

Australian Standard™

**Industrial automation systems and
integration—Physical device control—
Dimensional Measuring Interface
Standard (DMIS)**



This Australian Standard was prepared by Committee IT-006, Information Technology for Industrial Automation and Integration. It was approved on behalf of the Council of Standards Australia on 30 March 2004 and published on 3 June 2004.

The following are represented on Committee IT-006:

Association of Consulting Engineers Australia
Australian Electrical and Electronic Manufacturers Association
CSIRO Centre for Planning and Design
CSIRO Manufacturing & Infrastructure Technology
Department of Defence (Australia)
Institute of Instrumentation, Control and Automation Australia
Institution of Engineers Australia
Monash University
RMIT University
The University of Melbourne

Keeping Standards up-to-date

Standards are living documents which reflect progress in science, technology and systems. To maintain their currency, all Standards are periodically reviewed, and new editions are published. Between editions, amendments may be issued. Standards may also be withdrawn. It is important that readers assure themselves they are using a current Standard, which should include any amendments which may have been published since the Standard was purchased.

Detailed information about Standards can be found by visiting the Standards Web Shop at www.standards.com.au and looking up the relevant Standard in the on-line catalogue.

Alternatively, the printed Catalogue provides information current at 1 January each year, and the monthly magazine, *The Global Standard*, has a full listing of revisions and amendments published each month.

Australian Standards™ and other products and services developed by Standards Australia are published and distributed under contract by SAI Global, which operates the Standards Web Shop.

We also welcome suggestions for improvement in our Standards, and especially encourage readers to notify us immediately of any apparent inaccuracies or ambiguities. Contact us via email at mail@standards.org.au, or write to the Chief Executive, Standards Australia International Ltd, GPO Box 5420, Sydney, NSW 2001.

This Standard was issued in draft form for comment as DR 04109.

Australian Standard™

**Industrial automation systems and
integration—Physical device control—
Dimensional Measuring Interface
Standard (DMIS)**

First published as AS ISO 22093—2004.

COPYRIGHT

© Standards Australia International

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher.

Published by Standards Australia International Ltd
GPO Box 5420, Sydney, NSW 2001, Australia

ISBN 0 7337 5956 4

 **Documents**
Document Sharing Hub

PREFACE

This Standard was prepared by the Standards Australia Committee IT-006, Information Technology for Industrial Automation and Integration.

This Standard is identical with, and has been reproduced from, ISO 22093:2003, *Industrial automation systems and integration—Physical device control—Dimensional Measuring Interface Standard (DMIS)*.

The objective of this Standard is to provide a standard for the bi-directional communication of inspection data between computer systems and inspection equipment. The Dimensional Measuring Interface Standard (DMIS) is a vocabulary of terms, which establishes a neutral format for inspection programs and inspection results data.

The terms ‘normative’ and ‘informative’ are used to define the application of the annex to which they apply. A normative annex is an integral part of a standard, whereas an informative annex is only for information and guidance.

As this Standard is reproduced from an international standard, the following applies:

- a) Its number appears on the cover and title page while the international standard number appears only on the cover.
- b) In the source text ‘this International Standard’ should read ‘this Australian Standard’.
- c) A full point substitutes for a comma when referring to a decimal marker.

References to International Standards should be replaced by references to Australian or Australian/New Zealand Standards, as follows:

<i>Reference to International Standard</i>		<i>Australian Standard</i>	
ISO		AS ISO	
10303	Industrial automation systems and integration—Product data representation and exchange	10303	Industrial automation systems and integration—Product data representation and exchange
10303-1	Part 1: Overview and fundamental principles	10303.1	Part 1: Overview and fundamental principles
10303-11	Part 11: Description methods: The EXPRESS language reference manual	10303.11	Part 11: Description methods: The EXPRESS language reference manual
10303-12	Part 12: Description methods: The EXPRESS-I language reference model	10303.12	Part 12: Description methods: The EXPRESS-I language reference model
10303-21	Part 21: Implementation methods: Clear text encoding of the exchange structure	10303.21	Part 21: Implementation methods: Clear text encoding of the exchange structure
10303-22	Part 22: Implementation methods: Clear text encoding of the exchange structure	10303.22	Part 22: Implementation methods: Clear text encoding of the exchange structure
10303-31	Part 31: Conformance testing methodology and framework: General concepts	10303.31	Part 31: Conformance testing methodology and framework: General concepts
10303-41	Part 41: Integrated generic resources: Fundamentals of product description and support	10303.41	Part 41: Integrated generic resources: Fundamentals of product description and support

10303-42	Part 42: Integrated generic resources: Geometric and topological representation	10303.42	Part 42: Integrated generic resources: Geometric and topological representation
10303-43	Part 43: Integrated generic resources: Representation structures	10303.43	Part 43: Integrated generic resources: Representation structures
10303-44	Part 44: Integrated generic resources: Product structure configuration	10303.44	Part 44: Integrated generic resources: Product structure configuration
10303-45	Part 45: Integrated generic resources: Materials	10303.45	Part 45: Integrated generic resources: Materials
10303-46	Part 46: Integrated generic resources: Visual presentation	10303.46	Part 46: Integrated generic resources: Visual presentation
10303-47	Part 47: Integrated generic resources: Shape variation tolerances	10303.47	Part 47: Integrated generic resources: Shape variation tolerances
10303-49	Part 49: Integrated generic resources: Process structure and properties	10303.49	Part 49: Integrated generic resources: Process structure and properties
10303-101	Part 101: Integrated application resources: Draughting	10303.101	Part 101: Integrated application resources: Draughting
10303-104	Part 104: Integrated application resource: Finite element analysis	10303.104	Part 104: Integrated application resource: Finite element analysis
10303-105	Part 105: Integrated application resource: Kinematics	10303.105	Part 105: Integrated application resource: Kinematics
10303-201	Part 201: Application protocol: Explicit draughting	10303.201	Part 201: Application protocol: Explicit draughting
10303-202	Part 202: Application protocol: Associative draughting	10303.202	Part 202: Application protocol: Associative draughting
10303-203	Part 203: Application protocol: Configuration controlled design	10303.203	Part 203: Application protocol: Configuration controlled design
10303-204	Part 204: Application protocol: Mechanical design using boundary representation	10303.204	Part 204: Application protocol: Mechanical design using boundary representation
10303-207	Part 207: Application protocol: Sheet metal die planning and design	10303.207	Part 207: Application protocol: Sheet metal die planning and design
10303-209	Part 209: Application protocol: Composite and metallic structural analysis and related design	10303.209	Part 209: Application protocol: Composite and metallic structural analysis and related design
10303-210	Part 210: Application protocol: Electronic assembly, interconnection, and packaging design	10303.210	Part 210: Application protocol: Electronic assembly, interconnection, and packaging design
10303-212	Part 212: Application protocol: Electrotechnical design and installation	10303.212	Part 212: Application protocol: Electrotechnical design and installation

10303-214	Part 214: Application protocol: Core data for automotive mechanical design processes	10303.214	Part 214: Application protocol: Core data for automotive mechanical design processes
10303-216	Part 216: Application protocol: Ship moulded forms	10303.216	Part 216: Application protocol: Ship moulded forms
10303-224	Part 214: Application protocol: Mechanical product definition for process planning using machining features	10303.224	Part 214: Application protocol: Mechanical product definition for process planning using machining features
10303-225	Part 225: Application protocol: Building elements using explicit shape representation	10303.225	Part 225: Application protocol: Building elements using explicit shape representation
10303-227	Part 227: Application protocol: Plant spatial configuration	10303.227	Part 227: Application protocol: Plant spatial configuration
10303-232	Part 232: Application protocol: Technical data packaging core information and exchange	10303.232	Part 232: Application protocol: Technical data packaging core information and exchange

CONTENTS

Page

Introduction.....	xi
1 Scope.....	1
2 Conformance	2
2.1 DMIS conformance testing	2
2.2 Conformance testing services	2
2.3 CHFile	3
2.4 Complete standard conformance	3
2.5 Application profiles	3
2.6 Conformance levels	3
2.7 Conformance claim	3
3 Normative references.....	3
4 Terms and definitions	4
4.1 actuals	4
4.2 Automatically Programmed Tools (APT)	4
4.3 calibration sequence.....	4
4.4 carriage.....	4
4.5 characterization file.....	4
4.6 Computer Aided Design (CAD) system.....	5
4.7 Coordinate Measuring Machine (CMM).....	5
4.8 Dimensional Measuring Equipment (DME).....	5
4.9 Dimensional Measuring Interface Standard (DMIS).....	5
4.10 DMIS National Standards Committee (DNSC)	5
4.11 feature.....	5
4.12 filename	5
4.13 geometric compensation.....	5
4.14 High Level Language (HLL).....	5
4.15 Initial Graphics Exchange Specification (IGES).....	6
4.16 inner.....	6
4.17 In Process Verification (IPV)	6
4.18 input files	6
4.19 label	6
4.20 label type	6
4.21 label name	6
4.22 line reducible feature	6
4.23 measurement sequence	6
4.24 multiple carriage.....	6
4.25 nominal.....	7
4.26 orientation	7
4.27 orientational deviation	7
4.28 Orthonormal.....	7
4.29 outer.....	7
4.30 output files	7
4.31 part coordinate system (PCS)	7
4.32 plane reducible feature	7
4.33 point reducible feature.....	7
4.34 positional deviation.....	7
4.35 postprocessor	7
4.36 pre-processor	8
4.37 Quality Information System (QIS).....	8

	<i>Page</i>
4.38 ram	8
4.39 receiving system	8
4.40 rotary table system	8
4.41 selective processing	8
4.42 sensor	8
4.43 Standard Exchange Using Product Model Data (STEP)	8
4.44 statement	8
4.45 tolerance	8
4.46 tool holder	8
4.47 vendor	9
4.48 video inspection machine	9
4.49 work measuring zone	9
4.50 workpiece	9
5 Language reference	9
5.1 Syntax and structure	9
5.1.1 Characters	9
5.1.2 Words, label names, text strings, parameters, variables, and expressions	10
5.1.3 Variable assignments	15
5.1.4 DMIS command and definition statements	15
5.1.5 Delimiters, blank lines, spaces, and tabs	17
5.1.6 Line length	19
5.1.7 Programming comments	19
5.1.8 Operator input	19
5.1.9 Data output	20
5.1.10 Program structure	23
5.1.11 File structure	25
5.1.12 Programming considerations	25
5.2 Execution and control	26
5.2.1 Declaration statements	26
5.2.2 Definition statements	27
5.2.3 Program statement sequences	28
5.2.4 High Level Language (HLL)	31
5.3 Mathematics	36
5.3.1 Assignment operators	36
5.3.2 Features	38
5.3.3 Tolerances	47
5.3.4 Datums	49
5.3.5 Coordinate systems	49
5.4 Equipment control	63
5.4.1 Machine parameters	63
5.4.2 Rotary tables	67
5.4.3 Sensors and sensor-related	68
5.4.4 Carriages	70
5.4.5 Motion control	74
5.4.6 Measurement control	75
5.5 Characterization file	79
5.5.1 Usage	79
5.5.2 Characterization file format	80
5.5.3 Syntax for CHFIL1...ENDCH1 section	82
5.5.4 Example DMIS Characterization file grammar	88
6 Statement reference	94
6.1 ACLRAT	95
6.2 ALGDEF	96
6.3 ASSIGN	97
6.4 BADTST	98
6.5 BOUND	99
6.6 CALIB	100
6.7 CALL	102
6.8 CASE	104

	<i>Page</i>
6.9 CLMPID.....	105
6.10 CLMPSN	106
6.11 CLOSE	107
6.12 CMPNTGRP.....	108
6.13 CONST (input format 1)	109
6.14 CONST (input format 2)	111
6.15 CONST (input format 3)	112
6.16 CONST (input format 4)	113
6.17 CONST (input format 5)	116
6.18 CONST (input format 6)	117
6.19 CONST (input format 7)	119
6.20 CONST (input format 8)	121
6.21 CONST (input format 9)	122
6.22 CONST (input format 10)	123
6.23 CONST (input format 11)	124
6.24 CONST (input format 12)	125
6.25 CONST (input format 13)	126
6.26 CRGDEF	127
6.27 CRMODE.....	128
6.28 CROSCL	129
6.29 CRSLCT	130
6.30 CUTCOM.....	131
6.31 CZONE	132
6.32 CZSLCT	133
6.33 DATDEF	134
6.34 DATSET	135
6.35 DECL	137
6.36 DECPL	139
6.37 DELETE	140
6.38 DEVICE	141
6.39 DFTCAS	142
6.40 DISPLY.....	143
6.41 DMEHW.....	144
6.42 DMEID	145
6.43 DMESW.....	146
6.44 DMESWI.....	147
6.45 DMESWV	148
6.46 DMIS.....	149
6.47 DMISMD	150
6.48 DMISMN	151
6.49 DO	152
6.50 ELSE	153
6.51 ENDAT	154
6.52 ENDCAS	155
6.53 ENDDO.....	156
6.54 ENDFIL.....	157
6.55 ENDGO	158
6.56 ENDIF	159
6.57 ENDMAC.....	160
6.58 ENDMES	161
6.59 ENDSEL	162
6.60 ENDXTN.....	163
6.61 EQUATE.....	164
6.62 ERROR.....	165
6.63 EVAL	166
6.64 EXTENS	167
6.65 EXTFIL	168
6.66 FEAT/ARC (input format 1)	169
6.67 FEAT/ARC (input format 2)	171
6.68 FEAT/CIRCLE.....	173

