9

Initial Graphics Exchange Specification (IGES), Version 3.0

Brad Smith
Joan Wellington

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Manufacturing Engineering
Automated Production Technology Division
Gaithersburg, MD 20899

April 1986

U.S. DEPARTMENT OF COMMERCE

NATIONAL BUREAU OF STANDARDS

ACKNOWLEDGEMENT

IGES, Version 3.0, April 1986
First Printing

ERRATA

Page	Change required
xi	"2.2 ASCII Form" should indicate page 9
xii	"2.5.5." should read "2.5.5"
	3-8 delete the word "Additional"
	3-12 delete the word "Additional"
	4-10 should read "Font Code 1001"
	4-11 should read "Font Code 1002"
	4-38 should read "Elemental and Material Coordinate System"
	add "4-46 Nodal Load/Constraint...419"
xvii	Add page numbers as follows:
	A-1 426
	A-2 429
	A-3 435
	B-1 448
	B-2 449
	B-3 450
	B-4 451
	C-1 455
	C-2 456
	C-3 459
	C-4 460
	C-5 463
xviii	C-2 delete the word "File"
4	Sec. 1.4, third paragraph, second line: Change "ASCII format" to "ASCII form"
5	Thirdfull paragraph, fourth line: Change "such as color" to "such as line widening"
8	Last line: delete the word "also"
19	Table 2-1, Parameter 15: Change "Ten" to "Eleven"
25	Sec. 2.2.4.3, Last paragraph: Delete the sentence: "For those fields ... two descriptions given"
27	Table 2-2, Parameter 3: Add the word "Negated" before "Pointer to"

ERRATA (Continued)

Page	Change required
32	2.2.4.3.9.2, example: Change "(Sd,Yd,zd)" to read "(Xd,Yd,Zd)"
39-54	Table 2-3: The contents of this table are valuable as a reference but may not be entirely accurate and should be used with caution.
39	Table 2-3, first page, List of entity types: Add 128
	Table 2-3, first page, line 9c, comment: Change "00,01,05" to read "00,01,02,05"
	Table 2-3, first page, line 9d, comment: Add "See Note 1"
	Table 2-3, first page: Add "Note 1: Hierarchy is valid for entity types 118,120,122,130,140,144"
44	Table 2-3, sixth page, List of entity types: Add 312
	Table 2-3, sixth page, line 4, comment: delete "(Forms 20,21)"
	Table 2-3, sixth page, line 9b, comment: Add "For 312 cannot be 00"
48	Table 2-3, tenth page: Delete entire page
50	Table 2-3, twelfth page, line 9c, comment: Add "03 required for form 18"
51	Table 2-3, thirteenth page, line 9c, comment: Delete
53	Table 2-3, fourteenth page, line 5, comment: Delete
	Table 2-3, fourteenth page, line 5, notes: Delete heading and note 1.
54	Table 2-3, fifteenth page, line 9c, comment: Delete
76	Figure 2-12, top line: Delete
85	Sec 2.5.1: Replace with the attached Sec 2.5.1
93	Sec 2.5.4, first paragraph, fourth line: Delete "2 or"
	Swc 2.5.4, last paragraph, first line: Delete "logical"
99	Figure 2-23, Caption b) Change to : "Parent-Child Implicit Case"
115	Sec 3.3.9, parameter line 2: Change "DE" to "DE1"
	Sec 3.3.9, Parameter line N+1: Change "DE" to "DEN"

Page	Change required
130	Sec 3.7.3, parameter line 3: Change the overstruck "z" to a plain "z"
	Sec 3.7.3, parameter line 6: Change the overstruck "z" to a plain "z"
157	Sec 3.13.4, parameter line 1: Change "DEl" to "DE"
159	Figure 3-16, add the caption "Transformation Matrix Coordinate Systems"
171	Sec 3.16.2, parameter line 9+A+K: Change "XO" to "X0"
	Sec 3.16.2, parameter line 10+A+K: Change "YO" to "Y0"
	Sec 3.16.2, parameter line 11+A+K: Change "ZO" to "Z0"
175	Sec 3.17.2, parameter line 12+A+B+C: Change "U(O)" to "U(0)"
	Sec 3.17.2, parameter line 14+A+B+C: Change "V(O)" to "V(0)"
184	Sec 3.21.4, parameter line 2: Change "N1" to "N"
	Sec 3.21.4, parameter line 3: Change "DE" to "DE1"
	Sec 3.21.4, parameter line N1+2: Change "N1" to "N"
	Sec 3.21.4, parameter line N1+2: Change "DE" to "DEN"
	Sec 3.21.4, parameter line N1+3: Change "N1" to "N"
193	Sec 3.22.2, parameter line 5+NC: Change "X" to "X11"
	Sec 3.22.2, parameter line 6+NC: Change "y" to "y11"
	Sec 3.22.2, parameter line 7+NC: Change "z" to "z11"
	Sec 3.22.2, parameter line 8+NC: Change "RX" to "RX11"
	Sec 3.22.2, parameter line 9+NC: Change "RY" to "RY11"
	Sec 3.22.2, parameter line 10+NC: Change "RZ" to "RZ11"
	Sec 3.22.2, parameter line 7*NC-1: Change "X" to "X1NC"
	Sec 3.22.2, next parameter line: Change "y" to "y1NC"
	Sec 3.22.2, next parameter line: Change "z" to "z1NC"
	Sec 3.22.2, next parameter line: Change "RX" to "RX1NC"
	Sec 3.22.2, last parameter line: Change "X" to "XNN1"
194	Sec 3.22.2, first parameter line: Change "y" to "YNN1"
	Sec 3.22.2, next parameter line: Change "z" to "ZNN1"
	Sec 3.22.2, next parameter line: Change "RX" to "RXNN1"
	Sec 3.22.2, next parameter line: Change "RY" to "RYNN1"
	Sec 3.22.2, parameter line 10+NC+()*(): Change "RZ" to "RZNN1"
	Sec 3.22.2, next parameter line: Change "X" to "XNNNC"
	Sec 3.22.2, next parameter line: Change "Y" to "YNNNC"
	Sec 3.22.2, next parameter line: Change "z" to "ZNNNC"
	Sec 3.22.2, next parameter line: Change "RX" to "RXNNNC"
	Sec 3.22.2, next parameter line: Change "RY" to "RYNNNC"
	Sec 3.22.2, next parameter line: Change "RZ" to "RZNNNC"

Page	Change required
197	Sec 3.23.2, parameter line 1: Change "Nx" to "NX"
	Sec 3.23.2, parameter line 2: Change "Ny" to "NY"
	Sec 3.23.2, parameter line 3: Change "Nz" to "NZ"
	Sec 3.23.2, parameter line 4: Change "d" to "D"
200	Sec 3.24.2, parameter line 3: Change "B*PTR" to "BPTR" and change "C*" to "B"
	Sec 3.24.2, parameter line 5: Change "SoB*" to "SoB" (2 times)
202	Sec 3.25.2, parameter line 3+N1: Change "3+N1" to "4"
	Sec 3.25.2, parameter line 3+N1: Change "PTO1" to "PTO"
203	Sec 3.25.2, parameter line 4+N1: Change "4+N1" to "5"
	Sec 3.25.2, parameter line 3+N1+N2: Change "3+N1+N2" to "4+N2"
220	Sec 4.2.7.3, parameter line 6+N: Change "DE1" to "DEN"
221	Sec 4.2.8.2, parameter line 3: Change "DE" to "DE1"
	Sec 4.2.8.2, parameter line N+2: Change "DE" to "DEN"
229	Sec 4.2.9, step 4, third line: Change "test" to "text"
265	Sec 4.3.3.2, Name column: Insert "(" after "I" and add ")" to the end (Four times)
267	Sec 4.3.3.3.1, parameter line 2: Change "DE" to "DE1"
	Sec 4.3.3.3.1, parameter line 3: Change "DE" to "DE2"
	Sec 4.3.3.3.1, parameter line N+1: Change "DE" to "DEN"
285	Sec 4.3.3.3.10, parameter line 2: Change "DE" to "DE1"
	Sec 4.3.3.3.10, parameter line 3: Change "DE" to "DE2"
	Sec 4.3.3.3.10, parameter line N+1: Change "DE" to "DEN"
286	Sec 4.3.3.3.11, parameter line 2: Change "DE" to "DE1"
	Sec 4.3.3.3.11, parameter line 3: Change "DE" to "DE2"
	Sec 4.3.3.3.11, parameter line N+1: Change "DE" to "DEN"
288	Sec 4.3.3.3.12, parameter line 1: Change the name entry from "1" to "NTR"
292	Sec 4.3.3.13: Delete first four lines-- The Status Flag ... Hierarchy ... 00
	Sec 4.3.3.13, parameter line 8: Delete after "2=physical"
	Sec 4.3.3.13, parameter line 9: Delete after "2=fluid flow path"

Page	Change required
297	Sec 4.3.3.3.14.2, parameter line NS+4: Add Type "string" and Description "Signal Name"
305	Figure 4-28: Add Labels "ViewVolume" and "View Direction" appropriately to upper right hand example
344	Sec 4.3.7, paragraph 3, last line: Change "leel" to "level" Sec 4.3.7.2, Parameter line 1: Change name from "N" to "NP" Sec 4.3.7.2, Parameter line 1+N: Change name from "N" to "NP+1"
345	Sec 4.3.7.3.1, Parameter line 1: Change name from "Nl" to "NP" Sec 4.3.7.3.1, Parameter line 1+Nl: Change name from "Nl" to "NP+1"
349	Sec 4.3.7.3.5, parameter line 4: Change to read "Extension set by parameter 5" Sec 4.3.7.3.5, parameter line 6: Add "If" before "parameter 4"
356	Sec 4.3.7.3.11, Parameter line 1: Change name from "N" to "NP"
358	Sec 4.3.7.3.11: Immediately above parameter line 3, delete the vertical ellipsis and "n = Open Ended" Sec 4.3.7.3.11, parameter line 5: Change "TYP1" to "TYPI1" Sec 4.3.7.3.11: Immediately above parameter line 4+NI, delete the vertical ellipsis and "n = Open Ended" Sec 4.3.7.3.11, parameter line 4+NI: Change "TYPNI" to "TYPINI"
359	See amended page 359 attached.
361	Sec 4.3.7.3.11, PTYPE = 2, last matrix equation on page, second term in the third row of the 3x3 matrix: Change the subscript from "yz" to "yy"
373	Figure 4-39: Replace the illustration with the attached artwork
374	Sec 4.3.7.3.11, PTYPE = 12, on the line immediately following "ND = # of degrees of freedom": Insert "," between Z and M_x.

Page	Change required
383	Sec 4.3.7.3.11, PTYPE = 19: Enclose the three terms k_x, k_y, k_z with square brackets Sec 4.3.7.3.11, PTYPE = 19: Change "Now assuming no internal heat sources Q_I and the steady state where $\partial T/\partial t = 0$" to "Now assuming that there are no internal heat sources Q_I and no time dependencies $(\partial T/\partial t = 0)$." Sec 4.3.7.3.11, PTYPE = 19: Change "Consequently, integrating in one direction" to "then integrating the above equation in one direction yields" Sec 4.3.7.3.11, PTYPE = 19, second-to-end line in paragraph: Change "Where" to "where"
386	Sec 4.3.7.3.11, Example, Parameter line 1: Change name from "N" to "NP"
387	Sec 4.3.7.3.11, Example, Parameter line 1: Change name from "N" to "NP"
388	Sec 4.3.7.3.12, Parameter line 1: Change name from "N" to "NP" Sec 4.3.7.3.12, Parameter line 1+N1: Change name from "N" to "NP+1" and Change "NAMEN" to "NAMENP"
389	Sec 4.3.7.3.13, Parameter lines 3 and 4: Delete the word "constant" in the Type column
390	Sec 4.3.7.3.14, Parameter lines 2,3 and NP+1: Delete the word "constant" in the Type column Sec 4.3.7.3.14, Parameter line NP+1: Change name from "LN" to "LNP"
404	Sec 4.3.9.4.2, Parameter line 9: Change "PDR" to "PRD"
414	Sec 4.3.11.6, last equation: Change "O" to "0" four times
417	Sec 4.3.12, second line: Change "in referenced file." to "in a referenced file."
419	Sec 4.3.13, second paragraph: Change "4-45" to "4-46"
435	Appendix A, Figure A-3: Add label "View Volume" appropriately to upper right example

Page	Change required
474	Appendix D, Section D5, first line: Change "Section" to "section"
476	Appendix D, Section D6, last paragraph on page: Delete
479	Appendix E, last paragraph on page, third line: Change "into in the" to "into the"
481	Appendix F, last line: Insert "(" before "SMIT78)."
485	Appendix G: Ignore page in its entirety
497	List of References, line "ANSI78": Change "ANSI78" to "ANSI79" and change publication year from "1978" to "1979" List of References, line "ANSI79": Change "ANSI79" to "ANSI78" and reorder List of References, line "GORD 74": Change "GORD 74" to "GORD74"
504	Glossary, FLASH: Delete
515	Index of Topics, Annotation Entities: Move "General Symbol...254" to follow "General Note...223" Index of Topics, Associativity: Change External Reference File Index page references "90,269" to "90,281"
515	Index of Topics, Associativity: Change "Views Visible, Pen, Line Weight" to "Views Visible, Color, Line Weight"
516	Index of Topics, Attributes, Directory Entry: Change "Line Font Pattern Number" to "Line Font" Index of Topics: Add the following entries "Color Definition Entity...421" "Connect Point Entity...179"
517	Index of Topics: Add the following to "Entity" "Color Definition...421" "Connect Point...179" Delete "Entity" from "Finite Element Entity"

Page	Change required
518	Index of Topics: Add the following to "Entity" "General Symbol...254" "Implementor Defined...7" "Network Subfigure Definition...395" "Text Display Template...422" Delete "Entity" from "Rectangular Array Subfigure Instance Entity" Index of Topics: Change External Reference File Index Associativity page references "90,269" to "90,281"
519	Index of Topics: Add the following entries "Finite Element Entity...95,184" "General Symbol Entity...254" Index of Topics: Add the following entries to "Geometry" "Connect Point...179" "Curve on a Parametric Surface...198" "Finite Element...95,184" "Flash...167" "Nodal Displacement and Rotation...192" "Node...95,181" "Trimmed Surface...201"
520	Index of Topics, Mirror Flag: Change page reference to "223,423"
521	Index of Topics, Ordinate Dimension Entity: Change page reference to "245" Index of Topics, Property: Change "External Reference File Index" to "External Reference File List"
522	Index of Topics: Add the following entry "Signal String Associativity...297" Index of Topics: Add the following entries to "Structure Entities" "Circular Array Subfigure Instance Entity...401" "Color Definition Entity...421" "Drawing Entity...303" "Network Subfigure Definition Entity...395" "Network Subfigure Instance Entity...403" "Nodal/Constraint Entity...419" "Rectangular Array Subfigure Instance Entity...397" "Text Display Template Entity...422" Index of Topics, Structure Entities, External Reference File Index Associativity: Change page reference to "90,281" Index of Topics: Add the following Entry to "Surfaces" "Trimmed Parametric Surface...201"

Page   Change required

523   Index of Topics: Add the following entries
       "Text Display Template Entity...422"
       "Text Node Associativity...299"
       "Unordered Group Associativity...266,278"
       Index of Topics: Change "Views Visible, Pen, Line
       Weight Associativity" to "Views Visible, Color, Line
       Weight Associativity"

Section 2.5.1 Subfigures:

Subfigures have been provided to enable the use of a collection of entities many times within the model at various locations, orientations, and scales. In some cases, the collection itself is specified by a Subfigure Definition entity and each placement of the collection is specified by a Singular Subfigure Instance entity. The Network Subfigure Definition and Instance entity pair is similar in concept but has some special features to accommodate the notion of connect point in a network. (Figure 2-18 and Section 2.5.2 provide additional information about network subfigures.) In other cases, a Rectangular Array or a Circular Array Subfigure Instance entity specifies a base entity to be copied according to one of these two overall patterns.

Subfigures may be nested. For example, a Subfigure Definition entity may include a Singular Subfigure Instance entity as one entity in its collection. The notion of Depth of the Subfigure Definition entity is used to convey nesting information. Figure 2-17 illustrates two instances of a Subfigure Definition having a Depth value of N. That Subfigure Definition entity consists of two Geometry entities and one Subfigure Instance entity. The Subfigure Definition entity corresponding to this Instance entity then has a Depth value of N-1. A similar interpretation of Depth applies also to the Network Subfigure Definition and Instance entity pair. In these cases, the X,Y,Z location and the scale factor(s) in the Subfigure Instance entity help locate the Subfigure Definition entity into the definition space of the referring Subfigure Definition entity instead of into model space.

Thus, the processing sequence in these cases is as follows: Each entity in the Subfigure Definition is operated upon by its defining matrix and translation vector. Each entity is now located within the definition space of the Subfigure Definition entity. Then, the defining matrix and translation vector of the Subfigure Definition entity are applied. The entity collection of the Subfigure Definition entity is now located in the definition space of the Subfigure Instance entity. Next, the scale factor(s) located in the parameter data of the Subfigure Instance entity is (are) applied. This results in a scaling about the origin of the definition space of the Subfigure Instance entity. Next, the defining matrix and translation vector of the Subfigure Instance entity are applied. This locates the scaled entities either in model space or in the definition space of another Subfigure Definition entity. Finally, the X,Y,Z translation data located in the parameter data of the Subfigure Instance entity is applied. Note that this translation data can be relative to either model space or to the definition space of a Subfigure Definition entity. It will be relative to a definition space exactly when the Subfigure Instance entity is pointed to by another entity to which it is physically subordinate.

Index	Name	Type	Description
5+NI	NVALI1	Integer	Number of different values of the first independent variable
4+2*NI	NVALINI	Integer	Number of different values of the last independent variable
5+2*NI	VALI(1,1)	Real	First value of the first independent variable
4+3*NI	VALI(1,NVALI1)	Real	Last value of the first independent variable
$4+2 \cdot NI + \sum_{i=1}^{NI} NVALI_i$	VALI(NI,NVALINI)	Real	Last value of the last independent variable

$$M = 4 + 2 \cdot NI + \sum_{i=1}^{NI} NVALI_i$$

Index	Name	Type	Description
M+1	VALD(1,1)	Real	Value of the first dependent variable at the first data point
	VALD(J,K)	Real	Value of the j-th dependent variable at the k-th data point
N	VALD(ND,NVALINI)	Real	Value of the last dependent variable at the last data point

$$J = 1, \dots, ND$$

$$K = 1, \dots, \prod_{i=1}^{NI} NVALI_i$$

$$NP = M + ND * \prod_{i=1}^{NI} NVALI_i$$

$VALD(1,1) = f_1(VALI_1(1), VALI_2(1), \text{------} VALI_{NI}(1))$

$VALD(ND,1) = f_{ND}(VALI_1(1), \text{------} VALI_{NI}(1))$

$VALD(1,2) = f_1(VALI_1(2), \text{------} VALI_{NI}(1))$

$VALD(ND,2) = f_{ND}(VALI_1(2), \text{------} VALI_{NI}(1))$

$VALD(1,n) = f_1(VALI_1(NVALI_1) \text{------} VALI_{NI}(NVALI_{NI}))$

$VALD(ND,n) = f_{ND}(VALI_1(NVALI_1) \text{------} VALI_{NI}(NVALI_{NI}))$

Additional Pointers as required (see 2.2.4.2).

PAGE 373: FIGURE 4-39

Replace illustration at top of page with illustration shown here.

NBSIR 86-3359

INITIAL GRAPHICS EXCHANGE
SPECIFICATION (IGES), VERSION 3.0

Brad Smith
Joan Wellington

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Manufacturing Engineering
Automated Production Technology Division
Gaithersburg, MD 20899

April 1986

i

Special recognition is accorded to the following officers of the IGES
Organization who have contributed heavily to both the technical content
and the editorial review of this document.

Jeff	Altemueller	McDonnell Douglas AIS Co.
Raymond E.	Barker	Caterpillar Tractor Co.
William G.	Beazley	W. G. Beazley & Assoc.
Kalman	Brauner	The Boeing Commercial Airplane Co.
William C.	Burkett	McDonnell Aircraft Co.
Noel	Christensen	Allied Bendix Aerospace
Edward	Clapp	IBM Corporation
Robert	Colsher	IGES Data Analysis Corp.
Spencer	DePauw	Caterpillar Tractor Co.
Jim	Fleming	Cummins Engine Co. Inc.
William B.	Gruttke	McDonnell Douglas AIS Co.
Dennette A.	Harrod Jr	Applicon Corporation
Robert	Ivey	Westinghouse Electric Corp.
J.C.	Kelly	Sandia National Laboratories
Philip	Kennicott	General Electric Co.
Linda	Martino	IBM Corporation
Larry	O'Connell	Sandia National Laboratories
Curtis H.	Parks	General Dynamics Corp.
Kent	Reed	National Bureau of Standards
Patrick W.	Rourke	Newport News Shipbuilding
Douglas	Schenck	McDonnell Douglas
Fred	Stahl	IBM Corporation

THE FOLLOWING COMMITTEE MEMBERS HAVE CONTRIBUTED TO THIS DOCUMENT:

Joseph	Aebischer	Martin Marietta Data Systems
Jeff	Altemueller	McDonnell Douglas AIS Co
Bill	Anderson	General Dynamics Corp.
Robert E.	Anderson	NAVAIR Engineering Support Office
Colin	Ashley	PAFEC Ltd
Paul	Atallah	Electronic Data Systems Corp.
Dave	Barker	Hercules Aerospace
Raymond E.	Barker	Caterpillar Tractor Co.
William G.	Beazley	W. G. Beazley & Assoc.
Ronald	Belcher	McDonnell Douglas Automation Co.
Peter	Benjamin	Lockheed Missiles And Space Co.
Ted	Berenyi	Deere and Company
Dieter W.	Bergman	I P C
Kevin	Blackwell	Lawrence Livermore National Laboratory
Donald L.	Blank	Sperry Information Systems
Sandra	Bowen	Electronic Data Systems
James A.	Bradford	Allied Bendix Aerospace
Norman	Brainard	Electronic Data Systems
Kalman	Brauner	The Boeing Commercial Airplane Co.
Raymond A.	Brengs	Naval Ship R & D Center
David	Briggs	The Boeing Commercial Airplane Co.
Stan	Briggs	Naval Air Rework Facility
Ronald	Bryant	Xerox Corporation
William C.	Burkett	McDonnell Aircraft Co.
Richard	Butler	Computervision Corp.
Bradley K.	Call	PDA Engineering
Gerald E.	Campbell	Computervision Corp.
James	Carberry	NAVFAC
Lt Robert	Carringer	US Air Force ICAM Office
Dr. Nien-Hua	Chao	A T & T Bell Laboratories
Ken	Christensen	Ford Motor Company
Noel	Christensen	Allied Bendix Aerospace
Al	Cinque	M. Rosenblatt & Son
Edward	Clapp	IBM Corp.
Alan L.	Clark	Ford Motor Co.
Margaret	Cline	Lockheed Georgia
Robert	Clines	Martin Marietta Aerospace
Richard	Cochran	Pratt & Whitney
Charles E.	Cockrell	NASA Langley Research Center
Robert	Colsher	IGES Data Analysis Corp.
Richard	Costabile	Xerox Corporation
Gary A.	Cox	LTV Aerospace and Defense Company
Robert M.	Curry	TRW Electronics & Defense
Karlis	Dankers	TRW Electronics & Defense
Paul H.	Davis	Pratt & Whitney UTC
Anthony James	Day	Sikorsky Aircraft UTC
Mark	Demeranville	Sperry Defense Products Group
Spencer	DePauw	Caterpillar Tractor Co.
Karen	Deutsch	CADAM Inc
Richard N.	DiFronzo	Metagraphics Inc.
Ralph	Disa	Pratt & Whitney
George	Donnellan	Honeywell Electro-Optics Div.

