



BSI Standards Publication

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

National foreword

This British Standard is the UK implementation of ISO 22093:2011.

The UK participation in its preparation was entrusted to Technical Committee AMT/4, Industrial data and manufacturing interfaces.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2023
Published by BSI Standards Limited 2023

ISBN 978 0 580 71148 0

ICS 25.040.40

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 March 2023.

Amendments/corrigenda issued since publication

Date	Text affected
------	---------------

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

*Systèmes d'automatisation industrielle et intégration — Contrôle du
dispositif physique — Norme d'interface de mesurage dimensionnel
(DMIS)*





COPYRIGHT PROTECTED DOCUMENT

© ISO 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

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

4.37	positional deviation	8
4.38	post-processor	8
4.39	pre-processor	9
4.40	Quality Information System (QIS)	9
4.41	ram	9
4.42	receiving system	9
4.43	rotary table system	9
4.44	selective processing	9
4.45	sensor	9
4.46	Standard for the Exchange of Product Model Data (STEP)	9
4.47	statement	9
4.48	tolerance	9
4.49	tool holder	9
4.50	vendor	9
4.51	video inspection machine	10
4.52	vision inspection machine	10
4.53	work measuring zone	10
4.54	workpiece	10
5	Language reference	10
5.1	Syntax and structure	10
5.1.1	Characters	10
5.1.2	Numbers, words, label names, text strings, vectors, parameters, variables, and expressions	11
5.1.3	Variable assignments and use	19
5.1.4	DMIS command and definition statements	19
5.1.5	Delimiters, blank lines, spaces, and tabs	21
5.1.6	Line length	23
5.1.7	Programming comments	23
5.1.8	Operator input	23
5.1.9	Data output	24
5.1.10	Program structure	27
5.1.11	File structure	29
5.1.12	Programming considerations	30
5.2	Execution and control	30
5.2.1	Declaration statements	30
5.2.2	Definition statements	31
5.2.3	Program statement sequences	33
5.2.4	High Level Language (HLL)	35
5.3	Mathematics	41
5.3.1	Operators	41
5.3.2	Features	44
5.3.3	Tolerances	53
5.3.4	Key Characteristics	57
5.3.5	Datums	57
5.3.6	Coordinate systems	57
5.3.7	Measurement uncertainty	71
5.4	Equipment control	74
5.4.1	ZYZ Euler angles	74
5.4.2	Machine parameters	76
5.4.3	Rotary tables	80
5.4.4	Sensors and sensor-related	81
5.4.5	Carriages	84
5.4.6	Motion control	88
5.4.7	Measurement control	90
5.4.8	Axis configuration	99
5.5	Characterization file	100
5.5.1	Usage	100
5.5.2	Characterization file format	102
5.5.3	Syntax for CHFIL1...ENDCH1 section	104
5.5.4	Example DMIS characterization file grammar	109

6	Statement reference	119
6.1	ACLRAT	120
6.2	ALGDEF	122
6.3	ASSIGN	123
6.4	BADTST	124
6.5	BOUND	125
6.6	CALIB	126
6.7	CALL	128
6.8	CASE	130
6.9	CLMPID	131
6.10	CLMPSN	132
6.11	CLOSE	133
6.12	CMPNTGRP	134
6.13	CNFRMRUL	135
6.14	CONST (input format 1)	136
6.15	CONST (input format 2)	138
6.16	CONST (input format 3)	139
6.17	CONST (input format 4)	140
6.18	CONST (input format 5)	143
6.19	CONST (input format 6)	144
6.20	CONST (input format 7)	147
6.21	CONST (input format 8)	149
6.22	CONST (input format 9)	150
6.23	CONST (input format 10)	151
6.24	CONST (input format 11)	152
6.25	CONST (input format 12)	154
6.26	CONST (input format 13)	155
6.27	CONST (input format 14)	156
6.28	CONST (input format 15)	157
6.29	CRGDEF	158
6.30	CRMODE	159
6.31	CROSCL	160
6.32	CRSLCT	161
6.33	CUTCOM	162
6.34	CZONE	163
6.35	CZSLCT	164
6.36	DATDEF	165
6.37	DATSET	166
6.38	DATTRGDEF	168
6.39	DECL	169
6.40	DECPL	171
6.41	DELETE	172
6.42	DEVICE	173
6.43	DFTCAS	174
6.44	DISPLY	175
6.45	DMEHW	176
6.46	DMEID	177
6.47	DMESW	178
6.48	DMESWI	179
6.49	DMESWV	180
6.50	DMIS	181
6.51	DMISMD	182
6.52	DMISMN	183
6.53	DO	184
6.54	ELSE	185
6.55	ENDAT	186
6.56	ENDCAS	187
6.57	ENDDO	188
6.58	ENDFIL	189
6.59	ENDGO	190

6.60	ENDIF	191
6.61	ENDMAC.....	192
6.62	ENDMES	193
6.63	ENDSEL	194
6.64	ENDSIMREQT	195
6.65	ENDXTN.....	196
6.66	EQUATE	197
6.67	ERROR	198
6.68	EVAL	199
6.69	EXTENS	201
6.70	EXTFIL.....	202
6.71	FEAT/ARC (input format 1)	203
6.72	FEAT/ARC (input format 2)	205
6.73	FEAT/CIRCLE	207
6.74	FEAT/COMPOUND	209
6.75	FEAT/CONE	211
6.76	FEAT/CONRADSEGMNT	213
6.77	FEAT/CPARLN	215
6.78	FEAT/CYLNDR	217
6.79	FEAT/CYLRADSEGMNT	219
6.80	FEAT/EDGEPT	221
6.81	FEAT/ELLIPS	223
6.82	FEAT/ELONGCYL	225
6.83	FEAT/GCURVE	227
6.84	FEAT/GEOM.....	229
6.85	FEAT/GSURF	230
6.86	FEAT/LINE	232
6.87	FEAT/OBJECT	234
6.88	FEAT/PARPLN	236
6.89	FEAT/PATERN	238
6.90	FEAT/PLANE	240
6.91	FEAT/POINT.....	242
6.92	FEAT/RCTNGL	244
6.93	FEAT/REVSURF	246
6.94	FEAT/SPHERE	248
6.95	FEAT/SPHRADSEGMNT	250
6.96	FEAT/SYMLN	252
6.97	FEAT/TORRADSEGMNT	254
6.98	FEAT/TORUS	256
6.99	FEDRAT	258
6.100	FILDEF.....	260
6.101	FILNAM.....	261
6.102	FINPOS.....	262
6.103	FIXTID.....	263
6.104	FIXTSN.....	264
6.105	FLY	265
6.106	FROM.....	266
6.107	GEOALG.....	267
6.108	GEOM	271
6.109	GOHOME	272
6.110	GOTARG.....	273
6.111	GOTO.....	274
6.112	GROUP	277
6.113	IF	278
6.114	INCLUD.....	279
6.115	Intrinsic functions	280
6.116	ITERAT	287
6.117	JUMPTO	289
6.118	KEYCHAR.....	290
6.119	LITDEF (input format 1)	291