	<i>Page</i>
6.69 FEAT/CONE.....	175
6.70 FEAT/CPARLN	177
6.71 FEAT/CYLNDR	179
6.72 FEAT/EDGEPT	181
6.73 FEAT/ELLIPS.....	183
6.74 FEAT/GCURVE.....	185
6.75 FEAT/GEOM	187
6.76 FEAT/GSURF.....	188
6.77 FEAT/LINE	190
6.78 FEAT/OBJECT.....	192
6.79 FEAT/PARPLN	194
6.80 FEAT/PATERN	196
6.81 FEAT/PLANE	198
6.82 FEAT/POINT	200
6.83 FEAT/RCTNGL	202
6.84 FEAT/SPHERE	204
6.85 FEAT/TORUS.....	206
6.86 FEDRAT	208
6.87 FILDEF	209
6.88 FILNAM	210
6.89 FINPOS	211
6.90 FIXTID	212
6.91 FIXTSN	213
6.92 FLY	214
6.93 FROM	215
6.94 GECOMP.....	216
6.95 GEOALG	217
6.96 GEOM.....	219
6.97 GOHOME	220
6.98 GOTARG.....	221
6.99 GOTO	222
6.100 GROUP	223
6.101 IF.....	224
6.102 INCLUD	226
6.103 Intrinsic functions	227
6.104 ITERAT.....	231
6.105 JUMPTO.....	233
6.106 LITDEF (input format 1)	234
6.107 LITDEF (input format 2)	235
6.108 LOCATE.....	236
6.109 LOTID.....	238
6.110 MACRO	239
6.111 MATDEF.....	240
6.112 MEAS	242
6.113 MFGDEV	244
6.114 MODE.....	245
6.115 OBTAIN.....	246
6.116 OPEN	247
6.117 OPERID.....	249
6.118 OUTPUT	250
6.119 PARTID	253
6.120 PARTRV	254
6.121 PARTSN	255
6.122 PLANID	256
6.123 POP	257
6.124 PRCOMP.....	258
6.125 PREVOP.....	259
6.126 PROCID.....	260
6.127 PROMPT	261
6.128 PSTHRU.....	264

	<i>Page</i>
6.129 PTBUFF	265
6.130 PTMEAS.....	266
6.131 PUSH.....	267
6.132 QISDEF	268
6.133 RAPID	269
6.134 READ	270
6.135 RECALL.....	271
6.136 REFMNT.....	272
6.137 REPORT.....	273
6.138 RESUME.....	275
6.139 RMEAS (input format 1)	276
6.140 RMEAS (input format 2)	278
6.141 RMEAS (input format 3)	280
6.142 RMEAS (input format 4)	282
6.143 RMEAS (input format 5)	284
6.144 RMEAS (input format 6)	286
6.145 RMEAS (input format 7)	288
6.146 ROTAB.....	290
6.147 ROTATE.....	291
6.148 ROTDEF.....	293
6.149 ROTSET	294
6.150 SAVE.....	295
6.151 SCAN	296
6.152 SCNMOD.....	297
6.153 SCNPLN.....	298
6.154 SCNSET	299
6.155 SELECT	301
6.156 SENSOR	302
6.157 SNSDEF (input format 1)	305
6.158 SNSDEF (input format 2)	308
6.159 SNSDEF (input format 3)	310
6.160 SNSDEF (input format 4)	312
6.161 SNSDEF (input format 5)	314
6.162 SNSDEF (input format 6)	316
6.163 SNSDEF (input format 7)	318
6.164 SNSET.....	320
6.165 SNSGRP	323
6.166 SNSLCT	324
6.167 SNSMNT	326
6.168 TECOMP	327
6.169 TEXT	328
6.170 THLDEF	329
6.171 TOL/ANGL	330
6.172 TOL/ANGLB	331
6.173 TOL/ANGLR	332
6.174 TOL/CIRLTY	334
6.175 TOL/COMPOS	335
6.176 TOL/CONCEN.....	337
6.177 TOL/CORTOL	338
6.178 TOL/CPROFS	339
6.179 TOL/CRNOUT	341
6.180 TOL/CYLCTY	342
6.181 TOL/DIAM	343
6.182 TOL/DISTB	345
6.183 TOL/FLAT	347
6.184 TOL/GTOL	348
6.185 TOL/PARLEL	352
6.186 TOL/PERP	354
6.187 TOL/POS.....	356
6.188 TOL/PROFL	358

	<i>Page</i>
6.189 TOL/PROFP	359
6.190 TOL/PROFS	360
6.191 TOL/RAD	362
6.192 TOL/STRGHT	363
6.193 TOL/SYM	364
6.194 TOL/TRNOUT	365
6.195 TOL/USETOL	366
6.196 TOL/WIDTH	367
6.197 TOOLDF	368
6.198 TRANS	369
6.199 UNITS	371
6.200 VALUE	372
6.201 VFORM	380
6.202 WINDEF (input format 1)	381
6.203 WINDEF (input format 2)	382
6.204 WKPLAN	383
6.205 WRIST	384
6.206 WRITE	386
6.207 XTERN	387
6.208 XTRACT	388
Annex A (informative) DMIS example code segments	389
A.1 @ character, and use with a label	389
A.2 ASSIGN	389
A.3 BADTST	389
A.4 CALL	390
A.4.1 Example 1 of CALL	390
A.4.2 Example 2 of CALL	391
A.5 CLOSE	391
A.6 CONST (input format 8)	392
A.7 CONST (input format 9) and CONST (input format 10)	392
A.8 CRMODE	394
A.8.1 CRMODE/SEQNTL	394
A.8.2 CRMODE/SIMUL	395
A.8.3 CRMODE/SYNC	395
A.9 CZSLCT	395
A.10 DEVICE	397
A.11 DMISMN	397
A.12 DO	398
A.13 EQUATE	398
A.14 EVAL	399
A.15 FEAT/GEOM	399
A.15.1 Constructing the nearest feature nominal on the part by using FA(PtFeat)	399
A.15.2 Defining the closest feature nominal on the part by using FA(dummy)	399
A.16 FEAT/GSURF	400
A.17 GEOM	400
A.18 GOTARG	400
A.19 IF and JUMPTO	401
A.19.1 Example 1	401
A.19.2 Example 2	401
A.20 ITERAT	402
A.21 LITDEF (input format 1)	403
A.22 Macro definition	403
A.22.1 An example of a macro definition without an argument list:	403
A.22.2 An example of a macro definition and a CALL statement follows:	403
A.23 OBTAIN	404
A.24 OPEN	404
A.25 PROMPT, examples of numeric and alphanumeric input	404
A.25.1 Example 1 numeric input	404
A.25.2 Example 2 alphanumeric input	404

	<i>Page</i>
A.25.3 Example 3 multiple input and playing a .wav file.....	404
A.26 QISDEF	405
A.27 READ examples of delimited and formatted	406
A.27.1 An example of the READ statement for a delimited file	406
A.27.2 An example of the READ statement for a formatted file	406
A.28 ROTDEF rotary table definition.....	406
A.29 SELECT...CASE...ENDCAS...DFTCAS...ENDCAS...ENDSEL block example	407
A.30 Sensor, wrist, component group, and build examples	408
A.30.1 Example wrist, component, and sensor definition	408
A.30.2 Example component definitions	410
A.30.3 Example component, group, and probe definitions	411
A.30.4 Example component definition and calibration statements	412
A.30.5 Example complex wrist and component definitions	413
A.30.6 Example tool definition statements	416
A.30.7 Example sensor calibration statements	417
A.31 SNSDEF (input format 2)	417
A.32 TEXT	417
A.33 TOL/GTOL	417
A.33.1 Example 1, just GO/NOGO info desired:.....	418
A.33.2 Example 2, GO/NOGO and percentage of good points data desired:	418
A.33.3 Example 3, GO/NOGO and interference point/s info desired:	418
A.34 VALUE	418
A.35 Vector variable values	418
A.36 WRITE	419
A.37 XTRACT	420
Annex B (informative) Descriptive Figures	422
Annex C (normative) Standard characterization file	476
Annex D (informative) Characterization file extensions	560
D.1 Machine dependent parameters	560
D.2 User defined options.....	561
Annex E (informative) Scanning reference	562
E.1 Introduction	562
E.2 Technical briefing.....	562
E.2.1 Scan statements.....	562
E.2.2 Scan methods.....	563
E.3 Applicable DMIS vocabulary	564
E.3.1 The SCNSET statement	564
E.3.2 The FEDRAT/SCNVEL statement.....	564
E.3.3 The SCNPLN statement	565
E.3.4 The SCNMOD statement.....	566
E.3.5 The SCAN statement.....	566
E.4 Using the SCNSET statement to specify point spacing.....	567
E.5 Descriptive synopsis	568
E.6 Applying nominals to surface scans.....	570
E.7 Examples.....	571
E.7.1 Scanning a Cylinder.....	571
E.7.2 Scanning a cylinder with a rotary table	572
E.7.3 Scanning Inside a bore with an interruption	573
E.7.4 Scanning Sections Across a Surface using a generic curve	574
E.7.5 Scanning a generic surface	575
Index of Statements	577
Index of figures.....	579
Index of tables	583
Index of Statements by type.....	585
Branching and looping statements	585
Carriage statements	585

	<i>Page</i>
Datum statements	585
Feature statements.....	586
Feature construction statements.....	586
File and machine parameter statements.....	587
In process verification / Quality information system statements	587
Input / output statements.....	588
Macro statements	588
Measurement statements	588
Miscellaneous statements	588
Motion statements	589
Program flow statements	589
Rotary table statements.....	589
Sensor statements	589
Scanning statements	590
Tolerance statements.....	590
Variable statements.....	591

INTRODUCTION

DMIS provides the vocabulary to pass inspection programs to dimensional measuring equipment and to pass measurement and process data back to an analysis, collection, and/or archiving system. A piece of equipment which interfaces to others, using the DMIS vocabulary, may do so directly or it may have a pre-processor to convert its own native data formats into the DMIS format and/or a postprocessor to convert the DMIS format into its own data structure.

An environment making use of the DMIS formats as a data exchange standard is depicted in (Figure 1 — DMIS environment). As illustrated, an inspection program can be created by many different approaches. Inspection program creation can be assisted by CAD systems, non-graphical systems, automated systems, or constructed manually. A programming system may require a pre-processor, which converts the program into DMIS format. A DMIS inspection program can then be executed on dissimilar dimensional measuring equipment (DME)s. In (Figure 1 — DMIS environment), DME I has a DMIS pre-processor and post-processor which converts the DMIS data into its own unique data format. DME IV is utilizing DMIS as its native format and therefore no pre-processors or post-processors are required. Also, a host computer is being used to control DME II and DME III. The host has a post-processor which decodes the DMIS program and drives the two DMEs, either through DMIS formats, or through some user-defined data exchange format.

Resultant data may be passed back in DMIS format through various scenarios. For example, this data could be passed directly as DMIS or via a post-processor. Resultant data is typically passed to an analysis system and/or a storage system such as a Quality Information System (QIS).

The manual interface indicates that DMIS programs can be hand written, and results analyzed, without the use of computer aids. In addition, many other uses of the DMIS data exchange format could be applied.

The implementation of DMIS is dependent on individual users. DMIS simply defines a neutral data exchange format that can be transmitted via ASCII files from one DMIS supporting system to another. The method for the transmission, storage, and management of these files is user-dependent.

This version of DMIS incorporates the statements necessary to drive coordinate measuring machines, video inspection devices, and hybrid in-process verification systems for dimensional and functional inspection of discrete mechanical parts and electronic components. The intent of DMIS is to provide a standard of communication for all dimensional measuring equipment.

Future applications may expand capabilities to support:

- 1) robotics for both workpiece presentation and measurement, and
- 2) additional provisions for in-process verification and process adjustment in a closed loop manufacturing cell.

This First ISO edition of DMIS has been completely restructured from the previous DMIS 3.0 ANSI/CAM-I 101-1995 standard. This restructuring was required to conform with ISO/IEC Directives, Part 3. In addition to this restructuring, all approved changes since DMIS 3.0 ANSI/CAM-I 101-1995 was published have been incorporated. This release includes additional functionality in the areas of sensor definitions, sensor component grouping, complex alignment, CAD connectivity, soft functional gauging, quality information systems, relaxation of various length limitations, and operator interface using the PROMPT statement. The majority of approved changes are contained in clause 7; the remainder of the text has not changed significantly, although the restructuring tends to suggest otherwise. The DNSC acknowledges the support and participation of the European DMIS User group (EDUG) in the development of the DMIS 04.0 version. The significant changes are listed below:

(label name)	Allows the use of a period '.' in label names.
ACLRAT	Changes to the input format.