Ron	Downer	Hughes Aircraft Co. EDSG
Alan E.	Dragoo	McDonnell Douglas MIS Co
Marc W.	Durnin	Lockheed-Georgia Company
Colin R.	Earl	Automation Technology Products
Robert J.	Easterday	Martin Marietta Energy Systems
Richard	Eppes Jr	US Army Missile Command
Nick	Fkiaras	Tektronix Inc.
Jim	Fleming	Cummins Engine Co. Inc.
Henry H.	Fong	FDA Engineering
Danny	Forthman	Bruning CAD
Peter	Fosselman	Douglas Aircraft
Leland H.	Frayseth	Bechtel Petroleum
William R.	Freeman	Allied Bendix Aerospace
Richard	Fuhr	The Boeing Commercial Airplane Co.
Roger	Gale	D. Appleton Co.
Walter F.	Geisinger	Construction Specifications Institute
William	Gerrein	General Electric Co.
Albert J.	Gibbons	Westinghouse Electric Corp.
Mitchell	Gilbert	Grumman Aerospace Corp.
Burton	Gischner	General Dynamics - Electric Boat Div.
Rainer	Glatz	University of Karlsruhe
Scott A.	Gordon	NASA Goddard Space Flight Center
William H.	Gray	Martin Marietta Energy Systems
William B.	Gruttke	McDonnell Douglas AIS Co.
John	Hardt	Dana Corp.
C. Scott	Harris	Hughes Aircraft Co
Dennette A.	Harrod Jr	Applicon
Michael	Hastings	Control Data Corporation
Don	Hemmelgarn	International TechneGroup Inc
Edward C.	Hong	Electronic Data Systems
William A.	Hussong	Honeywell Inc. Avionics Division
Richard	Isler	Sandia National Laboratories
Robert	Ivey	Westinghouse Electric Corp.
Dwight L.	Jaeger	Los Alamos National Laboratory
Scott E.	Jorgenson	Control Data Corporation
Anthony	Joyce	Naval Facilities Engineering Command
Esfan	Kamvar	AT & T Bell Laboratories
Thomas	Keane	The Boeing Commercial Airplane Co.
J.C.	Kelly	Sandia National Laboratories
Debbie	Kengott	AutoTrol Technology Corp.
Philip	Kennicott	General Electric Co.
Stephen T.	Kertis	Naval Weapons Center
Gabor	Kiss	Bell Communications Research
Karen L.	Kontry	Electronic Data Systems Corporation
Sudhir	Kshirsagar	SDRC
Stanley J.	Kukla	General Dynamics
M.J.	Kutkus	Hughes Aircraft Co:
Harry	Ladd	DuPont
John	Lamoureux	IIT Research Institute
Deborah	LaFay	Westinghouse Electric Corp.
Lanse M.	Leach	US Military Academy
Kaiman	Lee	Naval Facilities Engineering Command
Bruce	Lepisto	Department of Defense

```
Olga            Lichten        IBM Corp.
Michael         Liewald        Hughes Aircraft Co.
Vincent         Lin            Automation Technology Products
Robert          Lipman         Naval Ship R & D Center
Lincoln         Little         Cummins Engine Co. Inc.
Julie           Long           Electronic Data Systems
Bill            Loye           Interconnics
Robert          MacLatchie     PlanPrint Company
Spencer         Magleby        General Dynamics Corp.
Bill            Marks          Hughes Aircraft Co
Linda           Martino        IBM Corp.
Steven          Mastrangelo    Micro Control Systems Inc.
Ralph J.        Mayer          Adra Systems, Inc.
Raphael         McBain         General Dynamics Corp.
Marie           McFarland      Hughes Aircraft Co.
Stanley         McMillen       Gerber Systems Technology Inc.
H. W. Guy       Meyer          Wisconsin Dept of Transportation
Thurber J.      Moffett        Northrup
Charles B.      Morrill        IBM Corp.
R.G.            Morrison Jr    McDonnell Aircraft Co.
Joseph A.       Mudd           Bechtel Petroleum Inc.
Roger           Nagel          Lehigh University
Paul A.         Nelson         Hughes Aircraft Co.
David           Norling        Boeing Computer Services
Fred J.         Norton         Lawrence Livermore National Laboratory
Prof. Horst     Nowacki        Technical University Berlin
Larry           O'Connell      Sandia National Laboratories
W. Rob          Oakes          Los Alamos National Laboratory
Jeffrey         Otten          General Electric Co
Connie          Panzica        General Motors BOC Hqts
Geoffrey H.     Parker         Intergraph Corp.
Curtis H.       Parks          General Dynamics Corp.
Klaus           Pasemann       Volkswagenwerg AG
Alan            Peltzman       Software Consultant
Steve           Peters         Pierce,Goodwin,Alexander
Freda           Phelps         Lawrence Livermore National Laboratory
Jon             Pittman        HOK Computer Services Corp.
Vladimir        Pochop         AutoDesk Inc.
Tim             Prohaska       Allied Bendix Aerospace
Lloyd           Purves         NASA Goddard Space Flight Center
Kent            Reed           National Bureau of Standards
Gaylen R.       Rinaudot       National Bureau of Standards
Phillip         Rosol          Martin Marietta Data Systems
Patrick W.      Rourke         Newport News Shipbuilding
Walter J.       Rygiel         Ford Motor Co.
Randy           Schmid         Hughes Aircraft Co EDSG
Dirk            Schroeter      Martin Marietta Orlando Aerospace
Arnold          Shak           Grumman Data Systems
Jerry           Shaver         Gerber Systems Technology
Chia Hui        Shih           SDRC
Marie           Skutch         Westinghouse Electric Corp.
Bradford        Smith          National Bureau of Standards
Keith M.        Smith          Summa Technologies Inc.
```

Jim	Snyder	Martin Marietta Energy Systems
Michael V.	Spinosa	Computervision Corp.
Fred	Stahl	IBM Corp.
Jeffrey	Star	University of California
John	Stedman	Mentor Graphics Corp.
Edward T.	Stickle	Martin Marietta Energy Systems
Charles L.	Stoddard	Pratt & Whitney
Randy	Terada	CADAM Inc.
Julia	Terry	Martin Marietta Energy Systems
David	Theilen	Allied Bendix Aerospace
Mark A.	Thiel	McDonnell Douglas Automation Co.
Alistair	Thompson	Cambridge Interactive Systems Ltd.
C.B. "Bill"	Thompson	Southern Company Services Inc
Paul	Thompson	Control Data
Vince E.	Thompson	Burns & McDonnell Engineering Co
Dietmar	Trippner	BMW
James	Tulenko	McDermott, Inc.
James A.	Turner	University of Michigan
Robert J.	Ventura	Martin Marietta Aerospace
Earl P.	Weaver	Ballistics Research Laboratory
Donald	Wechsler	The MITRE Corporation
Jerry A.	Weiss	McDonnell Aircraft Co.
Uwe	Weissflog	IBM Germany Dept 2081
Robert	White	Computervision Corp.
George	Whitehurst	NASA Langley Research Center
William	Whiteman	Enterprise Systems Design Inc.
Gary	Whitt	Intergraph Corp.
Peter	Wilson	General Electric Co
Robert M.	Wilson	Martin Marietta Energy Systems
Richard C.	Winfrey	Digital Equipment Corp.
Edward R.	Wittenberg	Ford Motor Co.
Deborah A.	Wright	United Technologies-Sikorsky Aircraft
Sheree	Yang	Ford Aerospace & Communications Co
Siu Fun	Yu	Hewlett-Packard Co.
John	Zimmerman	Allied Bendix Aerospace

FOREWORD

Version 3.0 of the Initial Graphics Exchange Specification greatly improves upon earlier publications of IGES Version 1.0 in 1980, ANSI Y14.26M in 1981, and IGES Version 2.0 in 1983 by extending and refining the specification in many areas. The refinements, which improve on the clarity and precision of the specification, are in many cases the result of considerable implementation experience gained with IGES across a large number of systems. The many technical extensions in this document reflect a desire to expand the specification's capability to communicate a wider range of product data developed and used by computer aided design and manufacturing systems. They are the result of two and one half years of technical work and were approved by IGES committee members in February 1985.

Despite the many changes to the earlier work, IGES Version 3.0 remains for all practical purposes upwards compatible with Version 2.0. This means that a translator fully conforming to Version 3.0 can correctly interpret IGES files written in accordance with prior versions of IGES. Minor technical changes do exist which do violate this objective or which would have violated this objective had any earlier translator been written to put out some of the more complex entity structures of Version 2.0. One example of a non-upwards compatible change is in the specification of connect points where the earlier approach was shown to be logically incorrect. Other changes introduced into Version 3.0 make obsolete the older entities, but it is believed that these older entities had never been implemented.

Considerable clarifications have been made in the 3.0 document in areas of the View and Drawing entities, in the Global parameters including default values, in the Units flag, in the Transformation Matrix pointer, in the Leader entity and in the Parametric Spline Curve and Surface entities.

New entity capability has been added in the geometry area for Offset Curves, Offset Surfaces and Curves on a Parametric Surface. A Trimmed Surface entity allows the definition of a surface boundary.

In the annotation area new capability exists for representing a larger range of annotation style of the base value and the tolerance limits. Additionally, a more compact definition for crosshatching is given.

Version 3.0 introduces greatly enhanced capability for user defined MACRO's essential for standard part libraries. By adding Labels, Branching and Calling arguments to the previous MACRO capability, the Version 3.0 MACRO is able to represent parametrized IGES constructs that are user defined. Coupled with another new extension for External File Reference, users will be able to implement large standard libraries of symbols or components.

Finally, Version 3.0 introduces a means of reducing IGES file size to one third of its previous size. Called the Compressed ASCII Format, the optional technique addresses storage size and telecommunications costs. Utility programs are included in an Appendix for two way conversion to the Compressed IGES physical file format.

TABLE OF CONTENTS

LIST OF FIGURES

1 GENERAL

1.1 Purpose

This document establishes information structures to be used for the digital representation and communication of product definition data. Use of the specification established herein permits the compatible exchange of product definition data used by various CAD/CAM (Computer Aided Design and Computer Aided Manufacturing) systems.

1.2 Field of Application

This Specification defines a file structure format, a language format, and the representation of geometric, topological, and non-geometric product definition data in these formats. Product definition data represented in these formats will be exchanged via a variety of physical media. The specific features and protocols for the communications media are the subject of other standards. The methodology for representing product definition data in this Specification is extensible and independent of the modeling methods used.

Section 2 defines the communications file structure and format. It explains the function of each of the sections of a file. The geometry data representation in Section 3 deals with two- and three-dimensional edge-vertex models and with simple surface representations. Section 4 specifies non-geometric representations, including common drafting practices, data organization methods, and data definition methods.

In Sections 3 and 4, the product is described in terms of geometric and non-geometric information, with non-geometric information being divided into annotation, definition, and organization. The geometry category consists of elements such as points, curves, and surfaces that model the product. The annotation category consists of those elements which are used to clarify or enhance the geometry, including dimensions, drafting notation, and text. The definition

1

category provides the ability to define specific properties or characteristics of individual or collections of data entities. The structure category identifies groupings of elements from geometric, annotation, or property data which are to be evaluated and manipulated as single items.

1.3   Concepts of Product Definition

This Specification is concerned with the data required to describe and communicate the essential engineering characteristics of physical objects as manufactured products. Such products are described in terms of their physical shape, their dimensions, and information which further describes or explains the product. The processes which generate or utilize the product definition data typically include design, engineering analysis, production planning, fabrication, material handling, assembly, inspection, marketing, and field service.

The requirements for a common data communication format for product definition can be understood in terms of today's CAD/CAM environment. Traditionally, engineering drawings and associated documentation are used to communicate product definition data. Commercial interactive graphics systems, originally developed as aids to producing these two-dimensional drawings, are rapidly developing sophisticated three-dimensional edge-vertex modeling capability. In parallel, extensive research work is being conducted in advanced geometric modeling techniques (e.g., parametric representations and solid primitives) and in CAM applications utilizing product definition data in manufacturing (e.g., NC machining and computer-controlled coordinate measurement). The result is rapid growth of CAD/CAM applications which should be able to exchange product definition data, but which usually employ incompatible data representations and formats. In addressing this compatibility problem, this Specification is concerned with needs and capabilities of current and advanced methods of CAD/CAM product definition development.

Product definition data may be categorized by their principal roles in defining a product. An example of such a categorization is presented in Figure 1-1. This Specification specifies communication formats (information structures) for subsets of the product definition.

o   **ADMINISTRATIVE**

        Product Identification
        Product Structure

o   **DESIGN/ANALYSIS**

        Idealized Models

o   **BASIC SHAPE**

        Geometric
        Topological

o   **AUGMENTING PHYSICAL CHARACTERISTICS**

        Dimensions and Tolerances
        Intrinsic Properties

o   **PROCESSING INFORMATION**

o   **PRESENTATIONAL INFORMATION**

FIGURE 1-1 CATEGORIES OF PRODUCT DEFINITION

3

1.4   Concepts of the File Structure

A format to allow the exchange of a product definition between CAD/CAM systems must, as a minimum, support the communication of geometric data, annotation, and organization of the data. The file format defined by this Specification treats the product definition as a file of entities, each entity being represented in an application-independent format, to and from which the native representation of a specific CAD/CAM system can be mapped. The entity representations provided in this Specification include forms common to the CAD/CAM systems currently available and forms which support the system technologies currently emerging.

The fundamental unit of information in the file is the entity. Entities are categorized as geometric and non-geometric. Geometric entities represent the definition of the physical shape and include points, curves, surfaces, and relations which are collections of similarly structured entities. Non-geometric entities typically serve to enrich the model by providing a viewing perspective in which a planar drawing may be composed and by providing annotation and dimensioning appropriate to the drawing. Non-geometric entities further serve to provide specific attributes or characteristics for individual or groups of entities and to provide definitions and instances for groupings of entities. The definitions of these groupings may reside in another IGES file. Typical non-geometric entities for drawing definition, annotation, and dimensioning are the view, drawing, general note, witness line, and leader. Typical non-geometric entities for attributes and groupings are the property and the associativity entities.

A file consists of five or six sections: Flag (in the case of the binary or compressed ASCII format), Start, Global, Directory Entry, Parameter Data, and Terminate. A file may include any number of entities of any type as required to represent the product definition. Each entity occurrence consists of a directory entry and a parameter data entry. The directory entry provides an index and includes descriptive attributes about the data, while the parameter data provides the specific entity definition. The directory data are organized in fixed fields and are consistent for all entities to provide simple access to frequently used descriptive data. The parameter data are entity specific and are variable in length and

format. The directory data and parameter data for all entities in the file are organized into separate sections, with pointers providing bi-directional links between the directory entry and parameter data for each entity. The Specification provides for groupings whose definitions will be found in a file other than the one in which they are used.

Each entity defined by the file structure of Section 2 has a specific assigned entity type number. While not all are assigned at this time, entity numbers 0001 through 0599 and 0700 through 5000 are allocated for specific assignment. Entity type numbers 0600 through 0699 and 10000 through 99999 are for implementor-defined (i.e. MACRO) entities. For user defined entities see section 1.5.7. The Index of Topics includes an alphabetical listing of entity types.

Some entity types include a form number as an attribute. The form number serves to further define or classify an entity within its specific type.

The entity set includes a provision for associativities and properties. The associativity provides a mechanism to establish relationships among entities, and to define the meaning of the relationship. The property allows specific characteristics, such as color, to be assigned to an entity or collection of entities. Each entity format includes a structure for an arbitrary number of pointers to associativities and properties. The file structure provides for both standard associativities and properties to be included in the Specification, and unique definitions which will be defined by the user.

1.5   Concepts of the Information Structures for Wire-Frame Model Descriptions

The wire-frame model refers to the entity set defined by Sections 3 and 4, and comprises an entity-based product definition file. The entity types, as described in 1.4, are categorized as geometric and non-geometric. In general, the geometric entities are defined independently of one another (surfaces are an exception). Features have been provided to define and compose relationships among entities to enhance the model. The non-geometric entities include structures in which an entity may be defined by a collection of other entities and structures which are independent.

5

Several entity types which are used to provide relations or definitions are essential to the file structure methodology of this Specification and are described below.

1.5.1 **Property Entity.** The PROPERTY entity allows non-geometric numeric or textual information to be related to any entity. Any entity occurrence may reference one or more property entity occurrences as required.

Property entities themselves may exist independently of other entities. In this case the property is defined to be a property of the level indicated in the level field of the directory entry (DE) of the property. This allows for a general property to apply to all entities of a given level or for the assignment of an applications function to a level. Because the level field in a DE is also allowed to point to a property of levels, properties may be applied to multiple levels.

1.5.2 **Associativity Entities.** The Associativity Entities are designed for use when several entities must be logically related to one another. Two types of entities are involved here: ASSOCIATIVITY DEFINITION and ASSOCIATIVITY INSTANCE. The associativity definition entity is used to specify the structure of the logical relationship, and the associativity instance entity is used to specify the information involved in a particular occurrence of the relationship.

Some associativities are defined as part of this Specification. These intrinsic definitions include GROUP and VIEWS VISIBLE associativities, and are defined in Section 4.3.

1.5.3 **View Entity.** A drawing or equivalent human-readable representation of the geometric model of a product is a two-dimensional projection of a selected subset of the model, together with non-geometric information such as text. The VIEW entity and VIEWS VISIBLE forms of associativity control such representations. These provide information for orientation, clipping, line removal, and other characteristics associated with individual views rather than with the model itself.

1.5.4 **Drawing Entity.** The DRAWING entity allows a set of views to be identified and arranged for human presentation. Note that the view and drawing entities contain only the rules and parameters for extracting drawings from the geometric model. The actual product definition is not duplicated in various views, eliminating risk of conflicting or ambiguous information.

1.5.5   **Transformation Matrix Entity.**   The TRANSFORMATION MATRIX entity allows translation and rotation to be applied as needed, to any entity in the construction of the model and to the development of views and drawings of the model.

1.5.6   **MACRO Entities.**   This Specification includes a MACRO DEFINITION entity for defining new entity types which may then be used in the defining file in the same manner as the intrinsically defined entities.   A language for defining these new entity types is specified in 4.3.6.

1.5.7   **Implementor Defined Entities.**   This specification allows implementors to include entities in their files that are not defined in this document but which have specific user meanings.   This feature supports the objective of the Specification to act as an archiving format where the receiving system is the same as the sending system. In this way, the implementor is able to archive those data forms which may be unique to a particular system.

From time-to-time, files with such implementor-defined entities are used with applications which attempt to edit the file.   In this situation, processing problems can arise because, without an entity definition, the editor cannot know which parameter values are pointers that have to be updated, and which are simply data values that should not be updated.

To avoid this problem, implementors should use MACRO definitions and instances of MACRO entities with entity type numbers in the range of 5001 to 9999 inclusively.   (See Section 4.3.6 for information on how to use the MACRO capabilities of the specification.) This means that for each different implementor-defined entity type, there will be a MACRO Definition Entity (Type 306).   In order to accomplish the desired result, all that needs to be present in the parameter data for these MACRO definitions is the first MACRO statement which defines the parameter list, and an ENDM statement to terminate the definition.

1.6   Appendices

As an aid to the implementor/user, a series of appendices is included with this Specification. Appendix A gives three part file examples. Appendix B describes an electrical-electronic product representation, and Appendix C, a plant flowsheet representation.   Appendix D provides explanation of spline representation and approaches for conversion techniques. Appendix E discusses the numerical stability of conic arcs. Appendix F provides mappings between color spaces. A set of FORTRAN utilities is documented in Appendix G to convert the ASCII Form of physical file structure to and from the Compressed ASCII Form. In addition, a List of References, a Glossary and an Index of Topics are also included.

2   DATA FORM

2.1   **General**

Two different formats are defined to represent IGES data. These formats are
ASCII (ANSI68, ANSI74, ANSI77), and Binary. The binary form uses a byte
oriented (bit string) structure which may be especially suited to the
transmission of large files.

The constants, free format rules, file organization, and information structure
are discussed in terms of displayed or printed ASCII form. Following this
discussion, the binary form is defined together with necessary changes in
constants and file structure.

2.2   **ASCII Form**

The ASCII form has two format types: a fixed (80 character) line length
format, and a compressed format. In the fixed line length ASCII format, the
entire file is partitioned into 80 character units beginning with the first
character in the file. These units are called lines. The term "column" refers
to the character position in a line. The file is divided into sections. The
section identification character shall occupy column 73 of each line.
Columns 74 through 80 are specified for the section sequence number of each
line. For the compressed ASCII format, the DIRECTORY ENTRY and
PARAMETER DATA sections are excepted from the above two rules, and will
be defined in a later section. The remaining columns are assigned to fields as
defined in the file section description. The term "record" refers to the set of
parameters for one entity within one file section. A record consists of one or
more lines.

2.2.1   **Sequence Numbers**

A sequence number is a string of from one to seven digits and is the means of
indexing lines within the various sections of the data file. The sequence
numbers for each section begin with 1 (0000001) and continue sequentially
without interruption to the value corresponding to the number of lines in the
section. A sequence number may have either leading zeros or leading blanks
and is right-justified in its field in the line (columns 74-80).

9

The sequence number is preceded in the line by a single letter code in column 73 identifying the section in which the line resides:

Section	Letter Code
Flag (optional)	B or C
Start	S
Global	G
Directory Entry	D
Parameter Data	P
Terminate	T

Letter codes "B" and "C" are used to signify binary (see section 2.4.2.1) and compressed ASCII (see section 2.3.1) information, respectively.

2.2.2   Constants

This specification defines five types of constants: integer (or fixed point), real (or floating point), string, pointer, and language statement. Regardless of whether the constants appear in a fixed or free format in the data file, certain rules apply to their formation and interpretation:

o   Blanks are only significant in string and language statement constants. A numeric field of all blanks is considered to denote the default value for that field. No blanks are allowed between the beginning of a numeric constant (i.e., its sign, first numeric digit, or decimal point) and the end of that constant (i.e., the last character position allocated in fixed format or the delimiter character in free format). Leading blanks in the parameters containing numeric constants are ignored. Blanks between the end of any constant and the delimiter following the constant are not allowed.

o   Numeric constants cannot contain embedded commas.

o   The absolute magnitude of an integer constant may not exceed the value $2^{**}(N-1)-1$, where N is the number of bits used to represent the integer value (Global parameter 7).

10

Similarly, the absolute magnitude and precision of a real constant may not exceed that indicated by Global parameters 8-9 (for single precision) and 10-11 (for double precision).

o   Only string and language statement constants may cross field/line boundaries. When such a constant does cross a boundary, it is considered to extend to the last usable column on the current line and then to continue with the first column on the succeeding line. The last usable column on lines in the Parameter Data section is column 64; on lines in all other sections it is column 72. A string constant may not be broken before the Hollerith delimiter (H).

o   A numeric constant may be either signed or unsigned. If signed, the leading plus or minus determines the sense of the constant. If unsigned, the sense is assumed to be non-negative.

2.2.2.1   Integer Constants

An integer constant (sometimes called a fixed point constant) is always an exact representation of an integer value. It may assume a positive, negative or zero value. It may assume only an integral value.

The form of an integer constant is an optional sign followed by a non-empty string of digits. The digit string is interpreted as a decimal number.

The following are examples of valid integer constants (assuming the value of Global parameter 7 is 32):

| 1 | 150 | 2147483647 | +3451 |
| 0 | -10 | -2147483647 | |

11

2.2.2.2   **Real Constants**

A real constant (sometimes called a floating point constant) is a processor approximation of the value of a real number. It may assume a positive, negative, or zero value. A real constant may be either a basic real constant, a basic real constant followed by an exponent, or an integer constant followed by an exponent.

A real constant may be of either single or double precision. The distinction is in the precision of the processor's representation of the real number which is specified in Global parameters 8 through 11. A double precision constant may be either a basic real constant followed by a double precision exponent, or an integer constant followed by a double precision exponent.

The form of a basic real constant is, in order, an optional sign, an integer part, a decimal point, and a fractional part. Both the integer part and the fractional part are strings of digits; either of these parts may be omitted but not both. A basic real constant is interpreted as a decimal number.

The form of a real exponent is the letter E followed by an optionally signed integer constant. A real exponent denotes a decimal power of ten by which the preceding constant is multiplied.

The form of a double precision exponent is the letter D followed by an optionally signed integer constant. A double precision exponent denotes a decimal power of ten by which the preceding constant is multiplied.

The following are examples of valid real constants:

256.091	0.	-0.58	+4.21
1.36E1	-13E-02	0.1E-3	1.E+4
145.98763D4	-2145.980001D-5	0.123456789D+09	-.43E2

2.2.2.3   **String Constants**

String constants are represented in the Hollerith form as specified in Appendix C of the current FORTRAN standard (ANSI78). A string constant is an arbitrary sequence of ASCII characters. Blanks, parameter delimiters, and record delimiters are treated simply as characters within the string. There is no limit on the length of a string constant.

The form of a string constant is a non-zero, unsigned integer constant (character count), followed by the letter H, followed by a string of characters consisting of the number of contiguous characters specified by the character count.

The following are examples of valid string constants:

3H123   10HABC.,;ABCD
8H0.457E03   12H HELLO THERE

2.2.2.4   **Pointer Constants**

A pointer constant is a non-empty string of from one to seven digits. Pointer constants are used to identify a line in either the same or a different section of the data file. The magnitude of the pointer constant corresponds to the sequence number of the referenced line, and the referenced file section is determined by the context of the reference. Pointer constants are unsigned except where they are alternative parameters in a field. Pointer constants whose magnitude requires fewer than seven digits may use leading zeros or leading blanks in fixed format fields.

2.2.2.5   **Language Statement Constants**

The language statement constant is an arbitrary character string made up of alphanumeric, punctuation, and blank characters from the ASCII character set. The language statement constant is not preceded by the

character count and the Hollerith delimiter H as is the string constant. Section 4.3.6 defines the syntax of the language statement constant as used for the MACRO entity. The length of the language statement constant is determined by means of the Parameter Data line count in the Directory Entry record for the entity (see Directory Entry parameter 14).

2.2.3   **Rules for Forming and Interpreting Free Formatted Data**

The data in several sections of a file may be entered in free format. The free format will apply to a range of columns of a line in the section and to the same range of columns of successive lines as needed. This free format feature allows the specification of parameters in the prescribed order without restricting the placement of the parameter to a particular location on a line. When free format is permitted, the following rules apply (in addition to those in section 2.2.2):

o   The parameter delimiter (Global parameter 1 - defaulting to a comma) is used to separate parameters.

o   The record delimiter (Global parameter 2 - defaulting to a semicolon) is used to terminate the record (i.e., to terminate a list of parameters).

o   When two parameter delimiters, or a parameter and record delimiter, appear adjacent to each other, or are separated by only blanks, the parameter is considered not to have been specified in the file and should be given its default value. Unless specifically noted, the default value for a numeric parameter is zero, and the default value for a string parameter is null. It is the responsibility of the preprocessor to ensure that these default values are reasonable for the particular parameter in question.

o   When a record delimiter appears before the list of parameters is complete, all remaining parameters should be given their default values (see above for a discussion of assigning default values).

o   The end of the data portion of the physical line (i.e., column 72 in the Global Section, and column 64 in the Parameter Data Section) is not to be construed to act as either a parameter delimiter or a record delimiter.

o   The parameter delimiter and record delimiter characters do not maintain their special significance when included within a string constant.

o   A numeric constant, including its trailing delimiter, cannot extend across a line boundary.

2.2.4   **File Structure**

The file contains six subsections which must appear in order as follows:

a.   Flag Section (optional)
b.   Start Section
c.   Global Section
d.   Directory Entry Section
e.   Parameter Data Section
f.   Terminate Section.

The Flag Section of the file is used, when present, to indicate that the file is in the Binary Form (2.4.2.1) or in the Compressed ASCII Form (2.3.1).

2.2.4.1   **Start Section.** The start section of the file is designed to provide a man-readable prolog to the file. There must be at least one start record, and all records in the section must have the letter S in column 73 and a sequence number in column 74 through 80 (See 2.2.1). The information in columns 1 through 72 is not formatted in any special way except that the ASCII character set must be used. An example of a start section is shown in Figure 2-1.

START SECTION

```
|1                                            72|73      80|
|   THIS SECTION IS A MAN READABLE              |S0000001|
|  PROLOG TO THE FILE. IT CAN CONTAIN           |S0000002|
|  AN ARBITRARY NUMBER OF LINES                 |S0000003|
|                       •                       |    •   |
|                       •                       |    •   |
|                       •                       |    •   |
|                       •                       |    •   |
|                       •                       |    •   |
|                       •                       |    •   |
|  USING ASCII CHARACTERS IN COLUMNS 1-72.      |S000000N|
|                                               |        |
```

FIG. 2-1 START SECTION

2.2.4.2   Global Section

The global section of the file contains the information describing the pre-processor and information needed by the post-processor to handle the file. All records in the global section shall contain the letter G in column 73 and a sequence number (See 2.2.1). The first two global parameters are used to define the parameter delimiter and record delimiter characters if necessary. The default characters are "comma" and "semicolon" respectively.

The parameters for the global section are input in free format as described in 2.2.3. As implied in 2.2.3, the global parameters will end with the record delimiter. If the global section specifies new delimiter characters, they take over immediately and are used in the global section as well as the rest of the file. This is possible, because the comma and semicolon delimiter specifications are the first two global parameters. The parameters in the global section are described in Table 2-1 and the paragraphs that follow. Unless explicitly stated, no defaults are provided.

TABLE 2-1 PARAMETERS IN THE GLOBAL SECTION

INDEX	FIELD TYPE	DESCRIPTION
1	String	Parameter delimiter character (default is comma)
2	String	Record delimiter character (default is semicolon)
3	String	Product identification from sending system
4	String	File name
5	String	System ID
6	String	Preprocessor version

7	Integer	Number of bits for integer representation
8	Integer	Maximum power of ten representable in a single precision floating point number on the sending system
9	Integer	Number of significant digits in a single precision floating point number on the sending system
10	Integer	Maximum power of ten representable in a double precision floating point number on the sending system
11	Integer	Number of significant digits in a double precision floating point number on the sending system
12	String	Product identification for the receiving system
13	Real	Model space scale (example: .125 indicates a ratio of 1 unit model space to 8 units real world)
14	Integer	Unit flag
15	String	Units. Ten values of the unit flag have been defined. (See 2.2.4.2.15).
16	Integer	Maximum number of line weight gradations (1-32768). Refer to the directory entry para-meter 12 (See 2.2.4.3) for use of this para-meter.
17	Real	Size of maximum line width in units. Refer to the directory entry parameter 12 (See 2.2.4.3) for use of this parameter.
18	String	Date & time of file generation 13HYYMMDD.HHNNSS where: YY is year (last 2 digits) MM is month (01-12) DD is day (01-31) HH is hour (00-23) NN is minute (00-59) SS is second (00-59)

19	Real	Minimum user-intended resolution or granularity of the model expressed in units defined by parameter 15 (example .0001)
20	Real	Approximate maximum coordinate value occurring in the model expressed in units defined by parameter 15. (Example: 1000.0 means for all coordinates $\lvert X \rvert$, $\lvert Y \rvert$, $\lvert Z \rvert \leq 1000$.)
21	String	Name of author
22	String	Organization
23	Integer	Integer value corresponding to the version of IGES used to create this file. (See 2.2.4.2.23 for Correspondence Table.)
24	Integer	Drafting standard in compliance to which the data encoded in this file was generated. (See 2.2.4.2.24)

2.2.4.2.1   **Parameter Delimiter Character.** This parameter indicates which character is used to separate parameter values in the Global and Parameter Data sections. Each occurrence of this character denotes the end of the current parameter and the start of the next parameter. Two exceptions exist: (1) string constants in which the delimiter character may be part of the string; (2) language statements in which the delimiter character may be a part of the language syntax. The default value is a comma. See 2.2.3.