6.120	LITDEF (input format 2)	292
6.121	LOCATE	293
6.122	LOTID	295
6.123	MACRO	296
6.124	MATDEF	297
6.125	MEAS	299
6.126	MFGDEV	302
6.127	MODE	303
6.128	OBTAIN	304
6.129	OPEN	305
6.130	OPERID	307
6.131	OUTPUT	308
6.132	PAMEAS	311
6.133	PARTID	313
6.134	PARTRV	314
6.135	PARTSN	315
6.136	PATH	316
6.137	PLANID	320
6.138	POP	321
6.139	PRCOMP	322
6.140	PREVOP	323
6.141	PROCID	324
6.142	PROMPT	325
6.143	PSTHRU	328
6.144	PTBUFF	329
6.145	PTMEAS	330
6.146	PUSH	332
6.147	QISDEF	333
6.148	RAPID	334
6.149	READ	335
6.150	RECALL	336
6.151	REFMNT	337
6.152	REPORT	338
6.153	RESUME	340
6.154	RMEAS (input format 1)	341
6.155	RMEAS (input format 2)	343
6.156	RMEAS (input format 3)	345
6.157	RMEAS (input format 4)	347
6.158	RMEAS (input format 5)	349
6.159	RMEAS (input format 6)	351
6.160	RMEAS (input format 7)	353
6.161	ROTAB	355
6.162	ROTATE	357
6.163	ROTDEF	359
6.164	ROTSET	360
6.165	SAVE	361
6.166	SCNMOD	362
6.167	SCNSET	363
6.168	SELECT	366
6.169	SENSOR	368
6.170	SIMREQT	371
6.171	SNSDEF (input format 1)	372
6.172	SNSDEF (input format 2)	375
6.173	SNSDEF (input format 3)	377
6.174	SNSDEF (input format 4)	379
6.175	SNSDEF (input format 5)	381
6.176	SNSDEF (input format 6)	383
6.177	SNSSET	385
6.178	SNSGRP	388
6.179	SNSLCT	389

6.180	SNSMNT	392
6.181	TECOMP	393
6.182	TEXT	394
6.183	THLDEF	395
6.184	TOL/ANGL	396
6.185	TOL/ANGLB	397
6.186	TOL/ANGLR	399
6.187	TOL/ANGLWRT	401
6.188	TOL/CIRLTY	403
6.189	TOL/COMPOS	404
6.190	TOL/CONCEN	406
6.191	TOL/CORTOL	408
6.192	TOL/CPROFL	410
6.193	TOL/CPROFS	412
6.194	TOL/CRNOUT	414
6.195	TOL/CYLCTY	416
6.196	TOL/DIAM	417
6.197	TOL/DISTB	419
6.198	TOL/DISTWRT	421
6.199	TOL/FLAT	423
6.200	TOL/GTOL	425
6.201	TOL/PARLEL	429
6.202	TOL/PERP	431
6.203	TOL/POS	433
6.204	TOL/PROFL	435
6.205	TOL/PROFP	437
6.206	TOL/PROFS	439
6.207	TOL/RAD	441
6.208	TOL/STRGHT	443
6.209	TOL/SYM	445
6.210	TOL/TRNOUT	447
6.211	TOL/USETOL	449
6.212	TOL/WIDTH	451
6.213	TOOLDF	453
6.214	TRANS	454
6.215	UNCERTALG	456
6.216	UNCERTSET	457
6.217	UNITS	458
6.218	VALUE	459
6.219	VFORM	469
6.220	WINDEF (input format 1)	470
6.221	WINDEF (input format 2)	471
6.222	WKPLAN	472
6.223	WRIST	473
6.224	WRITE	475
6.225	XTERN	476
6.226	XTRACT	477
Annex A (informative) DMIS example code segments		479
A.1	@ character, and use with a label	479
A.2	ASSIGN	479
A.3	BADTST	479
A.4	CALL	480
A.4.1	Example 1 of CALL	480
A.4.2	Example 2 of CALL	481
A.5	CLOSE	481
A.6	CONST (input format 8)	482
A.7	CONST (input format 9) and CONST (input format 10)	482
A.8	CRMODE	484
A.8.1	CRMODE/SEQNTL	484

A.8.2	CRMODE/SIMUL.....	484
A.8.3	CRMODE/SYNC.....	485
A.9	CZSLCT	485
A.10	DEVICE	487
A.11	DMISMN	487
A.12	DO.....	487
A.13	EQUATE.....	488
A.14	EVAL.....	488
A.15	FEAT/GEOM	489
A.16	FEAT/GSURF	489
A.17	GEOM	489
A.18	GOTARG.....	490
A.19	IF and JUMPTO	490
A.19.1	Example 1	490
A.19.2	Example 2	491
A.20	ITERAT.....	491
A.21	KEYCHAR	492
A.22	Macro definition	496
A.22.1	An example of a macro definition without an argument list:.....	496
A.22.2	An example of a macro definition and a CALL statement follows:.....	496
A.23	OBTAIN.....	496
A.24	OPEN	496
A.25	PROMPT, examples of numeric and alphanumeric input.....	497
A.25.1	Example 1 numeric input	497
A.25.2	Example 2 alphanumeric input	497
A.25.3	Example 3 multiple input and playing a .wav file	497
A.26	QISDEF.....	498
A.27	READ examples of delimited and formatted	498
A.27.1	An example of the READ statement for a delimited file	498
A.27.2	An example of the READ statement for a formatted file.....	498
A.28	ROTDEF rotary table definition	498
A.29	SELECT...CASE...ENDCAS...DFTCAS...ENDCAS...ENDSEL block example	500
A.30	Sensor, wrist, component group, and build examples.....	501
A.30.1	Example wrist, component, and sensor definition.....	501
A.30.2	Example component definitions	502
A.30.3	Example component, group, and probe definitions.....	504
A.30.4	Example component definition and calibration statements.....	505
A.30.5	Example complex wrist and component definitions	506
A.30.6	Example tool definition statements	509
A.30.7	Example sensor calibration statements.....	510
A.31	SNSDEF (input format 2)	510
A.32	STR().....	510
A.33	TEXT	511
A.34	TOL/GTOL	511
A.34.1	Example 1, just GO/NOGO info desired:.....	511
A.34.2	Example 2, GO/NOGO and percentage of good points data desired:.....	512
A.34.3	Example 3, GO/NOGO and interference point/s info desired:	512
A.35	VALUE	512
A.36	Vector variable values	512
A.37	WRITE.....	512
A.38	XTRACT	514
Annex B	(informative) Descriptive Figures	515
Annex C	(normative) Standard characterization file	575
Annex D	(informative) Characterization file extensions.....	681
D.1	Machine dependent parameters	681
D.2	User defined options	682
Annex E	(informative) Scanning reference	683
E.1	Introduction	683