ALGDEF	New statement added to define an algorithm, and assign to it a label name.
BADTST	New statement for trapping specific errors.
Characterization file	The format of the characterization file has changed to an Extended Backus Naur Form. Refer to paragraph 5.5 and sub-paragraphs, and also to Annex C.
CMPNTGRP	New statement added to group of components for a sensor, as a single component.
CONST (input format 4)	Extended capabilities.
CONST (input format 8)	New statement for constructing an offset feature.
CONST (input format 9)	New statement for soft functional gauging.
CONST (input format 10)	New statement for soft functional gauging.
CONST (input format 11)	New statement to copy a feature actual and transform it into another coordinate system.
CONST (input format 12)	New statement to construct a circle on an actual cone.
CONST (input format 13)	New statement to construct a nominal point from CAD geometry by locating the nearest point.
CROSCL	New statement to enhance use of rotary tables.
DECL	Added a VECTOR variable type, and multi-dimensional arrays.
DECPL	New statement to define decimal precision of output.
DMESW	Added DELAY as a minor word option.
DMISMD	The DMIS version is now mandatory with this statement.
DMISMN	The DMIS version is now mandatory with this statement.
EQUATE	New statement to equate two coordinate systems.
ERROR	Several changes to enhance error trapping.
EXTENS	New statement to define sensor extensions.
EXTFIL	The SYS minor word has been removed for the next major release of this standard.
FEAT/ARC (input format 1)	Removed the restriction that the arc must lie in a plane parallel to the current working plane.
FEAT/ARC (input format 2)	Removed the restriction that the arc must lie in a plane parallel to the current working plane.
FEAT/EDGEPT	New statement to define an edge point.
FEAT/GEOM	New statement to define a feature using previously defined geometry.
FEAT/TORUS	New statement to define a torus feature. The addition of this statement also caused changes to the CONST (input format 1) statement, the GEOALG statement, the MEAS statement, the RMEAS (input format 5) statement, the TOL/DIAM statement, the TOL/GTOL statement, and paragraph 5.3.2.1.1.

FEDRAT	Changes to the input format.
Figure Identifiers:	All Figure Identifiers have been renamed to conform to the ISO format.
FILNAM	The DMIS version is now mandatory with this statement.
FLY	New statement added to set or disable continuous motion mode on the controller.
GECOMP	New statement to turn geometric compensation on or off.
GEOM	New statement for defining CAD geometry associated with features.
GOTO	Extended the capabilities for the GOTO statement.
GROUP	New statement to define a group of previously calibrated sensors.
Intrinsic Functions:	Several changes to return system information.
ITERAT	New statement for complex alignments.
Label name Length:	Extended to 64 Characters. Refer to paragraph 5.1.2.2.
LOCATE	New statement for complex alignments.
MATDEF	New statement to specify parameters for functional mating.
POP	New statement to recall, from memory, various system information.
PROMPT	New statement to prompt the operator for input.
PUSH	New statement to save, in memory, various system information.
QISDEF	New statement for use in Quality Information Systems(QIS).
REFMNT	New statement to define a mounting reference point for a sensor mount component.
REPORT	Several minor words added, for enhanced reporting capability.
RESUME	New statement to recover from errors.
RMEAS (input format 7)	New statement for measurement of edge points.
SENSOR	New statement to define a sensor component capable of making measurement.
SNSDEF (input format 6)	XRAY sensor definition, and all references to XRAY sensors will be removed in the next DMIS major release.
SNSDEF (input format 7)	New input format added to build a sensor from wrists and components.
SNSGRP	New statement for defining a group of sensor components.
SNSLCT	Modified the format to specify wrist angles; Tip numbers; Feature axes; i,j,k, ang_tol; Sensor groups.
Statement Length	The statement length limit was 256 characters, this has been removed. See paragraph 5.1.6.
TECOMP	Added the capability to pass the thermal expansion coefficient of the part to the output file.

THLDEF	Added wrists and extensions to the tool holder/changer definitions.
TOL/ANGLR	Modified the input format to allow a second datum feature for orientation tolerances. Also added the capability to specify a parallel plane tolerance zone for all orientation tolerances on cylindrical tolerance features.
TOL/COMPOS	Modified to comply with ASME Y14.5M-1994.
TOL/CORTOL	The minor word RADIUS has been replaced with RADIAL. The minor word RADIUS was a typographical error. The RADIUS and RADIAL minor words have the same definition in this case. Both the RADIUS and RADIAL minor words will coexist for one DMIS version. The RADIUS minor word will be removed in the next DMIS major release.
TOL/GTOL	New statement for soft functional gauging.
TOL/PARLEL	Modified the input format to allow a second datum feature for orientation tolerances. Also added the capability to specify a parallel plane tolerance zone for all orientation tolerances on cylindrical tolerance features.
TOL/PERP	Modified the input format to allow a second datum feature for orientation tolerances. Also added the capability to specify a parallel plane tolerance zone for all orientation tolerances on cylindrical tolerance features.
TOL/POS	The minor word RADIUS has been replaced with RADIAL. The minor word RADIUS was a typographical error. The RADIUS and RADIAL minor words have the same definition in this case. Both the RADIUS and RADIAL minor words will coexist for one DMIS version. The RADIUS minor word will be removed in the next DMIS major release.
TOL/PROFS	An option in var_3; dev,AVGDEV was changed to AVGDEV,dev.
UNITS	A major change to allow multiple UNITS statements within a program. See paragraph 5.4.1.7.
VALUE	Major modifications have been made to gain access to the tolerance value when more than one value is possible, and to return various information.
VFORM	Added capabilities with var_3.
WRIST	New statement to define wrist rotations and offsets.
XTRACT	New statement to extract bounded features from complete curves or surfaces.

AUSTRALIAN STANDARD

Industrial automation systems and integration — Physical device control — Dimensional Measuring Interface Standard (DMIS)

1 Scope

The objective of this International Standard is to provide a standard for the bi-directional communication of inspection data between computer systems and inspection equipment. The Dimensional Measuring Interface Standard (DMIS) is a vocabulary of terms, which establishes a neutral format for inspection programs and inspection results data.

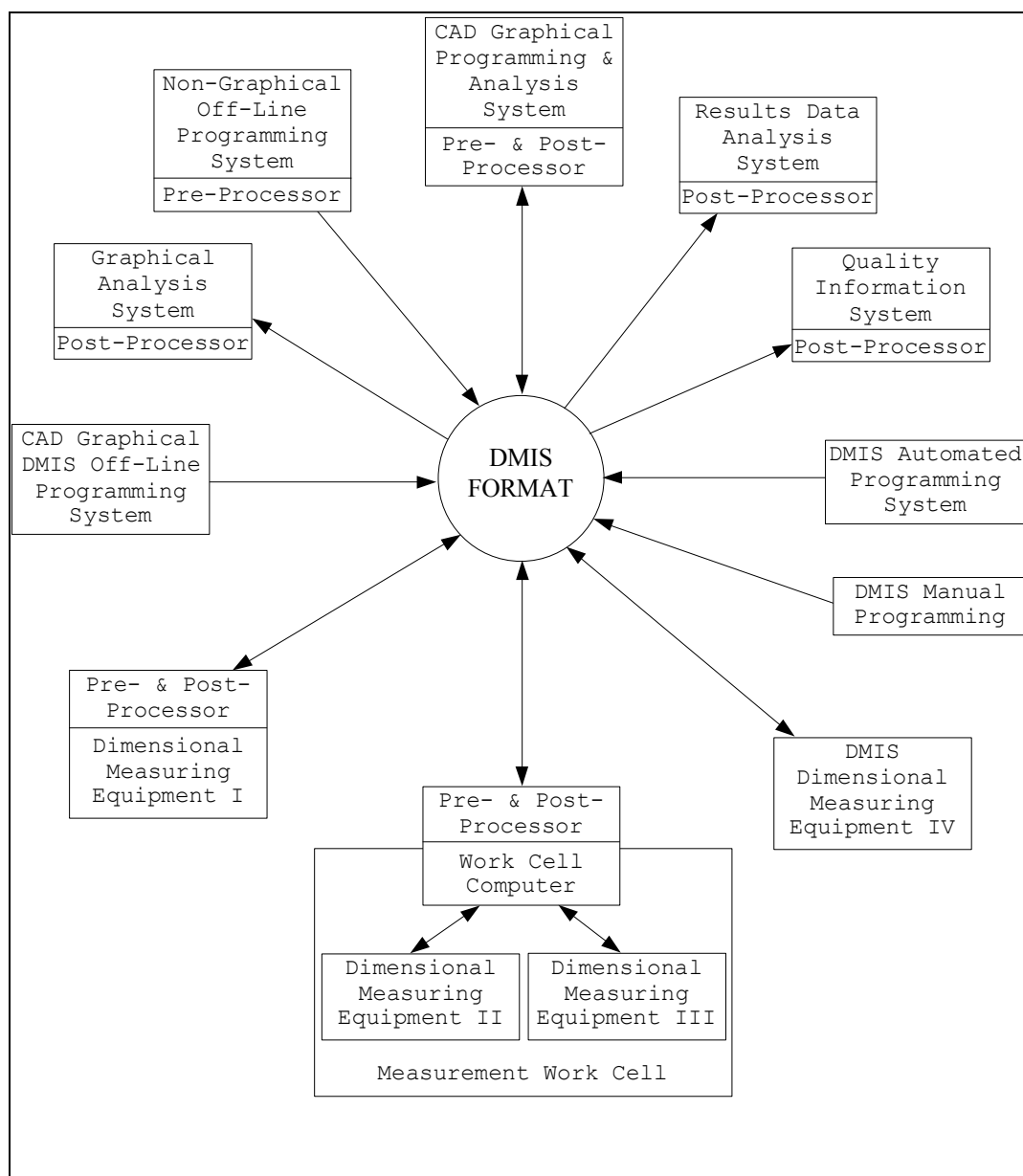


Figure 1 — DMIS environment

While primarily designed for communication between automated equipment, DMIS is designed to be both man-readable and man-writable, allowing inspection programs to be written and inspection results to be analyzed without the use of computer aids. With the enhancement of the High Level Language extensions, DMIS can function and be implemented as a DME language.

2 Conformance

The primary purpose of DMIS is to allow organizations to exchange and store measurement data among different dimensional measurement devices and computer applications both within their organizations, as well as with other organizations. DMIS is widely used and available for a broad range of measurement systems and applications. However, a DMIS file that is created by one DMIS product may not be fully or correctly interpreted by another DMIS product. Successful DMIS interchange can only be achieved if DMIS applications faithfully implement 1) the DMIS specification and 2) the appropriate formally recognized DMIS Application Profiles. DMIS is a large and complex standard. Vendors do not need to implement the entire standard. They implement functional subsets. These subsets are known as profiles.

The primary benefit of any DMIS profile is the ability to insure interoperability through the use of validation tools against DMIS instances and certification services for applications. Once an application has been certified through a testing service, behavior of that application is predictable under the constraints of the profile.

2.1 DMIS conformance testing

DMIS conformance testing is a way of determining if a DMIS compliant product correctly implements the DMIS specification with its associated application profile.

Strictly speaking, this DMIS specification is solely an exchange file format. However, the term "DMIS" is often used to include a generator (a program which produces the DMIS), an interpreter (a program which reads the DMIS), as well as the metafiles (the actual DMIS input and output files). Together, the generator, metafiles, and interpreter form a total DMIS system.

Conformance of DMIS is defined in terms of conformance to a particular profile of DMIS. Thus, the DMIS specification in conjunction with an application profile is necessary in order to test conformance of a total DMIS system.

Testing DMIS for conformance entails one or many of the following:

- a) verifying that the metafiles are syntactically correct,
- b) verifying that a generator produces conforming metafiles which accurately and correctly represent the intended results,
- c) verifying that an interpreter can correctly and completely read the metafile and produce the intended results, and
- d) verifying that the DMIS characterization file is syntactically correct and that it accurately represents the capabilities of the application.

An Application Profile conforms to DMIS if it adheres to all syntactic requirements defined in this Standard.

Application software conforms to DMIS if it interprets all conforming DMIS profiles (as defined above) according to all required semantics prescribed by this Standard.

2.2 Conformance testing services

Conformance Testing Services will be recognized by the DNSC to develop test suites for validation of DMIS characterization files and to test implementations for conformance to one or many DMIS Application Profiles.

2.3 CHFile

A vendor's DMIS characterization file represents its compliance to the DMIS specification. Conformance testing will validate that the characterization file is syntactically correct and that it accurately represents the capabilities of the intended generator or interpreter.

2.4 Complete standard conformance

DMIS is a large and complex standard. Vendors do not need to implement the entire standard. However it is possible that a DMIS application may conform to the entire standard.