2.2.4.2.2   **Record Delimiter.** This parameter indicates which character is used to denote the end of a list of parameters in the Global section and each Parameter Data section entry. Each occurrence of this character denotes the end of the current parameter list. Two exceptions exist: (1) string constants in which the delimiter character may be part of the string; (2) language statements in which the delimiter character may be a part of the language syntax. The default value is a semicolon. See 2.2.3.

2.2.4.2.3   **Product Identification From Sender.** This is the name of the identifier which is used by the sender to reference this product.

2.2.4.2.4   **File Name.** This is the name of the IGES file.

2.2.4.2.5   **System ID.** This parameter is an identification code which should uniquely identify the system which generated this file. It includes both the name of the system and the version of software on that system.

2.2.4.2.6   **Preprocessor Version.** This parameter identifies the version of the preprocessor software which created this file.

2.2.4.2.7   **Number of Bits for Integer Representation.** This parameter indicates how many bits are present in the integer representation of the sending system. This parameter sets limits on the range of values for integer parameters in the file.

2.2.4.2.8   **Single Precision Magnitude.** This parameter indicates the maximum power of ten which may be represented as a single precision floating point number on the sending system.

2.2.4.2.9   **Single Precision Significance.** This parameter indicates the number of decimal digits of significance which can be accurately represented in the single precision floating point representation on the sending system.

2.2.4.2.10   **Double Precision Magnitude.** This parameter indicates the maximum power of ten which may be represented as a double precision floating point number on the sending system.

2.2.4.2.11   **Double Precision Significance.** This parameter indicates the number of decimal digits of significance which can be accurately represented in the double precision floating point representation on the sending system.
Example: For an IEEE 754-1985 (IEEE85) floating point representation with 32 bits, the magnitude and significance parameters have the values 38 and 6, respectively; for a representation with 64 bits, the values are 308 and 15, respectively.

2.2.4.2.12   **Product Identification for the Receiver.** This is the name or identifier which is intended to be used by the receiver to reference this product.

2.2.4.2.13   **Model Space Scale.** The ratio of model space to real world space.

2.2.4.2.14   **Unit Flag.** An integer value denoting the measuring system used in the file. The values in the file are assumed to be:

       Unit flag   =   1   (Inches)
                    =   2   (Millimeters)
                    =   3   (See Parameter 15 for name of units)
                    =   4   (Feet)
                    =   5   (Miles)
                    =   6   (Meters)
                    =   7   (Kilometers)
                    =   8   (Mils, i.e. 0.001 inch)
                    =   9   (Microns)
                    =   10   (Centimeters)
                    =   11   (Microinches)

This is the controlling definition of units. A value of '3' should only be used when it is intended to transfer data to a system using the same units, in

which case parameter 15 must be used to provide additional information as to those units.

2.2.4.2.15 <u>Units.</u> A string constant naming the model units in the system, e.g.:

2HIN or 4HINCH	(model units are inches)
2HMM	(model units are millimeters)
2HFT	(model units are feet)
2HMI	(model units are miles)
1HM	(model units are meters)
2HKM	(model units are kilometers)
3HMIL	(model units are mils)
2HUM	(model units are microns)
2HCM	(model units are centimeters)
3HUIN	(model units are microinches)

When the unit flag is given a value of 3, the string constant naming the desired unit should conform to MIL-STD-12D, (DOD12D) or ANSI/IEEE 260, (IEEE260).

2.2.4.2.16 <u>Maximum Number of Line Weight Gradations.</u> This is the number of equal subdivisions of line thickness.

2.2.4.2.17 <u>Size of Maximum Line Width in Units.</u> This is the actual width of the thickest line possible in the (scaled) file.

2.2.4.2.18 <u>Date and Time of File Generation.</u> This is a time stamp of when the file was created.

2.2.4.2.19 <u>Minimum User-Intended Resolution.</u> This parameter indicates the smallest distance in model space units that the system should consider as discernable. Coordinate locations in the file which are less than this distance apart should be considered to be coincident.

2.2.4.2.20 <u>Approximate Maximum Coordinate Value.</u> This is the upper bound on the values of all coordinate data actually occurring in this model. The absolute magnitude of each of the coordinates in this model is less than or equal to this value.

2.2.4.2.21 **Name of Author.** The name of the person responsible for the generation of the data contained in this file.

2.2.4.2.22 **Organization.** The organization or group with whom the author is associated.

2.2.4.2.23 **Version Number.** Each version of IGES is assigned a unique integer value corresponding to that version. This field contains the integer value corresponding to the version of IGES used to create this file. The correspondences are specified in the table below.

VALUE	IGES VERSION
1	1.0
2	ANSI Y14.26M - 1981
3	2.0
4	3.0

If no valid integer value is entered in this field, the default value of 3 (corresponding to IGES Version 2.0) should be assumed.

2.2.4.2.24 **Drafting standard code.** The drafting standard according to which the data in this file was generated.

0	None	-	No standard specified
1	ISO	-	International Organization for Standardization
2	AFNOR	-	French Association for Standardization
3	ANSI	-	American National Standards Institute
4	BSI	-	British Standards Institute
5	CSA	-	Canadian Standards Association
6	DIN	-	German Institute for Standardization
7	JIS	-	Japanese Institute for Standardization

2.2.4.3   Directory Entry Section

The directory entry section has one directory entry for each entity in the
file. The directory entry for each entity is fixed in size and contains twenty
fields of eight characters each spread across two consecutive eighty charac-
ter lines. Data are right justified in each field. With the exception of the
fields numbered 10, 16, 17, 18, and 20, entries in all fields in this section will
be either integer constants or pointer constants. In this section, the word
"number" is sometimes used in place of the phrase "integer constant."

The purposes of the directory entry section are to provide an index for the
file and to contain attribute information for each entity. The order of the
directory entries within the directory entry section is arbitrary with the
exception that a definition entity must precede all of its instances.

Within the directory entry section, a field consisting wholly of blanks is to be
considered to have not been specified and should be given a default value
where possible. Default values are not allowed in fields 1, 2, 10, 11, 14, and
20. The actual values to be assigned as defaults will vary depending on the
entity type. This rule does not apply to compressed ASCII.

Some of the fields in the directory entry can contain either an attribute value
directly, or a pointer to a set of such values. In these fields, a positive value
indicates an integer constant, while for a negative value, the absolute value
should be taken and the result interpreted as a pointer constant.

Since zero is not a valid pointer constant value, a zero in a field requiring a
pointer is not permitted unless a specific interpretation has been allowed for
in this Specification.

Table 2-2 and the following paragraphs describe each directory entry field.
For those fields accommodating either an attribute value or a poil .er, there
are two descriptions given. Figure 2-2 gives an abbreviated listing of the
fields making up the directory entry for each entity.

DIRECTORY ENTRY (DE) SECTION

	8|9 16|17	24|25	32|33	40|41	48|49	56|57	64|65	72|73	80
	ENTITY TYPE	PARA-METER DATA	STRUCTURE	LINE FONT PATTERN	LEVEL	VIEW	TRANSFOR-MATION MATRIX	LABEL DISPLAY	STATUS NUMBER	SEQUENCE NUMBER
LINE 1	#	▲	#,▲	#,▲	#,▲	0,▲	0,▲	▲	#	D, #
	1	2	3	4	5	6	7	8	9	10
	ENTITY TYPE	LINE WEIGHT	COLOR	PARA-METER LINE COUNT	FORM	RESERVED	RESERVED	ENTITY LABEL	ENTITY SUB-SCRIPT	SEQUENCE NUMBER
LINE 2	#	#	#,▲	#	#				#	D.#+1
	11	12	13	14	15	16	17	18	19	20

- INTEGER
▲ - POINTER
#,▲ - INTEGER OR POINTER (POINTER HAS NEGATIVE SIGN)
0 - ZERO

FIG. 2-2 DIRECTORY ENTRY (DE) SECTION

TABLE 2-2 DIRECTORY ENTRY FIELD DESCRIPTION

NO.	FIELD NAME	MEANING AND NOTES
1	Entity Type Number	Identifies the entity type.
2	Parameter Data	Pointer to the first line of the parameter data record for the entity. The letter P is not included.
3	Structure	Pointer to the directory entry of the definition entity that specifies this entity's meaning. The letter D is not included. The integer values 0, 1, and 2 are permissible in this field but should be disregarded. (See 2.2.4.3.3)
4	Line Font Pattern	Line font pattern or pointer to the directory entry of a line font definition entity.
5	Level	Number of the level upon which the entity resides, or a pointer to the directory entry of a property entity (form 1) which contains a list of levels upon which the entity resides.
6	View	Pointer to the directory entry of a view entity, or pointer to a views visible associativity instance, or integer zero (default).
7	Transformation Matrix	Pointer to the directory entry of a transformation matrix (type number 124) used in defining this entity; zero (default) implies the identity transformation matrix and zero translation vector will be used.

8	Label Display Associativity	Pointer to the directory entry of a label display associativity (Entity 402 Form 5).

9	Status Number	Provides four two-digit status values which are entered from left to right in the status number field in the order given below.

1-2 Blank Status

     00   Visible

     01   Blanked

3-4 Subordinate Entity Switch

     00   Independent

     01   Physically Dependent

     02   Logically Dependent

     03   Both (01) and (02)

5-6 Entity Use Flag

     00   Geometry

     01   Annotation

     02   Definition

     03   Other

     04   Logical/Positional

     05   2D Parametric

7-8 Hierarchy

     00   Global top down

     01   Global defer

     02   Use hierarchy property

10	Section Code and Sequence Number	Physical count of this line from the beginning of the directory entry section, preceded by the letter D (odd number).

11	Entity Type Number	(Same as Field 1.)

12	Line Weight Number	System display thickness; given as a gradation value in the range of 0 to the maximum (parameter 16 of the global section).
13	Color Number	Color number or pointer to the directory entry of a color definition entity.
14	Parameter Line Count Number	Number of lines in the parameter data record for this entity.
15	Form Number	Certain entities have different interpretations. These interpretations are uniquely identified by a form number. Possible form numbers are listed within each entity description.
16-17	Reserved for future use	
18	Entity Label	Up to eight alphanumeric characters (right justified).
19	Entity Subscript Number	1 to 8 digit unsigned number associated with the label.
20	Section Code and Sequence Number	Same meaning as field 10 (even number).

2.2.4.3.1   **Entity Type Number.** The integer number indicating the type of entity.

2.2.4.3.2   **Parameter Data.** This is the sequence number of the first parameter data record for this entity. The letter P is not included.

2.2.4.3.3   **Structure.** Non-negative integer values are permitted in this field, but should be disregarded. (In versions prior to Version 3.0, non-negative integers were used in this field to designate version numbers.) For a negative value, the absolute value of this field is interpreted as a pointer to the structure definition entity which specifies the schema for this entity type number.

2.2.4.3.4   **Line Font Pattern.** This indicates a display pattern to be used to display a geometric entity. A positive value indicates that the receiving system's corresponding version of the solid, dashed, phantom and centerline fonts should be used. A negative value indicates that the absolute value should be interpreted as a pointer to a line font definition entity (type 304) which provides the information specifying the display pattern.

> 1 = Solid   3 = Phantom
>
> 2 = Dashed   4 = Centerline

2.2.4.3.5   **Level.** This value specifies a graphic display level or levels to be associated with this entity. A positive value indicates the graphic level this entity exists on. A negative value indicates the absolute value is a pointer to a property entity (type 406, form 1) which contains a list of levels to be associated with the entity. This feature allows an entity to exist on multiple graphic levels.

2.2.4.3.6   **View.** Three options exist. This value is either a pointer to the directory entry of a view entity (type 410), a pointer to an associativity instance (type 402, form 3 or 4, views visible), or the integer zero (default). The first option applies when the entity is visible in a single view. The second option applies when the entity is visible in more than one view. (Form 4 applies when the display characteristics of the entity are view dependent.) The third option applies when the entity is displayed with the same characteristics in all views.

2.2.4.3.7   **Transformation Matrix.** This value is either a pointer to the directory entry of a Transformation Matrix entity (type 124) or the integer zero (default). Zero implies the identity rotation matrix and zero translation vector will be used. The Transformation Matrix entity provides Form Numbers according to the form of the transformation matrix. See Section 3.14.

2.2.4.3.8   **Label Display Associativity.** This is a pointer to the directory entry of a label display associativity (type 402, form 5) which defines how the entity's label and subscript are to be displayed in different views.

2.2.4.3.9   **Status Number.** This value contains four pieces of information which are concatenated into a single integer number. The four two-digit values are concatenated from left to right in the order given below.

2.2.4.3.9.1 **Blank Status.** This value defines whether the entity is meant to be visible on the output device of the receiving system. A value of 00 implies the entity is to be displayed and a value of 01 implies the entity is not to be displayed.

2.2.4.3.9.2 **Subordinate Entity Switch.** This value indicates whether or not the entity is referenced by other entities in the file; and, if so, what type of relationship exists.   An entity can be independent, physically dependent, logically dependent, or physically and logically dependent. The values are defined as follows:

> 00: Independent. The entity is not referenced (i.e., pointed to) by any other entities in the file. It can exist alone in the native database.

> 01: Physically Dependent. This entity (the child) is referenced by another entity (the parent) in the file. The child cannot exist unless the parent exists. The matrix pointed to by the entity (as a child) must be applied to the entity's definition in order to determine its location in the parent's definition space.

Entity A is subordinate to entity B if, and only if, the parameter data entry of entity B contains a pointer to entity A. This implies that entities are NOT subordinate to the view (or views visible associativity) entity that defines the view within which the entity is displayed.

31

The structure formed by a parent entity and its physically subordinate components is indivisible and may therefore be considered to form a single entity.

The following are examples of physically subordinate entities:

o   A leader line entity pointed to by a subordinate entity.
o   A line entity pointed to by a plane entity.
o   A circular arc entity pointed to by a composite curve entity.
o   A composite curve entity pointed to by a subfigure definition (note that the subfigure definition would NOT point to the constituent entities of the composite curve).

Consider the following example:

o   Entity A is physically subordinate to entity B.
o   Entity A points to a transformation matrix M1.
o   Transformation matrix M1 points to a transformation matrix M2.
o   Entity B is subordinate to a subfigure definition entity C.
o   Entity B points to a transformation matrix M3.
o   Entity C is instanced in a subfigure instance D.
o   The parameter data of entity D specifies its scale factor as Sd and position as (Sd,Yd,Zd).
o   Entity D points to a transformation matrix M4.
o   Entity D points to a view entity E.
o   The view scale factor defined in the parameter data of entity E is Se.
o   Entity E occurs within a drawing F at drawing coordinates (Fx,Fy).
o   Entity E points to a transformation matrix M5.

In order to obtain the drawing space coordinates of entity A, the following operations are performed:

1.   The coordinates of entity A are transformed by M1.
2.   The coordinates resulting from the preceeding step are transformed by M2.
3.   The coordinates resulting from the preceeding step are transformed by M3.

32

4. The coordinates resulting from the preceeding step are scaled by Sd.

5. The coordinates resulting from the preceeding step are transformed by M4.

6. The coordinates resulting from the preceeding step are translated by the vector (Xd,Yd,Zd). The coordinates resulting from this step are the model space coordinates of entity A.

7. The coordinates resulting from the preceeding step are transformed by M5.

8. The coordinates resulting from the preceeding step are scaled by the scale factor Se.

9. The coordinates resulting from the preceeding step are translated by the vector (Fx,Fy).

02: Logically Dependent. This entity (the child) can exist alone in the native database, but is referenced by some sort of logical grouping entity, or entities (the parents), such as associativities. The matrix pointed to by the parent entity has no effect on the location of the child.

An example of a logically subordinate entity is that of a line entity pointed to by a group associativity entity.

03: Both Physically and Logically Dependent. This entity is physically dependent upon another entity in the file and is subject to the physical dependency rules described above. This entity is also referenced by one or more logical grouping entities, and is also subject to the logical dependency rules described above. Additionally, an entity cannot be physically and logically dependent upon the same parent entity.

The case of an entity being both logically and physically subordinate refers to the fact that the transformation matrix pointed to by a parent entity is not applied to its logically subordinate children.

An example of a logically and physically subordinate entity is that of a line entity occuring in a subfigure definition and pointed to by a group associativity entity.

33

2.2.4.3.9.3 **Entity Use Flag.** This value indicates the intent of the entity. It classifies the entity as intending to serve in the following manners:

00   Geometry. This is the default value. The entity is used to define the geometry of the structure of the product.

01   Annotation. The entity is used to add annotation or description to the file. This includes geometric entities used to form annotation or description.

02   Definition. The entity is used in definition structures of the file. It is not intended to be valid outside of the other entities which reference the definition structure. An example is the entities in a subfigure definition which are intended to be valid in the subfigure instances that reference the subfigure definition. This class includes all entities in the 300 entity type number range.

03   Other. The entity is being used for other purposes such as defining structural features in the file. This category corresponds roughly to the 400 range, but there are exceptions. For example, a subfigure instance (408) could define geometry, thus having an entity use flag = 0 or it could define a drawing format, thus having an entity use flag = 01. An associativity instance would ordinarily have the value 03. Exceptions include associativities concerned with display where it would have the value 01. The view and drawing entities have value 01 (annotation). Transformation depends on its use: If used only for annotation (e.g., defining a view) the value is 01; if used for defining geometry or for defining geometry and annotation, value is 00.

04   Logical/Positional.   The entity is used as a logical and/or positional reference by other entities.   This usage does not prevent the entity from referencing other entities or having its own attributes. Some entities which may be instanced in this way are nodes, connect points, and points when their primary use is as a reference.

05   2D Parametric. The entity takes its values in two dimensional XY parameter space considered as a subset of three dimensional XYZ space by ignoring the Z coordinate.   The transformation matrix from definition space to parameter space must be 2-dimensional (i.e., in entity 124, section 3.14.8, $T_3=R_{13}=R_{31}=R_{32}=R_{23}=0.0$ and $R_{33}=1.0$).   In addition, the coordinates do not have units of length (i.e., the model space scale and units conversion do not apply).   This is intended for use in defining curves on surfaces.

2.2.4.3.9.4 **Hierarchy.**   This value indicates the relationship between entities in a hierarchical structure and is used to determine which entity's directory entry attributes should be applied.   It applies to line font, view, entity level, blank status, line weight, and color number. Three values are provided:

00:   All the above directory entry attributes will apply to entities physically subordinate to this entity.

01:   None of the above directory entry attributes of this entity will apply to physically subordinate entities.   The physically subordinate entities will use their own directory entry attributes.

02:   Individual setting of each of above direct entry attributes are allowed. A hierarchy property entity form 10 (see 4.3.7.3.10) specifies whether 00 or 01 is applied for each directory entry attribute to physically subordinate entities.

Example: If an entity A has 00 in its DE status digits 7 and 8, all entities subordinate to A will have the attributes assigned to A. Consequently, the attributes assigned to all entities subordinate to A are ignored.

If an entity A has 01 in its DE status digits 7 and 8, the entities immediately subordinate to A will retain their own status. Consequently, the attributes assigned to A are ignored.

2.2.4.3.10 **Sequence Number.** A number which specifies the position of the DE line in the directory entry section. The sequence number of the first DE line for any entity is always odd and the sequence number of the second line is always even.

2.2.4.3.11 **Entity Type Number.** This is the same as Field 1.

2.2.4.3.12 **Line Weight Number.** This value denotes the thickness (or width) with which an entity should be displayed. A specific series of possible thicknesses are specified by global parameters 16 and 17. The largest thickness possible is that specified in global parameter 17 and is denoted by setting this value equal to the value in global parameter 16. The smallest thickness possible is equal to the result of dividing global parameter 17 by global parameter 16 and is denoted by setting this value equal to 1. Thicknesses between the smallest and largest thickness are available in increments equal to the smallest possible thickness and are denoted by setting this value equal to the integer number of (adjacent) increments required.

Thus, display thickness is:
Line Weight Number * (Global parameter 17/Global parameter 16)

A value of 0 indicates that the default line weight display of the receiving system is to be used.

2.2.4.3.13 **Color Number.** Field 13 is a color number and is used for specifying color when the precise shade is unimportant or is a pointer to a more precise color definition. It is up to the receiving system to generate colors which approximately fit the following description.

Color No.	Color
0	No color assigned
1	Black
2	Red
3	Green
4	Blue
5	Yellow
6	Magenta
7	Cyan
8	White

If the color number is negative, its absolute value is a pointer to the directory entry of a color definition entity (type 314).

2.2.4.3.14 **Parameter Line Count Number.** This is the number of lines in the parameter data section which contain the parameter data record for this entity.

2.2.4.3.15 **Form Number.** This value indicates an individual interpretation of the entity to be used when processing the parameter data for this entity. Some entity types allow multiple interpretations of their parameter data. This parameter along with the entity type number uniquely identify the interpretation of the parameter data.

2.2.4.3.16 **Reserved Field.** This field is reserved for future use and should be left blank.

2.2.4.3.17 **Reserved Field.** Same as Field 16.

2.2.4.3.18 **Entity Label.** This is the application-specified alphanumeric identifier or name for this entity. It is used in conjunction with the entity subscript number (Field 19) to provide the application-specified alphanumeric identifier for the entity.

2.2.4.3.19 **Entity Subscript Number.** This is a numeric qualifier for the entity label (Field 18).

2.2.4.3.20 **Sequence Number.** Same as 2.2.4.3.10

2.2.4.3.21 Table 2-3 summarizes DE Field Requirements for mechanical product entities.

Each DE field is placed in one of three following categories.

Ignore

The field has no meaning for this entity. The preprocessor should set the field to 0 or blank. The postprocessor should ignore the field altogether.

Valid: May be defaulted to

0 or blank is a valid setting for this field. If the field is blank, the postprocessor will interpret it a 0 or blank as specified. Other valid settings must be set explicitly by the preprocessor.

Valid: Non-zero required

A non-zero value must be set by the preprocessor. Zero or blank are not valid settings for this field, therefore no default is allowed.

TABLE 2-3 ENTITY DE FIELD REQUIREMENTS

Entity Type(s): 100, 104, 106 (Forms 1-3, 11-13, 63) 110, 112, 116, 126, 114, 108, 118, 120, 122, 130, 140, 142, 144

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font			X	Ignored for 116, 106 (Forms 1, 2, 3)
5. Level		0		
6. View ptr.		0		0 means display in all views
7. Matrix ptr.		0		
8. Label ptr.		0		
9a. Blank S		00		
9b. Subord Sw		00		
9c. Entity use		00		00, 01, 05 are the only valid values. 01 implies the entity is used as annotation rather than defining geometry.
9d. Hierarchy	X			
10. Seq. num.			X .	
11. TYPE			X	
12. Line wt.		0		A value of 0 means to use the receiving system's default line weight.
13. Color		0		A value of 0 means any color may be used.
14. PD Count			X	
15. Form		0		Non-zero required for 106
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

39

DATA FORM - ASCII - Directory Entry
                  Section

Entity Type(s): 102

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font			X	Note 1
5. Level		0		Note 1
6. View ptr.		0		Note 1
7. Matrix ptr.		0		
8. Label ptr.		0		
9a. Blank S		00		Note 1
9b. Subord Sw		00		
9c. Entity use		00		
9d. Hierarchy		00		
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.		0		Note 1
13. Color		0		Note 1
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Notes:

1. *This field is ignored if the hierarchy value for field is 01.*

Entity Type(s): 124 (Forms 0,1)

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.		0		
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw	X			
9c. Entity use	X			
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 124 (Forms 10,11,12)

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.		0		
8. Label ptr.	X			
9a. Blank S		00		If not blanked then display coordinate system in color indicated in DE field 13
9b. Subord Sw		00		01 is used to stack coordinate systems, 02 is used to group coordinate systems
9c. Entity use	X			
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X.	
12. Line wt.	X			
13. Color		0		
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 200 Series except 212 and 214

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font			X	
5. Level		0		
6. View ptr.		0		
7. Matrix ptr.		0		
8. Label ptr.		0		
9a. Blank S		00		
9b. Subord Sw		00		
9c. Entity use			X	Must be 01
9d. Hierarchy		00		
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.		0		
13. Color		0		
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 212. 214 and 106 (Forms 20, 21, 31-40)

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font			X	Ignored for 106 (Forms 20, 21)
5. Level		0		
6. View ptr.		0		
7. Matrix ptr.		0		
8. Label ptr.		0		
9a. Blank S		00		
9b. Subord Sw		00		
9c. Entity use			X	Must be 01
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.		0		
13. Color		0		
14. PD Count			X	
15. Form		0		Non-zero required for 106
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 302, 306, 310

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.	X			
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 304

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font			X	
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.		0		
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form			X	
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 308, 320 Subfigure definition

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font			X	Note 1
5. Level	X			Note 1
6. View ptr.	X			
7. Matrix ptr.		0		
8. Label ptr.		0		
9a. Blank S	X			
9b. Subord Sw		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy		00		
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.		0		Note 1
13. Color		0		Note 1
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Notes:

1. This field is ignored if the hierarchy value for the field is 01.

Entity Type(s): 312

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.	X			
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw			X	Must be 02
9c. Entity use			X	Must be 02
9d. Hierarchy			X	Must be 01
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 314

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.	X			
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color		0		
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 402

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure			X	Ignored for form numbers to 5000
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.	X			
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		00 required for Forms 3 & 4
9c. Entity use	X			
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form			X	
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 404

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.	X			
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		Must be independent
9c. Entity use			X	Must be 03
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Entity Type(s): 406

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level		0		
6. View ptr.	X			
7. Matrix ptr.	X			
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		See Note 1
9c. Entity use	X			
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form			X	
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Notes:

1. *The setting for this field should be 02. The specification currently states a setting of 00 implies the property should be applied to all entities occurring at the same level.*

Entity Type(s): 408, 412, 414, 420

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level		0		See Note 1
6. View ptr.		0		
7. Matrix ptr.		0		
8. Label ptr.		0		
9a. Blank S		00		
9b. Subord Sw		00		
9c. Entity use		00		
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

Notes:

1. *The specification currently states entities within a subfigure are displayed at the same level as the instance.*

53

Entity Type(s): 410

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-Zero Required	COMMENT
1. TYPE			X	
2. PD ptr.			X	
3. Structure	X			
4. Line Font	X			
5. Level	X			
6. View ptr.	X			
7. Matrix ptr.		0		
8. Label ptr.	X			
9a. Blank S	X			
9b. Subord Sw		00		Dependent if 404 points to it. Independent if not.
9c. Entity use			X	Must be 03
9d. Hierarchy	X			
10. Seq. num.			X	
11. TYPE			X	
12. Line wt.	X			
13. Color	X			
14. PD Count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Seq. num.			X	

2.2.4.4   **Parameter Data Section**

The parameter data section of the file contains the parameter data
associated with each entity.   The following information is true for all
parameter data.

2.2.4.4.1   Parameter data are placed in free format (see 2.2.3) with the first field
always containing the entity type number.   Therefore, the entity type
number and a parameter delimiter (default is comma) precede parameter
one of each entity.   The free format part of a parameter line ends in
column 64. Column 65 shall contain a blank.   Columns 66 through 72 on all
parameter lines contain the sequence number of the first line in the
directory entry of the entity for which parameter data is being presented.
Column 73 of all lines in the parameter section shall contain the letter P
and columns 74 through 80 shall contain the sequence number (See 2.2.1).