Annex F (informative) Tolerance application	684
F.1 Application of TOL/ANGLB and TOL/ANGLWRT	684
F.2 Application of TOL/DISTB and TOL/DISTWRT	684
F.3 Tolerance application	685
F.3.1 Feature combinations for angle tolerances	686
F.3.2 Feature combinations for distance tolerances	687
F.3.3 Tolerance application	688
Annex G (informative) Deleted statements	691
G.1 Introduction	691
G.1.1 Listing order	691
G.2 DNSC Oak Ridge, Tenn. U.S.A. August 1998	691
G.2.1 RADIUS	691
G.3 DNSC Auburn Hills, Mich. U.S.A. March 1999	691
G.3.1 EXTFIL	691
G.4 DNSC Markham, Ont. Canada July 1999	691
G.4.1 SNSDEF (input format 6) Defining an XRAY sensor	691
G.5 DNSC Orlando, Florida February 2003	691
G.5.1 SCAN	692
G.5.2 SCNPLN	692
G.6 DNSC Brighton, Mich. U.S.A. October 2003	692
G.6.1 PATTERN	692
G.7 DNSC Arlington, Texas U.S.A. April 2004	692
G.7.1 CALL	692
G.8 DSC Troy, Michigan U.S.A. April 2009	692
G.8.1 GECOMP	692
Index of statements	693
Index of figures	696
Index of tables	701
Index of statements by type	702
Branching and looping statements	702
Carriage statements	702
Datum statements	702
Feature statements	703
Feature construction statements	703
File and machine parameter statements	704
In process verification / quality information system statements	704
Input / output statements	705
Macro statements	705
Measurement statements	705
Miscellaneous statements	706
Motion statements	706
Program flow statements	706
Rotary table statements	706
Sensor statements	707
Scanning statements	707
Tolerance statements	707
Variable statements	708

Foreword

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

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

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

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 22093 was prepared by the American National Standards Institute (ANSI) (as DMIS 5.2) and was adopted, under a special "fast track procedure", by Technical Committee ISO/TC 184, *Automation systems and integration*, Subcommittee SC 1, *Physical device control*.

This second edition cancels and replaces the first edition (ISO 22093:2003), which has been technically revised.

The significant changes are listed below:

Annex G: New annex to list deleted statements. Because of the confusion and ambiguity created if a new statement is created to replace or enhance an existing statement, this annex was added.

FEAT/LINE: Changed the definitions for the endpoints of the line to explicitly state that the first point is the starting point and the second point is the ending point.

Intrinsic functions: Changed CONCAT(str,str var_2) to CONCAT(str var_2).

Changed MN(x,x var_4) to MN(x var_4)

Changed MX(x,x var_4) (to MX(x var_4)

FEAT/OBJECT: removed ambiguity with input & output values.

DATDEF: the syntax did not allow for a single datum target, though this is explicitly allowed by Note 3 text.

Clause 5.3.1: added normative text for vector operators.

CONST (input format 6): added Ellipse to the var_1 options.

Clause 5.1.6 Removed the 80 character line length limitation and changed it to a max line length of 65,536 characters and the line is terminated with a carriage return(CR) and line feed(LF).

IF: Modified the var_x options.

TOL/ANGLWRT: Modified the var_1 options.

SCNSET:	Added the minor word DEFLECTION with a deflection value.
VALUE:	Added the minor word DEFLECTION to the var_1 option.
CONST (input format 15):	Removed search_radius from a var_3 option.
CALIB:	Added the ability to recalibrated sensors.
CRGDEF:	Added the ability to define a carriage without volume parameters.
FROM:	Added DME, RAM and SCALE options.
ROTAB:	Added the ability to position a rotary table relative to a feature direction.
Annex C:	The EBNF has been updated to reflect all changes to DMIS. Additionally several typographical errors have been corrected.
GECOMP:	The GECOMP statement has been added to Annex G and removed from DMIS.
KEYCHAR:	New major word which defines a key characteristic that associates nominal feature(s) and nominal tolerance(s) with an optional key characteristic criticality designation and assigns a unique label to it.

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

1 Scope

This International Standard defines a neutral language for communication between information systems and Dimensional Measurement Equipment (DME) called the Dimensional Measuring Interface Standard (DMIS). DMIS is an execution language for measurement part programs and provides an exchange format for metrology data such as features, tolerances, and measurement results.

DMIS conveys the product and equipment definitions along with the process and reporting information necessary to perform dimensional measurements that employ coordinate metrology. DMIS contains product definitions for nominal features, feature constructions, dimensional and geometric tolerances, functional datums, and part coordinate systems. It also communicates equipment definitions for various measurement sensors, measurement resources, and machine parameters. DMIS instructs the DME's motions and measurements for product acceptance or verification and for manufacturing process validation and control. Furthermore, DMIS guides the analysis of coordinate data to report and tag measurement results that ascertain product/process quality.

Finally, to aid in its implementation, application functional subsets of DMIS have been defined that ensure successful interoperability and to validate DMIS conformance. Also, DMIS addresses the associativity of DMIS product definitions with CAD information.

While primarily designed for communication between automated equipment, DMIS is designed to be both human-readable and human-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 complete DME language.

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 input and output 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 equipments. 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 or UTF8 files from one DMIS supporting system to another. The method for the transmission, storage, and management of these files is user-dependent.