2.5 Application profiles

Currently, there are no formally recognized DMIS Application Profiles. However the DNSC encourages the formulation, agreement, and submission of Application Profiles, such as Prismatic, Sheet Metal, or Multiple Carriages, for recognized approval from the DNSC.

2.6 Conformance levels

Each Application Profile will define three levels of adherence.

- ∞ # Level 1 – is essential to meeting the Profile's goals
- ∞ # Level 2 – is important to meeting the Profile's goals
- ∞ # Level 3 – is beneficial to meeting the Profile's goals.

2.7 Conformance claim

Upon positive notification from a recognized DMIS Conformance Testing Service, a DMIS application may make a conformance claim. An example of a claim is: OurProduct version x.y from DMISVendor conforms to DMIS Prismatic Profile, Level 2.

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

NOTE 1 The emerging STEP (Standard Exchange for Product Model Data) International Standard defines the computer-interpretable representation and unambiguous exchange of product information throughout the life of a product. As this community progresses toward a complete product data definition exchange standard, DMIS will evolve to maintain compatibility with their developments. DMIS has been designed with this compatibility and growth path in mind.

ISO/IEC 14977:1996, *Information technology — Syntactic metalanguage — Extended BNF*

American Standard Code for Information Interchange (ASCII)

NOTE 2 All DMIS vocabulary consists of characters from the ASCII 255-character set.

ANSI/ASME B89.4.1-1997, *Methods for Performance Evaluation of Coordinate Measuring Machines*

NOTE 3 All terminology that refers to Coordinate Measuring Machines is used in accordance with this International Standard.

ANSI/CAM-I 101 1990, *Dimensional Measuring Interface Standard*

NOTE 4 There are references to prior versions of DMIS in this document to provide for backward compatibility with the language.

ANSI Y14.26M-1987, *Initial Graphics Exchange Specification*

NOTE 5 Development of DMIS has been monitored by the community, which has developed and maintained the Initial Graphics Exchange Specification (IGES).

APT: ANSI X3.37-1987, *Programming Language APT*

NOTE 6 DMIS is similar in syntax to the syntax of the numerical control programming language APT (Automatically Programmed Tools). Some of the words in DMIS are taken from the APT vocabulary. It should be noted, however, that some words in DMIS, which are identical to words in APT, may have different meanings or a different syntax. The APT vocabulary should not be used as a reference when interpreting the meaning of DMIS vocabulary words.

ASME Y14.5M - 1994, *Dimensioning and Tolerancing*

NOTE 7 All ASME Y14.5M - 1994 geometric dimensions and tolerances are incorporated in DMIS.

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ANSI/ASME B89.4.1-1997 and the following apply.

4.1 actuals

referring to features or tolerances, are the results computed from measured or constructed features

NOTE Actuals, then, are output by the measuring device and are not defined until a measurement or construction has occurred.

4.2 Automatically Programmed Tools (APT)

computer language for numerical controlled (NC) machines

4.3 calibration sequence

sequence of statements contained in a CALIB...ENDMES block

NOTE A calibration sequence may be either a rotary table calibration sequence initiated with the CALIB/RTAB statement or a sensor calibration sequence initiated with the CALIB/SENS statement.

4.4 carriage

moving component of a machine that carries the ram

4.5 characterization file

file created by a vendor that lists the DMIS statements and functions it supports

4.6

Computer Aided Design (CAD) system

computerized system for design and analysis

NOTE With respect to this standard, CAD systems are used to develop inspection part programs and to analyze inspection results.

4.7

Coordinate Measuring Machine (CMM)

computerized inspection device for gathering discrete point data and analyzing dimensions and tolerances

NOTE Data is usually acquired using a touch probe or scanning probe as a sensor.

4.8

Dimensional Measuring Equipment (DME)

class of equipment used to inspect parts and evaluate dimensions and tolerances

NOTE DMEs include, but are not limited to, coordinate measuring machines, video inspection equipment, optical comparators, robotic measuring devices, theodolites, photogrammetry, and laser-based measuring devices.

4.9

Dimensional Measuring Interface Standard (DMIS)

name given to this standard

NOTE Reference is often made to DMIS vocabulary, DMIS inspection programs, DMIS files, and so forth. These are data files, which conform to the formats and structure set forth in this standard.

4.10

DMIS National Standards Committee (DNSC)

committee accorded responsibility for the maintenance and support of the DMIS National Standard

NOTE The DNSC is an accredited standards committee operating under ANSI- approved CAM-I procedures as the accredited organization.

4.11

feature

point or group of points, which are identified and referred to as a feature

NOTE Usually, features are geometric elements of a part such as points, lines, planes, and circles.

4.12

filename

computer readable file

NOTE A 'filename' can be a filename and extension or it can contain a full path specification.

4.13

geometric compensation

software specific to the DME which provides for increased accuracy by mapping the DME volume

4.14

High Level Language (HLL)

collection of statements providing advanced capabilities

NOTE These statements are identified in paragraph 5.2.4.

4.15**Initial Graphics Exchange Specification (IGES)**

specification for the exchange of CAD data

4.16**inner**

signifies that probing occurs from inside, and material lies exterior to, the envelope of the defining feature

4.17**In Process Verification (IPV)**

computerized system linking inspection and process verification to the manufacturing process

4.18**input files**

inspection programs input to the dimensional measuring equipment

NOTE Unless otherwise noted, input files (and input programs) are inspection programs input to the dimensional measuring equipment.

4.19**label**

that part of a DMIS statement which fully identifies a feature, tolerance, coordinate system, sensor, output data format, datum, mating, macro routine, text string, or program line, etc.

NOTE A label consists of a label type followed by a label name enclosed in parentheses except in the case of a label identifying a program line which contains only a label name enclosed in parentheses with no label type.

4.20**label type**

that portion of a DMIS label which specifies the type of a DMIS entity which can be a nominal or actual feature, nominal or actual tolerance, nominal or actual coordinate system, a nominal or actual sensor, an output data format, a datum, a mating, etc., or one of several text string types including a clamp id, clamp serial number, a dimensional measuring device id, a DME software id, etc.

4.21**label name**

that portion of a DMIS label enclosed in parentheses which identifies an individual DMIS entity out of all DMIS entities of the same label type

4.22**line reducible feature**

feature that can be reduced to a geometric line

NOTE Line reducible features are given in (Table 7 — Reducible features).

4.23**measurement sequence**

sequence of statements contained in either a MEAS...ENDMES block or a RMEAS...ENDMES block

4.24**multiple carriage**

DME having two or more carriages

4.25**nominal**

referring to a feature or tolerance, this is the "as-designed" value defined in the product definition of the part

NOTE The nominal is known prior to measurement and is the value, to which the actuals are compared for computing out-of-tolerance conditions.

4.26**orientation**

angular relationship of an entity which can be a nominal or actual feature, nominal or actual sensor, or nominal or actual coordinate system

4.27**orientational deviation**

angular difference between the nominal and actual vectors of the measurement plane and the feature

4.28**Orthonormal**

orthogonal with the integral of the square of each function over a specified interval equal to one

4.29**outer**

signifies that probing occurs from outside, and material lies interior to, the envelope of the defining feature

4.30**output files**

inspection results output from the dimensional measuring equipment

NOTE Unless otherwise noted, output files are inspection results output from the dimensional measuring equipment.

4.31**part coordinate system (PCS)**

datum reference frame associated with the part to be measured

4.32**plane reducible feature**

feature that can be reduced to a geometric plane

NOTE Plane reducible features are given in (Table 7 — Reducible features).

4.33**point reducible feature**

feature that can be reduced to a geometric point

NOTE Point reducible features are given in (Table 7 — Reducible features).

4.34**positional deviation**

scalar difference between the nominal and actual locations of the measurement plane and the feature

4.35**postprocessor**

computer program, which converts a file in DMIS format to a machine-specific format

4.36**pre-processor**

computer program, which converts a data file in a source machine specific format to a file in DMIS format

4.37**Quality Information System (QIS)**

computerized system for the analysis and storage of inspection data

4.38**ram**

moving component of a machine that carries the sensor

4.39**receiving system**

CAD system, QIS, or other computerized system which receives a DMIS results file that is output from a DME

4.40**rotary table system**

application of a rotary table(s) as an adjunct to a DME

4.41**selective processing**

inspection process having the required conditional branching to alter the control of a program based on inspection results or pre-set variables

NOTE

This capability is provided for by incorporation of HLL extensions.

4.42**sensor**

device used on a DME to acquire measurement data

NOTE

For a CMM, the sensor is typically a touch probe. For a video inspection machine, the sensor is typically a camera. Other sensor types include infrared, lasers, non-contact electrical capacitance, and X-rays.

4.43**Standard Exchange Using Product Model Data (STEP)**

specification for the exchange of product model data

4.44**statement**

DMIS definition or command

4.45**tolerance**

measure of acceptable deviation from nominal conditions

NOTE

Geometric tolerances in DMIS are as defined in the ASME Y14.5M-1994 Standard for Dimensioning and Tolerancing.

4.46**tool holder**

device that holds or stores sensors for automatic sensor changing

4.47**vendor**

someone who provides hardware or software that interfaces to others through the DMIS vocabulary

4.48**video inspection machine**

computerized inspection device, which acquires data for analyzing dimensions and tolerances

NOTE The sensor is typically a camera.

4.49**work measuring zone**

measurement volume of a machine as specified by the supplier

NOTE More than one work-measuring zone can be specified for a given machine. Parallel systems, for example, would have a work-measuring zone defined for each carriage.

4.50**workpiece**

object or part to be inspected

5 Language reference**5.1 Syntax and structure**

The DMIS vocabulary is similar in syntax to the APT NC programming language, with major and minor words separated by the slash character "/". Output data formats are similar in syntax to the input data formats. Translators for the DMIS vocabulary can be simple single pass interpreters or complex multiple pass compilers, depending on the implementation method which each individual vendor chooses.

There are two basic types of DMIS statements: process-oriented command statements and geometry-oriented definition statements. Process command statements consist of motion statements, machine parameter statements, and other statements which are unique to the inspection process itself. Definition statements, on the other hand, are used to describe geometry, tolerances, coordinate systems, and other types of data, which may be included in a CAD database. Typically, part models do not include all of the data needed in the DMIS interface, so supplementary data must be added manually. The evolution of CAD systems, though, is in the direction of complete part models, and DMIS has been designed to be compatible with this growth path.

The DMIS vocabulary consists of ASCII characters, which are combined to form words, labels, parameters and variables. These are then combined to form definition statements and command statements. These statements are combined to form program blocks. Program blocks, along with more statements, are combined to form entire DMIS programs.

5.1.1 Characters

In this specification, ASCII characters are represented by the corresponding decimal (not hexadecimal) numbers from the ASCII character table.

When DMIS statements are interpreted, upper and lower case alpha characters are considered to be the same. All of the following, for example, are interpreted as being the same by the DMIS system:

∞ # **TEXT**

∞ # **Text**

∞ # **text**

∞ # **teXT**

The only exception to this rule is that upper and lower case characters are significant in text strings passed with the TEXT, FILNAM, and other statements where text is enclosed with apostrophes. Note also that datum labels can only be upper case alphanumeric characters.

Numerical data included in DMIS can be positive or negative. When negative, the number is preceded by a minus sign character '-'. Positive numbers are either preceded by a plus sign character '+', or by no sign. Numeric integer values do not require the decimal point '.'. Numerical data can be delimited by a colon character ':' when signifying angles in degrees, minutes, and seconds, for example 4:03:47.00.

5.1.2 Words, label names, text strings, parameters, variables, and expressions

Characters are combined to form vocabulary words, labels, parameters, variables, and expressions. Vocabulary words are classified as either major words or minor words. Variables can be character, integer, or real data types, these types are given in 6.35.

5.1.2.1 Words

5.1.2.1.1 Major words

Major words consist of a minimum of two alphanumeric characters. A major word may either be a DMIS statement in itself, or it may indicate a class of statements. When it indicates a class of statements, the major word is modified by one or more minor words. Major words are listed in (Table 1 — DMIS major words). Characterization file major words are listed in (Table 2 — Characterization file major words).

5.1.2.1.2 Minor words

Minor words consist of a minimum of one alphanumeric character. Some minor words are preceded by a minus sign character. A minor word is used to modify a major word or to describe certain parameters in the statement. Minor words are listed in (Table 3 — DMIS minor words). Characterization file minor words are listed in (Table 4 — Characterization file minor words).