2.2.4.4.2   Two groups of parameters are defined at the end of the specified
parameters for each entity.   The first group of parameters may contain
pointers to Associativity Instances, General Notes, and/or Text Template
entities.   The second group of parameters may contain pointers to one or
more properties.   The pointers to associativity instances are "back
pointers" in that they point back from a member of an associativity
instance to the associativity instance.   These pointers may be required by
the associativity definition.

If the given entity references associated text, a pointer to the General
Note may be included in the first group of pointers.   If so, the indicated
General Note specifies the string constant and the indicated display
parameters.

If instead, the given entity contains a string constant to be displayed, a
pointer to a Text Template entity may be included in the first group of
pointers.   The Text Template entities provide display parameters for text
supplied by the entity which points to the template.

Either group of parameters, or both, may be empty or may be defaulted to no pointers.

When present, the pointers comprising these parameters are added after all the other specified (or defaulted) parameters, but ahead of the record delimiter as follows:

Let NV = last parameter number

Index	Name	Type	Description
.	.	.	.
.	.	.	.
NV+1	NA	Integer	Number of pointers to Associativity Instances/Text Entities
NV+2	DE	Pointer	
.	.	.	
.	.	.	Associativity Instance/Text Entity List
NV+NA+1	.	.	
NV+NA+2	NP	Integer	Number of Pointers to Properties
NV+NA+3	DE	Pointer	
.	.	.	
.	.	.	Property List
NV+NA+NP+2	.	.	

2.2.4.4.3 Any desired comment may be added after the record delimiter. Note that additional lines may be used for this purpose by keeping the directory entry pointer in columns 65-72 constant and include them in the count of parameter lines for the entity (DE field 14). Figure 2-3 shows a parameter data section.

PARAMETER DATA (PD) SECTION

	64	65	66 72	73 80
ENTITY TYPE NUMBER FOLLOWED BY PARAMETER DELIMITER PARAMETERS SEPARATED BY PARAMETER DELIMITERS			DE POINTER	P0000001
PARAMETERS SEPARATED BY PARAMETER DELIMITERS RECORD DELIMITER			DE POINTER	P0000002
• • •			• • •	• • •
ENTITY TYPE NUMBER FOLLOWED BY PARAMETER DELIMITER ETC.			DE POINTER	P0000000N

DE POINTER: THE SEQUENCE NUMBER OF THE FIRST DIRECTORY ENTRY LINE FOR THIS ENTITY

FIG. 2-3 PARAMETER DATA SECTION

57

2.2.4.5   **Terminate Section**

There is only one line in the terminate section of the file. It is divided into
ten fields of eight columns each. The terminate section must be the last
line of the file. It has a "T" in column 73 and columns 74 through 80 contain
the sequence number with a value of one (1).

The fields on the terminate line contain the character representing the
section type and the last sequence number used in each of the previous
sections. The fields are defined below and shown in Figure 2-4.

FIELD	COLUMNS	SECTION
1	1-8	Start Section
2	9-16	Global Section
3	17-24	Directory Entry Section
4	25-32	Parameter Section
5-9	33-72	(not used)
10	73-80	Terminate Section

TERMINATE SECTION

| 1 | 8 | 9 | 16 | 17 | 24 | 25 | 32 | 33 | 40 | 41 | 48 | 49 | 56 | 57 | 64 | 65 | 72 | 73 | 80 |

S0000020 G0000003 D0000500 P0000026I ◄———————— NOT USED ————————► T0000001
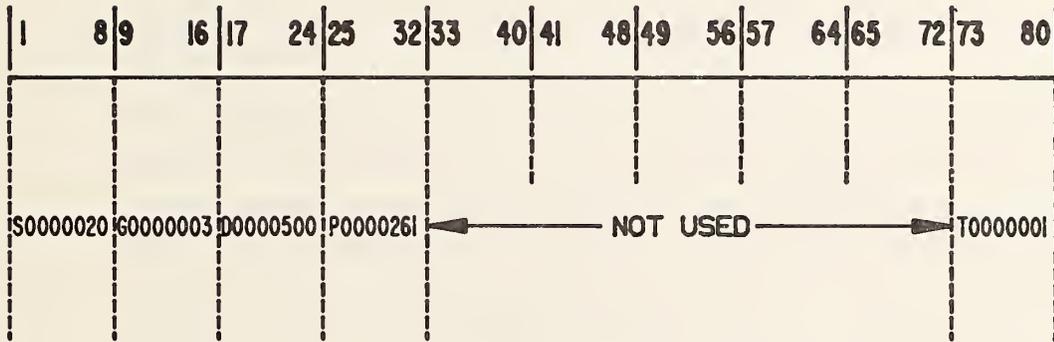
FIG. 2-4 TERMINATE SECTION

THIS PAGE LEFT BLANK

2.3 Compressed ASCII Form

The format described here is intended to serve as an alternative to the ASCII Form when the size of a file is a problem. The Compressed ASCII Form is intended to be simply converted to and from the regular ASCII Form. As an example of such a conversion, the National Bureau of Standards has software available to convert from ASCII to Compressed ASCII and from Compressed ASCII to regular ASCII (See Appendix G).

2.3.1 File Structure

A single line shall precede the START section with the character "C" in character position 73 to identify the file as compressed ASCII. The START, GLOBAL and TERMINATE sections remain the same as the ASCII Form, while the DE and PD sections are combined into a single DATA section. The PD portion of the DATA section is written in free format in a manner similar to the present PD section. Each line is of variable length and is to terminate before character position 65, thus assuring that character position 65, if it existed (i.e., if the line is read into a fixed 80 character buffer) would always contain the blank character. The lines corresponding to a single entity begin with one or more lines giving the data presently contained in the DE record.

The "neo-DE" records begin with the letter "D", followed, without intervening blanks, by an integer equal to the present sequence number of the first DE record. See Figure 2-5. The D⟨sequence number⟩ group of characters is followed by one or more DE field specifiers. The DE field specifier consists of the symbol "@" (commercial at) followed by an integer giving the DE field being specified. The @ field number group is followed by a character sequence consisting of the symbol "_" (underline), followed by the value of the field.

All DE field numbers remain the same as with the present DE record. Fields 2, 10, 11, and 20 are not specified, as they are either redundant or meaningless in the Compressed ASCII form. When several DE

fields are being specified, additional lines can be used. The sequence of field specifiers must be broken only between complete specifiers, thus assuring that new records will begin with the symbol "@".

Field values will be specified only when they change. Thus, a field retains its value from entity to entity unless it is changed, and only the first entity in a file is assured of containing a complete set of fields and values. No separation mark is used between @⟨field number ⟩⟨value groups⟩ but the DE field specifier record(s) end with a record delimiter (default ";").

```
|I                                                               80|
|                                                                  |
| START SECTION                                                  | |
| GLOBAL SECTION                                                 | |
| DI23@4_2...ADDITIONAL FIELD DESCRIPTORS...          |            |
| ...USING AS MANY LINES OF UP TO 72 CHAR...        |              |
| ...EACH  ....     ı |                                            |
| ...PD RECORD BELONGING TO   DE @I23 ...          |               |
| ...ON LINES UP TO 64 CHAR. ... ı |                              |
| DI25 ....                                      |                 |
| ...PD RECORD ...       ı |                                       |
| TERMINATE SECTION                              |                 |
|                                                                  |
```

NOTE: DEFAULT RECORD DELIMITER ASSUMED
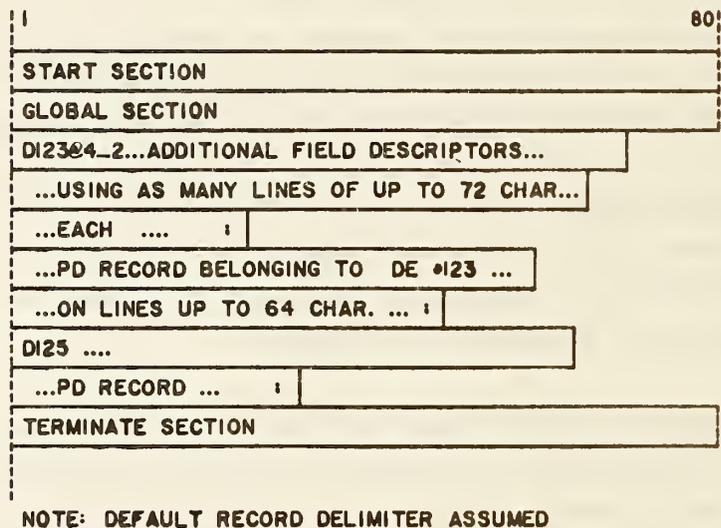
FIGURE 2-5 COMPRESSED ASCII FORM

The PD records remain the same as in the ASCII Form with the exception that they terminate at the end of parameter data (i.e., before character position 65).

2.4   **Binary Form**

The format defined in Section 2.2, referred to as the ASCII Form, has 80 character fixed length record lines. This section describes a bit stream binary representation of data which may be used as an alternative format to the ASCII Form.

This data is transportable by user selected communication protocols with the data treated as "transparent" or bit stream data. All entity parameterizations and data organization are otherwise identical to the ASCII Form.

2.4.1   **Constants.** The following constants need to be represented in the Binary Form:

- o   integer numbers
- o   real numbers
- o   string constants
- o   pointers
- o   language constants

A control byte will precede each value or set of values of the same type, unless otherwise specified. The control byte will specify the format of the following value or set of values, the quantity of subsequent values with that format, and whether values other than the initial value following the control byte are present. If the control byte indicates that values subsequent to the initial value of the set are absent, all subsequent values, up to the quantity indicated are assumed to have the same value as the initial value following the control byte.

The repetition portion of the control byte is unsigned and biased by 1 so that the true quantity of numbers to which the repetition field applies is one more than the unsigned value of the field.

The format of the control byte is as shown in Figure 2-6.

2.4.1.1   Integer Numbers.

The structure of an integer number shall be a sign bit followed by a two's complement integer of length i-1 as shown in Figure 2-7.

Two lengths, i, of integer data can be selected by the system which generates the file.

The length of single precision data is i_s and the length of double precision data is i_d , defined in section 2.4.2.1.

2.4.1.2   Real Numbers.

The structure of a real number shall be a sign bit followed by a biased exponent value of NX bits which is a power of 2 and a binary fraction of NF bits. (NX and NF are defined in section 2.4.2.1) The value of the number is the sign applied to the fractional part multiplied by two raised to the power specified by the exponent part. The sign field consists of one bit. A sign of 0 indicates a positive number and a sign of 1 indicates a negative number. The exponent field consists of NX bits and is interpreted as an unsigned integer, BX, often referred to as the biased exponent. The value of the exponent is its unbiased value X which is obtained by deducting the bias $B=2**(NX-1)$.

CONTROL BYTE FORMAT:

P/A	REPETITION	FORMAT

1 BIT | 4 BITS | 3 BITS
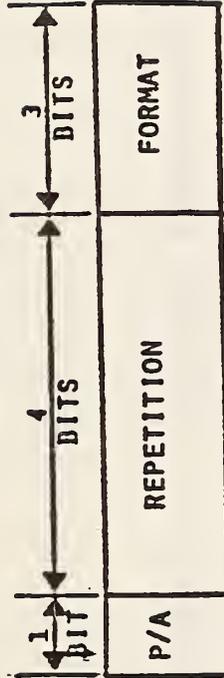
P/A: = 0 IF ONLY FIRST OF A SET OF REPEATED VALUES IS PHYSICALLY PRESENT

= 1 IF ALL EXPECTED VALUES ARE PHYSICALLY PRESENT

REPETITION: (NUMBER OF FOLLOWING VALUES - 1) TO WHICH THIS CONTROL BYTE APPLIES

FORMAT: = 0 IF DEFAULT VALUE IS TO BE USED

= 1 IF SINGLE LENGTH INTEGER

= 2 IF DOUBLE LENGTH INTEGER

= 3 IF SINGLE PRECISION FLOATING POINT

= 4 IF DOUBLE PRECISION FLOATING POINT

= 5 IF POINTER

= 6 IF TEXT STRING

FIGURE 2-6 FORMAT OF CONTROL BYTE

65

THE PAD OF ZEROES IS INCLUDED ONLY IF THE LENGTH, I IS NOT A MULTIPLE OF 8 BITS

CONTROL BYTE	SIGN	VALUE	PAD
8 BITS	1 BIT	I-1 BITS	8N-I
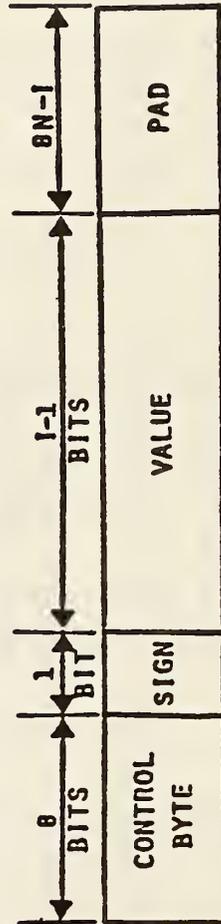
FIGURE 2-7 INTEGER PRIMITIVE FORMAT

The fraction field consists of NF bits interpreted as the low order bits of a normalized (NF+1)-bit fraction part, F. The fraction lies between 0.5 (inclusive) and 1.0 (exclusive). Since the most significant bit of a normalized fraction is always 1 it is not explicitly represented.

Numbers with a non-zero biased exponent have a value given by:

$$(-1)^{SIGN} * 2^{(BX-B)} * F$$

The structure of a real number is shown in Figure 2-8.

Two lengths of real data can be selected by specifying the length of each exponent (NX) and the length of each fractional portion (NF).

2.4.1.3   String Constants.

Following the control byte will be a character count with a length of i_s, defined in section 2.4.2.1. Where the character count exceeds the capability of an i_s length integer, the string is broken up into substrings. In order to indicate that another substring follows the current string, a negative character count is used. The number of characters in the substring is the absolute value of the character count. A positive character count indicates the last substring.

THE PAD OF ZEROES FROM 1 TO 7 BITS IS INCLUDED ONLY IF THE LENGTH OF THE FLOATING
POINT NUMBER (1+NX + NF) IS NOT A MULTIPLE OF EIGHT BITS
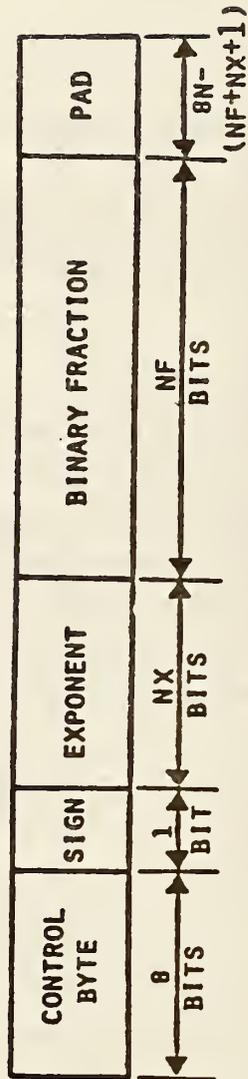
FIGURE 2-8 REAL NUMBER PRIMITIVE FORMAT

The structure of the string constant is shown in Figure 2-9.

2.4.1.4   Pointers.

The structure of a pointer shall be a 32 bit integer. The pointer shall contain the relative byte position of the entity byte count of the DE or PD entity to which it is pointing. A pointer to the first DE entity will have a value of 1. A pointer to the second DE entity will have a value equal to the number of bytes of the first DE entity plus one. A pointer to the first PD entity will have a value of 1. Pointers with values of zero or negative are not actual pointers but may have a default meaning depending upon the context. For example, a defining matrix value of zero would imply that the identity rotation matrix and zero translation vector are used. This case might also be handled by using the control byte, instead, to indicate a default value.

2.4.1.5   Language Primitives.

Language primitives are the string constants of the MACRO definition entity which, in the ASCII Form, are not preceded by nH and are terminated with a record delimiter. In Binary Form the format of language primitives will be identical to string constants. Each language primitive (MACRO statement) will be an individual string constant.

2.4.2   File Structure.

The general file structure is as shown in Figure 2-10 and is comprised of the following six sections:

- o   Binary information section
- o   Start section
- o   Global section
- o   Directory entry section
- o   Parameter Data section
- o   Terminate section

Following each section is zero, one or many 8-bit null padding characters. These characters do not belong to the section and have no meaning. They are provided to assist the creator of a file with

FOR $N_1 > 0$

CONTROL BYTE	NUMBER OF CHARACTERS (N_1)	ASCII CHAR 1	ASCII CHAR 2		ASCII CHAR N_1
8 BITS	I_S BITS	8 BITS	8 BITS		8 BITS

FOR $N_J < 0, \ldots, N_K < 0, N_R > 0$

CONTROL BYTE	NUMBER OF CHARACTERS (N_M)	ASCII CHAR 1	ASCII CHAR 2		ASCII CHAR (N_M)	NUMBER OF CHARACTERS N_R	ASCII CHAR	ASCII CHAR		ASCII CHAR

$\overset{8 \text{ BITS}}{\longleftrightarrow}$ $\overset{\sum\limits_{M=J}^{K}(8N_M + I_S)}{\longleftrightarrow}$ $\overset{I_S}{\longleftrightarrow}$ $\overset{8N_R}{\longleftrightarrow}$

REPEAT FOR $N_M < 0$ ($J \leq M \leq K$)
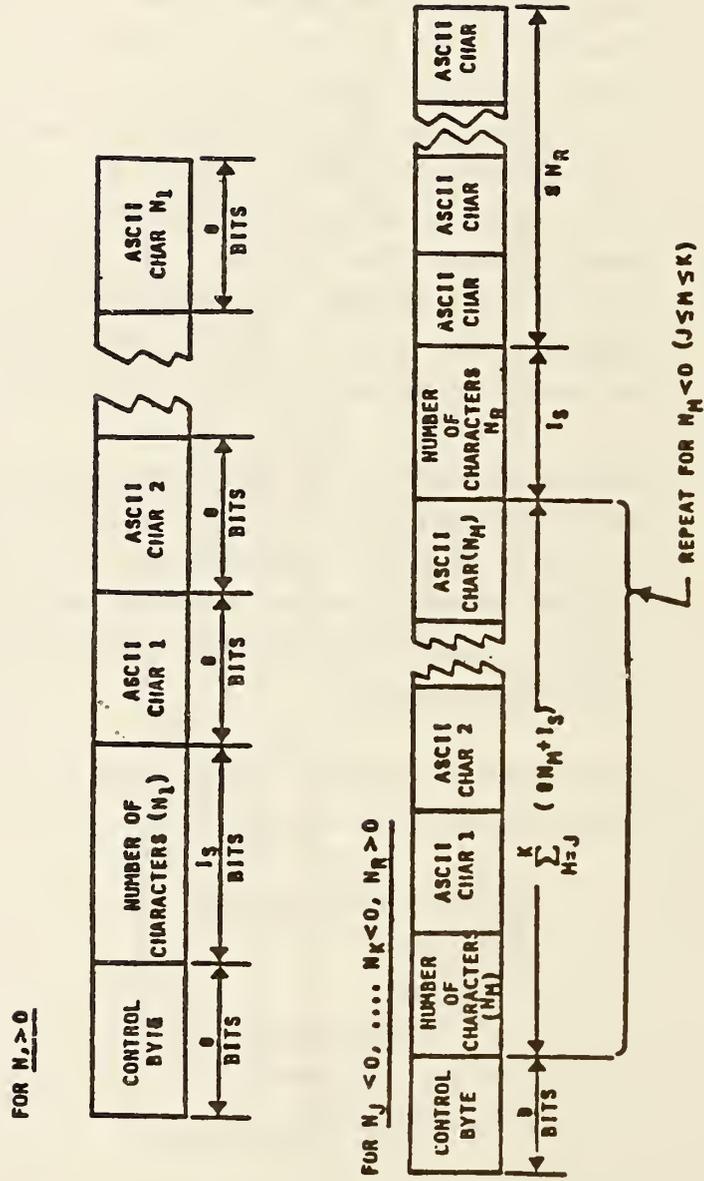
FIGURE 2-9 STRING CONSTANT PRIMITIVE FORMAT

70

```
┌─────────────────────────────────────┐
│     BINARY INFORMATION SECTION      │
├─────────────────────────────────────┤
│            START SECTION            │
├─────────────────────────────────────┤
│           GLOBAL SECTION            │
├─────────────────────────────────────┤
│                                     │
│       DIRECTORY  ENTRY   SECTION    │
│                                     │
├─────────────────────────────────────┤
│                                     │
│                                     │
│         PARAMETER SECTION           │
│                                     │
├─────────────────────────────────────┤
│          TERMINATE SECTION          │
└─────────────────────────────────────┘
```
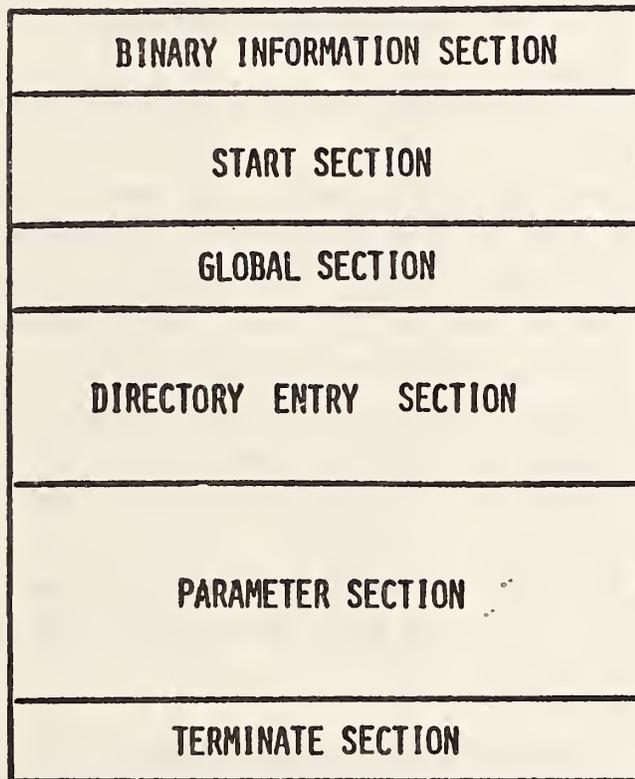
FIGURE 2-10 BINARY GENERAL FILE STRUCTURE

physical system limitations such as word or sector boundaries.

Following the terminate section of the file shall be zero, one, or many null padding characters followed by an 8-bit end of information designator, the ASCII letter E. Any information following the letter E shall be ignored.

2.4.2.1   **Binary Information Section.**   The format of the binary information section is as shown in Figure 2-11. It is comprised of the following data items, all of which are integers unless otherwise specified.

o   Binary information section identifier consisting of the ASCII letter B.

o   Binary information section byte count.   This byte count excludes the 5 bytes required for the section identifier and section byte count.   This byte count also excludes any null padding characters. The value of this byte count will be 75.

o   Length i_s of single length integer primitives.

o   Length i_d of double length integer primitives.

o   Length NX_s of exponent of single precision real primitives.

o   Length NF_s of binary fraction of single precision real primitives.

o   Length NX_d of exponent of double precision real primitives.

o   Length NF_d of binary fraction of double precision real primitives.

o   ASCII letter B.

o   Binary information section displacement. This is the byte count of the total length of the binary information section including all null padding characters.   This length is the actual length from the initial B of the binary information section up to but not including the S of the start section.

o   ASCII letter S.

o   Start section displacement.   This is the byte count of the total length of the start section including all control bytes and null padding characters.   This length is the actual length from the initial S of the start section up to but not including the G of the

FIGURE 2-11 FORMAT OF BINARY INFORMATION SECTION

global section.

o   ASCII letter G.

o   Global section displacement. This is the byte count of the total length of the global section including all control bytes and null padding characters. This length is the actual length from the initial G of the global section up to but not including the D of the directory entry section.

o   ASCII letter D.

o   Directory entry section displacement. This is the byte count of the total length of the descriptive entity section including all control bytes and null padding characters. The length is the actual length from the initial D of the directory entry section up to but not including the P of the parameter data section.

o   ASCII letter P.

o   Parameter data section displacement. This is the byte count of the total length of the parameter data section including all control bytes and null padding characters. This length is the actual length from the initial P of the parameter data section up to but not including the T of the terminate section.

o   ASCII letter T.

o   Terminate section displacement. This is the byte count of the total length of the terminate section including all null padding characters. This length is the actual length from the initial T of the terminate section up to but not including the letter E of the end of information designator.

o   31 unassigned bytes.

o   ASCII letter B.

o   6 ASCII blanks or zeroes.

o   ASCII character 1.

No control bytes are applied to this section. Thus the characters in the equivalent of columns 73 through 80 of the binary information section are similar in format to the section identification of the ASCII Form and can be used to determine if a file is ASCII or binary. If the file contains an S in column 73 of its first 80 bytes, it is ASCII (or C if compressed ASCII). If it contains a B, it is binary.

2.4.2.2   **Start Section.** The format of the start section is as shown in Figure 2-12. It is comprised of the following data items:

   o   A start section identifier consisting of the ASCII letter S

   o   Byte count for the start section. The byte count excludes the 5 bytes required for the start section identifier and section byte count. This byte count also excludes any null padding characters.

   o   One or more language or text primitives which are logically equivalent to columns 1 through 72 of the ASCII Form. There is no required physical correspondence between the ASCII Form and language/text primitives. One language/text primitive may contain the equivalent of several complete or partial ASCII records. Carriage return characters may be embedded in the language/text primitives. Control bytes only apply to the language and text primitives. No control bytes precede the section identifier and byte count.

2.4.2.3   **Global Section.** The format of the global section is as shown in Figure 2-13. The global section is comprised of the following data items:

   o   Global section identifier consisting of the ASCII letter G

   o   Global section byte count. This byte count excludes the 5 bytes required for the global section identifier and the section byte count. This byte count also excludes any null padding characters.

   o   24 global parameters.

Control bytes apply only to the 24 global parameters.

The global parameters have the same sequence and meaning as the ASCII Form global parameters with the exception that global parameters 1 (parameter delimiter character), 2 (record delimiter), 7 (number of bits for integer representation), 8 (single precision magnitude), 9 (single precision significance), 10 (double precision magnitude), and 11 (double precision significance) shall be ignored in Binary Form. The binary information section shall supersede these global parameters.

THE FORMAT OF THE LANGUAGE/TEXT PRIMITIVES IS AS SHOWN IN FIGURE 2-3

• THESE FIELDS DO NOT HAVE CONTROL BYTES
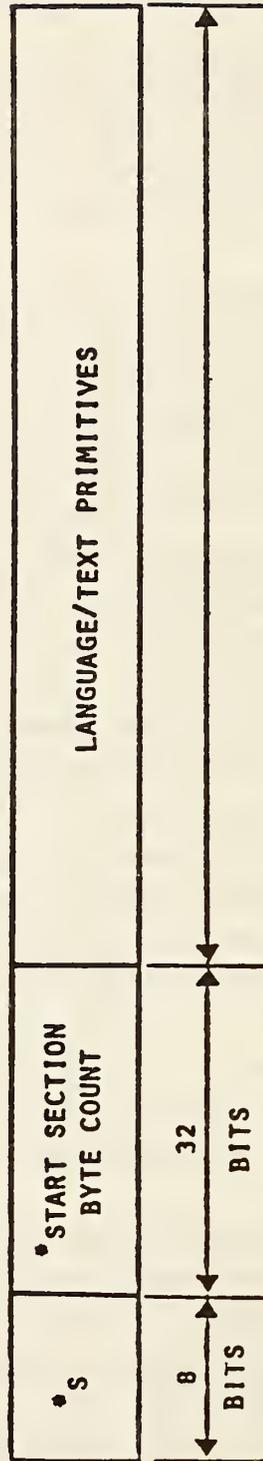
FIGURE 2-12 FORMAT OF START SECTION

FIGURE 2-13 GLOBAL SECTION FORMAT

2.4.2.4 **Directory Entry Section.** The format of the directory entry section is as shown in Figure 2-14. The directory entry section is comprised of the following data items:

- o Directory entry section identifier consisting of the ASCII letter D

- o Directory entry section byte count. This byte count excludes the 5 bytes required for the section identifier and section byte count. This byte count also excludes any null padding characters.