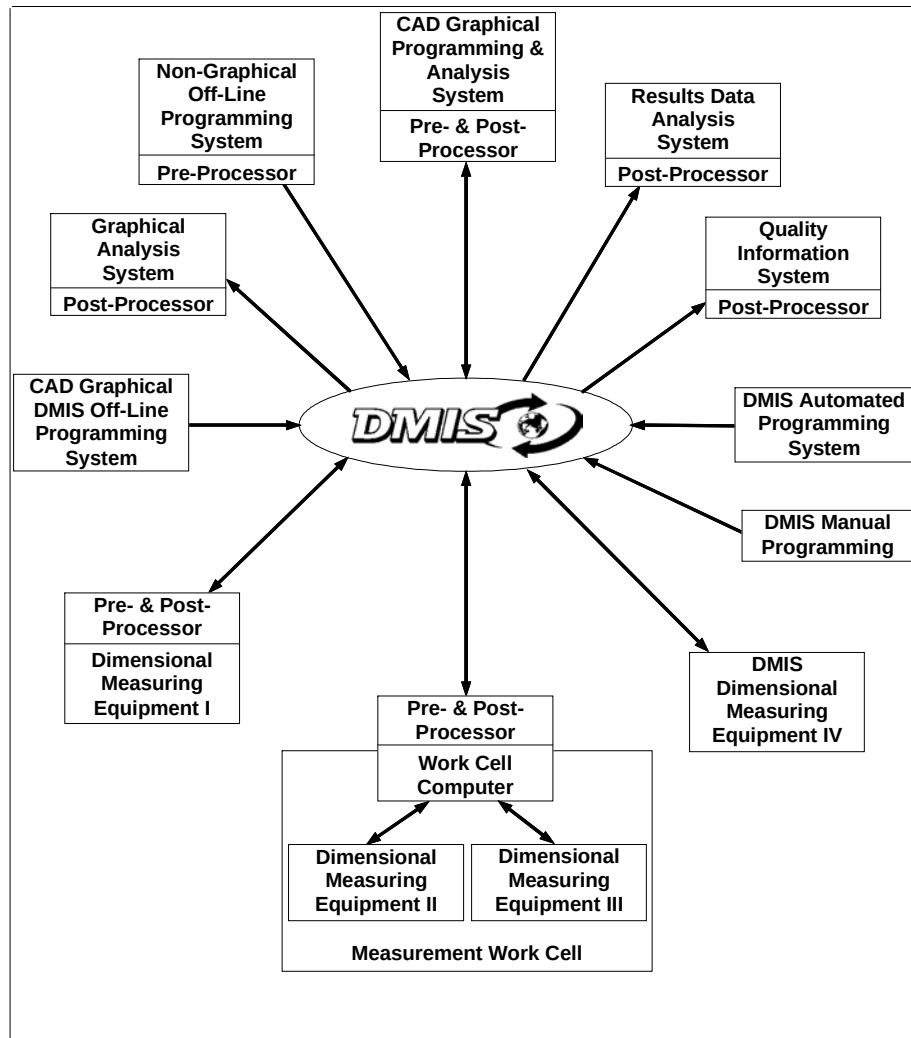


Figure 1 — DMIS environment

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 coupled with any addenda. DMIS is a large and complex standard. Vendors do not need to implement the entire standard. They implement functional characterized subsets. A characterized subset involves an application profile at a specific conformance level along with zero or many associated addenda at their specified conformance level.

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, behaviour 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 application 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 syntactically to DMIS if it interprets all conforming DMIS application profiles.

Application software conforms semantically to DMIS if it interprets all metafiles that conform to DMIS application profiles according to all required semantics prescribed by this standard.

2.2 Conformance testing services

Conformance testing services will be recognized by the DSC to utilize 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 two application profiles that have been defined along with their two character identifier, Prismatic (PM) and Thin Walled (TW), to each of the three levels, as well as seven (7) addenda that expand the levels of conformance. The addenda along with their two character identifier are: Rotary Tables (RT), Multi Carriage (MC), Contact Scanning (CS), In-Process Verification (IP), Quality Information Systems (QI), Measurement Uncertainty (MU), and Soft Gaging (SG).

2.6 Conformance levels

Each application profile or addendum will define three levels of conformance.

- Level 1 – is essential (required) 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 Program Conformance Designation

Each DMIS program or DMIS module shall designate its conformance classes via a collection of one or more pairs of application profile / addenda with conformance level. An example of a DMIS program conformance designation within a DMISMN statement is:

DMISMN, 'DMIS Program Interoperability', 5.2, PM, 2, RT, 1, MC, 1

2.8 Conformance claim

Upon positive notification from a recognized DMIS conformance certification service, a DMIS application may make a conformance claim.

An example of a conformance claim is:

MyProduct version x.y from DMISVendor conforms to DMIS Prismatic Profile, Level 2 with Rotary Table, Level 1.

3 Normative references

The following normative documents contain provisions that, through reference in this text, constitute provisions of this international standard. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this international standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid international standards.

American Standard Code for Information Interchange (ASCII)

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

ANSI/ASME B89.4.1-1997

All terminology that refers to Coordinate Measuring Machines is used in accordance with this standard.

ANSI/CAM-I 101 1990

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

ANSI Y14.26M-1987

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

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

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

Dimensional Mark-up Language Specification for the transfer of dimensional inspection results (DML).

<http://www.dmisstandard.com/DML/>

ISO 10303 TC184/SC4

The emerging Standard Exchange for Product Model Data (STEP) 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 11562:1996 Geometrical Product Specifications (GPS) – Surface texture: Profile method – Metrological characteristics of phase correct filters.

ISO/IEC 14977:1996 (E) Information technology - Syntactic metalanguage - Extended BNF

4 Terms and definitions

The following terms and acronyms appear throughout this standard. When they are used, it is only within the context of the limited definitions set forth in this section.

4.1

actuals

Referring to features or tolerances, the actuals are the results computed from measured or constructed features. Actuals are determined by the measuring device and do not exist until a measurement or construction has occurred.

4.2

Automatically Programmed Tools (APT)

A computer language for numerical controlled (NC) machines.

4.3

calibration sequence

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

The moving component of a machine that carries the ram.

4.5

characterization file

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

4.6

Computer Aided Design (CAD) system

A computerized system for design and analysis. 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)

A computerized inspection device for gathering discrete point data and analyzing dimensions and tolerances. Data is usually acquired using a touch probe or scanning probe as a sensor.

4.8

Dimensional Measuring Equipment (DME)

A class of equipment used to inspect parts and evaluate dimensions and tolerances. 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)

This is the name given to this standard. 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)

The DMIS National Standards Committee (DNSC) was the committee formed in 1990 and accorded responsibility for the maintenance and support of DMIS until June of 2005 at which time the committee name changed to the DMIS Standards Committee (DSC). The DNSC was an accredited standards committee operating under ANSI-approved DMSC procedures as the accredited organization.

4.11

DMIS Standards Committee (DSC)

The DMIS Standards Committee (DSC) is the committee accorded responsibility for the maintenance and support of DMIS. The DSC is an accredited standards committee operating under ANSI-approved DMSC procedures as the accredited organization.

4.12

Dimensional Mark-up Language (DML)

The Dimensional Mark-up Language Specification is a neutral, non-proprietary, open format developed by the Automotive Industry Action Group (AIAG) that provides interoperability between the CMM, Coordinate Measuring Machines. The language is in a XML Schema Format.

4.13

DML Common Space

The coordinate system to which all features and transformations are referenced within DML.

4.14

Extended Backus-Naur Form (EBNF)

A syntactic metalanguage which is a notation for defining the syntax of a language by use of a number of rules. Each rule names part of the language (called a non-terminal symbol of the language) and then defines its possible forms.

4.15

feature

A feature is a point or group of points, which are identified and referred to as a feature. Usually, features are geometric elements of a part such as points, lines, planes, and circles.

4.16**filename**

The term filename is used to indicate a computer readable file. A 'filename' can be a filename and extension or it can contain a full path specification.

4.17**geometric compensation**

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

4.18**High Level Language (HLL)**

A collection of statements providing advanced capabilities that are identified in clause 5.2.4.

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

A specification for the exchange of CAD data.

4.20**inner**

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

4.21**In Process Verification (IPV)**

A computerized system linking inspection and process verification to the manufacturing process.

4.22**input files**

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

4.23**label**

A label is 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. 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.

4.24**label type**

The label type is 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.

4.25**label name**

The label name is 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.

4.26**line reducible feature**

A line reducible feature is a feature that can be reduced to a geometric line. Line reducible features are given in (Table 8 — Reducible features).

4.27

measurement sequence

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

4.28

multiple carriage

A DME having two or more carriages.

4.29

nominal

Referring to a feature or tolerance, this is the "as-designed" value defined in the product definition of the part. The nominal is known prior to measurement and is the value to which the actuals are compared for computing out-of-tolerance conditions.

4.30

orientation

The angular relationship to a coordinate system of an entity that can be a nominal or actual feature, nominal or actual sensor, or nominal or actual coordinate system.

4.31

orientational deviation

(RMEAS): The angular difference between the nominal and actual vectors of the measurement plane and the feature.