Table 1 — DMIS major words

ACLRAT	VA()= ALGDEF	varname=ASSIGN
BADTST	BOUND	CALIB
CALL	CASE	CI()= CLMPID
CS()= CLMPSN	CLOSE	SG()= CMPNTGRP
CONST	CR()= CRGDEF	CRMODE
CROSCL	CRSLCT	CC()= CUTCOM
CZ()= CZONE	CZSLCT	DATDEF
D()/DA()= DATSET	DECL	DECPL
DELETE	DID()= DEVICE	DFTCAS
DISPLY	DMEHW	DI()= DMEID
DMESW	DS()= DMESWI	DV()= DMESWV
DMIS	DMISMD	DMISMN
DO	ELSE	ENDAT
ENDCAS	ENDDO	ENDFIL
ENDGO	ENDIF	ENDMAC
ENDMES	ENDSEL	ENDXTN
EQUATE	ERROR	EVAL

SX()= EXTENS	EXTFIL	F()/FA()= FEAT
FEDRAT	VF()= FILDEF	FILNAM
FINPOS	FI()= FIXTID	FS()= FIXTSN
FLY	FROM	GECOMP
GEOALG	G()= GEOM	GOHOME
GOTARG	GOTO	GSA()= GROUP
IF	INCLUD	varname=ITERAT
JUMPTO	VL()= LITDEF	D()/DA()= LOCATE
LI()= LOTID	M()= MACRO	MA()= MATDEF
MEAS	MD()= MFGDEV	MODE
varname=OBTAIN	OPEN	OP()= OPERID
OUTPUT	PN()= PARTID	PR()= PARTRV
PS()= PARTSN	PL()= PLANID	POP
PRCOMP	PV()= PREVOP	PC()= PROCID
varname=PROMPT	PSTHRU	PTBUFF
PTMEAS	PUSH	Q()= QISDEF
RAPID	READ	RECALL
RM()= REFMNT	R()= REPORT	RESUME
RMEAS	ROTAB	D()/DA()= ROTATE
RT()= ROTDEF	ROTSET	SAVE
SCAN	SCNMOD	SCNPLN
SCNSET	SELECT	SS()= SENSOR
S()/SA()= SNSDEF	SNSET	SGS()= SNSGRP
SNSLCT	SNSMNT	TECOMP
TEXT	TH()= THLDEF	T()/TA()= TOL
TL()= TOOLDF	D()/DA()= TRANS	UNITS
varname=VALUE	V()= VFORM	VW()= WINDEF
WKPLAN	SW()= WRIST	WRITE
XTERN	XTRACT	

Table 2 — Characterization file major words

CHFIL1	ENDCH1
CHFIL2	ENDCH2
CHFIL3	ENDCH3
CHFILE	ENDCHF

Table 3 — DMIS minor words

2D	3D	4POINT	A	ABSL	ACEL
ACLRAT	ACT	ADJUST	ALGOR	ALL	ALLSA
AMT	AN	AND	ANGDEC	ANGDMS	ANGL
ANGLB	ANGLE	ANGLR	ANGRAD	APPEND	APPRCH
ARC	ATTACH	AUTO	AVG	AVGDEV	BACK
BADTST	BEZIER	BF	BND	BOOL	BOUND
BOX	BSPLIN	BUILD	BUTTON	CART	CCW
CHAR	CHECK	CHORD	CIRCLE	CIRLTY	CLRSRF
CM	CODE	COG	COMAND	COMM	COMMON
COMPOS	CONCEN	CONE	CONT	CONTIN	CORTOL
COUNT	CPARLN	CPROFS	CRAD	CRMODE	CRNOUT
CROSCL	CRSLCT	CURENT	CURVE	CW	CYCLE
CYLCTY	CYLNDR	CZSLCT	DATE	DATSET	DEFAULT
DELAY	DELETE	DEPTH	DEV	DFTCAS	DIAM
DIM	DIRECT	DISK	DIST	DISTB	DME
DMIS	DMISMD	DMISMN	DOUBLE	DRAG	EDGE LN
EDGEPT	EDIT	ELLIPS	END	ENDCAS	ENDDO
ENDSEL	ENTITY	EQ	ERR	ERRMODE	ERROR
EXCEPT	EXTERN	EXTREM	FALSE	FDATA	FEATUR
FEDRAT	FEET	FILNAM	FINPOS	FIXED	FLAT
FOCUSN	FOCUSY	FORCE	FZ	GCURVE	GE
GECOMP	GEOALG	GLOBAL	GOTO	GRID	GROUP
GSURF	GT	HIGH	HIST	HUMID	ILLEGALTOUCH
INCH	INCR	INDEX	INFRED	INNER	INPUT
INTGR	INTOF	INTOL	IPM	IPMM	IPS
IPSS	KEEP	L	LASER	LE	LEFT
LIMIT	LINE	LIST	LMC	LN2LN	LOCAL
LONG	LOW	LSTSQR	LT	M	MAJOR
MAN	MATRIX	MAX	MAXINS	MCS	MESACL
MESVEL	MIDLI	MIDPL	MIDPT	MIN	MINCIR
MINCON	MINMAX	MINOR	MM	MMC	MMPS
MMPSS	MNTLEN	MODE	MODEL	MOVEPT	MPM
MPMM	N	NAME	NE	NEXT	NOM
NOMINL	NONCON	NONE	NOROT	NOT	NOTOUCH
NOTRAN	NURBS	OBJECT	OBLQ	OFF	OFFSET
ON	OPER	OR	ORIENT	OUTER	OUTFIL
OUTOL	OUTPUT	OVERWR	PARAM	PARLEL	PARPLN
PARTO	PATERN	PAUSE	PCENT	PCS	PECK
PERP	PERPTO	PICTURE	PIXBTN	PLANE	PLOT

POINT	POL	POS	POSACL	POSVEL	PP
PRBRAD	-PRBRAD	PRCOMP	PRINT	PROBE	PROFL
PROFP	PROFS	PROG	PROJCT	PROJLI	PROJPT
PT2PT	PT2LN	PT2PL	PTBUFF	PTDATA	PTMEAS
QUERY	R	RAD	RADIAL	RADIUS	RAWDAT
RCTNGL	REAL	REPORT	RES	RETRCT	RFS
RIGHT	ROTACL	ROTNUL	ROTORG	ROTTOT	ROTVEL
ROUND	RPM	RPMM	RPTSYC	RTAB	SCALEX
SCALEY	SCNMOD	SCNVEL	SEARCH	SENS	SEQNTL
SGAGE	SHORT	SIMUL	SINGLE	SIZE	SNS
SNSSET	SNSLCT	SNSMNT	SOUND	SPART	SPAWN
SPHERE	START	STAT	STOP	STOR	STRGHT
STROBE	SURF	SYM	SYNC	SYS	TANGPL
TANTO	TECOMP	TEMP	TEMPC	TEMPF	TEMPWC
TEMPWF	TERM	TEXT	THRU	TIME	TITLE
TORUS	TR	TRIGER	TRMATX	TRNOUT	TRUE
UNBND	UNITS	USERDF	USETOL	VEC	VECBLD
VECTOR	VERSION	VERTEX	VIDEO	WAIT	WIDTH
WKPLAN	XAXIS	XDIR	-XDIR	XORIG	XRAY
XVEC	XYAXIS	XYDIR	XYPLAN	XYZAXI	XYZDIR
YAXIS	YDIR	-YDIR	YORIG	YZAXIS	YZDIR
YZPLAN	ZAXIS	ZDIR	-ZDIR	ZORIG	ZVEC
ZXAXIS	ZXDIR	ZXPLAN			

Table 4 — Characterization file minor words

CMM	ENDSPT
FORMA	FORMB
FORMC	FORMD
FORME	FORMF
FORMG	FULL
NONE	NS

5.1.2.2 Label names

Labels consist of two components, a one to three character label type (for example F, TA, DAT...), followed immediately by a label name enclosed within parentheses. Labels are assigned in the inspection program to name features, tolerances, coordinate systems, sensors, output data formats, datums, macro routines, text strings, and program statements, etc; and each of these entities has a unique label type that is defined by DMIS. All label names, except for datum labels, are from one to sixty-four (64) characters where the only permissible characters are alphanumeric (A-Z, a-z, 0-9), dash '-', period '.', and underscore '_'. Datum label names are from one to two upper-case alpha characters for single datums, and two to four upper-case alpha characters with a dash '-' in between for compound datums. All label names are enclosed in parentheses. Label names (except for feature nominals) can be issued only once and cannot be redefined in a particular program; and labels with the same label name (within parentheses) but with a different label type are unique.

In addition to directly specifying label names, the indirect reference '@' operator is available. At most one level of indirect reference may be used. With the indirect reference method, the '@' character precedes the name of a CHAR variable that contains a label that is valid for the label type. If the character variable was declared as multi-dimensional, its name must be followed by an array index. Refer to example A.1.

Labels must not use DMIS reserved words and are not case sensitive.

Jump targets are a special type of label that have an implied label type and are used for example in ERROR, JUMPTO or RESUME statements.

5.1.2.3 Text strings

Text strings are used extensively in the use of DMIS. A text string is a grouping of printable ASCII characters, that is enclosed with apostrophes. All printable ASCII characters are allowed within a text string, but when an apostrophe is required it must be preceded by an additional apostrophe, for a total of two apostrophes. For example if an operator prompt requesting a Supplier's part number were "Enter the Supplier's part number". An appropriate statement would be:

```
TEXT/QUERY, (Sup_Part), 20, AN, L, 'Enter the Supplier''s part number'
```

In the word "Supplier's" there are two apostrophes preceding the "s" this indicates that a single apostrophe is to be contained in the literal string, and that the apostrophe is not the end of the text string. This must be two individual apostrophe characters (' ') not a double quote character (").

If a text string is longer than the maximum allowable line length of 80 characters, the line continuation character (a single dollar sign "\$") must be the last character (before the carriage return and line feed characters) in the line to be continued. A single apostrophe must be the last character to designate the end of the text string. An appropriate statement would be:

```
TEXT/QUERY, (Sup_Part), 20, AN, L, 'Enter the Supplier''s part number for the $
parts that we are going to measure in this group. Do not enter more than $
a single part number. All parts must have the same part number.'
```

The literal string resulting from this TEXT/QUERY statement would be held in the text label (Sup_Part) as:

Enter the Supplier's part number for the parts that we are going to measure in this group. Do not enter more than a single part number. All parts must have the same part number.

5.1.2.4 Parameters

Parameters are values that can be transferred with various statements. Parameters can be used in several instances, such as being passed to a macro with the CALL statement, or placed in variables with the OBTAIN statement. The characteristics of a parameter vary with the statement and context in which they are used.

5.1.2.5 Variables

All DMIS variables are declared and include Boolean, character string, double, integer, long, real, and vector data types. Variable names consist of a string of alphanumeric and underscore characters. The first character of a name must be a letter, and names may have up to sixteen(16) characters. Variable names must not use DMIS reserved words (major words, minor words, etc.) and are not case sensitive. Variables may be explicitly declared as local, global, or common. All variables without an explicitly declared scope in the main program are global to the entire program. Variables without an explicitly declared scope in a routine are local to that routine. Multi-dimensional variable arrays may be declared. The variable array index (base) begins at one. String variables are used as dummy labels in macro routines. Further information about variables can be found throughout section 5.2.

An expression can be substituted for any parameter (for example a numeric value, a literal text string, or label) of the proper data type in any DMIS statement.

EXAMPLE

```

$$
$$ The following example demonstrates substituting a character expression for a
$$ literal text string parameter
$$
TEXT/OPER,CONCAT('The current time is: ',TIME())

```

5.1.2.6 Expressions

A DMIS expression is a variable, constant, literal string, arithmetic expression or logical expression and may contain open and closed parentheses. An arithmetic expression uses the arithmetic operators described in paragraph 5.3.1.1. A logical expression evaluates to .TRUE. or .FALSE.. It contains relational and/or logical operators as described in paragraphs 5.3.1.2 and 5.3.1.3.

5.1.3 Variable assignments

DMIS provides for the assignment of previously declared variables. Variables may be obtained from a definition parameter (for example, OBTAIN), set to a measurement result (for example, VALUE), or assigned a value (for example, ASSIGN).

A previously declared and assigned variable can be substituted for any parameter (for example a numeric value, a literal text string, or label) of the proper data type in any DMIS statement. A character string (type CHAR) variable can be substituted for any single quoted (apostrophe) string in the DMIS syntax.

EXAMPLE

```

$$
$$ The following example demonstrates substituting a character variable for a
$$ literal text string parameter
$$
DECL/GLOBAL,CHAR,80,MYTEXT
MYTEXT=ASSIGN/'This is my text message to the operator'
TEXT/OPER,MYTEXT

```

5.1.4 DMIS command and definition statements

Major words, minor words, parameters, labels, text, and variables are combined to form either command statements or definition statements.

5.1.4.1 Command statements

Command statements direct the DME or receiving system to perform some function. Often a command statement contains a reference to a previous definition statement. In this case, the definition statement is used in the execution of the command statement.

The major word indicates the command statement or command statement type. This is followed by a slash character "/" and one or more minor words, parameters, labels, text, and/or variables.

In some cases, a major word by itself is valid. In this case, no slash, minor words, parameters, labels, text, or variables occur in the statement.

Example:

DFTCAS

In other cases, a major word and a minor word, with no parameters, labels, text, or variables, is a valid DMIS statement.

Example:

BADTST/ON

Sometimes a major word with parameters and no minor words, labels, text, or variables may be required.