- o For each directory entry, the following 17 data fields are present:

  - entity byte count, which is the length in bytes, including control bytes, of the subsequent 16 data fields.
  - entity type
  - parameter data pointer
  - structure
  - line font number or pointer
  - level number or pointer
  - view pointer
  - transformation matrix pointer
  - label display associativity pointer
  - status number
  - line weight
  - color number or pointer
  - form number
  - reserved field 1
  - reserved field 2
  - entity label
  - entity subscript
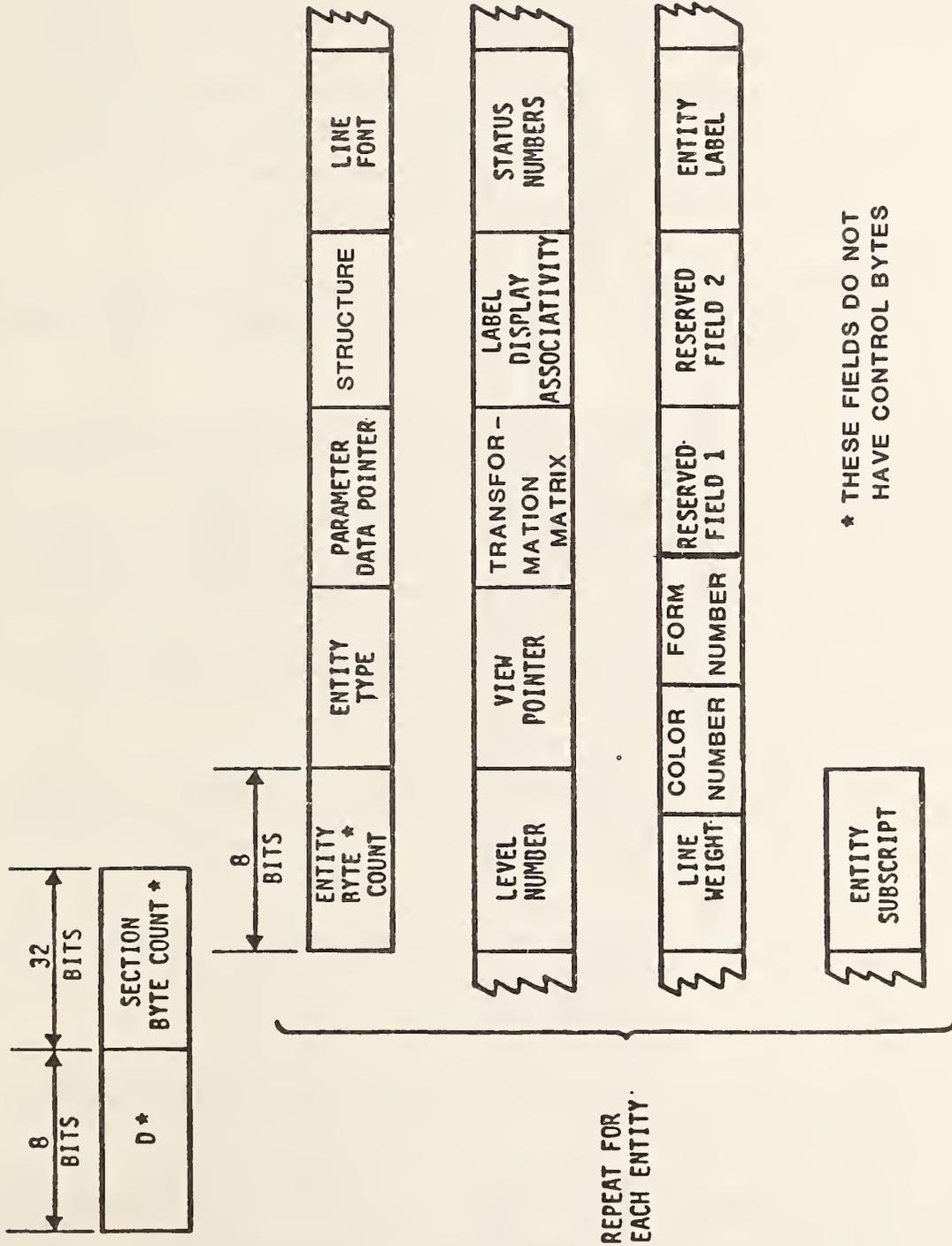
FIGURE 2-14 FORMAT OF DE SUBRECORD

Control bytes apply only to the last 16 data fields.

The directory entry data fields, except for the entity byte count, are identical to and have the same sequence as fields in the ASCII Form. Within a single file, the length of DE record for each entity (in bytes) shall be consistent. If in the future additional fields are required, it is preferable to increase the number of fields for each directory entry and add any new fields subsequent to existing fields.

2.4.2.5 **Parameter Section.** The format of the parameter data section is as shown in Figure 2-15. The parameter data section is comprised of the following data items:

- o Parameter data section identifier consisting of the ASCII letter P

- o Parameter data section byte count. This byte count excludes the 5 bytes required for the section identifier and section byte count. This byte count also excludes any null padding characters.

- o For each parameter data entity, the following data fields are required:

  - entity byte count, which is composed of the lengths, including control bytes, of all subsequent data fields for this entity.
  - entity type
  - directory entry pointer (relative to directory entry section)
  - parameter data

Control bytes apply only to the entity type, directory entry pointer and parameter data fields.

The parameter data entity fields, except for the entity byte count, are identical to and have the same sequence as the the ASCII Form.

80
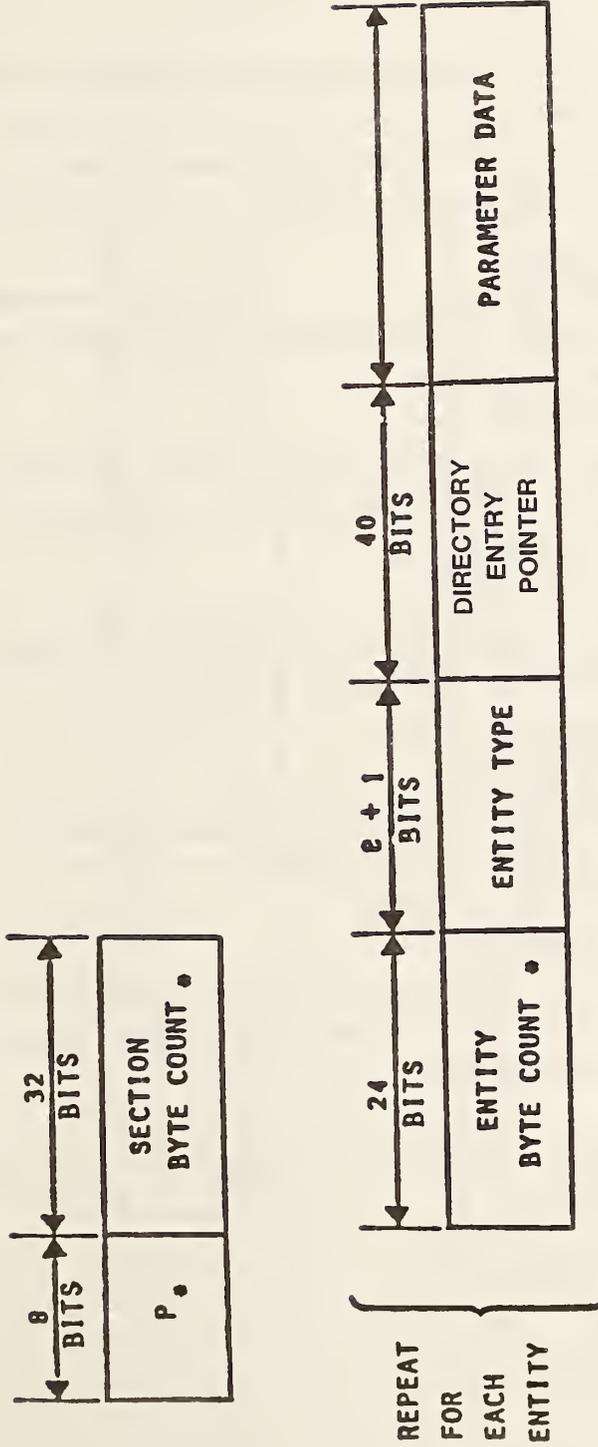
FIGURE 2-15 FORMAT OF PARAMETER SECTION

* THESE FIELDS DO NOT HAVE CONTROL BYTES

2.4.2.6 **Terminate Section.** The format of the terminate section is as shown in Figure 2-16. The terminate section is comprised of the following data items:

- o Terminate section identifier consisting of the ASCII letter T
- o Terminate section byte count. This byte count excludes the 5 bytes required for the section identifier and section byte count. This byte count also excludes any null padding characters.
- o ASCII letter B
- o Binary identification section byte count, including the section identifier, and section byte count, but excluding any null padding characters.
- o ASCII letter S
- o Start section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.
- o ASCII letter G
- o Global section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.
- o ASCII letter D
- o Directory entry section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.
- o ASCII letter P
- o Parameter data section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.

The terminate section has no control bytes applied to any of its data.
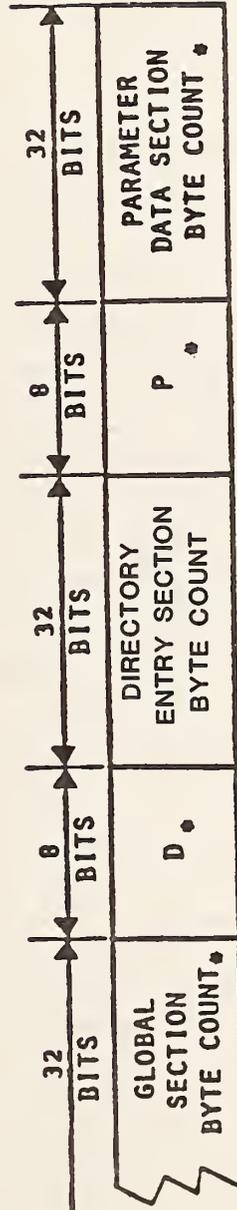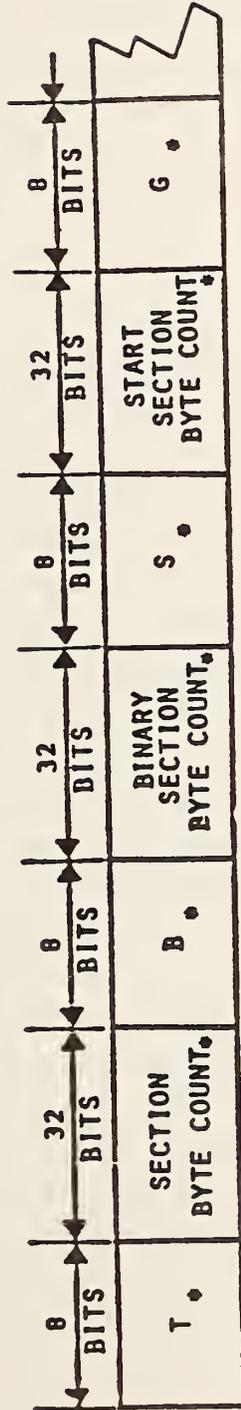
FIGURE 2-16 FORMAT OF TERMINATE SECTION

* THESE FIELDS DO NOT HAVE CONTROL BYTES

THIS PAGE LEFT BLANK

2.5   **Specific File Structures**

Individual entities of the file structure are described in Sections 3 and 4
of this document. Complete files of these entities must be structured
uniformly to convey specialized information between applications.
Some of the ways in which those entities tie together to form
relationships are described in this Section.

2.5.1   Subfigures

The concept of subfigures is supported by defining a collection of
entities once and then instancing that collection multiple times in the
model at various locations, orientations, and scales. The entity use flag
defines how the entity will be used in the model. The structure consists
of a Subfigure Definition Entity (type 308) which points to a collection
of entities (see Figure 2-17). One or more subfigure instance entities
may point to the subfigure definition entity and specify an X,Y,Z
location and scale factor. Rotation of the instance is specified by
having the subfigure instance point to a transformation matrix entity.

Two types of subfigure definitions and four types of subfigure instances
are defined. The Subfigure Definition Entity can be instanced: (1)
individually by the Singular Subfigure Instance Entity, (2) as a
rectangular grid of instances by the Rectangular Array Subfigure
Instance Entity or, (3) as a circularly located set of instances by the
Circular Array Subfigure Instance Entity. Items which have locations
specified for logical connections (represented by one or more Connect
Point Entities) in their subfigure definitions, and thus instances, or
which require independent scale factors in the X,Y, and Z axes shall be
defined by a Network Subfigure Definition Entity and (4) instanced by
the Network Subfigure Instance Entity (see Section 2.5.2 and Figure 2-
18).

85
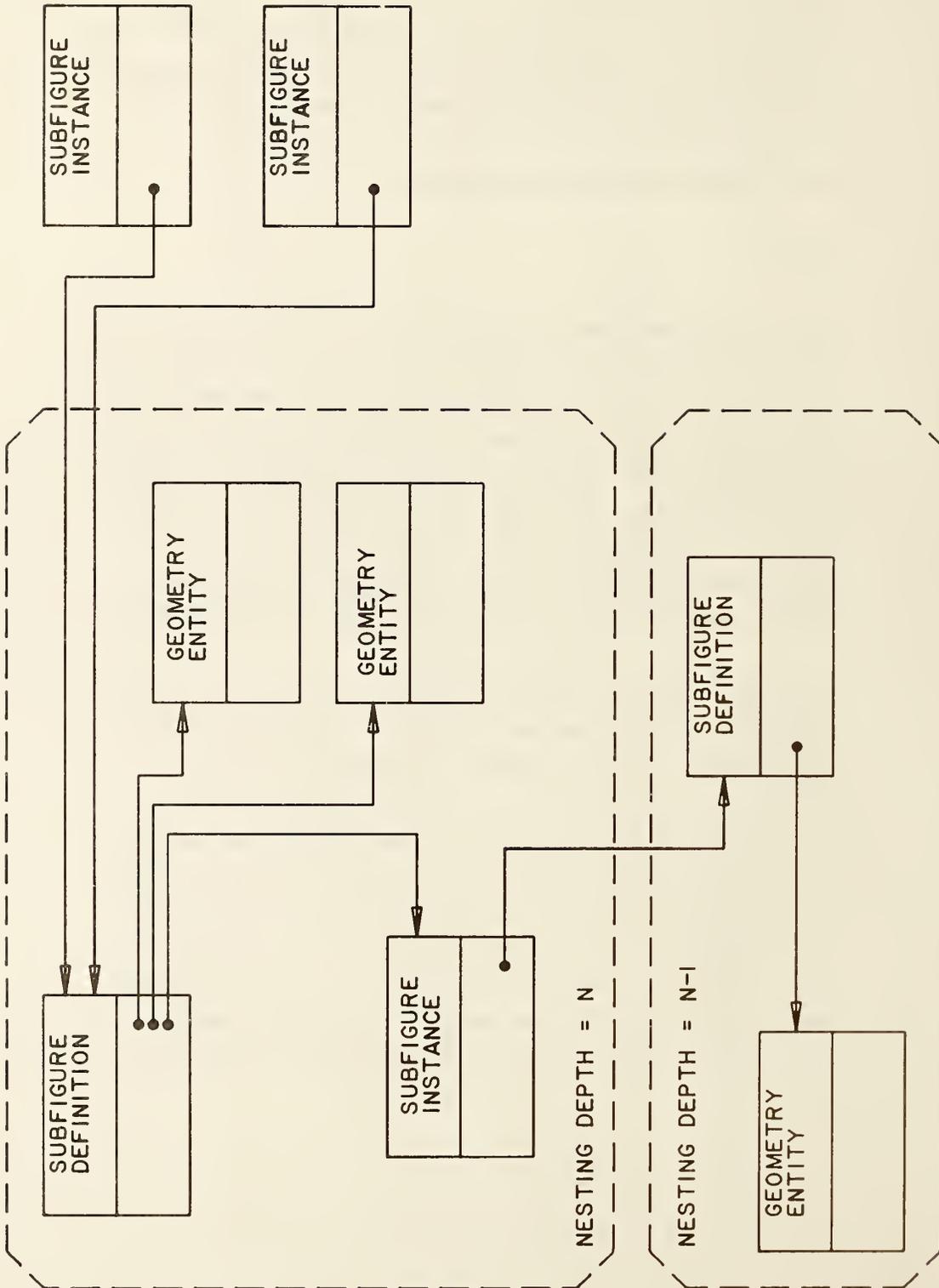
SUBFIGURE STRUCTURES
FIGURE 2-17

GENERAL CONNECTIVITY POINTER DIAGRAM

FIGURE 2-18

LEGEND
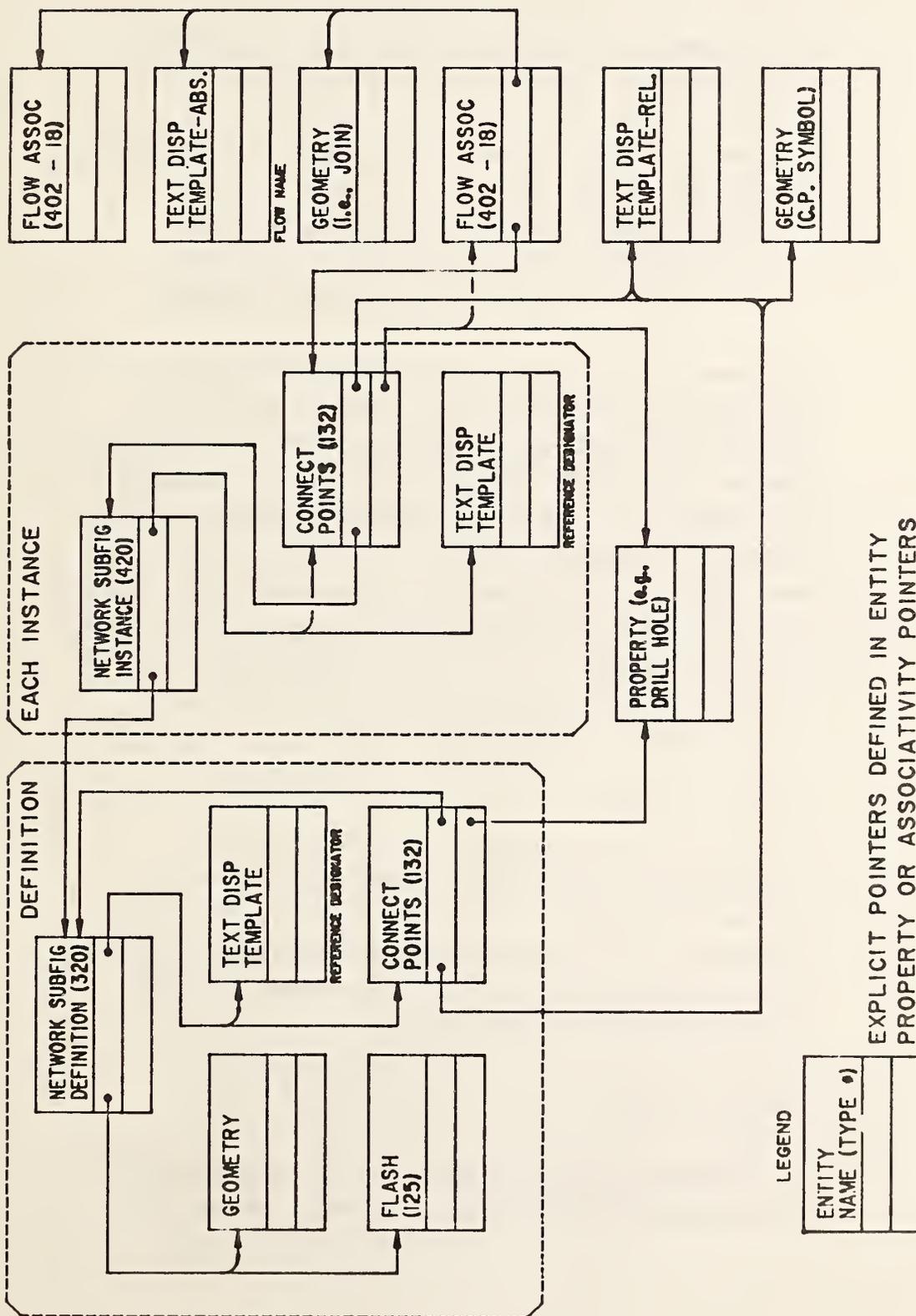
ENTITY
NAME (TYPE #)

EXPLICIT POINTERS DEFINED IN ENTITY
PROPERTY OR ASSOCIATIVITY POINTERS

87

2.5.2   Connectivity

The following file structure shall be used to define logical (and the
location for physical) connections between objects.

A formed connection between two or more objects requires the data to
represent the following:
>   1.   the exact location of each connection point
>   2.   the flow path formed and its identification (if any)
>   3.   the physical connection between the objects (if any)

These objects may include electrical or mechanical components such as
transistors, pipes and valves, and air conditioning ductwork. Each
connection formed defines a flow path between the objects, allowing a
fluid (electricity, water, or air) to flow from one object to another.
The Network Subfigure (Definition and Instance) entities are used to
represent the objects to be connected. The Connect Point Entity is
used to represent the exact location of connection. The term "link" will
refer to the logical representation of the flow path (signal) formed, and
"flow-name" will refer to the flow path identifier. The term "join" will
refer to the file entity or entities which represent the physical
connection (geometries between the items).

2.5.2.1   Connectivity Entities

The entities used to implement connectivity include the Network
Subfigure Definition (type 320) and Network Subfigure Instance (type
420) entities, the Flow Associativity (type 402 form 18), the Connect
Point Entity (type 132), and the Text Display Template (type 312;
absolute = form 0, incremental = form 1).

2.5.2.2   Entity Relationships

A flow path (signal) may be formed between items by a link which
references the items' connect points (entities) to be related. This

creates an associativity among the connect points and thus the entities connected. The flow-name may be used to uniquely identify the particular signal formed. The join may be used to provide a graphical representation of the flow path. In electrical applications the join will be represented by geometric entities such as line, arc, subfigure, copious data, etc. In a piping application, an example of a join represented might be the section of pipe between a valve and a tank. The logical constructs (link and flow-name) shall be implemented by the Flow Associativity Entity which in turn identifies (by pointer) the entities which form the join.

In electrical applications, for example, the items to be connected are components (i.e., resistor, 16-pin dual in-line package, etc.), or integrated circuit cells, represented and instanced by Network Subfigures. Each pin (or signal port) is a potential connection point in a flow path, thus each Network Subfigure has a Connect Point for each pin (or port). When such a subfigure is instanced, its connect points must also be instanced. An instanced Connect Point, when added to a flow path is different from its definition which shall not be a member of any flow path. See Figure 2-18 for the basic entity relationships. For more information see Appendix B.

2.5.2.3   Information Display

The Network Subfigures, representing electrical components for example, often contain text describing the component and its pins. The Text Display Template allows text, embedded in another entity to be displayed without redundant specification of the text string. The Text Display Template may be used to display reference designators and pin numbers. The absolute form, within a network subfigure, is recommended for the reference designator text. Each instance of the subfigure need only supply the text string. The pin number can be represented in the incremental form. All the pin numbers on a given side of a package outline having the same X, Y, and Z offsets relative to the pin whose number is to be displayed may use the same Text Display Template definition.

2.5.2.4   Additional Considerations

The situation is exactly the same for both logical and physical product representations. The only differences arise in the subfigure and join entities used. One file may contain both schematic and physical representations of a product. The Flow Associativity Entity contains a type flag to indicate the connection type (logical or physical). In this case, one Flow Associativity would represent the logical connection and a second the physical connection. The two associativities would be related by the pointers provided in the Flow Associativity.

2.5.3   MACROs

A MACRO capability is provided for defining new entities in terms of other entities. See Section 4.3.6. Two specific applications for the MACRO capability are parametric designs and standard parts. The structure consists of MACRO definition entities which are pointed to by MACRO instance entities as shown in Figure 2-19. Each MACRO definition is assigned a unique MACRO entity type number. MACRO instance entities use this number as their entity type number. MACRO instance entities also use the third field of their directory entry to point to the directory entry of the MACRO definition (or to an external reference entity which points to a library file containing the MACRO definition).

2.5.4   External Reference Linkage

Linkages between entities can occur not only within a file, but also between entities in different files. Two entities shall be used in a referencing file to establish this linkage: the External Reference Entity (Type 416) which provides the actual linkage to the referenced file, and the External Reference File List Property Entity (Type 406 Form 12) which provides a list of the names of all the files referenced. Further, only directly referenced files shall be in this property's parameter list. Each file name listed in the parameter data of this property must match the name in the fourth global parameter of a referenced file.

