`, that specifies the lower bounds for `FPTR`.

- Changes to the intrinsic module `IEEE_ARITHMETIC` for conformance with ISO/IEC 60559:2020: The new functions `IEEE_MAX`, `IEEE_MAX_MAG`, `IEEE_MIN`, and `IEEE_MIN_MAG` perform the operations maximum, maximumMagnitude, minimum, and minimumMagnitude in ISO/IEC 60559:2020. The functions `IEEE_MAX_NUM`, `IEEE_MAX_NUM_MAG`, `IEEE_MIN_NUM`, and `IEEE_MIN_NUM_MAG` now conform to the operations maximumNumber, maximumMagnitudeNumber, minimumNumber and minimumMagnitudeNumber in ISO/IEC 60559:2020; the changes affect the treatment of zeros and NaNs.
- Program units and procedures:
A procedure can be specified to be a [simple procedure](#); a [simple procedure](#) references or defines nonlocal variables only via its [dummy arguments](#). [Conditional arguments](#) provide [actual argument](#) selection in a [procedure reference](#).

This document is organized in 19 clauses, dealing with 8 conceptual areas. These 8 areas, and the clauses in which they are treated, are:

| | |
|-------------------------------|--------------------|
| High/low level concepts | Clauses 4, 5, 6 |
| Data concepts | Clauses 7, 8, 9 |
| Computations | Clauses 10, 16, 17 |
| Execution control | Clause 11 |
| Input/output | Clauses 12, 13 |
| Program units | Clauses 14, 15 |
| Interoperability with C | Clause 18 |
| Scoping and association rules | Clause 19 |

It also contains the following nonnormative material:

| | |
|---|---------|
| Processor dependencies | Annex A |
| Deleted and obsolescent features | Annex B |
| Extended notes ISO/IEC 1539-1 | Annex C |
| Index | Index |

<https://standards.iteh.ai/catalog/standards/sist/b91563f9-a6f2-4000-bfd2-7fdedd8803b0/iso-iec-1539-1>