Example:

FROM/12.341,427.284,183.294

In still other cases, a major word along with minor words, parameters, labels, text, and/or variables is required.

Example:

CALL/EXTERN,DME,'bolt_circle',WAIT,5,20

The basic structure of a command statement is as follows:

MAJOR_WORD/MINOR_WORD(S),PARAMETER(S)

5.1.4.2 Definition statements

Definition statements describe the detailed information for specific cases of geometry, tolerance, machine specifics, and other conditions pertaining to inspection part programming. The basic structure of a definition statement is similar to that of a command statement, except that the major word is preceded by a label and an equal sign "=". A definition statement describes one specific set of conditions and assigns a label name to it for easy reference. The label name is then used in another statement to refer to the object defined.

5.1.4.3 Command and definition statement examples

As an example of the various statements described above, consider the following list of valid DMIS statements:

```
FINPOS/ON
F(CIRCLE_1)=FEAT/CIRCLE,INNER,$
CART,10,10,5,0,0,1,4
MEAS/CIRCLE,F(CIRCLE_1),3
GOTO/10,10,5
PTMEAS/CART,12,10,5,-1,0,0
PTMEAS/CART,8,10,5,1,0,0
PTMEAS/CART,10,12,5,0,-1,0
ENDMES
```

- ∞ # The first line is a command statement consisting of only a major word and a minor word, with no parameters. This directs the fine positioning feature of the DME to be enabled.
- ∞ # The second line is a definition statement containing a major word, a label, minor words, and parameters. It describes the size, location, and orientation of a feature that is a circle, and assigns to it the name "CIRCLE_1". With the line continuation character "\$", this is continued on the third line.
- ∞ # The fourth line is a command statement consisting of a major word with a minor word and parameter which directs the DME to measure a circle. This command statement also has a pointer to the definition of the circle.
- ∞ # The next line is a command statement consisting of only a major word and parameters that direct the DME to move to the given position.
- ∞ # The next three lines are command statements consisting of a major word with a minor word and parameters that direct the DME to take point measurements at the given locations.
- ∞ # The last line is a command statement that is a major word by itself, which indicates that the measurement sequence is complete.

A list of major words is given in (Table 1 — DMIS major words), and a list of minor words is given in (Table 3 — DMIS minor words). These words, when combined properly with the appropriate parameters, make up the valid DMIS vocabulary. A detailed description of each command statement and definition statement in the DMIS vocabulary is given in Section 6.

5.1.5 Delimiters, blank lines, spaces, and tabs

5.1.5.1 Delimiters

Slashes, commas, parentheses, apostrophes, quotation marks, and brackets are used as delimiters in DMIS.

5.1.5.1.1 Slash

The forward slash character '/' is used to separate major and minor words. A major word can either be a statement by itself, or can have some modifying minor words or parameters. When a major word is used by itself, it has no slash character:

MAJOR_WORD

Example:

ENDMES

When a major word is followed by a minor word, the slash follows the major word and precedes the minor word.

MAJOR_WORD/MINOR_WORD

Example:

SCAN/PAUSE

When a major word is followed by one or more parameter(s), the slash follows the major word and precedes the first parameter.

MAJOR_WORD/parameter(s)

Example:

FROM/302,481,183

In the output of GCURVE, GSURF, and RAWDAT, each data point is preceded by a slash without major or minor words.

The slash character is also used as the division operator in numeric expressions.

5.1.5.1.2 Comma

Commas ',' are used as general delimiters to separate minor words and parameters:

MAJOR_WORD/MINOR_WORD1,MINOR_WORD2,param1,param2

5.1.5.1.3 Parentheses

Parentheses '(' and ')' are used as delimiters. A label name is preceded with a left parenthesis and followed by a right parenthesis.

For example, if a circle has a label name of CIRCLE_1, it's nominal is denoted by the label:

F(CIRCLE_1)

All labels have balanced parentheses. That is, a left parenthesis is always followed by a right parenthesis.

Below is an example of an unconditional branch to a jumptarget, which is labelled "BEGIN":

```
GOTARG/.....
GOTO/.....
ENDGO
JUMPTO / (BEGIN)
SNSLCT/.....
(BEGIN)
MEAS/CIRCLE,F(CIRCLE_1),5
PTMEAS/.....
PTMEAS/.....
```

The JUMPTO statement causes an unconditional branch to the statement labelled BEGIN, which precedes the MEAS/CIRCLE statement.

Parentheses are also used in arithmetic, relational, logical, and character expressions, refer to paragraphs 5.3.1.1, 5.3.1.2, 5.3.1.3, and 5.3.1.4.

5.1.5.1.4 Apostrophe

The apostrophe '-' is used to delimit the start and end of a text string.

For example:

```
FILNAM/'041380385 test dated 1999/10/17',04.0
```

Use a total of two apostrophes, one before the one required, when an apostrophe is required within a text string.

For example:

```
TEXT/OPER,'Use Paul''s setup instructions from the last job.'
```

This results in the following message sent to the screen:

Use Paul's setup instructions from the last job.

The apostrophe '-' is also used to indicate the dummy parameters in a macro definition that will be replaced with labels when the macro routine is called. Refer to paragraph 5.2.4.3.2 for an example of a macro routine.

5.1.5.2 Blank lines, spaces, and tabs

Blank lines and spaces have no significance in DMIS. Blank lines are insignificant and are ignored during translation. Spaces are not allowed within DMIS words, label names, parameters and variables.

Spaces and tabs may be used:

- a) between tokens of the language (DMIS words, label names, variable names, numbers, and delimiters).
- b) inside text strings passed in TEXT, FILNAM, and other statements using 'text'.
- c) inside comments.

Spaces and tabs may not be used anywhere else. Spaces and tabs have no significance where they are correctly used, except in text strings.

Blank lines, spaces, and tabs may add clarity in manual programming or analysis, but they are ignored by computer programs in the translation of a DMIS file (except spaces and tabs in text strings).

For example:

```
F(CIR_1)=FEAT/CIRCLE, INNER,CART, 10, 10, 5, 0, 0, 1, 25
MEAS / CIRCLE, F(CIR_1), 5
```

is interpreted the same as:

```
F(CIR_1)=FEAT/CIRCLE, INNER, CART, 10, 10, 5, 0, 0, 1, 25
MEAS/CIRCLE, F(CIR_1), 5
```

5.1.6 Line length

A DMIS file consists of records with a variable line length. Each line is terminated with a carriage return(CR) and line feed(LF). A single dollar sign '\$' acts as a line continuation symbol.

DMIS statements have a variable line length, with a maximum of 80 characters per line, including the CR and LF. The statement length can be unlimited. When the length of a statement is greater than 80 characters, the statement must be continued on the next line using a single dollar sign '\$' as the last character (before the carriage return(CR) and line feed(LF) characters) in the line to be continued.

For example:

```
DISPLY/PRINT, V(OUT_1), TERM, $
V(OUT_2)
```

Is interpreted the same as:

```
DISPLY/PRINT, V(OUT_1), TERM, V(OUT_2)
```

And:

```
Q(MyLabel)=QISDEF/'This is a $100.00 program'$
, '100.00'
```

Is interpreted the same as:

```
Q(MyLabel)=QISDEF/'This is a $100.00 program', '100.00'
```

5.1.7 Programming comments

Programming comments are lines of text inserted into a DMIS inspection program to aid in program debugging and to document portions of the program. They are not to be interpreted by any automated system.

Programming comments are signified by two dollar signs '\$\$' as the first two characters in a line. The computer program executing or converting DMIS commands will ignore all lines, which begin with a double dollar sign.

The following are examples of a programming comment:

```
$$ This is a programming comment to be
$$ ignored by the DMIS system.
$$ In this section of the program, the datums
$$ will be established.
$$ This part of the program will be performed
$$ in manual mode.
```

5.1.8 Operator input

5.1.8.1 The TEXT statement

Various forms of text are supported in the DMIS vocabulary. Header data are supported through the TEXT/QUERY and TEXT/RES statements. This allows each user to create header data to individual requirements.

The TEXT/QUERY statement is passed from the input program to the DME and requires a response from an operator or some other system. A part number, for example, may either be keyed in by an operator or read in through a bar code scanner. The statement specifies a label for the query, the number of characters allowed for the response, whether it is alphanumeric, numeric, or any printable ASCII character, and whether it is right or left justified. When the response is entered, it is passed to the output file as a TEXT/RES statement with the same

label as the corresponding query. The other parameters (length, type, and justification) are also passed to the output file. Typical queries in the input file might appear as follows:

```
TEXT/QUERY, (name), 40, A, L, 'Enter Operator Name: '
TEXT/QUERY, (today), 10, AN, L, 'Enter today' 's date: '
```

Here, "name" and "today" are text labels. The labels are followed by length, type, and justification parameters, and then by the message, which will be output to the operator. The corresponding responses in the output file might appear as follows:

```
TEXT/RES, (name), 40, A, L, 'John Smith'
TEXT/RES, (today), 11, AN, L, '2000/06/14'
```

The responses "John Smith" and "2000/06/14" have been typed in by the operator at run time.

Other forms of the TEXT statement allow text messages to be output to the screen (for the operator) and/or to the output file. Text may also be conditionally sent to the operator such that it will only be output if the DME executes a statement in manual mode. This allows operator instructions to be programmed and suppressed unless needed.

5.1.8.2 The PROMPT statement

The PROMPT statement provides a richer mechanism for operator prompting; and adds the ability to initialize program variables from operator input.

5.1.9 Data output

Output data from the DME can be sent to any of several devices. The data to be sent to each device can be specified to be in DMIS format, in vendor format, or both. Output of some DMIS statements is caused by the execution of an OUTPUT statement, whereas other DMIS statements are passed directly to all opened DMIS output files when executed. Each individual statement reference gives that statement's particular DMIS output characteristics.

Example: The ROTDEF statement is passed to the output file when executed.

Output devices can be classified into two categories, those controlled by the DME and those controlled by the user. Input devices are controlled by the user.

5.1.9.1 Default output devices

The DME default device(s) to which output data will be sent is controlled by the DISPLY statement. This statement either turns all output off or specifies one or more of the following as output devices:

- ∞ # TERM (display terminal at the DME)
- ∞ # PRINT (printer)
- ∞ # STOR (magnetic storage such as a floppy or hard disk)
- ∞ # COMM (an auxiliary communications port)

The DISPLY statement must appear in the DMIS input program before any statements causing output. If the VFORM statement is referenced, it must be defined prior to its use in the DISPLY statement. The DMIS output file must have a FILNAM statement as its first line. When output is required on multiple devices, they must all be identified in the first DISPLY statement to insure that the FILNAM statement will appear on all selected devices. As illustrated below the DISPLY statement can appear more than once in a DMIS program. It is important to note, however, that the last DISPLY statement is always active, and all previous settings are lost. To activate or reset a previous setting, a new DISPLY statement must be issued.

```
$$ Sample DMIS Input Program
DMISMN/'Sample DMIS 4.0 file' 2000/02/04',04.0
```

```
V (Vendor_output)=VFORM/ALL
DISPLY/PRINT,DMIS,STOR,V (Vendor_output)
FILNAM/'Sample DMIS Input Program',04.0
```

executable statements

```
DISPLY/TERM,DMIS
```

executable statements

```
DISPLY/PRINT,DMIS,STOR,V (Vendor_output)
```

executable statements

```
ENDFIL
```

When STOR is used in the DISPLY statement, the data will be directed to the storage device. The filename chosen for the storage device is DME-dependent.

5.1.9.2 User defined input/output devices

The user device(s) to which output data will be sent, or from which input data is received, is controlled by the DEVICE and OPEN statements. The DEVICE statement identifies the device type (printer, terminal, communications port or storage device) and specifies a label for it. The OPEN statement opens the previously defined system device and establishes the connection's input/output attributes.

A device opened with the SNS, PCS, FEATUR, or RTAB parameter is accessed only with the SAVE or RECALL statements. Such devices are used for reading and writing DME specific format system information.

Existing output storage devices can be appended to or overwritten. If the file specified in an OPEN statement being opened for either appending or overwriting does not exist, a new file is opened.

Output will go to a device opened with the FDATA parameter in either DMIS or vendor format as specified in the OPEN statement. The text and version from the FILNAM statement is written to new DMIS output files, that is, DMIS output files which do not already contain the FILNAM statement as their first line of output. The FILNAM statement is also passed to existing DMIS output files when they are opened for overwriting. If user defined DMIS output files are already open when the FILNAM statement is executed, the FILNAM statement is passed to those files if they do not yet contain the FILNAM statement as their first line of output. User defined DMIS output files to which the FILNAM statement is passed are storage devices initially opened with the OPEN/DID(label),FDATA,DMIS,OUTPUT,OVERWR statement and which may be subsequently re-opened with the OPEN/DID(label),FDATA,DMIS,OUTPUT,APPEND statement after being closed.

The ENDFIL statement will close all open devices. Open storage devices such as files are not deleted when closed. If the open device is a DMIS output file then the ENDFIL statement is passed to that output file before it is closed.