90

DIRECTORY ENTRY SECTION

| 306 | | | | | | | | | | MACRO DEFINITION

| 601 | | | | | | | | | | MACRO INSTANCE

| 601 | | | | | | | | | | MACRO INSTANCE

PARAMETER DATA SECTION

```
306,MACRO,601,XC,YC,ZC,...
LET...
SET...
  .
  .
ENDM:
```
MACRO DEFINITION

```
601,0.,0.,0.,...:
```
MACRO INSTANCE

```
601,1.75,40.,0.,...:
```
MACRO INSTANCE
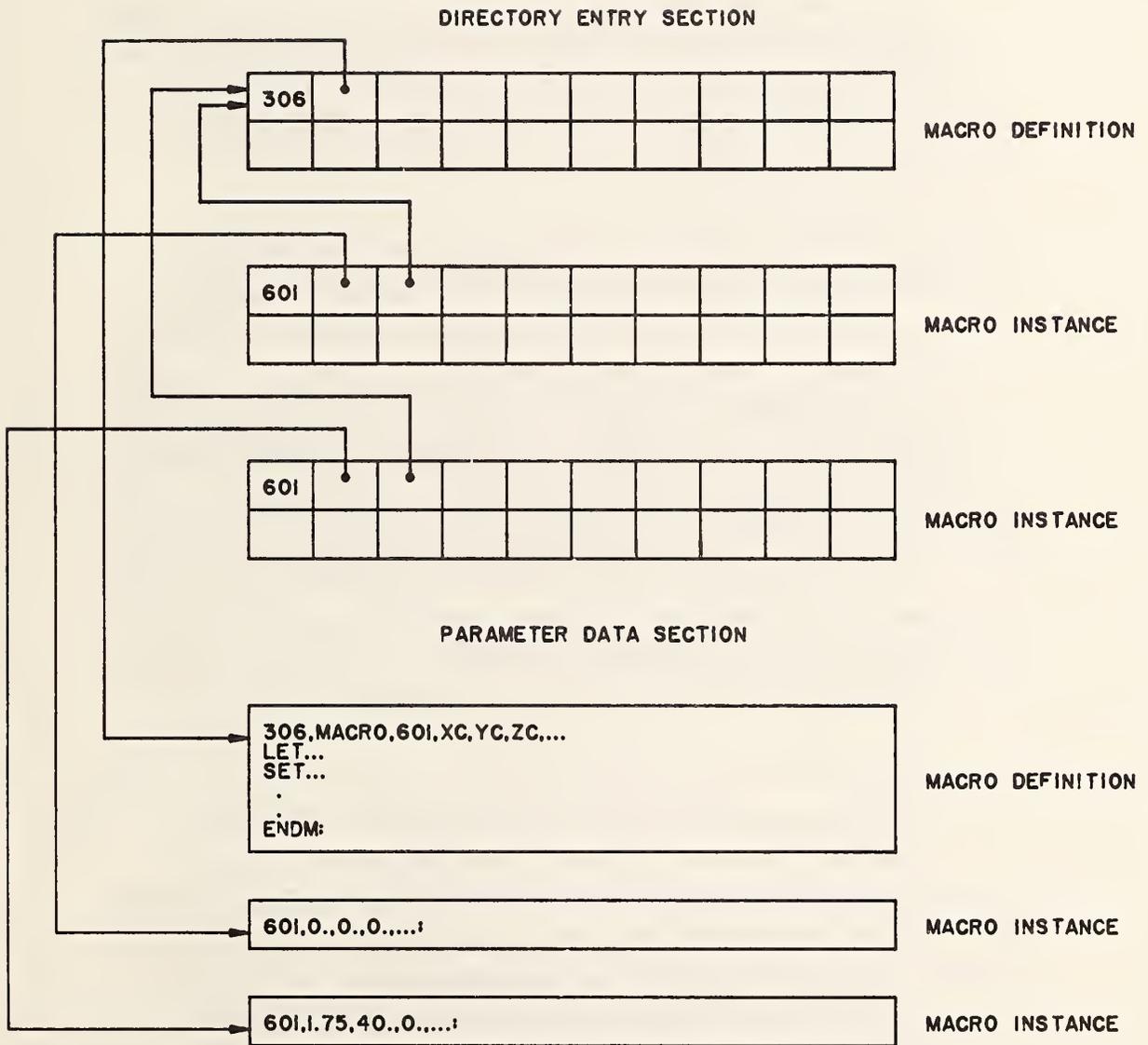
MACRO DEFINITION / INSTANCE STRUCTURE

FIGURE 2-19

91

An External Reference File Index associativity (type 402 form 12) is
required in the referenced file when the type 416 Form 0 or 2 is used
(i.e., more than one referenced entity in the referenced file).

This associativity provides a directory to the referenced entities within
its file, and both relate a symbolic name to the directory entry of an
entity within the file (see Figure 2-20). All symbolic names used within
a set of files linked by references must be unique. Definitions may be
nested, and a symbolic name used need be unique only on the nesting
level on which it is used.

Because of the intricacy of the linkages, an example follows (refer to
Figure 2-20). Consider a file containing a subfigure instance entity
(type 408). The first item in its parameter data record is a pointer to
the subfigure definition entry in the DE section of the file. In the case
that the subfigure definition entity (type 308) is to be contained in a
library file, this first parameter is a pointer to an external reference
entity (type 416). The external reference entity so referenced will have
in its parameter data record the name of the file which is to contain
the definition and the symbolic name of the definition itself. The file
name is the fourth global parameter in the referenced file. The
symbolic name is a string which identifies the appropriate referenced
definition.

In the case of a library file which contains several definitions, each of
which are expected to be referenced by other files, the External
Reference Associativity (type 402 form 12) provides a "table of
contents" of the available definitions in the file. The parameter data
record of this associativity contains pairs of data: the symbolic name
associated with the definition (the same one used in the Type 416
entity's parameter data record), and a pointer to the directory entry
record which contains the desired definition.

In the case that the entire external file is to be included (i.e., a super-subfigure), Form 1 of the Type 416 entity is used which does not contain a symbolic name in the parameter data record. In a similar manner, the referenced file does not contain an associativity Type 402 Form 2 or 12 entity; it is unneeded since the entire file is to be used.

In either case, the External Reference File Index Property (type 406 Form 12) will be found in the referencing file. The parameter data record contains a simple list of the file names of the various external files referenced by this file. Once again, the file name used is that in the fourth global parameter of the referenced file. Note that this list contains only those file names that are directly referenced; it gives no information about files which may be referenced in turn by those files used by this file.

A limitation of external referencing is that the backpointers (in the "backpointers to associativities" addition to an entity's parameters) cannot be used. If a pointer is required in each direction, separate external reference mechanisms must exist in each file (e.g., the double linkage between files A and B in Figure 2-20).

A preprocessor implementor should use the external logical reference mechanism with care because of the burden placed on the postprocessor.

2.5.5   Drawings and Views

This Specification provides a mechanism for associating models and drawings so that there is consistency between them. The mechanism is based on the existing practices of some CAD/CAM graphic systems to define the views of a part on a drawing in terms of a single 3-dimensional (3-D) model.

The Drawing Entity (type 404) specifies a drawing of a given size within a special drawing space coordinate system. This entity can refer to one or more View entities (type 410) which will specify the projection from
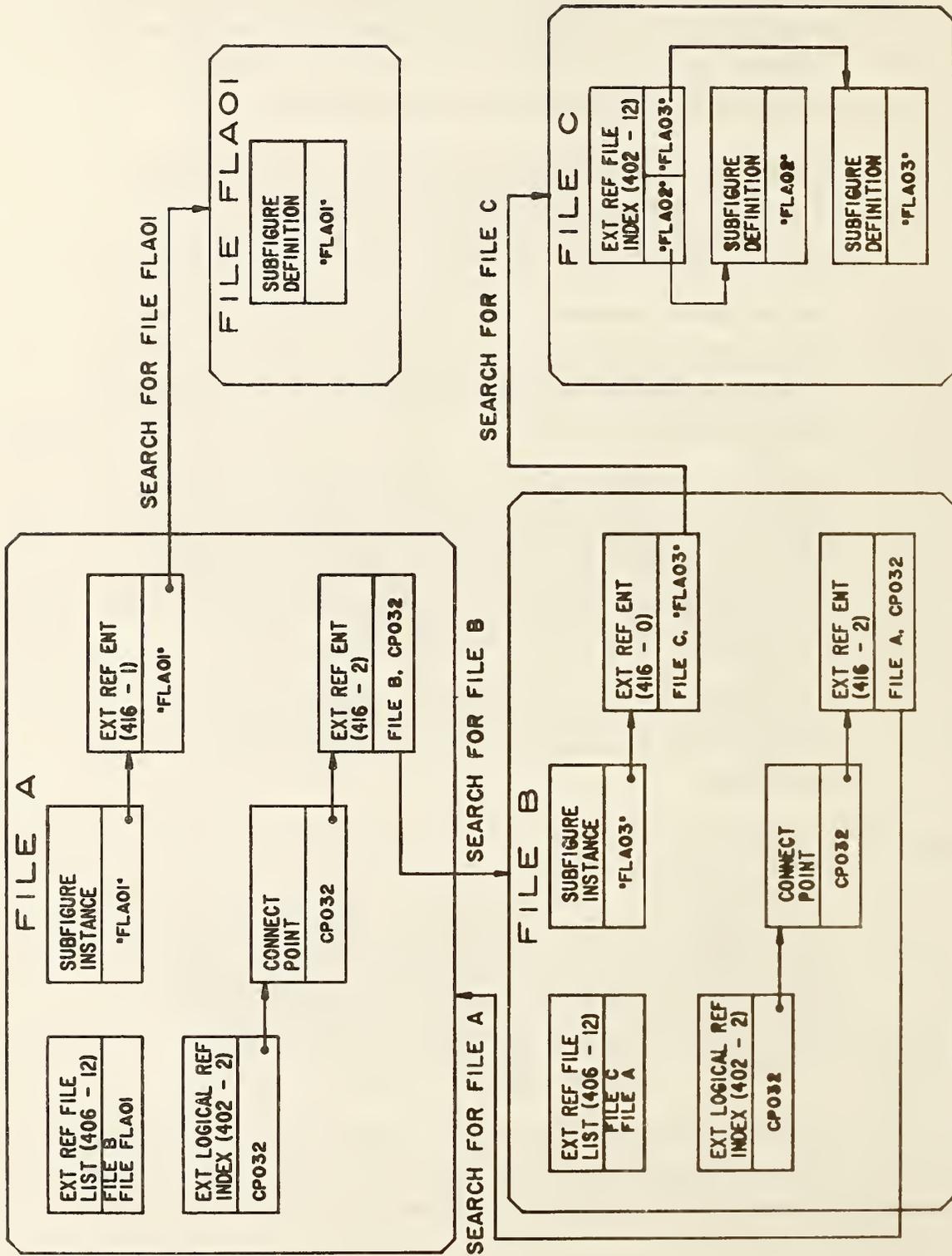
EXTERNAL LINKAGES
FIGURE 2-20

3-D model space to the 2-D drawing space. Annotation entities such as
dimensioning can be defined directly in the drawing coordinate system,
or can be defined in the 3-D model space and then be included in
individual views. More than one drawing entity may be included in a
file.

In addition to being used in conjunction with the Drawing entity, the
view-specific display of parts of the model can be used to communicate
hidden lines, phantom lines, etc.

Graphic systems which do not have the ability to define drawing and
views of models in this manner are not required to preprocess this
construct into a file, but all systems with postprocessors must be able
to process the drawing and view entities in received files.

2.5.6   Finite Element Modeling

This Section defines the entities and their relationships (pointers)
required to support the Finite Element Modeling (FEM) application and
to display results of analysis on those systems which support finite
element analysis postprocessing.

The entities available for exchanging FEM data are illustrated in
Figures 2-21 and 2-22. The left side of Figure 2-21 illustrates the
relationships between the entities that define the model's parametric
attributes. The right side illustrates the addition of the analysis
results. Figure 2-22 illustrates the FEM entities used to define an
example beam structure with accompanying material properties, a load,
and a constraint. The entities defined in support of such analysis are
the Element, Node, Load/Constraint, Tabular Data Property, and Nodal
Displacement and Rotation.

Element (type 136) defines a finite element to be used in the finite
element model. Several finite elements are defined in the
Specification. Examples of an element are: BEAM, CTRIA, and DAMP.
Specifically, the element entity specifies the topology type, number of

95
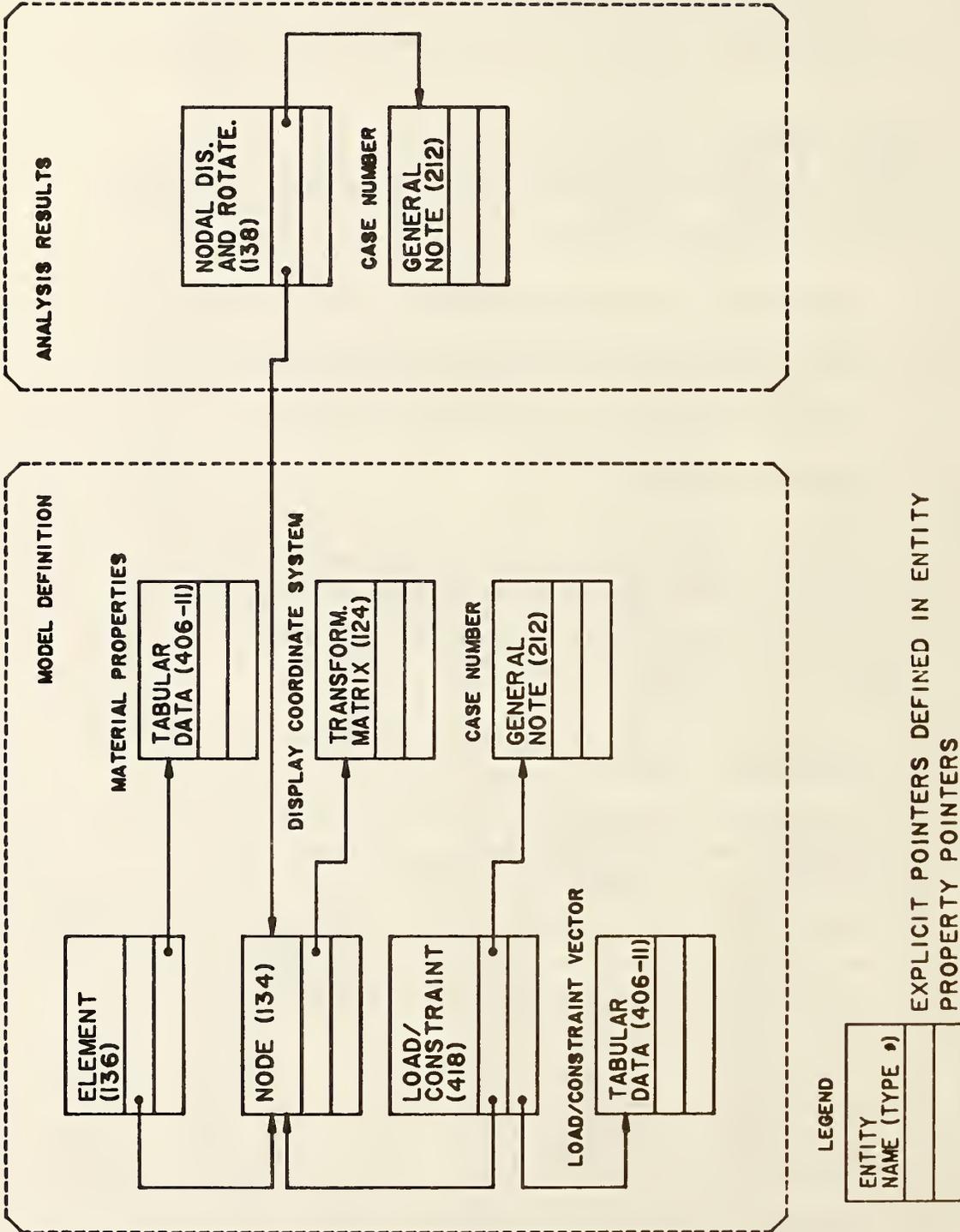
FIGURE 2-21   FINITE ELEMENT MODELING FILE STRUCTURE

96
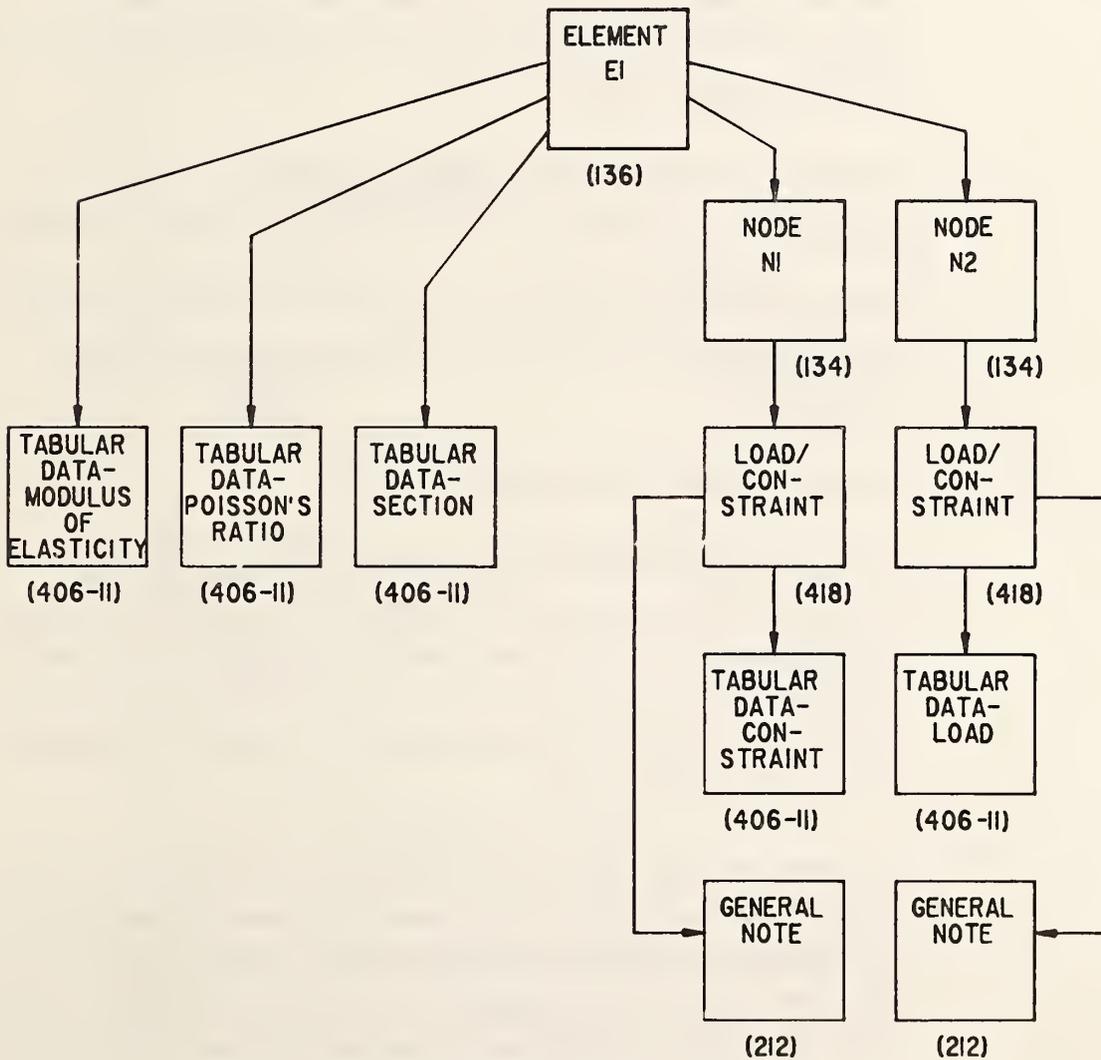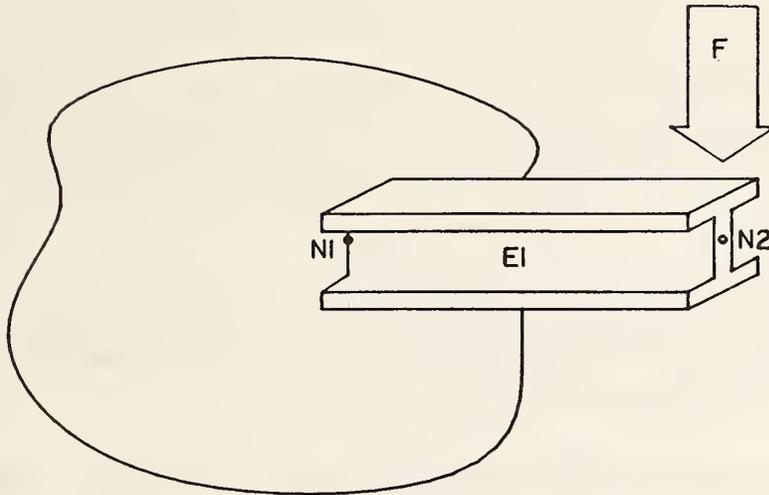
FINITE ELEMENT MODELING LOGICAL STRUCTURE
FIGURE 2-22

97

nodes, and the element type name. Pointers locate the defining nodes and the material properties of the element. The connectivity of the nodes is implied in the order of the contained pointers and topology type.

Node (type 134) defines the grid points or nodes of the element. It contains the spatial values that define the node and a pointer to the coordinate system upon which it is defined.

Load/Constraint (type 418) is an entity that points to a node. It defines either a load or a constraint as applied to that node. It also contains a pointer to general note entities that define the load case. Property pointers point to the tabular data entity that contains the values of the load or constraint vector.

Tabular Data Property (type 406 form 11) contains the material property data of the elements and the load/constraint data as required.

Nodal Displacement and Rotation Entity (type 138) is used to contain the analysis results in the form of incremental variations in node position.

2.5.7   Multiple Transformation Entities

There are only two cases in which entities can be operated on by multiple transformation entities. The first is the explicit case in which an entity points to a transformation entity through its directory entry field 7, and that transformation entity, in turn, points to an additional transformation entity through its directory entry field 7. This structure is illustrated in Figure 2-23a.

The other case is an implicit one in which two entities are in a parent/child relationship, and each points to a transformation entity through its respective directory entry field 7. A parent/child relationship occurs when one entity (the parent) is pointing to another entity (the child). This structure is illustrated in Figure 2-23b.
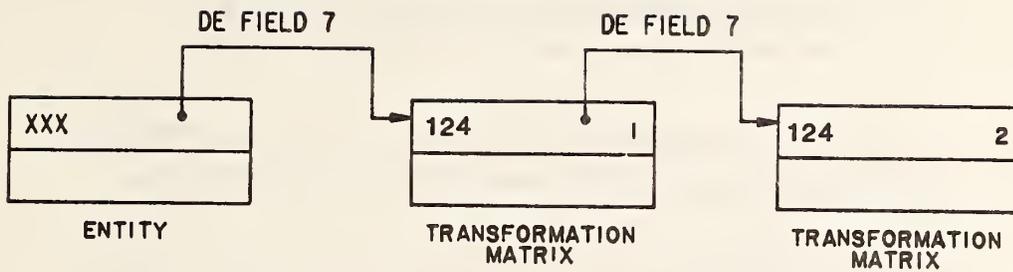
a) EXPLICIT CASE
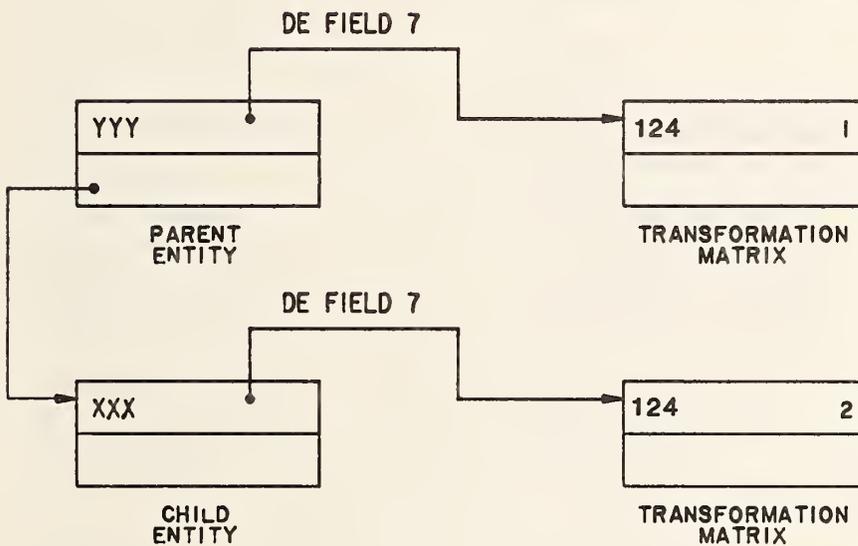
b) PARENT-CHILD CASE

MULTIPLE TRANSFORMATION CASES

FIGURE 2-23

99

In the cases illustrated by Figure 2-23 the points represented by entity XXX are operated upon by matrix 2 and from that point on are transformed like the points in entity YYY, using matrix 1.

A parent/child relationship between entities may also be created with a Single Parent Associativity instance entity (type 402, Form 9).

When the specific parent/child relationships shown in Table 2-4 occur the implicit relation rule shall apply. Each of the relationships in Table 2-4 ordinarily results in the subordinate entity switch of the child entity being set to 01 (physically dependent). The exception is the case in which a preprocessor wishes to actually instance the child entity. In this case the child's subordinate entity switch is set to 02 (logically dependent), and the matrix pointed to by the parent has no effect on the location of the child (see Section 2.2.4.3.9.2).

TABLE 2-4

PHYSICAL PARENT/CHILD RELATIONSHIPS

Parent	Child
Composite Curve	all constituents.
Plane	bounding curve
Point	display symbol
Ruled Surface	rail curves
Flash	defining entity
Surface of Revolution	axis, generatrix
Tabulated Cylinder	directrix
Offset Curve	base curve
Offset Surface	surface
Trimmed Surface	surface
Angular Dimension	all subordinate entities
Diameter Dimension	all subordinate entities
Flag Note	all subordinate entities
General Label	all subordinate entities
Linear Dimension	all subordinate entities
Ordinate Dimension	all subordinate entities
Point Dimension	all subordinate entities
Radius Dimension	all subordinate entities
General Symbol	all subordinate entities
Sectioned Area	all boundary curves
Entity Label Display	all leaders
Connect Point	display symbol, Text Display Templates
Drawing	all annotation entities
Subfigure Definition	all associated entities
Network Subfigure Definition	all associated entities, Text Display Templates and Connect Points
Nodal Display and Rotation	all General Notes and Nodes
any entity with entity use flag =00 or 01	all General Notes in text pointer field

101

THIS PAGE LEFT BLANK

3   GEOMETRY

3.1   **General**

This section gives information concerning the geometry entity types available to be used in the entity-based product definition file. Descriptions of the various directory entry fields were given in Section 2.2.4.3. The meanings of these fields remain the same across all entities. In this section those entities making extended use of field 15 in the directory entry (Form Number) are indicated and the various options are listed. The parameter data record for each entity is also described in this section. The fields for this record vary from entity to entity.

3.1.1   **Coordinate Systems.**

This section introduces a model space concept and a definition space concept. Model space is three-dimensional Euclidean space, the space in which the "model" (or product) being represented resides. The model space X, Y, Z coordinate system is a right-handed Cartesian coordinate system. It is fixed relative to the model.

Definition space is also three-dimensional Euclidean space, but has its own right-handed Cartesian XT, YT, ZT coordinate system. In contrast to model space where a single fixed coordinate system exists, the definition space coordinate system may vary from entity to entity. The origin of a definition space coordinate system may be any point in model space, and the orientation may be arbitrary with respect to model space. It is assumed that the unit of length is always the same in both the model space and the definition space coordinate systems.

The definition space concept allows the use of a temporary coordinate system in positioning certain geometric entities into model space. This concept plays a simplifying role that is most apparent in connection with those entities which can be contained within a single plane. Use of definition space entails initially describing an entity in definition space, and then converting this to a model space description. Thus, an orthogonal matrix and a translation vector are used to generate model space coordinates from definition space coordinates. The orthogonal matrix used for this purpose is called the defining matrix; both it and the translation vector are treated within the Transformation Matrix entity.

The value of the determinant of an orthogonal matrix is always plus or minus one. In case the determinant is one, there are two equivalent points of view that can be taken concerning how the geometric entity is related to model space from its definition space description. In order to simplify the discussion of these that follows, the translation vector is assumed to be the zero vector. This implies that the origin of the definition space coordinate system coincides with the origin in the model space coordinate system.

The first point of view imagines that the two coordinate systems are initially coincident (that is, X axis to XT axis, etc.) but that the XT, YT, ZT coordinate frame is free to rotate relative to the X, Y, Z frame. The geometry entity is then considered to be defined relative to the XT, YT, ZT frame, and the defining matrix then rotates this frame, geometry included, so that the geometry entity is positioned as desired relative to the X, Y, Z frame.

The second point of view imagines that the XT, YT, ZT frame is initially situated so that the geometry entity within definition space is positioned in the desired manner relative to model space. The defining matrix then leaves the geometry entity fixed, but rotates the XT, YT, ZT frame. At the completion of the rotation, the XT, YT, ZT frame becomes the X, Y, Z frame. The result is that the geometry entity is then positioned as desired relative to the X, Y, Z frame.

It is to be emphasized that the discussion here pertains to a single defining matrix whose action in transforming coordinates can be viewed intuitively in two ways. Each point of view stresses the temporary nature of the XT, YT, ZT system, insofar as what is ultimately of interest is the relationship of the geometry entity to the X, Y, Z frame.