4.32

outer

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

4.33

output files

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

4.34

part coordinate system (PCS)

A datum reference frame associated with the part to be measured.

4.35

plane reducible feature

A plane reducible feature is a feature that can be reduced to a geometric plane. Plane reducible features are given in (Table 8 — Reducible features).

4.36

point reducible feature

A point reducible feature is a feature that can be reduced to a geometric point. Point reducible features are given in (Table 8 — Reducible features).

4.37

positional deviation

(RMEAS): The scalar difference between the nominal and actual locations of the measurement plane and the feature.

4.38

post-processor

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

4.39**pre-processor**

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

4.40**Quality Information System (QIS)**

A computerized system for the analysis and storage of inspection data.

4.41**ram**

The moving component of a machine that carries the sensor.

4.42**receiving system**

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

4.43**rotary table system**

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

4.44**selective processing**

An inspection process having the required conditional branching to alter the control of a program based on inspection results or pre-set variables. This capability is provided for by incorporation of HLL extensions.

4.45**sensor**

A device used on a DME to acquire measurement data. 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 and non-contact electrical capacitance.

4.46**Standard for the Exchange of Product Model Data (STEP)**

A specification for the exchange of product model data.

4.47**statement**

Any DMIS definition or command.

4.48**tolerance**

A measure of acceptable deviation from nominal conditions. Geometric tolerances in DMIS are as defined in the ASME Y14.5M-1994 Standard for Dimensioning and Tolerancing.

4.49**tool holder**

A device that holds or stores sensors for automatic sensor changing.

4.50**vendor**

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

4.51

video inspection machine

A computerized inspection device which acquires data for analyzing dimensions and tolerances. The sensor is typically a camera.

4.52

vision inspection machine

A computerized inspection device which acquires data for analyzing dimensions and tolerances. The sensor is typically a camera or group of cameras.

4.53

work measuring zone

The measurement volume of a machine as specified by the supplier. 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. The work measuring zone is also referred to as the working volume.

4.54

workpiece

An 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.

5.1.2 Numbers, words, label names, text strings, vectors, parameters, variables, and expressions

Characters are combined to form numbers, vocabulary words, labels, text strings, parameters, variables, and expressions.

5.1.2.1 Literal numbers

Literal numbers in DMIS input and output files consist of an optional sign, followed by one or more digits, optionally with a decimal point before the digits, between two digits, or after the last digit (+7.40, for example). Digits are the characters 0 (ASCII 48) through 9 (ASCII 57). A sign is a plus sign (+, ASCII 43) or a minus sign (-, ASCII 45). A decimal point is ASCII 46. Scientific notation (such as 3.7E02) is not allowed. All numbers are interpreted as decimal numbers. Only base ten numbers are used in DMIS.

Literal numbers 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.

A literal integer is a literal number that has no decimal point. A literal real number is a literal number that has a decimal point. A literal integer may be used, however, wherever a real number is required in DMIS.

The literal real numbers +0.0 and -0.0 represent the same number as 0.0; similarly for the literal integers +0, -0, and 0.

Every literal number is positive, negative, or zero. The literal numbers +0.0 and +0 are zero, not positive. The literal numbers -0.0 and -0 are zero, not negative.

Numerical data for angles can be represented as degrees, minutes and seconds separated by the colon character (:, ASCII 58), for example 4:03:47.00. Numerical data for times can be represented as hours, minutes and seconds separated by the colon character, for example 4:03:47. These representations cannot be used literally with mathematical operators or functions (COS(3 * 4:03:47.00), for example, is not allowed). Angles and times represented using a colon are not considered to be literal numbers. They are structured data with the usual rules for what characters are allowed; for example, the number of minutes must consist of exactly two characters, the first of which is a digit between 0 and 5, and the second of which is any digit.

5.1.2.2 Words

5.1.2.2.1 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).

Table 1 — DMIS major words

ACLRAT	ENDSEL	PREVOP
ALGDEF	ENDSIMREQT	PROCID
ASSIGN	ENDXTN	PROMPT
BADTST	EQUATE	PSTHRU
BOUND	ERROR	PTBUFF
CALIB	EVAL	PTMEAS
CALL	EXTENS	PUSH

CASE	EXTFIL	QISDEF
CLMPID	FEAT	RAPID
CLMPSN	FEDRAT	READ
CLOSE	FILDEF	RECALL
CMPNTGRP	FILNAM	REFMNT
CNFRMRUL	FINPOS	REPORT
CONST	FIXTID	RESUME
CRGDEF	FIXTSN	RMEAS
CRMODE	FLY	ROTAB
CROSCL	FROM	ROTATE
CRSLCT	GEOALG	ROTDEF
CUTCOM	GEOM	ROTSET
CZONE	GOHOME	SAVE
CZSLCT	GOTARG	SCNMOD
DATDEF	GOTO	SCNSET
DATSET	GROUP	SELECT
DATTRGDEF	IF	SENSOR
DECL	INCLUD	SIMREQT
DECPL	ITERAT	SNSDEF
DELETE	JUMPTO	SNSSET
DEVICE	KEYCHAR	SNSGRP
DFTCAS	LITDEF	SNSLCT
DISPLY	LOCATE	SNSMNT
DMEHW	LOTID	TECOMP
DMEID	MACRO	TEXT
DMESW	MATDEF	THLDEF
DMESWI	MEAS	TOL
DMESWV	MFGDEV	TOOLDF
DMIS	MODE	TRANS
DMISMD	OBTAIN	UNCERTALG
DMISMN	OPEN	UNCERTSET
DO	OPERID	UNITS
ELSE	OUTPUT	VALUE
ENDAT	PAMEAS	VFORM
ENDCAS	PARTID	WINDEF
ENDDO	PARTRV	WKPLAN
ENDFIL	PARTSN	WRIST
ENDGO	PATH	WRITE
ENDIF	PLANID	XTERN
ENDMAC	POP	XTRACT

ENDMES	PRCOMP	
--------	--------	--

Table 2 — Characterization file major words

CHFIL1	ENDCH1
CHFIL2	ENDCH2
CHFIL3	ENDCH3
CHFILE	ENDCHF

5.1.2.2.2 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 3 — DMIS minor words

-PRBRAD	CYLCY	INTOF	PART	SNS
-XDIR	CYLNDR	INTOL	PARTO	SNSET
-YDIR	CYL RADSEGMENT	IP	PATTERN	SNSLCT
-ZDIR	CZSLCT	IPM	PAUSE	SNSMNT
2D	DATA	IPMM	PCENT	SOUND
2RC	DATTRG	IPS	PCS	SPART
3D	DEFAULT	IPSS	PECK	SPH
4POINT	DEFLECTION	JOINTCONFIG	PERP	SPHERE
ABOVE	DELAY	KEEP	PERPTO	SPHRADSEGMENT
ABSL	DELETE	LAMBDA	PICTURE	SPLINE
ACEL	DEPTH	LASER	PITCH	START
ACLRAT	DEV	LE	PIXBTN	STAT
ACT	DFTCAS	LEFT	PLANE	STDDEV_LIMIT
ADJUST	DIAM	LEFTY	PLOT	STOP
ALGOR	DIM	LIMIT	PM	STOR
ALL	DIRECT	LINE	POINT	STRGHT
ALLAXESTOUCH	DISK	LINEAR	POL	STROBE
ALLSA	DIST	LIST	POS	SUPPORTED
ALPHA	DISTANCE	LMC	POSACL	SURF
AMT	DISTB	LN2LN	POSVEL	SURFACE
AND	DME	LOCAL	PP	SYM
ANGDEC	DMIS	LONG	PRBRAD	SYMPN
ANGDMS	DMISMD	LOW	PRCOMP	SYNC
ANGL	DMISMN	LOWPASS	PRINT	SYS