The CLOSE statement can be used to close devices opened with the OPEN statement before the end of a DMIS program, that is, before the ENDFIL statement is encountered. When a storage device such as a file is closed, it can be deleted by using the DELETE parameter. The storage device will not be deleted if either the KEEP or END parameters are used, or when no parameter is specified. The KEEP parameter is of general use for all storage devices. The device or file is simply closed without being deleted. The END parameter is of particular use with DMIS output files. When the CLOSE/DID(label),END statement is executed, an ENDFIL statement is passed to the storage device as if the end of the DMIS program had been reached, that is, as if the ENDFIL statement had been encountered.

```
DMISMN/'Sample program with DEVICE statements',04.0
V (V1)=VFORM/NOM,ACT,DEV,AMT
DISPLY/PRINT,V (V1),STOR,DMIS
FILNAM/'DEVICE statements program',04.0
$$ The DMIS default output file is opened and the FILNAM statement is
$$ passed to the file. The vendor format printer is opened.
```

executable statements

\$\$ Any DMIS output is passed to the default DMIS output file as
 \$\$ statements are executed.

DID(mydata)=DEVICE/STOR,'/user/dmis/myfile.dat'

OPEN/DID(mydata),FDATA,DMIS,OUTPUT,OVERWR

\$\$ The user defined DMIS output file is opened and the FILNAM statement
 \$\$ is passed to the file.

executable statements

\$\$ Any DMIS output is passed to both the default DMIS output file and
 \$\$ the user defined DMIS output file as statements are executed.

CLOSE/DID(mydata),KEEP

\$\$ The user defined DMIS output file is closed (no ENDFIL statement is
 \$\$ passed to the file).

executable statements

\$\$ Any DMIS output is now passed only to the default DMIS output file
 \$\$ as statements are executed.

OPEN/DID(mydata),FDATA,DMIS,OUTPUT,APPEND

\$\$ The user defined DMIS output file is reopened.

executable statements

\$\$ Again, any DMIS output is passed to both the default DMIS output
 \$\$ file and the user defined DMIS output file as statements are
 \$\$ executed.

CLOSE/DID(mydata),END

\$\$ The ENDFIL statement is written to the user defined DMIS output file
 \$\$ before being closed.

ENDFIL

\$\$ The ENDFIL statement is written to the default DMIS output file
 \$\$ before it is closed. The vendor format print device is closed.

Below are additional examples of user defined output devices. They demonstrate how the DEVICE statement interacts with the OPEN, CLOSE and VFORM statements.

Example 1.

Two output devices, c:\users\reports\test_report_1.txt and c:\users\reports\test_report_2.txt, are opened simultaneously. Due to the OVERWR Minor word option, both files will be overwritten should they already exist. The output of the measured point will be directed to DID(filestore1) in a user defined vendor format defined by the VFORM statement. In addition, the output of the point is also directed to output device DID(filestore2) in DMIS output format.

After the output has been performed, the sample DMIS program closes both devices. Device DID(filestore1) is closed with the KEEP minor word option which keeps the file, unlike the DELETE minor word option which would remove the file after closing. Device DID(filestore2) is closed using the END minor word option which causes the ENDFIL statement to be written into the DMIS output file after which the file is closed.

FILNAM/'Some_File_Name',04.0

V(allout)=VFORM/ALL

DID(filestore1)=DEVICE/STOR,'c:\users\reports\test_report_1.txt'

DID(filestore2)=DEVICE/STOR,'c:\users\reports\test_report_2.txt'

OPEN/DID(filestore1),FDATA,V(allout),OUTPUT,OVERWR

OPEN/DID(filestore2),FDATA,DMIS,OUTPUT,OVERWR

F(circle_1)=FEAT/POINT,CART,100,100,20,0,0,1

MEAS/POINT,F(circle),1

ENDMES

OUTPUT/FA(circle_1),TA(..),TA(..)

```
CLOSE/DID(filestore1),KEEP
CLOSE/DID(filestore2),END
```

Example 2.

In this example, every time the do-loop is re-executed, a new output file is created. The result of this sample code is that the files "\\data\\mydrive\\spc_001_file.spc through "\\data\\mydrive\\spc_130_file.spc will be created. Each of these output files will contain one measurement output of FA(label) once this sample program is completed.

```
FILNAM/'This_File_Name',04.0
DID(incremental_file)=DEVICE/INCR,'\\data\\mydrive\\spc_???_file.spc'
V(act_dev)=VFORM/ACT,DEV
DECL/INTGR,i
DO/i,1,130,1
    OPEN/DID(incremental_file),FDATA,V(act_dev),OUTPUT,APPEND
    F(Point_1)=FEAT/POINT,CART,100,100,20,0,0,1
        MEAS/POINT,F(Point_1),1
        ENDMES
    OUTPUT/FA(Point_1),TA(..),TA(..)
    CLOSE/DID(incremental_file),KEEP
ENDDO
```

5.1.9.3 Output format

The data format in which the output data will be sent can be either DMIS format, a format unique to the DME vendor, or both. The DMIS format for output data is similar in syntax to the format for input data. Vendor formatted data varies widely with each vendor and is not standard.

The VFORM statement specifies the data content of vendor formatted output data for example nominals, actuals, deviations, etc. The actual physical format of these data, however, is left to each individual vendor. If the VFORM statement specifies nominals, actuals and deviations, for example, one DME may output nominals, then actuals, and then deviations. Another DME may output deviations, then nominals, and then actuals. One DME may output the data in a tabular format. Another DME may output data all in a single column. The DMIS vocabulary does not control the format of vendor data; it specifies the data content only.

5.1.10 Program structure

DMIS program structure generally involves only one program file. However, the DMIS high level language (HLL) extensions allow modular programming. Modular programming provides the capability to reuse modules of existing code. As a result, a DMIS part program could consist of multiple files that as a whole define a complete inspection part program. Modularity can be obtained by using external file declarations (EXTFIL), include statements (INCLUDE), and modular (DMISMD) files. DMIS statements can be combined to form program units and blocks, Program Main, and Program Module.

5.1.10.1 Program units

Program units are designated by a file. There are two types of program units.

5.1.10.1.1 Program main

The program main unit is designated by the DMISMN statement and terminates with an ENDFIL statement.

5.1.10.1.2 Program module

The program module unit is designated by the DMISMD statement and terminates with an ENDFIL statement.

External DMIS files (that is, modules) containing declarations, procedures and functions for external reference by the program main unit involves the following:

```
DMISMD/'My_Module_ID',04.0
FILNAM/'My_Filename',04.0
```

External Declarations

Data Declarations

Macro Definitions

DMIS Statement Sequence

```
ENDFIL
```

The general structure of a DMIS high level language (HLL) module typically involves the following:

```
DMISMD/'SQRKEY_Feature_Definitions',04.0
FILNAM/'/user/dmis/defn/sqrkey',04.0
DECL/COMMON,REAL,X,Y,Z
DECL/GLOBAL,INTGR,sm_num
M(spechole)=MACRO/x,y,z
```

executable statements

```
ENDMAC
```

```
M(crunchem)=MACRO/crunch_numbr,num1,num2
```

executable statements

```
ENDMAC
```

```
ENDFIL
```

5.1.10.2 Program blocks

A program block is a logically grouped series of statements that perform some function. A program block is identified by the first and last statements in the block. There are multiple types of program blocks in DMIS.

5.1.10.2.1 Calibration sequences

Calibration sequences begin with the CALIB statement and are terminated with the ENDMES statement.

5.1.10.2.2 Measurement sequences

Measurement sequences begin with the MEAS or RMEAS statement and are terminated with the ENDMES statement.

5.1.10.2.3 Motion sequences

Motion sequences direct a series of non-measurement moves. They begin with a GOTARG statement and are terminated with an ENDGO statement.

5.1.10.2.4 IF Conditional branching

The IF...ENDIF block evaluates a logical expression and branches based on the test evaluation. An IF block begins with an IF statement, may contain an optional ELSE statement, and ends with an ENDIF statement.

5.1.10.2.5 Macro definitions

Macro routines are subroutines, which can be executed with different values in the place of parameters. Macros begin with a MACRO definition statement and are terminated with an ENDMAC statement.

5.1.10.2.6 Select by case branching

The SELECT...ENDSEL block executes statements according to the results of one or more CASE tests. A SELECT block begins with a SELECT statement, contains multiple CASE blocks with an optional DFTCAS block statement, and terminates with an ENDSEL statement.

5.1.10.2.7 Case statement(s)

The CASE...ENDCAS block executes statement(s) bounded between a CASE and an ENDCAS statement dependent on tests on the SELECT argument.

5.1.10.2.8 Default case statement

The DFTCAS...ENDCAS block executes statement(s) bounded between a DFTCAS and an ENDCAS statement, only if none of the CASE clauses are activated in the SELECT block.

5.1.10.2.9 Looping sequence

The DO...ENDDO block provides the capability of repeating a sequence of instructions based on an initial and limit value at a specified increment. A DO loop begins with a DO statement and terminates with an ENDO statement.

5.1.10.2.10 External file declaration

The external file declaration block declares file(s) containing program units that are used within the local DMIS program. A external file declaration begins with a XTERN statement and terminates with an ENDXTN statement.

5.1.11 File structure

Programs (Input Files) and output files consist of a combination of command statements, definition statements, and program blocks.

5.1.11.1 Program file or input file

A DMIS inspection program consists of command statements, definition statements, and program blocks. DMIS Programs always have a DMISMN statement as the first executable statement in the program and an ENDFIL statement as the last executable statement in the program.

5.1.11.2 Results file or output file

Dimensional measuring equipment DMIS output files are similar in syntax to the input files. Measurement results are passed in the form of actual feature and tolerance definitions that are similar in format to the input file (nominal) definition statements. In addition, certain command statements are passed to the output file to indicate machine parameter settings at the time of measurement, that are also similar in format to the input file command statements. The DMIS output file(s) always has a FILNAM statement as the first line and an ENDFIL statement as the last line.

5.1.12 Programming considerations

5.1.12.1 Program structure

Some general programming considerations should be kept in mind when generating an inspection program using the DMIS vocabulary. These considerations include: program declaration, program structure, entity definitions, branching, modality, and default settings.

5.1.12.2 Program declarations

The declarations in DMIS programs include external files, variables, and macro declarations. Details as given in paragraph 5.2.1 and sub paragraphs.

5.1.12.3 DMIS statements

The body of a DMIS program has specific structure, defined as follows.

5.1.12.3.1 Definition statements

All features, tolerances, and sensors are defined in the DMIS input program prior to their use in other statements.

5.1.12.3.2 Modal and non-modal statements

Some statements in the DMIS vocabulary are either modal or non-modal.

Modal statements usually set some machine parameter or other condition that remains in effect until the statement is reissued. Non-modal statements are in effect only for a single execution of the statement. As a result, when branching in a program, caution should be exercised to ensure that modal settings are properly set both before and after the branch occurs. Branches are caused by the IF and JUMPTO statements.

5.1.12.4 Default settings

There are no default settings in the specification. As a result, users must be aware that initial DME settings or residual setting(s) from preceding operations may be in effect.

5.2 Execution and control

5.2.1 Declaration statements

The declarations in DMIS programs include external files, variables, and macro definitions.

5.2.1.1 External declaration statements

DMIS provides for the declaration of external files that are used within the local DMIS program unit or DMIS module (DMISMN, DMISMD). These files may contain external programs (that is, DMISMD), shell scripts, DME routines, and DMIS macros that may be invoked by the CALL/EXTERN statement. All external files must be declared within an XTERN...ENDXTN external declaration block. An XTERN statement starts the external declaration block. An ENDXTN statement ends the external declaration block. The external declaration block includes EXTFIL statements that declare a file pathname containing program definitions for the DMIS program unit.

Synopsis of an external declaration block is as follows:

```
XTERN
EXTFIL/var_1, 'pathname'
ENDXTN
```

5.2.1.2 Variable declaration statements

All DMIS variables must be declared with the DECL statement. DMIS variable types include Boolean, integer, long integer, real, double real, character string, and vector. Variable names consist of a string of 1 to 16 characters of alphanumeric and underscores. The first character of a name must be an alpha character. Variable names must not use DMIS reserved words (major words, minor words, definition labels, and function names) and are not case sensitive. Typically, a variable declaration section is used to declare one or more variables of a specific data type for the DMIS program unit DMISMN, DMIS module DMISMD and macros.

Synopsis of a variable declaration section is as follows:

```
DECL/var_1 var_2 var_3
```

A previously declared and assigned variable can be substituted for any parameter (that is, numeric value, or label) of the proper data type in any DMIS statement. When such a variable appears in a statement, any DMIS output text shall contain the value at the time of execution.

```
DECL/CHAR,10,myclamp
myclamp=ASSIGN/ 'CLAMP12'
CI (CLAMP) =CLMPID/myclamp
```

5.2.2 Definition statements

5.2.2.1 Algorithmic definitions

The algorithmic definitions are used to define specific algorithms to particular inspections or features. The ALGDEF statement primarily supports video inspection devices with additional sensor capabilities as they relate to different vision algorithms and filters. The intent of the GEOALG statement is to designate a specific substitute feature data fitting algorithm (for example, least squares, minmax, minimum circumscribed, maximum inscribed) for creating feature actuals.