In a case when the geometry entity to be located within model space can be contained within a single plane, it can be seen that the definition space concept can be used in such a way that the geometry entity as initially described in definition space can be considered to lie in the XT, YT-plane (i.e., the plane ZT=0). From this, it is then convenient to also allow entities to be situated in definition space in any plane parallel to the XT, YT plane (i.e., ZT=arbitrary constant).

As indicated in 1.5.5, each entity in this section is acted upon by a transformation matrix. This implies that each entity makes use of the definition space concept, i.e., is defined initially in definition space, and then transformed into model space. Thus the complete definition of a geometry entity, with respect to model space, involves the Transformation Matrix entity. However, in some instances, it may very well be that the transformation matrix will leave all coordinates unchanged. This will be the case exactly when the defining matrix is the identity rotation matrix and the translation vector is the zero vector. (In this situation, a convention is provided to prevent unnecessary processing. See the explanation given in 2.2.4.3.7 for Field 7 of the directory entry.)

3.1.2   Directionality.

Within model space, all curves are directed. Such curves have associated end points, i.e., start point and terminate point. For each entity type, the manner of assigning direction is discussed within the description of each individual entity.

Within the entity descriptions that follow, some refer to a "counterclockwise direction" with respect to a sense of rotation in the XT, YT plane. Since the XT, YT plane is located within three dimensional XT, YT, ZT space, this phrase is ambiguous unless a viewing direction is specified from which to view the rotation within the plane. The viewing direction is taken to be from the positive ZT axis looking "down" upon the XT, YT plane. Then, if a clock were imagined to be lying "face up" in the XT, YT plane, i.e., so as to be readable from the chosen viewing direction along the ZT axis - the phrase "counterclockwise direction" refers to the sense of rotation which is opposite the sense of rotation of the hands of the clock. This same notion of the meaning of counterclockwise carries over to any plane that is parallel to the XT, YT plane.

3.1.3   **Geometric Entities.**

Entity numbers from 100 through 199 are reserved for geometry entities. The following entity type numbers have been assigned:

Entity Type Number	Entity Type
100	Circular Arc
102	Composite Curve
104	Conic Arc
106	Copious Data
	Centerline
	Linear Path
	Section Line
	Simple Closed Area
	Witness Line
108	Plane
110	Line
112	Parametric Spline Curve
114	Parametric Spline Surface
116	Point
118	Ruled Surface
120	Surface of Revolution
122	Tabulated Cylinder
124	Transformation Matrix
125	Flash
126	Rational B-Spline Curve
128	Rational B-Spline Surface
130	Offset Curve
132	Connect Point
134	Node
136	Finite Element
138	Nodal Displacement and Rotation
140	Offset Surface
142	Curve on a Parametric Surface
144	Trimmed Parametric Surface

3.2   **Circular Arc Entity**

A circular arc is a connected portion of a parent circle which consists of more than one point. The definition space coordinate system is always chosen so that the circular arc lies in a plane either coincident with or parallel to the XT, YT plane.

3.2.1   A circular arc determines unique arc end points and an arc center point (the center of the parent circle). By considering the arc end points to be enumerated and listed in an ordered manner, start point first, followed by terminate point, a direction with respect to definition space can be associated with the arc. The ordering of the end points corresponds to the ordering necessary for the arc to be traced out in a counterclockwise manner. This convention serves to distinguish the desired circular arc from its complementary arc (complementary with respect to the parent circle). Refer to Section 3.1.2 for information relating to use of the term counterclockwise.

3.2.2   The direction of the arc with respect to model space is determined by the original counterclockwise direction of the arc within definition space, in conjunction with the action of the transformation matrix on the arc.

3.2.3   In the event that a parameterization is required but not given, the default parameterization is:

$$C(t) = (X1 + R*\cos t, Y1 + R*\sin t, ZT)$$
$$\text{for } t2 \leqslant t \leqslant t3$$

where, for i = 2 and 3,
(i) $R = \text{sqrt}((Xi{-}X1)**2 + (Yi{-}Y1)**2)$
(ii) ti is such that $(R*\cos ti, R*\sin ti) = (Xi - X1, Yi - Y1)$
and
$$0 \leqslant t2 < 2*PI$$
$$0 \leqslant t3 - t2 \leqslant 2*PI$$

3.2.4   Examples of the circular arc entity are shown in Figure 3-1. In Example 3 of Figure 3-1, the solid arc is defined using point A as the start point and point B as the terminate point. If the complementary dashed arc were desired, the first endpoint listed in the parameter data entry would be B, and the second would be A.

3.2.5   **Directory Data**
ENTITY TYPE NUMBER :   100

3.2.6   **Parameter Data**

Index	Name	Type	Description
1	ZT	Real	Parallel ZT displacement of arc from XT, YT plane
2	X1	Real	Arc center abscissa
3	Y1	Real	Arc center ordinate
4	X2	Real	Start point abscissa
5	Y2	Real	Start point ordinate
6	X3	Real	Terminate point abscissa
7	Y3	Real	Terminate point ordinate

Additional Pointers as required (see 2.2.4.4.2).

FIG. 3-1 EXAMPLES OF THE CIRCULAR ARC ENTITY
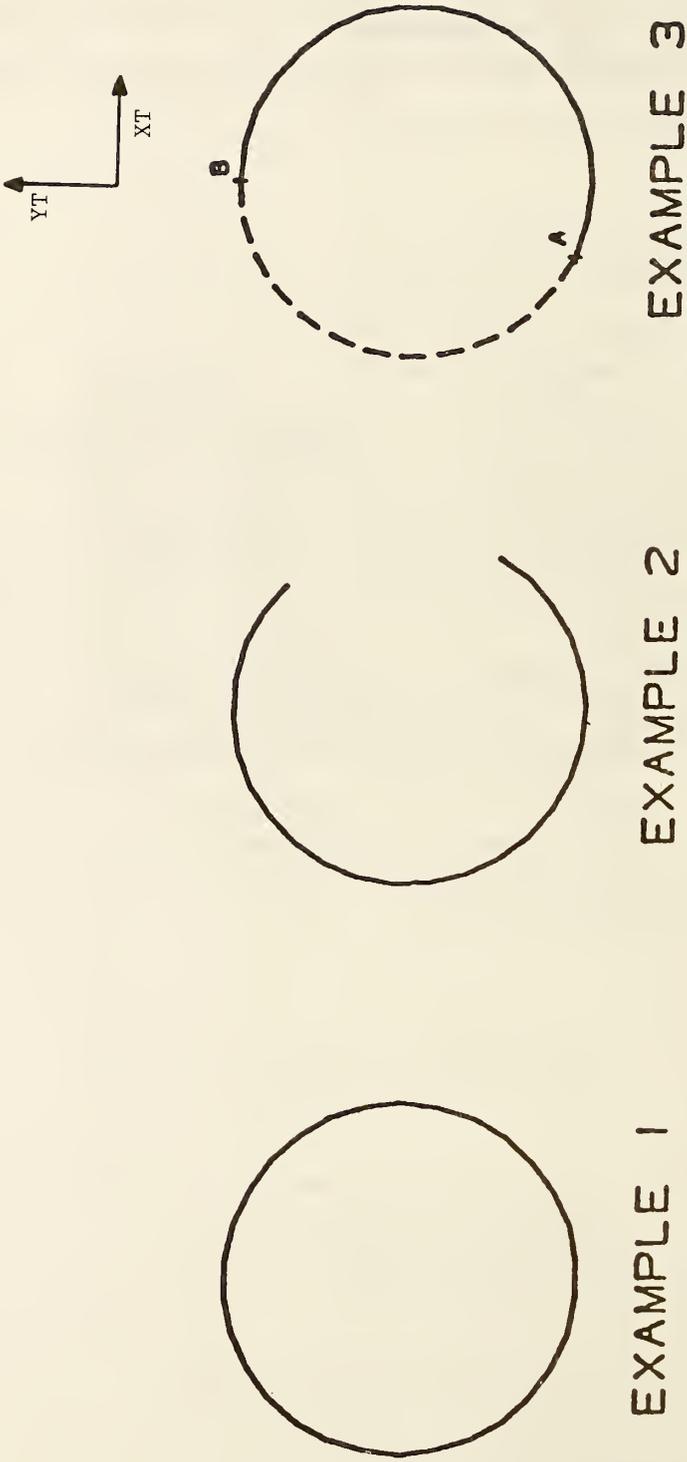
EXAMPLE 1

EXAMPLE 2

EXAMPLE 3

3.3   Composite Curve Entity

A composite curve is a connected curve that results from the grouping of certain individual constituent entities into a logical unit.

3.3.1   A composite curve is defined as an ordered list of entities of the following types: point, line, circular arc, conic arc, parametric spline, rational B-spline, and connect point. The list of entities appears in the parameter data entry. There, each entity to appear in the defining list is indicated by means of a pointer to the directory entry of that entity. The order within the defining list is derived from the order of the listing of these pointers.

3.3.2   Each constituent entity has its own transformation matrix and display attributes. Each constituent entity may have text or properties associated with it. Because the constituent entities are subordinate to the composite entity, the Subordinate Entity Switch (digits 3-4 in directory entry field 9) of each constituent entity should indicate a physical dependency.

3.3.3   A composite curve is a directed curve, having a start point and a terminate point. The direction of the composite curve is induced by the direction of the constituent curve entities (i.e., those constituent entities other than the point entity) in the following way: The start point for the composite curve is the start point of the first curve entity appearing in the defining list. The terminate point for the composite curve is the terminate point of the last curve entity appearing in the defining list. Within the defining list itself, the terminate point of each constituent curve entity has the same coordinates as the start point of the succeeding curve entity.

3.3.4   The point and connect point entities are included as allowable entity types so that properties or general notes can be attached to either the start point or the terminate point of any constituent curve entities in the defining list.

A logical connection relationship can be indicated by having two composite curves or a composite curve and a network subfigure reference the connect point entity. For the special case of the logical connection of a connect point on one subfigure instance to a connect point on another subfigure instance a
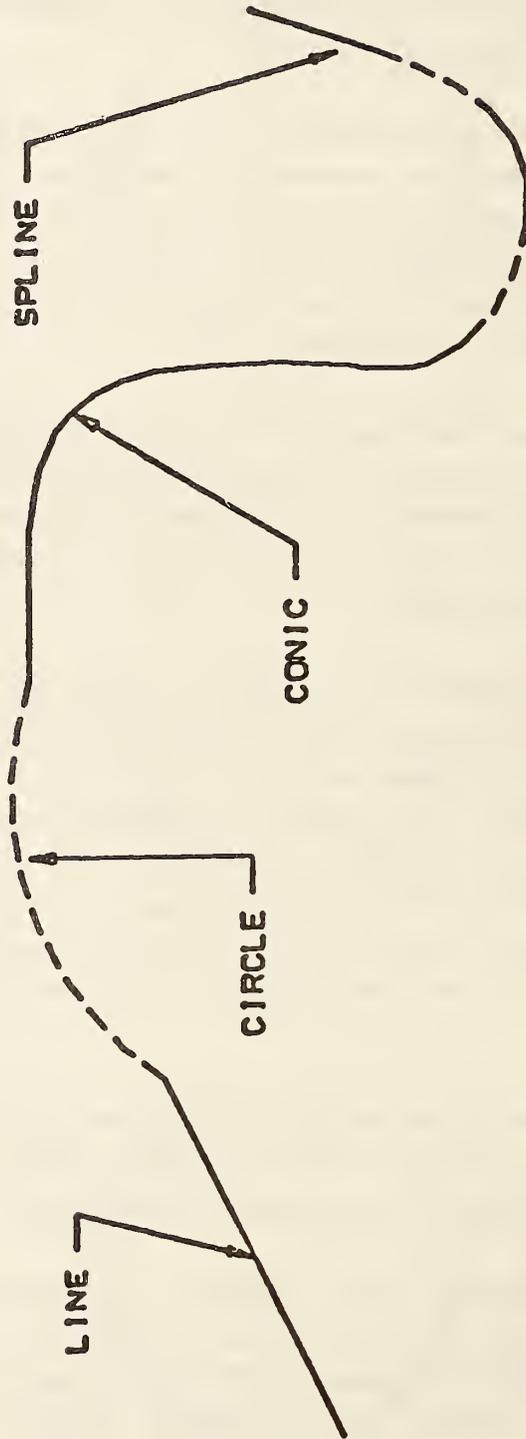
FIG. 3-2 EXAMPLE OF THE COMPOSITE CURVE ENTITY

composite curve is allowed whose list contains only two connect point entities with no intervening curve entity. There are certain restrictions regarding the use of the point entity in a composite entity. They are:

a. Two point or connect point entities cannot appear consecutively in the defining list unless they are the only entities in the composite curve.

b. If a point or connect point entity and a curve entity are adjacent in the defining list, then the coordinates of the point or connect point entity must agree with the coordinates of the terminate point of the curve entity whenever the curve entity precedes the point or connect point entity, and must agree with the coordinates of the start point of the curve entity whenever the curve entity follows the point or connect point entity.

c. A composite curve cannot consist of a point entity alone or a single connect point.

3.3.5   In the event that a parametrization is required but not given, the default parametrization of the composite curve is obtained from the parametrization of the constituent curves as defined below. As point and connect point entities do not contribute to the parametrization of a composite curve, they are not considered in the definition below.

Let

C	be the composite curve;
N	be the number of constituent curves ($N \geqslant 1$);
CC(i)	be the i-th constituent curve, for each i such that $1 \leqslant i \leqslant N$;
PS(i)	be the parametric value of the start of CC(i) ;
PE(i)	be the parametric value of the end of CC(i) ;
T(0)	be 0.0 ;
T(i)	be the sum from j=1 to j=i of (PE(j) - PS(j)), for each i such that $1 \leqslant i \leqslant N$.

Then

(1)   the parametric values of C range from T(0) to T(N) ; and

(2)   C(u) = CC(i) (u - T(i-1) + PS (i)) where u is a parametric value such that
       $T(i-1) \leqslant u \leqslant T(i)$.

A composite curve consisting solely of point and/or connect point entities, will
not be given a parametrization.

3.3.6   In this section, an example of a parametrization of a composite curve entity is
given.

Let N=3 and for each i such that $1 \leqslant i \leqslant 3$, let CC(i) be the i-th constituent curve
of the composite curve C. Assume the parametric values of the start and end
points of each CC(i) are given by the table

i	PS(i)	PE(i)
1	0.0	0.4
2	3.3	3.5
3	0.0	0.3

Then T(0) = 0.0, T(1) = 0.4, T(2) = 0.6, T(3) = 0.9, and the composite curve C, is
defined from 0.0 to 0.9.

This situation described is illustrated by
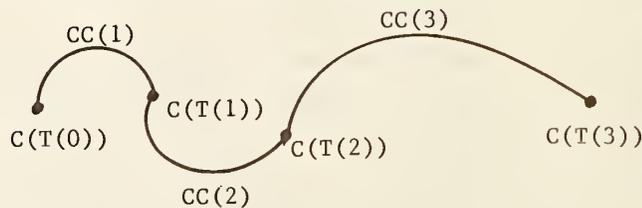
The curve combining CC(1), CC(2), and CC(3) represents the composite curve C.

3.3.7   An example of a composite curve entity is shown in Figure 3-2

3.3.8   **Directory Data**
ENTITY TYPE NUMBER :   102

3.3.9   **Parameter Data**

Index	Name	Type	Description
1	N	Integer	Number of entities
2	DE	Pointer	
.	.	.	Pointers to directory entries for the constituent entities
.	.	.	
.	.	.	
N+1	DE	Pointer	

Additional Pointers as required (see 2.2.4.4.2).

3.4   <u>Conic Arc Entity</u>

A conic arc is a bounded connected portion of a parent conic curve which consists of more than one point. The parent conic curve is either an ellipse, a parabola, or a hyperbola. The definition space coordinate system is always chosen so that the conic arc lies in a plane either coincident with or parallel to the XT, YT plane. Within such a plane, a conic is defined by the six coefficients in the following equation.

$$A*XT^2 + B*XT*YT + C*YT^2 + D*XT + E*YT + F = 0$$

3.4.1   Each coefficient is a real number. The definitions of ellipse, parabola, and hyperbola in terms of these six coefficients are given below.

3.4.2   A conic arc determines unique arc endpoints. A conic arc is defined within definition space by the six coefficients above and the two endpoints. By considering the conic arc endpoints to be enumerated and listed in an ordered manner, start point followed by terminate point, a direction with respect to definition space can be associated with the arc. In order for the desired elliptical arc to be distinquished from its complementary elliptical arc, the direction of the desired elliptical arc must be counterclockwise. In the case of a parabola or hyperbola, the parameters given in the parameter data section uniquely define a portion of the parabola or a portion of a branch of the hyperbola; therefore, the concept of a counterclockwise direction is not applied. (Refer to Section 3.1.2 for information concerning use of the term "counterclockwise".)

3.4.3   The direction of the conic arc with respect to model space is determined by the original direction of the arc within definition space, in conjunction with the action of the transformation matrix on the arc.

3.4.4   The definitions of the terms ellipse, parabola, and hyperbola are given in terms of the quantities Q1, Q2, and Q3. These quantities are:

$$Q1 = \text{determinant of} \begin{bmatrix} A & B/2 & D/2 \\ B/2 & C & E/2 \\ D/2 & E/2 & F \end{bmatrix}$$

$$Q2 = \text{determinant of} \begin{bmatrix} A & B/2 \\ B/2 & C \end{bmatrix}$$

$$Q3 = A + C$$

3.4.5   A parent conic curve is

An ellipse if Q2>0 and Q1 * Q3<0.

A hyperbola if Q2<0 and Q1 $\neq$ 0.

A parabola if Q2 = 0 and Q1 $\neq$ 0.

An example of each type of conic arc is shown in Figure 3-3.

3.4.6   Those entities which can be represented as various degenerate forms of a conic equation (Point and Line) must not be put into the Entity Type 104; more appropriate Entity Types exist for these forms.

Because of the numerical sensitivity of the implicit form of the conic description, a receiving system not using that form as its internal representation for conics need not be expected to correctly process conics in this form unless they are put into a standard position in definition space. A conic arc entity is said to be in a standard position in definition space provided each of its axes is parallel to either the XT axis or YT axis and provided it is centered about the ZT axis. For a parabola, use the vertex as the origin. The conic is moved from this position in definition space to the desired position in space with a transformation matrix (Entity type 124).

The form number is regarded as purely informational by such a postprocessor.

Further details may be found in Appendix E.
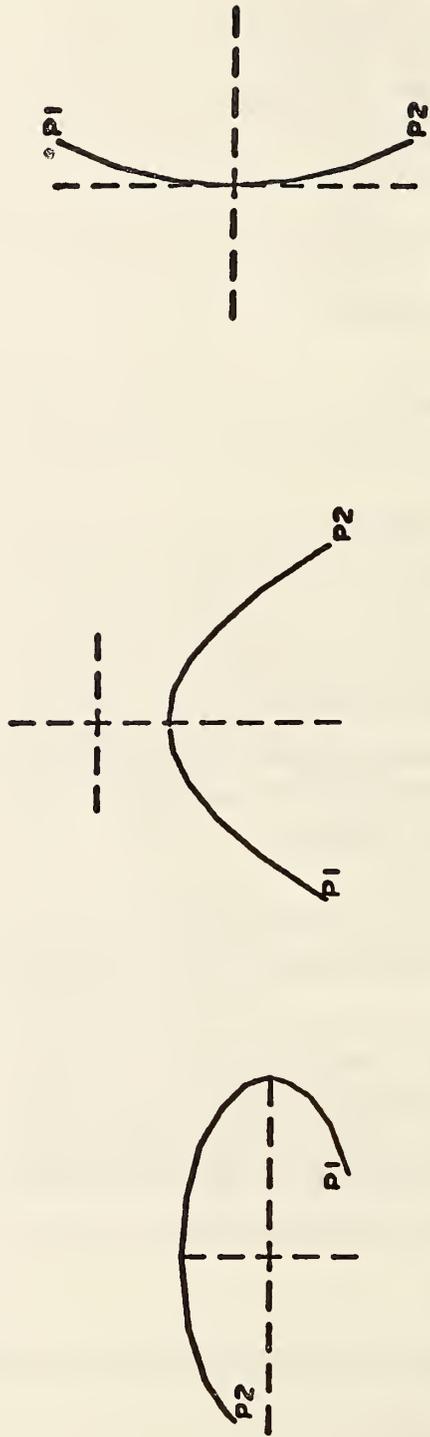
FIG. 3-3 EXAMPLES OF THE CONIC ARC ENTITY

118

3.4.7   In the event that a parameterization is required but not given, the default parameterization is:

Parabola

case A and E $\neq$ 0.0

if X1 $<$ X2
> $C(t) = (t, -(A/E)*t**2, ZT)$ for t1 $\leqslant$ t $\leqslant$ t2
> where, for i = 1 and 2, ti = Xi.

if X2 $<$ X1
> $C(t) = (-t, -(A/E)*t**2, ZT)$ for t1 $\leqslant$ t $\leqslant$ t2
> where, for i = 1 and 2, ti = -Xi.

case C and D $\neq$ 0.0

if Y1 $<$ Y2
> $C(t) = (-(C/D)*t**((2, t, ZT)$ for t1 $\leqslant$ t $\leqslant$ t2
> where, for i = 1 and 2, ti = Yi.

if Y2 $<$ Y1
> $C(t) = (-(C/D)*t**2, -t, ZT)$ for t1 $\leqslant$ t $\leqslant$ t2
> where, for i = 1 and 2, ti = -Yi.

Ellipse

$C(t) = (a*cos\ t, b*sin\ t, ZT)$

> for t1 $\leqslant$ t $\leqslant$ t2

where
> $a = sqrt(-F/A)$
> $b = sqrt(-F/C)$

and, for i = 1 and 2, ti is such that
> (i) $(a*cos\ ti, b*sin\ ti, ZT) = (Xi, Yi, ZT)$
> (ii) $0 \leqslant t1 \leqslant 2*PI$
> (iii) $0 \leqslant t2 - t1 \leqslant 2*PI$

Hyperbola

case F*A $<$ 0.0 and F*C $>$ 0.0

let
> $a = sqrt(-F/A)$
> $b = sqrt(F/C)$
> and, for i = 1,2
> > ti is such that
> > (i) $(a*sec\ ti, b*tan\ ti, ZT) = (Xi,Yi,ZT)$
> > (ii) $- PI/2 < t1,t2 < PI/2$

if t1 $<$ t2

$\quad\quad$ C(t) = (a*sec t, b*tan t, ZT) $\quad\quad\quad\quad\quad\quad$ for 1 $\leqslant$ t $\leqslant$ t2

if t2 $<$ t1

$\quad\quad$ C(t) = (a*sec(-t), b*tan(-t), ZT) $\quad\quad\quad\quad$ for -t1 $\leqslant$ t $\leqslant$ -t2

case F*A $>$ 0.0 and F*C $<$ 0.0

let
$\quad\quad$ a = sqrt(F/A)
$\quad\quad$ b = sqrt(-F/C)
$\quad\quad$ and, for i = 1,2
$\quad\quad\quad\quad$ ti is such that
$\quad\quad\quad\quad\quad\quad$ (i) $\quad$ (a*tan ti, b*sec ti, ZT) = (Xi,Yi,ZT)
$\quad\quad\quad\quad\quad\quad$ (ii) - PI/2 $<$ t1,t2 $<$ PI/2

if t1 $<$ t2

$\quad\quad$ C(t) = (a*tan t, b*sec t, ZT) $\quad\quad\quad\quad\quad\quad$ for t1 $\leqslant$ t $\leqslant$ t2

if t2 $<$ t1

$\quad\quad$ C(t) = (a*tan(-t), b*sec(-t), ZT) $\quad\quad\quad\quad$ for -t1 $\leqslant$ t $\leqslant$ -t2

3.4.8 $\quad$ Field 15 of the directory entry accommodates a Form Number. For this entity, the options are as follows:

FORM	Meaning
0	Form of parent conic curve must be determined from the general equation.
1	Parent conic curve is an ellipse (See example 1, Figure 3-3).
2	Parent conic curve is a hyperbola (See example 2, Figure 3-3).
3	Parent conic curve is a parabola (See example 3, Figure 3-3).

3.4.9   **Directory Data**
ENTITY TYPE NUMBER :   104

3.4.10   **Parameter Data**

Index	Name	Type	Description
1	A	Real	Conic Coefficient
2	B	Real	Conic Coefficient
3	C	Real	Conic Coefficient
4	D	Real	Conic Coefficient
5	E	Real	Conic Coefficient
6	F	Real	Conic Coefficient
7	ZT	Real	ZT Coordinate of plane of definition
8	X1	Real	Start Point Abscissa
9	Y1	Real	Start Point Ordinate
10	X2	Real	Terminate Point Abscissa
11	Y2	Real	Terminate Point Ordinate

Additional Pointers as required (see 2.2.4.4.2).

121

3.5   Copious Data Entity

This entity stores data points in the form of pairs, triples, or sextuples. An interpretation flag value signifies which of these forms is being used. This value is one of the parameter data entries. The interpretation flag is abbreviated below by the letters IP.

Data points within definition space which lie within a single plane are specified in the form of XT, YT coordinate pairs. In this case, the common ZT value is also needed. Data points arbitrarily located within definition space are specified in the form of XT, YT, ZT coordinate triples. Data points within definition space which have an associated vector are specified in the form of sextuples; the XT, YT, ZT coordinates are specified first, followed by the i, j, k coordinates of the vector associated with the point. (Note that, for an associated vector, no special meaning is implicit.)

Field 15 of the directory entry accommodates a Form Number. For this entity, the options are as follows:

FORM	Meaning
1	Data points in the form of coordinate pairs. All data points lie in a plane ZT= constant. (IP=1)
2	Data points in the form of coordinate triples. (IP=2)
3	Data points in the form of sextuples. (IP=3)
11	Data points in the form of coordinate pairs which represent the vertices of a planar, piecewise linear curve (piecewise linear string is sometimes used). All data points lie in a plane ZT=constant. (IP=1)
12	Data points in the form of coordinate triples which represent the vertices of a piecewise linear curve (piecewise linear string is sometimes used). (IP=2)
13	Data points in the form of sextuples. The first triple of each sextuple represents the vertices of a piecewise linear curve (piecewise linear string is sometimes used). The second triple is an associated vector. (IP=3)
20	Centerline Entity through points (IP=1)

21	Centerline Entity through circle centers (IP=1)
31	Section Entity Form 31 (IP=1)
32	Section Entity Form 32 (IP=1)
33	Section Entity Form 33 (IP=1)
34	Section Entity Form 34 (IP=1)
35	Section Entity Form 35 (IP=1)
36	Section Entity Form 36 (IP=1)
37	Section Entity Form 37 (IP=1)
38	Section Entity Form 38 (IP=1)
40	Witness Line Entity (IP=1)
63	Simple Closed Area Entity (IP=1)

The linear path is an ordered set of points in either 2- or 3-dimensional space. These points define a series of linear segments along the consecutive points of the path. The segments may cross or be coincident with each other. Paths may close, i.e., the first path point may be identical to the last.

The linear path is implemented as two forms of the copious data block (entity number 106). Form 11 is for 2-dimensional paths and form 12 is for 3-dimensional paths. This entity will be closely associated with properties indicating functionality and fabrication parameters, such as Line Widening.

Refer to the centerline and witness line entities in Section 4 of this specification for examples of Form Numbers 20, 21 and 40. Each of these annotation entities contains a description of how the associated copious data are to be interpreted. Forms 31-38 provide for the transfer of graphical information and are defined here for compatability with previous versions of the specification. The Sectioned Area Entity (type 230) provides a more compact method for transferring this information.