ANGLB	DML	LSTSQR	PRNTCHAR	TANGPL
ANGLE	DOUBLE	LT	PROBE	TANTO
ANGLR	DRAG	MACH	PROFL	TECOMP
ANGRAD	EDGELN	MAJOR	PROFP	TEMP
APPEND	EDGEPT	MAN	PROFS	TEMPC
APPRCH	EDIT	MATRIX	PROG	TEMPF
ARC	ELIMINATE	MAX	PROJCT	TEMPWC
ATTACH	ELLIPS	MAXINS	PROJLI	TEMPWF
AUTO	ELONGCYL	MC	PROJPT	TERM
AVG	END	MCS	PT2LN	TEXT
AVGDEV	ENDCAS	MESACL	PT2PL	THRU
AXDIR	ENDDO	MESVEL	PT2PT	TIME
AXIAL	ENDSEL	METER	PTBUFF	TITLE
BACK	ENTITY	MIDLI	PTDATA	TORRADSEGMNT
BADTST	EQ	MIDPL	PTMEAS	TORUS
BANDPASS	ERR	MIDPT	QI	TR
BELOW	ERRMODE	MIN	QUERY	TRIGER
BEZIER	ERROR	MINCIR	RAD	TRMATX
BF	EXCEPT	MINCON	RADIAL	TRNOUT
BND	EXTERN	MINMAX	RADIUS	TRUE
BOOL	EXTREM	MINOR	RAWDAT	TW
BOUND	FALSE	MM	RCTNGL	UNBND
BOX	FDATA	MMC	REAL	UNCERT
BSPLIN	FEAT	MMPS	RECFILT	UNITS
BUILD	FEATUR	MMPSS	REPORT	UNKNOWN
BUTTON	FEDRAT	MNTLEN	REQUNCERT	UNSUPPORTED
CART	FEET	MODE	RES	USERDF
CCW	FILNAM	MODEL	RETRCT	USETOL
CHAR	FILTER	MOVEPT	RETRIEVE	VEC
CHECK	FINPOS	MPM	REVSURF	VECBLD
CHORD	FIRST	MPMM	RFS	VECTOR
CIRCLE	FIXED	MU	RIGHT	VENDOR
CIRCULAR	FLAT	NAME	RIGHTY	VERSION
CIRLTY	FLIP	NE	ROTACL	VERTEX
CLRSRF	FOCUSN	NEXT	ROTARY	VIDEO
CM	FOCUSY	NODATA	ROTNUL	WAIT
CODE	FORCE	NOFLIP	ROTORG	WIDTH
COG	FORM	NOM	ROTTOT	WKPLAN
COMAND	FX	NOMINL	ROTVEL	XAXIS
COMM	FZ	NONCON	ROUND	XDIR

COMMON	GAUSS	NONE	RPM	XORIG
COMP	GCURVE	NOROT	RPMM	XVEC
COMPOS	GE	NOT	RPTSYC	XYAXIS
COMPOUND	GEOALG	NOTOUCH	RTAB	XYDIR
CONCEN	GLOBAL	NOTRAN	RULE	XYPLAN
CONE	GOTO	NUMERIC	RULEINTOL	XYZAXI
CONRADSEGMNT	GRID	NURBS	RULEOUTOL	XYZDIR
CONT	GROUP	OBJECT	RULEUNDET	YAXIS
CONTIN	GSURF	OBLQ	RULEUNSUP	YDIR
CORTOL	GT	OFF	RY	YORIG
COUNT	HEADCS	OFFSET	SCALEX	YZAXIS
CPARLN	HEADTOUCH	ON	SCALEY	YZDIR
CPROFL	HELICAL	OPEN	SCNACL	YZPLAN
CPROFS	HIGH	OPER	SCNMOD	ZAXIS
CRAD	HIGHPASS	OPTIMAL	SCNVEL	ZDIR
CRITICAL	HIST	OR	SEARCH	ZORIG
CRMODE	HUMID	ORIENT	SENS	ZVEC
CRNOUT	ILLEGALTOUCH	OUTER	SENSOR	ZXAXIS
CROSCL	INCH	OUTFIL	SEQNTL	ZXDIR
CRSLCT	INCR	OUTOL	SF	ZXPLAN
CT	INDEX	OUTPUT	SGAGE	
CURRENT	INFRED	OVERWR	SHORT	
CURVE	INNER	PARAM	SIMUL	
CW	INPUT	PARLEL	SINGLE	
CYCLE	INTGR	PARPLN	SIZE	

Table 4 — Characterization file minor words

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

5.1.2.3 DMIS reserved words

DMIS reserved words consist of DMIS major words defined in (Table 1 — DMIS major words), DMIS minor words defined in (Table 3 — DMIS minor words) and intrinsic function names listed in (Table 6 — Intrinsic function words).

5.1.2.4 Labels

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 and datum target labels, are from one to sixty-four (64) characters where the only permissible characters are the ASCII printable characters (code 32 to 126) except for the following:

```
Code 34 Double quote -----"
Code 36 Dollar sign -----$
Code 39 Apostrophe-quote----'
Code 40 Left parenthesis -----(
Code 41 Right parenthesis -----)
Code 64 Commercial at-----@
Code 91 Left bracket-----[
Code 93 Right bracket -----]
```

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 ASCII code 45, dash-hyphen-minus '-' in between for compound datums (for example AA-ZZ). Datum target label names are from one to two upper-case alpha characters followed by a numeric (for example A1, A2, BB3).

All label names are enclosed in parentheses. Leading spaces and trailing spaces within a label name are ignored. For example, FA(Hole 1) is equivalent to FA(Hole 1). 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 name 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. The array index (base) begins at the number 1. Refer to example A.1.

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

5.1.2.5 Text strings

Text strings are used extensively in DMIS. A text string is a grouping of UTF8 printable characters (code 32 to 126), that is enclosed with apostrophes. All UTF8 printable 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. Space, UTF8 code 32, is considered as a printable character. An empty string is specified by using only two apostrophes with no characters in between, that is ". 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, PRNTCHAR, LEFT, '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 very long, it may be divided into pieces and placed on two or more lines using the line continuation character (a single dollar sign "\$"). If the line continuation character is used to divide a text string, it must be the last character other than spaces and tabs (before the carriage return and line feed) in the line to be continued. If a text string is divided using '\$', any spaces and tabs following the '\$' at the end of the line are not part of the text string. 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,PRNTCHAR,LEFT,'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.