5.2.2.2 Boundary definitions

The BOUND statement assigns boundaries to features or tolerances. The feature(s) and/or tolerance definition(s) referenced in the BOUND statement must occur in the file prior to the BOUND statement. Also, the BOUND statement must occur prior to the measurement of a feature that is bounded. When a tolerance is bounded, the BOUND statement must occur prior to the EVAL or OUTPUT statements for the feature to which the tolerance is applied.

5.2.2.3 Geometry definitions

The geometry definitions are used to define specific geometry with CAD data information.

5.2.2.4 Macro definitions

DMIS provides for the definition of macro routines to perform repetitive functions without having to reprogram all of the statements. A macro is declared and defined between the MACRO...ENDMAC block. Then a macro is available for invoking by using a CALL statement. Typically, a macro definition section defines macros for use within the DMIS program unit or DMIS module (DMISMN, DMISMD). Synopsis of a macro definition section is as follows:

```
M(label)=MACRO/argument_list
```

```
external declarations
```

```
data declarations
```

```
DMIS statement sequences
```

```
ENDMAC
```

5.2.2.5 Manufacturing device definitions

Statements defining manufacturing devices and tools, MFGDEV and TOOLDF, are associated with a compensation statement, CUTCOM, to allow process adjustment based on inspection results. These statements provide the necessary data to close the loop for manufacturing cells requiring in process verification.

5.2.2.6 Mating definitions

The mating definitions are used to define mating information between either features and geometry or between features and other features.

5.2.2.7 Quality information system definitions

The Quality Information System (QIS) definitions are used to define quality information features. This will allow the management and use of these features. In support of QIS requirements, the statements are included to define the identification of such entities as: DMEs, DME software versions, operators, part numbers, revision levels, inspection plans, fixtures, clamps, previous operations, etc.

5.2.2.8 Tolerance definitions

The tolerance definitions in DMIS are generic, in that there are no pointers to the feature to which the tolerance is applied. The association between the tolerance and the feature is managed with the EVAL and OUTPUT statements. The tolerance definition need not occur in a DMIS file immediately following the feature definition, but the tolerance definition must occur before being used by either an EVAL or OUTPUT statement.

5.2.3 Program statement sequences

5.2.3.1 Selective processing

Selective processing is the capability to allow program flow control to be altered based on either the results of measurements, or the values of externally supplied parameters. This capability can be obtained by using the HLL extensions. This includes using the OBTAIN and ASSIGN statements along with the HLL conditional branching statements (for example, IF, SELECT).

The general structure of a DMIS high level language (HLL) module typically involves the following:

```
DMISMD/'SQRKEY_Feature_Definitions',04.0
FILNAM/'/user/dmis/defn/sqrkey',04.0
DECL/COMMON,REAL,X,Y,Z
DECL/GLOBAL,INTGR, sm_num
M(spechole)=MACRO/x,y,z

    executable statements

ENDMAC
M(crunchem)=MACRO/crunch_numbr,num1,num2

    executable statements

ENDMAC
ENDFIL
```

DMIS high level language (HLL) program structure generally involves the following:

```
DMISMN/'Prog_ID',04.0
FILNAM/'Filename',04.0

    External File Declarations

    Data Declarations

    Macro Definitions

    DMIS Statement Sequence

ENDFIL
```

An example of a general structure of a DMIS high level language (HLL) program typically involves the following:

```

DMISMN/'SQRKEY_Main_Program',04.0
FILNAM:'/user/dmis/prgm/sqrkey.dmi',04.0
$$ External File Declaration
XTERN
EXTFIL/DMIS,'/user/dmis/techniques'
EXTFIL/SYS,'/user/algorithms/circles.alg'
ENDXTN
$$ Data Declarations
DECL/COMMON,INTGR,Z_DIM,A[10]
$$ Macro Definitions
M(cir_meas)=MACRO/x,y,z,diam

    executable statements

ENDMAC
M(cyl_meas)=MACRO/x,y,z,diam,len

    executable statements

ENDMAC
$$ DMIS Body Statements

    executable statements

ENDFIL

```

5.2.3.2 Programs and modules

The body of an inspection program (that is, feature definitions, tolerance definitions, bound statements, and output statements) must be sequenced properly. In general, entities must be defined prior to their use. All DMIS input programs must begin with a DMISMN statement and all DMIS modules must begin with a DMISMD statement. Both must end with an ENDFIL statement. A DMISMN statement indicates to the receiving system that the program is the DMIS input main program. In a DMIS input main program, the ENDFIL statement indicates that the end of the main program has been reached. In a DMIS input module, the ENDFIL statement transfers program execution back to the calling program at the line following the CALL statement. DMIS input modules that are not part of the main DMIS program can be called. All modules will have a DMISMD statement as their first line of executable code. This allows for the separation of modules and main programs and accommodates a more flexible approach.

5.2.3.3 Evaluation

The EVAL and OUTPUT statements are commonly used immediately following a measurement sequence to evaluate features with tolerances. DME's with memory constraints may require feature actuals to be saved after measurement and later recalled for evaluation or output. When the SAVE statement is used for a feature and the PTBUFF/ON statement was used prior to the time of measurement, the feature actual and its data points are saved. If the PTBUFF/OFF statement was used prior to the time of measurement, then only the feature actuals are saved.

For example, two features are required to verify parallelism and one of them might have been measured prior to the EVAL or OUTPUT statements. Since it is required that the feature actual(s) exists, because of possible memory constraints, the feature actual should be saved by the SAVE statement.

The use of the EVAL statement is optional in determining the calculation of a tolerance to a feature because the evaluation of a tolerance to a feature can also be performed with the OUTPUT statement. The EVAL and OUTPUT statements are commonly used immediately following a measurement sequence but can occur anywhere following the measurement sequence providing the feature actual was saved with the SAVE statement. When the SAVE statement is used for features, it is the feature actuals that are saved, not the data points.

The EVAL statement can be used to generate a tolerance actual, that is, a TA(label), that can then be obtained with the OBTAIN statement and used in a selective processing test instead of being sent to an output file with the OUTPUT statement. The EVAL and OUTPUT statements can support several tolerance labels for one to two feature labels. Two feature labels are required for relationship tolerances for example with the , TOL/ANGLB, and TOL/DISTB statements.

When executed, the OUTPUT and/or EVAL statement causes the evaluation of the feature actual(s), FA(label), and the tolerance actual, TA(label). Additionally, the OUTPUT statement causes data to be output. The OUTPUT statement may specify a report format, which is additional information to be sent to the output file. If this is the case, the REPORT statement format must have been defined prior to the OUTPUT statement. Evaluation means the results are available for further use.

Output statements specify output data content, format, and the device(s) to which the data will be sent. Output statements can also be used to send instructions to the receiving system.

5.2.3.4 Measurement

When executed, the CALIB, MEAS or RMEAS statements cause a feature to be measured. They must occur after the feature is defined, after the BOUND statement (if the feature is bounded) and prior to the EVAL and OUTPUT statements for that feature.

5.2.3.5 Miscellaneous

Miscellaneous statements provide for the communication of user-specific data for special applications including the DMIS, DMEHW, and DMESW statements.

5.2.3.6 Example program statement sequences

As an example of the program statement sequences described above, consider the following:

```

$$ group 1) -----
R(REP_1)=REPORT/DATE
F(CYL_1)=FEAT/CYLINDR, INNER, CART, 0, 0, 5, 0, 0, 1, 8
F(PLN_1)=FEAT/PLANE, CART, 0, 0, 0, 0, 0, -1
F(PLN_2)=FEAT/PLANE, CART, 0, 0, 10, 0, 0, 1
F(PLN_3)=FEAT/PLANE, CART, 0, 0, 10.25, 0, 0, 1
T(PERPX)=TOL/PERP, .05, RFS, DAT(A), RFS
T(dia_1)=TOL/DIAM, -.002, +.002

$$ group 2) -----
BOUND/F(CYL_1), F(PLN_1), F(PLN_2)
BOUND/T(PERPX), F(PLN_1), F(PLN_3)

$$ group 3) -----
MEAS/CYLINDR, F(CYL_1), 12

    executable statements

ENDMES
SAVE/FA(CYL_1)

$$ group 4) -----
EVAL/FA(CYL_1), T(PERPX)
$$ (OPTIONAL STATEMENT)

$$ group 5) -----
OUTPUT/FA(CYL_1), TA(PERPX), TA(dia_1), R(REP_1)

$$ group 6) -----
F(NEXT_HOLE)=FEAT/CIRCLE, INNER, CART, 5, 4, 3, 0, 0, 1, 8
MEAS/CIRCLE, F(NEXT_HOLE), 7
ENDMES

```

The statements are split into six groups for the purpose of illustration.

In group 1, the feature definitions and tolerance definitions may be listed in any order, providing they precede:

- any required BOUND statements,
- EVAL statements, or
- OUTPUT statements.

The BOUND statements for features and tolerance, group 2, when required, must occur after the referenced definitions.

Once the feature definitions and required BOUND statements are issued, the feature can be measured in group 3.

The optional EVAL statement can occur immediately after the measurement sequence or at the end of the file.

The OUTPUT statement, group 5, may be listed in any order with other DMIS statements, but must follow the MEAS sequence for the corresponding feature measurement, and must follow the required tolerance definition. The OUTPUT statement can occur at the end of the file.

Although the REPORT and TOL statements are shown in group 1, they can be defined anywhere in groups 1, 2, or 4, as long as they appear prior to the OUTPUT statement.

5.2.4 High Level Language (HLL)

5.2.4.1 Functionality

The primary benefits to DMIS for the HLL and related I/O functions are as follows:

- ∞ # Provide structure to DMIS programs.
- ∞ # Incorporate more standard data types.
- ∞ # Permit variable arrays.
- ∞ # Permit variable assignments for example, PI=ASSIGN/ 3.1416
- ∞ # Permit arithmetic assignments for example, A=ASSIGN/PI * (R**2)
- ∞ # Permit logical assignments for example, XX=ASSIGN/(DIAM .GT. 1.25)
- ∞ # Incorporate numeric, character, and system intrinsic functions for example, COS, CONCAT, TIME
- ∞ # Provide selective processing
- ∞ # Allow nested IFs.
- ∞ # Allow unconditional JUMPTOs
- ∞ # Incorporate value branching for example, SELECT...CASE...ENDCAS...ENDSEL
- ∞ # Incorporate DO looping
- ∞ # Incorporate external calling for example, CALL/EXTERN
- ∞ # Permit modular programming
- ∞ # Permit inclusion of existing statements for example, INCLUD
- ∞ # Expand Input / Output for example, DEVICE, OPEN, CLOSE, READ, and WRITE

5.2.4.2 Branching and conditionals

The DMIS vocabulary allows for conditional branching in an inspection program based on the results of a measured value, obtained value, variable, arithmetic expression or logical expression. This is performed with the logical IF...ELSE...ENDIF block, DO...ENDDO block, and SELECT...CASE...ENDCAS...ENDSEL block branching statements. Unconditional branching is performed with the JUMPTO, or CALL statements.

5.2.4.2.1 IF conditional branching

The IF...ELSE...ENDIF block statements provide the capability to transfer control of the program based on a logical expression. A typical IF...ELSE...ENDIF block will consist of the following:

IF/(logical expression)

executable statement(s)

ELSE

executable statement(s)

ENDIF

If the logical expression evaluates to .TRUE., control of the program is passed to the statements following the IF statement and executed until an ELSE or ENDIF statement is encountered. If the logical expression evaluates to .FALSE. control of the program is passed to the statements following the next occurrence of an ELSE statement. If an ENDIF statement is encountered before an ELSE statement, the conditional is ended. ELSE statements are optional. Nested IF statements are allowed and must have a corresponding ENDIF statement.

5.2.4.2.2 SELECT conditional branching

The SELECT...CASE...ENDCAS...ENDSEL block statements provide the capability to execute statements according to the results of one or more Boolean tests.

A typical SELECT...CASE...ENDCAS...ENDSEL block will consist of the following:

SELECT/arg

CASE/arg_1

executable statement(s)

ENDCAS

CASE/arg_2

CASE/arg_3

executable statement(s)

ENDCAS

DFTCAS

executable statement(s)

ENDCAS

ENDSEL

Based on the value of the SELECT argument, each CASE argument is compared. If the comparison is true then the subsequent executable statement(s) are performed until the next ENDCAS statement is encountered. If no CASE argument equates to the SELECT argument and if the optional DFTCAS is present, then the statement(s) between the DFTCAS...ENDCAS block are executed. An ENDSEL statement designates the end of the SELECT...ENDSEL block.

All arguments in a SELECT...CASE...ENDCAS...DFTCAS...ENDCAS...ENDSEL block must be declared as INTGR, LONG, REAL, DOUBLE, or CHAR.