5.1.2.6 Parameters

Parameters that appear in DMIS statements are separated from each other and from minor words by commas. The type of each parameter is either an integer number, a real number, a character string or a label. Parameter types are defined in section 6, the Statement Reference section of this document.

A parameter's value can be set in a DMIS statement by an expression evaluating to the appropriate type. A numeric expression evaluating to an integer number may be used to replace a real number parameter. A numeric expression evaluating to a real number cannot be used to directly replace an integer number valued parameter. The INT() intrinsic function can be used to create an integer valued expression from a real number.

Otherwise, substituting a parameter in a DMIS statement of the wrong type causes an error to be flagged by the DME.

For example, the following statements will cause errors to occur because the wrong literal type is used:

```
GOTO/'1','2','3'
MEAS/POINT,F(pnt),2.3
TEXT/OPER,12*5
```

Use of the wrong variable type as parameters will cause similar errors. If we have declared the following variables:

```
DECL/REAL,RVAL
DECL/INTGR,IVAL
DECL/CHAR,80,CVAL
```

Then the following statements will cause errors to occur:

```
MEAS/CIRCLE,F(cir),RVAL
GOTO/0,0,CVAL
```

However, the following statement is correct:

```
GOTO/0,0,IVAL
```

5.1.2.7 Variables

All DMIS variables are declared before being used and include Boolean, character string, double, integer, long, real, and vector data types. Variable names consist 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 (refer to clause 5.1.2.3) 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 module or macro are local to that module or macro. Multi-dimensional variable arrays may be declared. The variable array index (base) begins at the number 1. Further information about variables can be found throughout section 5.2.

A previously declared and assigned variable can be substituted for any parameter (that is, numeric value, Boolean value, vector value, character string, or label name) 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

Variables declared with the DECL statement have no particular value until explicitly assigned a value.

5.1.2.7.1 Numeric types and storage limits

DMIS supports the following numeric variable types: DOUBLE, REAL, LONG and INTGR. DOUBLE and REAL are floating point representations of real numbers and LONG and INTGR are representations of integer numbers.

DMIS places the following storage limits for these numeric types:

DOUBLE – a minimum of 8 bytes of storage

REAL – a minimum of 4 bytes of storage

LONG – a minimum of 4 bytes of storage giving a minimum range –2147483648 to 2147483647

INTGR – a minimum of 2 bytes of storage giving a minimum range –32768 to 32767

The actual number ranges for REAL and DOUBLE that the above storage requirements allow may be system dependent. Because the numeric limits that these storage limits imply will represent numbers that are well beyond anything encountered in normal metrology, DMIS does not specify specific numeric limits for the DOUBLE and REAL storage types.

5.1.2.8 Expressions

A DMIS expression is formed from operators, operands and parentheses and can represent a simple data reference or a computation. Operands include literal numbers and text strings, variables and intrinsic function calls. The simplest expression consists of a single operand. More complex expressions consist of combinations of operands and operators or combinations of operands, operators and parentheses.

Expressions are referred to by the type of their result. A numeric expression evaluates to a number and may use the numeric operators described in section 5.3.1.1. A logical expression evaluates to a Boolean value (.TRUE. or .FALSE.) and may use the relational and/or logical operators described in sections 5.3.1.2 and 5.3.1.3. A vector expression evaluates to a vector and may use the vector operators described in clause 5.3.1.4.

A character expression evaluates to a character string and can only be a single operand: either a literal text string, a character variable or a call to an intrinsic function returning a character string; there are no character operators in DMIS.

An expression of the proper type can be substituted for any parameter in any DMIS statement. For example, a numeric expression can be substituted for a numeric parameter in a DMIS statement, and a character expression may replace any text string parameter in a DMIS statement.

The numeric type to which a numeric expression evaluates depends on the types of the operands in the expression. If all the operands are of the same type, the result will be of that type. If operands of different types are used then the type of the result will be determined by the highest ranking operand using the following ranking: type DOUBLE has the highest ranking, followed by REAL, followed by LONG and then by INTGR with the lowest ranking.

In a numeric expression containing sub-expressions enclosed by parentheses, the numeric type of each sub-expression is determined by the operands within the parentheses.

EXAMPLE 1:

```
$$
$$ The following example demonstrates a complex expression.
$$ The numeric type of the division in parentheses will be integer even though
$$ a real number appears outside the parentheses in the same expression.
```

\$\$
4.0+ (2/3)

In the above example of a complex expression the numeric type of the division in parentheses will be integer even though a real number appears outside the parentheses in the same expression.

EXAMPLE 2:

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

5.1.3 Variable assignments and use

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 name) 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 to it for easy reference. The label 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 statement is continued on the third line.
- The fourth line is a command statement consisting of a major word with a minor word and a 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

Apostrophes, brackets, commas, parentheses, and slashes are used as delimiters in DMIS.

5.1.5.1.1 Apostrophe

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

EXAMPLE

```
FILNAM/'041380385 test dated 2005/10/17',05.1
```

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

EXAMPLE

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

This results in the following message sent to the display device:

Use Paul's setup instructions from the last job.

5.1.5.1.2 Brackets

Brackets '[' and ']' are used as delimiters. An array index or a series of array indices (separated by commas) is preceded with a left bracket and followed by a right bracket.

5.1.5.1.3 Comma

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

```
MAJOR_WORD/MINOR_WORD1,MINOR_WORD2,param1,param2
```

5.1.5.1.4 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, its 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 clauses 5.3.1.1, 5.3.1.2, 5.3.1.3, and 5.3.1.4.

5.1.5.1.5 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:

PRCOMP/OFF

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.

5.1.5.2 Blank lines, spaces, and tabs

Blank lines have no significance in DMIS. Spaces and tabs have no significance except inside character strings and (spaces only) inside label names. Blank lines are insignificant and are ignored during translation. Spaces and tabs are not allowed within DMIS words, parameters and variables.

Spaces but not tabs may be used inside label names (but leading and trailing spaces are not part of a label name).

Spaces and tabs may be used:

- a) between tokens of the language (DMIS words, variable names, numbers, and delimiters).
- b) inside text strings passed in TEXT, FILNAM, and other statements using 'text'.
- c) before or inside programming comments.
- d) after a line continuation character.

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

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 in label names and 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 contains statements of variable lengths. Each line is terminated with a carriage return(CR) and line feed(LF). There is no limit to the length of a statement. A statement can be on one line or split across multiple lines by using the line continuation character which is a single dollar sign '\$'. A '\$' used as a line continuation must be the last visible character on the line. Space and tabs (which are not visible) may occur after the '\$' but before the carriage return and line feed that end the line. The line continuation character has no effect in a programming comment line. Each line is terminated with the system appropriate new line symbol. The maximum length of a line in a DMIS file will not exceed 65,536 characters including the new line symbol.

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 non-whitespace characters in a line. Spaces and tabs are allowed prior to the \$\$ to allow DMIS part programs to become structured per the programmer's desire. 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 name for the query, the number of characters allowed for the response, whether it is alphanumeric, numeric, or any printable UTF8 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 name 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, ALPHA, LEFT, 'Enter Operator Name: '  
TEXT/QUERY, (today), 10, PRNTCHAR, LEFT, 'Enter today's date: '
```

Here, "name" and "today" are label names. The label names 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, ALPHA, LEFT, 'John Smith '  
TEXT/RES, (today), 11, PRNTCHAR, LEFT, '2005/06/14 '
```

The responses "John Smith" and "2005/06/14" 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. When a variable or expression is used as a parameter to a given input statement, it is the value of that variable or expression that appears in the output statement.

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 5.1 file'          2007/02/04',05.1
V(Vendor_output)=VFORM/ALL
DISPLY/TERM,DMIS,PRINT,DMIS,STOR,V(Vendor_output)
FILNAM/'Sample DMIS Input Program',05.1

```

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, DML or RTAB parameter is accessed only with the SAVE or RECALL statements. Such devices are used for reading and writing DME specific or DML 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(Iname),FDATA,DMIS,OUTPUT,OVERWR statement and which may be subsequently re-opened with the OPEN/DID(Iname),FDATA,DMIS,OUTPUT,APPEND statement after being closed.

Output to a device opened with the DML parameter will include all F(), FA(), T(), TA(), D(), DA() information accumulated since the OPEN statement and ending at the SAVE statement. The DML Common Space is defined by the DA() specified parameter in the SAVE statement. Input from the RECALL statement will populate and define all F(), FA(), T(), TA(), D(), DA() information contained within the DML file. The DML Common Space will be equated to the coordinate system specified by the DA() parameter in the RECALL statement.

The ENDFIL statement at the end of the DMISMN main module 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(Iname),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',05.1
```

```
V(V1)=VFORM/NOM,ACT,DEV,AMT
```

```
DISPLY/PRINT,V(V1),STOR,DMIS
```

```
FILNAM/'DEVICE statements program',05.1
```

```
$$ 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',05.1
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(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(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(Point_1) once this sample program is completed.

```
FILNAM/'This_File_Name',05.1
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.

5.1.9.4 DML Import/Export Capability

DML is an XML format definition tailored to the needs of dimensional results for discrete manufacturing. The purpose is to exchange results between applications that generate or use dimensional information. A typical scenario is where an inspection device collects dimensional data and sends the information to an SPC package for process analysis or a database for long term storage.

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, including the program main unit and program module units.

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

A 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 involve the following:

```
DMISMD/'My_Module_ID',05.1
FILNAM/'My_Filename',05.1
```

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',05.1
FILNAM/'/user/dmis/defn/sqrkey',05.1
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. Refer to clause 5.4.6 and sub-clauses for rules and examples.

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 initial and limit values at an optionally 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 an 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 clause 5.2.1 and sub clauses.

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

Statements in the DMIS vocabulary are either modal or non-modal refer to Table 13 — Modal statements.

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 JUMPTO, IF, DO, SELECT, CALL, ITERAT, ERROR and RESUME 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. All DMIS variables must be declared before they are used. 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, Boolean value, character string, vector value, or label name) 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 calculating resultant features from the measured data.

The feature actual or "FA" that results from a measurement sequence always contains all individual point information. If point buffering was enabled with the PTBUFF statement when the feature was measured, all individual point information will be accessible. The effect of a substitute data fitting algorithm will never remove measured data but may cause the calculation of a feature using less than all of the measured data.

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 with the GEOM statement.

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 in a MACRO...ENDMAC block. Once defined a macro is available for invoking by using a CALL statement. Typically, a macro definition section is used to define macros for use within the DMIS program unit or DMIS module (DMISMN, DMISMD). Synopsis of a macro definition section is as follows:

M(lname)=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

Mating definitions are used to define mating information between either features and geometry or between features and other features with the MATDEF statement.

5.2.2.7 Quality information system definitions

Quality Information Systems (QIS) definitions are provided to support quality information management. A QIS definition defines a QIS variable. QIS variables may be used with the OBTAIN, PROMPT and REPORT statements. QIS variables may be user-defined with the QISDEF statement or predefined using the statements listed in Table 5 — Predefined QIS variable types.

Example of user-defined QIS variable

Q(boss1)=QISDEF/'supervisor','Jill'

Example of predefined QIS variable

OP(oper_id)=OPERID/'Jack'

Table 5 — Predefined QIS variable types

Statement	Label Prefix	Meaning
CLMPID	CI	part holding clamp id
CLMPSN	CS	part holding clamp serial number
CUTCOM	CC	compensation or process adjustment
DMEID	DI	dimensional measuring device id
DMESWI	DS	DME software id
DMESWV	DV	DME software version
FIXTID	FI	part holding fixture id
FIXTSN	FS	part holding fixture serial number
LOTID	LI	lot id
MFGDEV	MD	manufacturing device
OPERID	OP	operator id

PARTID	PN	part id
PARTRV	PR	part revision level
PARTSN	PS	part serial number
PLANID	PL	inspection plan id
PREVOP	PV	previous operation
PROCID	PC	inspection procedure
TOOLDF	TL	tool used on a manufacturing device

QIS variables can not be referenced using just the variable name, the type identifier and the variable name must be used.

Example of variable reference:

```
Q(boss1)=QISDEF/'supervisor','Jill'
```

Would be referenced as:

```
DECL/CHAR,32,msg,supe
supe=OBTAIN/Q(boss1),2
msg=ASSIGN/CONCAT('Supervisor: ',supe,' supervised.')
TEXT/OUTFIL,msg
```

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 specified with a TOL statement 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).

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.

The EVAL statement can be used to generate a tolerance actual, that is, a TA(Iname), from which data 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, TOL/DISTB and TOL/SYM statements.

When executed, the OUTPUT or EVAL statements cause the evaluation of the feature actual(s), FA(Iname), and the tolerance actual, TA(Iname). 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:

```

$$ group 1) -----
R(REP_1)=REPORT/DATE
F(CYL_1)=FEAT/CYLNDR, INNER, CART, 0, 0, 5, 0, 0, 1, 8
F(PLN_1)=FEAT/PLANE, CART, 0, 0, 0, 0, 0, -1
F(PLN_2)=FEAT/PLANE, CART, 0, 0, 10, 0, 0, 1
F(PLN_3)=FEAT/PLANE, CART, 0, 0, 10.25, 0, 0, 1
T(PERP_X)=TOL/PERP, .05, RFS, DAT(A), RFS
T(dia_1)=TOL/DIAM, -.002, +.002
$$ group 2) -----
BOUND/F(CYL_1), F(PLN_1), F(PLN_2)
BOUND/T(PERP_X), F(PLN_1), F(PLN_3)
$$ group 3) -----
MEAS/CYLNDR, 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)

```

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

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

- a) any required BOUND statements,
- b) EVAL statements, or
- c) 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. Further, specialized conditional branching is achieved with the ERROR, RESUME and ITERAT 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...ENDSEL block, which begins with a SELECT statement, provides the capability to execute statements according to the results of one or more equality tests.

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

```
SELECT/arg
CASE/arg_1

    executable statement(s)

ENDCAS
CASE/arg_2

    executable statement(s)

ENDCAS
CASE/arg_3

    executable statement(s)

ENDCAS
DFTCAS
```