A simple closed area is a bounded region of XY coordinate space represented by a set of points that forms a series of connected linear segments. These segments must form a closed loop, i.e., the first point of the boundary of the area and the last point must be identical. No segments of this entity are allowed to intersect or be coincident except for the closing of the entity at the initial and final

points. This entity will be closely related to properties that indicate functionality of closed regions, such as Region Fill and Region Restriction.

The area is implemented as Form 63 of entity 106, the copious data block.

3.5.1   Directory Data
ENTITY TYPE NUMBER :   106

3.5.2   Parameter Data

Index	Name	Type	Description
1	IP	Integer	Interpretation Flag
			IP=1 x,y pairs, common z
			IP=2 x,y,z coordinates
			IP=3 x,y,z coordinates and
			i,j,k vectors
2	N1	Integer	Number of n-tuples
For IP=1 (x,y pairs, common z):			
3	ZT	Real	Common z displacement
4	X1	Real	First data point abscissa
5	Y1	Real	First data point ordinate
.	.	.	.
.	.	.	.
.	.	.	.
3+2N	YN	Real	Last data point ordinate
For IP=2 (x,y,z triples):			
3	X1	Real	First data point x value
4	Y1	Real	First data point y value
5	Z1	Real	First data point z value
.	.	.	.
.	.	.	.
.	.	.	.
2+3N	ZN	Real	Last data point z value

For IP=3 (x,y,z,i,j,k sextuples):

3	X1	Real	First data point x value
4	Y1	Real	First data point y value
5	Z1	Real	First data point z value
6	I1	Real	First data point i value
7	J1	Real	First data point j value
8	K1	Real	First data point k value
.	.	.	.
.	.	.	.
.	.	.	.
2+6N	KN	Real	Last data point k value

Additional pointers as required (see sec. 2.2.4.4.2).

3.6   Plane Entity

The plane entity can be used to represent an unbounded plane, as well as a bounded portion of a plane. In either of the above cases, the plane is defined within definition space by means of the coefficients A, B, C, D, where at least one of A, B, and C is non-zero and

$$A*XT + B*YT + C*ZT = D$$

for each point lying in the plane, and having definition space coordinates (XT, YT, ZT).

3.6.1   The definition space coordinates of a point, as well as a size parameter, can be specified in order to assist in defining a system-dependent display symbol. These values are parameter data entries six through nine, respectively. This information, together with the four coefficients defining the plane, provides sufficient information relative to definition space in order to be able to position the display symbol. (In Examples 1 and 3 of Figure 3-4, the dashed curve and the crosshair together constitute the display symbol.) Setting the size parameter to zero indicates that a display symbol is not intended.

3.6.2   The case of a bounded portion of a fixed plane is indicated by the existence of a pointer to a closed curve lying in the plane. This is parameter five. The only allowed coincident points for this curve are the start point and the terminate point. Setting this value to zero indicates the case of an unbounded plane.

3.6.3   The case of a bounded portion of a fixed plane minus some portion(s) of that plane, such as those shown in Figure 3-5, are expressed through the use of the Single Parent Associativity (Type 402, Form 9) where the outer closed curve defines the parent bounded plane and each internal closed curve defines some child bounded plane to be subtracted from the parent. Each of these planes (parent and child) is a separate plane entity in the IGES file and has a backpointer to the associativity structure. The child plane entity will have a subordinate entity switch class of 01 (Physically Dependent).
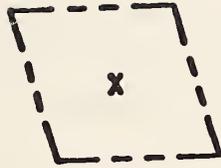
EXAMPLE 3
(UNBOUNDED)
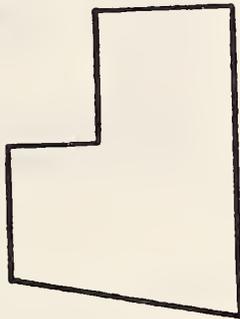
EXAMPLE 2
(BOUNDED)
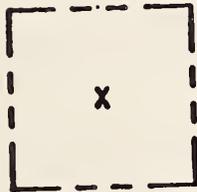
EXAMPLE 1
(UNBOUNDED)

FIG. 3-4 EXAMPLES OF THE PLANE ENTITY
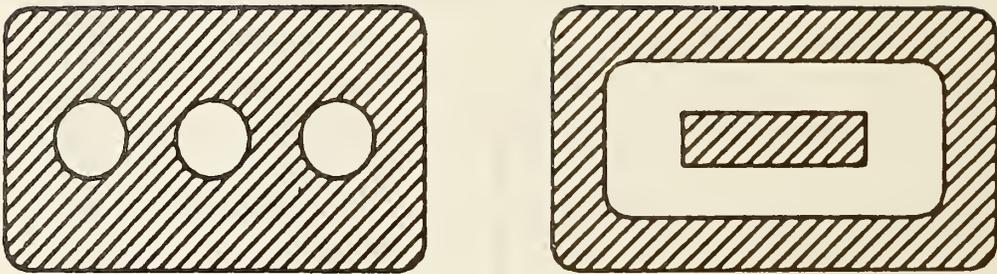
FIG. 3-5 SINGLE PARENT ASSOCIATIVITY AS USED WITH A COLLECTION
OF BOUNDED PLANES

3.6.4   Field 15 of the directory entry accommodates a Form Number. For this entity, the options are as follows:

FORM   Meaning

+1   Bounded planar portion is considered positive.

-1   Bounded planar portion is considered negative (hole).

3.6.5   Directory Data
ENTITY TYPE NUMBER :   108

3.6.6   Parameter Data

Index	Name	Type	Description
1	A	Real	
2	B	Real	Coefficients of Plane
3	C	Real	
4	D	Real	
5	PTR	Pointer	Pointer to directory entry of closed curve entity or zero
6	X	Real	XT coordinate of location point for display symbol
7	Y	Real	YT coordinate of location point for display symbol
8	Z	Real	ZT coordinate of location point for display symbol
9	SIZE	Real	Size parameter for display symbol

Additional Pointers as required (see 2.2.4.4.2).

3.7   Line Entity

A line is a bounded, connected portion of a parent straight line which consists of more than one point.

A line is defined by its end points. Each end point is specified relative to definition space by a triple of coordinates. With respect to definition space, a direction is associated with the line by considering the start point to be listed first and the terminate point second.

The direction of the line with respect to model space is determined by the original direction of the line within definition space, in conjunction with the action of the transformation matrix on the line. Examples of the line entity are shown in Figure 3-6.

3.7.1   In the event that a parameterization is required, the default parameterization is:

$$C(t) = P_1 + t(P_2 - P_1) \quad \text{for} \quad 0 \leqslant t \leqslant 1$$

3.7.2   **Directory Data**
ENTITY TYPE NUMBER :   110

3.7.3   **Parameter Data**

Index	Name	Type	Description
1	X1	Real	Start Point P1
2	Y1	Real	
3	Z1	Real	
4	X2	Real	Terminate Point P2
5	Y2	Real	
6	Z2	Real	

Additional Pointers as required (see 2.2.4.4.2).

EXAMPLE 3

EXAMPLE 2

EXAMPLE 1

FIG. 3-6 EXAMPLES OF THE LINE ENTITY

131

3.8　　　**Parametric Spline Curve Entity**

(Consult Appendix D for additional mathematical details)

The parametric spline curve is a sequence of parametric polynomial segments. The CTYPE value in Parameter 1 indicates the type of curve as it was represented in the sending (pre-processing) system before conversion to this entity.

3.8.1　　The N polynomial segments are delimited by the breakpoints T(1), T(2), ...,T(N+1). The coordinates of the points in the i-th segment of the curve are given by the following cubic polynomials (the coefficients D, or C and D will be zero if the polynomials are of degrees 2 or 1, respectively):

$$X(u) = AX(i) + BX(i) \cdot s + CX(i) \cdot s^2 + DX(i) \cdot s^3$$
$$Y(u) = AY(i) + BY(i) \cdot s + CY(i) \cdot s^2 + DY(i) \cdot s^3$$
$$Z(u) = AZ(i) + BZ(i) \cdot s + CZ(i) \cdot s^2 + DZ(i) \cdot s^3$$

where
$$T(i) \leqslant u \leqslant T(i+1), i=1,...,N$$
$$s = u - T(i)$$

In order to avoid degeneracy, for each i at least one of the nine real coefficients, BX(i), CX(i), DX(i), BY(i), CY(i), DY(i), BZ(i), CZ(i), and DZ(i) must be non-zero.

3.8.2　　If the spline is planar, it must be parametrized in terms of the X and Y polynomials only. The Z polynomial will then be zero except for each i, the AZ(i) term which indicates the Z-depth in definition space.

3.8.3　　The parameter H is used as an indicator of the smoothness of the curve. If H=0, the curve is continuous at all breakpoints. If H=1, the curve is continuous and has slope continuity (see section 6.3 of FAUX79) at all breakpoints. If H=2, the curve is continuous and has both slope and curvature continuity at all breakpoints (see section 6.3 of Faux79).

3.8.4   To enable determination of the terminate point and derivatives without computing the polynomials, the Nth polynomials and their derivatives are evaluated at $u = T(N+1)$. These data are divided by appropriate factorials and stored following the polynomial coefficients. For example, the name TPY3 will be used to designate $1/3!$ times the third derivative of the Y polynomial for the Nth segment evaluated at $u=T(N+1)$, the parameter value corresponding to the terminate point. Note that these data are redundant as they are derived from the data defining the Nth polynomial segment.

3.8.5   An example of a parametric spline is shown in Figure 3-7. Additional examples are shown in Figure 3-8.

3.8.6   <u>Directory Data</u>
         <u>ENTITY TYPE NUMBER</u> :   112

3.8.7   <u>Parameter Data</u>

<u>Index</u>	<u>Name</u>	<u>Type</u>	<u>Description</u>
1	CTYPE	Integer	Spline Type (1=Linear 2=Quadratic 3=Cubic 4=Wilson-Fowler 5=Modified Wilson-Fowler 6=B Spline)
2	H	Integer	Degree of continuity with respect to arc length
3	NDIM	Integer	2=planar 3=non-planar
4	N	Integer	Number of segments
5 . . . 5+N	T(1) . . . T(N+1)	Real	Break points of piecewise polynomial

133

CURVE = (X(U), Y(U), Z(U)), FOR T(1) ≤ U ≤ T(N+1)

N = 3 SEGMENTS
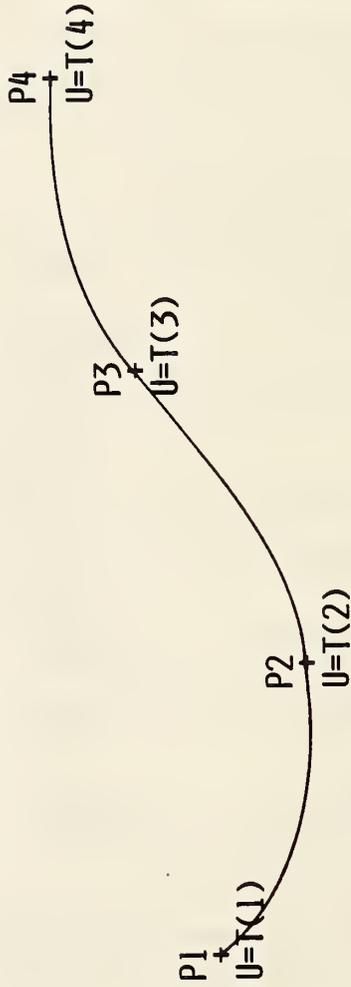
P1 = (AX(1), AY(1), AZ(1))
P2 = (AX(2), AY(2), AZ(2))
P3 = (AX(3), AY(3), AZ(3))
P4 = TP0 = (TPX0, TPY0, TPZ0)
FIRST DERIVATIVE AT P4 = TP1 = (TPX1, TPY1, TPZ1)

FOR SEGMENT NUMBER 2:

$X(U) = AX(2) + BX(2)*(U-T(2)) + CX(2)*(U-T(2))2 + DX(2)*(U-T(2))3$
$Y(U) = AY(2) + BY(2)*(U-T(2)) + CY(2)*(U-T(2))2 + DY(2)*(U-T(2))3$
$Z(U) = AZ(2) + BZ(2)*(U-T(2)) + CZ(2)*(U-T(2))2 + DZ(2)*(U-T(2))3$

FIGURE 3-7   EXAMPLE OF THE PARAMETRIC SPLINE CURVE ENTITY
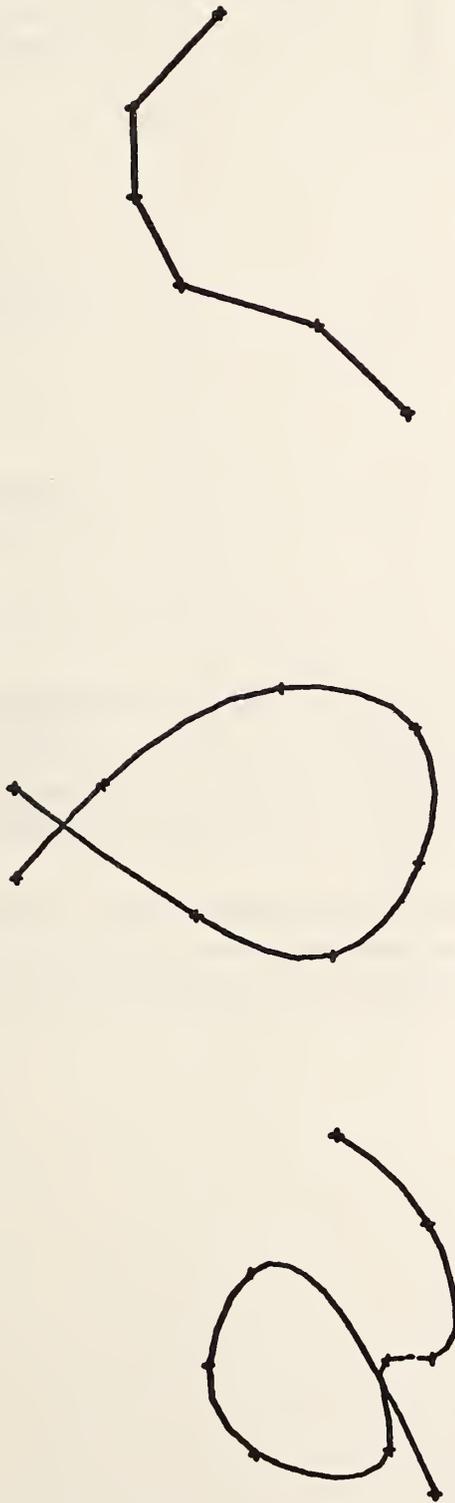
EXAMPLE 3
(LINEAR)

EXAMPLE 2

EXAMPLE 1

FIG. 3-8 EXAMPLES OF PARAMETRIC SPLINE CURVE ENTITY

135

Index	Name	Type	Description
6+N	AX(1)	Real	X coordinate polynomial
7+N	BX(1)		
8+N	CX(1)		
9+N	DX(1)		
10+N	AY(1)		Y coordinate
11+N	BY(1)		polynomial
12+N	CY(1)		
13+N	DY(1)		
14+N	AZ(1)		Z coordinate
15+N	BZ(1)		polynomial
16+N	CZ(1)		
17+N	DZ(1)		
.			Subsequent X, Y, Z polynomials concluding with the twelve coefficients of the Nth polynomial segment.
.			
.			

(The parameters that follow comprise the evaluations of the polynomials of the Nth segment and their derivatives at the parameter value u=T(N+1) corresponding to the terminate point. Subsequently these evaluations are divided by appropriate factorials.)

6+13*N	TPX0	Real	X value
	TPX1		X first derivative
	TPX2		X second derivative/2!
	TPX3		X third derivative/3!
	TPY0		Y value
	TPY1		
	TPY2		
	TPY3		
	TPZ0		Z value
	TPZ1		
	TPZ2		
	TPZ3		

Additional Pointers as required (see 2.2.4.4.2)

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

3.9   Parametric Spline Surface Entity

(Consult Appendix D for additional mathematical details)

The parametric spline surface is a grid of parametric polynomial patches. PTYPE in the Parameter Data Section indicates the type of patch under consideration.

3.9.1   The MxN grid of patches is defined by the u breakpoints TU(1), ... , TU(M+1) and the v breakpoints TV(1), ... , TV(N+1). The coordinates of the points in each of the patches are given by the general bicubic polynomials (given here for the (i, j) Patch).

$$X(u,v) = AX(i,j)+BX(i,j)\cdot s+CX(i,j)\cdot s^2+DX(i,j)\cdot s^3$$
$$+ EX(i,j)\cdot t+FX(i,j)\cdot t\cdot s+GX(i,j)\cdot t\cdot s^2+HX(i,j)\cdot t\cdot s^3$$
$$+ KX(i,j)\cdot t^2+LX(i,j)\cdot t^2\cdot s+MX(i,j)\cdot t^2\cdot s^2+NX(i,j)\cdot t^2\cdot s^3$$
$$+ PX(i,j)\cdot t^3+QX(i,j)\cdot t^3\cdot s+RX(i,j)\cdot t^3\cdot s^2+SX(i,j)\cdot t^3\cdot s^3$$

$$Y(u,v) = \cdots$$
$$Z(u,v) = \cdots$$

where

$$TU(i)\leqslant u\leqslant TU(i+1), \quad i=1, ..., M$$
$$s=u-TU(i)$$

and
$$TV(j)\leqslant v\leqslant TV(j+1), \quad j=1, ..., N$$
$$t=v-TV(j).$$

3.9.2   Parameters with the following indices shall be ignored by post-processors:

for k=1,2,3,...,M   7+M+N+48* (k*N+ (k-1))

through

6+M+N+48* (k*(N+1))

(i.e., the (N+1) row of patches)

and

$7+M+N+48*$ $(M*(N+1))$

    through

$6+M+N+48*(M+1)*(N+1)$

(i.e., the $(M+1)$ column of patches).

To maintain upward compatibility with previous versions of IGES, the pre-processors must either enter a real number for each of these parameters or a series of parameter delimiters (see 2.2.3). These values act as place holders in the parameter list. These parameters were intended to handle first, second, and third partial derivatives of the Nth row and M^{th} column of patches along the outer edge or boundary. However, these parameters can be computed by the receiving system, as needed, from the other parameter values contained in this entity, and therefore are not needed.

SURFACE: (X(U,V),Y(U,V),Z(U,V))
M:6
N:5

U:TU(M+1).
V:TV(N+1)

U:TU(M+1).
V:TV(1)

U:TU(1),
V:TV(N+1)

U:TU(1),
V:TV(1)

6,5

6,1

4,1

3,1

2,1

2,5

2,4

2,3

2,2

1,2

1,1

X(U,V)=AX(2,3)+...
Y(U,V)=AY(2,3)+...
Z(U,V)=AZ(2,3)+...
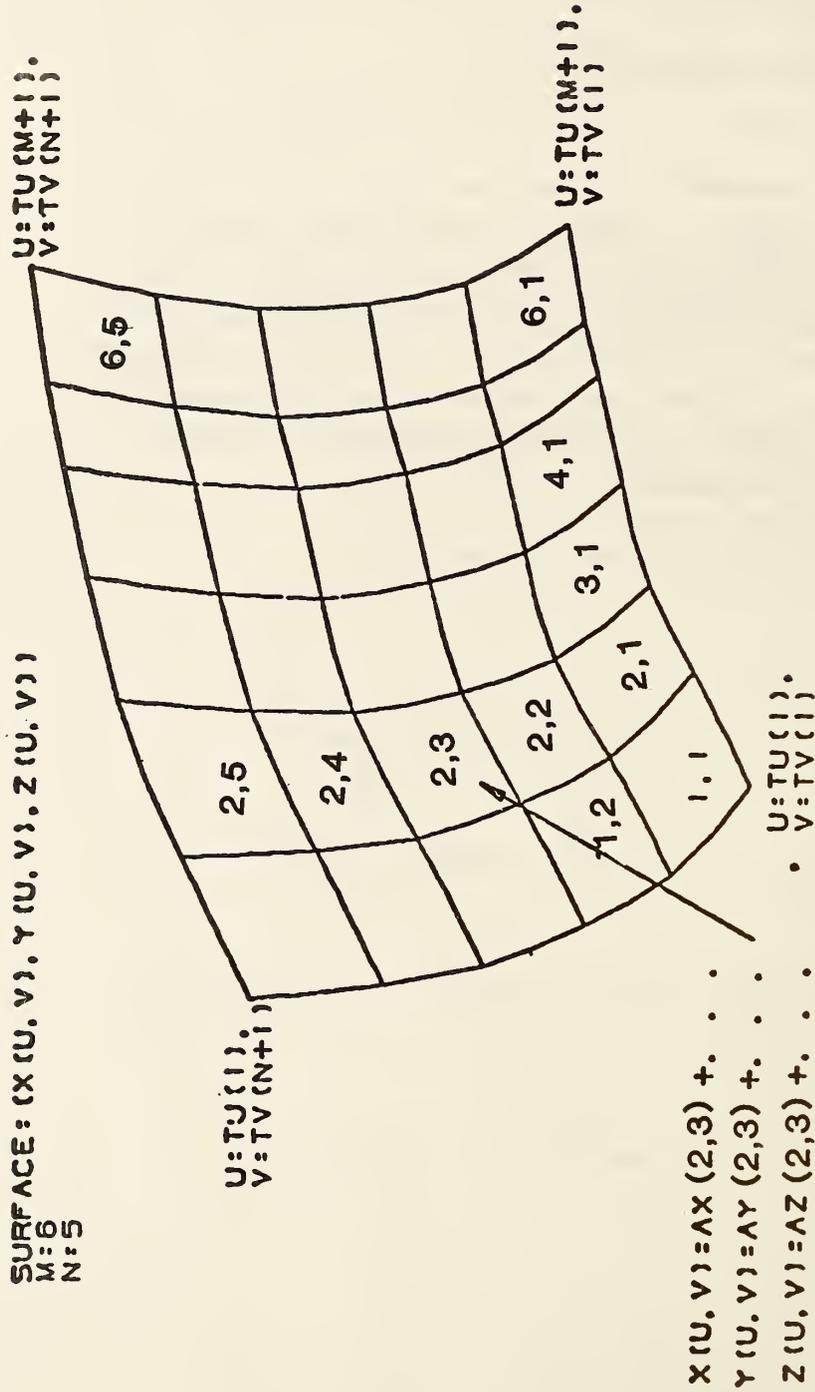
FIGURE 3-9 EXAMPLE OF THE PARAMETRIC SPLINE SURFACE ENTITY

140

3.9.3   An example of the bicubic surface is shown in Figure 3-9.

3.9.4   **Directory Data**
ENTITY TYPE NUMBER :   114

3.9.5   **Parameter Data**

Index	Name	Type	Description
1	CTYPE	Integer	Spline Boundary Type (1=Linear 2=Quadratic 3=Cubic 4=Wilson-Fowler 5=Modified Wilson-Fowler 6 = B spline)
2	PTYPE	Integer	Patch Type (1=Cartesian Product 0=Unspecified)
3	M	Integer	Number of u segments
4	N	Integer	Number of v segments
5	TU(1)	Real	Breakpoints in u (u values of grid lines)
.	.		
.	.		
.	.		
5+M	TU(M+1)		

Index	Name	Type	Description
6+M	TV(1)	Real	Breakpoints in v (v values of grid lines)
.	.		
.	.		
.	.		
6+M+N	TV(N+1)		
7+M+N	AX(1,1)	Real	X Coefficients of (1,1) Patch
.	.		
.	.		
.	.		
22+M+N	SX(1,1)		
23+M+N	AY(1,1)		Y Coefficients of (1,1) Patch
.	.		
.	.		
.	.		
38+M+N	SY(1,1)		
39+M+N	AZ(1,1)		Z Coefficients of (1,1) Patch
.	.		
.	.		
.	.		
54+M+N	SZ(1,1)	Real	
55+M+N	AX(1,2)	Real	Coefficients of (1,2) Patch
.	.	.	.
.	.	.	.
.	.	.	.
102+M+N	SZ(1,2)	Real	
.	.	.	
.	.	.	
.	.	.	
7+M+N+48*(N-1)	AX(1,N)	Real	Coefficients of (1,N) Patch
.	.	.	
.	.	.	
.	.	.	
6+M+N+48*N	SZ(1,N)	Real	
7+M+N+48*N		Real	Arbitrary Values
.	.	.	.
.	.	.	.
.	.	.	.
6+M+N+48*(N+1)		Real	
7+M+N+48*(N+1)	AX(2,1)	Real	Coefficients of (2,1) Patch
.	.	.	
.	.	.	
.	.	.	
6+M+N+48*(N+2)	SZ(2,1)	Real	
.	.	.	
.	.	.	
.	.	.	

Index	Name	Type	Description
7+M+N+48*(2*N)	AX(2,N)	Real	Coefficients of (2,N) Patch
.	.	.	.
.	.	.	.
.	.	.	.
6+M+N+48*(2*N+1)	SZ(2,N)	Real	
7+M+N+48*(2*N+1)		Real	Arbitrary Values
.	.	.	.
.	.	.	.
6+M+N+48*(2*N+2)		Real	
.			
.			
7+M+N+48*[(J-1)*(N+1)+K-1]	AX(J,K)	Real	Coefficients of (J,K) Patch
.	.	.	.
.	.	.	.
.	.	.	.
6+M+N+48*[(J-1)*(N+1)+K]	SZ(J,K)	Real	
.	.	.	
.	.	.	
7+M+N+48*[(M-1)*(N+1)+N-1]	AX(M,N)	Real	Coefficients of (M,N) Patch
.	.	.	.
.	.	.	.
6+M+N+48*[(M-1)*(N+1)+N]	SZ(M,N)	Real	
7+M+N+48*[(M-1)*(N+1)+N]		Real	Arbitrary Values
.		.	.
.		.	.
6+M+N+48*[(M-1)*(N+1)+(N+1)]		Real	
7+M+N+48*[M*(N+1)]		Real	Arbitrary Values
.		.	.
.		.	.
6+M+N+48*[M*(N+1)+(N+1)]		Real	

Additional Pointers as required (see 2.2.4.4.2).

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

3.10   Point Entity

A point is defined by its coordinates in definition space. Examples of the point entity are shown in Figure 3-10.

3.10.1   Directory Data
ENTITY TYPE NUMBER: 116

3.10.2   Parameter Data

Index	Name	Type	Description
1	X	Real	Coordinates
2	Y	Real	of point
3	Z	Real	
4	PTR	Pointer	Pointer to directory entry of subfigure instance specifying the display symbol. If zero, no display symbol specified.

Additional Pointers as required (see 2.2.4.4.2).

EXAMPLE 3

EXAMPLE 2

EXAMPLE 1

FIG. 3-10   EXAMPLES OF THE POINT ENTITY

3.11   **Ruled Surface Entity**

A ruled surface is formed by moving a line connecting points of equal relative arc length (Form 0) or equal relative parametric value (Form 1) on two parametric curves from a start point to a terminate point on the curves. The parametric curves may be points, lines, circles, conics, parametric splines, rational B-splines, composite curves, or any parametric curves defined in this Specification (both planar and non-planar).

3.11.1

Form 0.

In this case, DE1 and DE2 specify the defining rail curves, but their given parametrizations are not the ones used to generate the ruled surface. Instead, their arc length reparametrizations, C1 and C2 (respectively), are used.

Form 1.

In this case, DE1 and DE2 specify the defining rail curves, C1 and C2 (respectively). Moreover, their given parametrizations are the ones used to generate the ruled surface.

For both Form 0 and Form 1.

In either case, the two curves are expressed parametrically by the functions $(C1_X(t), C1_Y(t), C1_Z(t))$ and $(C2_X(s), C2_Y(s), C2_Z(s))$, with range $a \leqslant t \leqslant b$ and $c \leqslant s \leqslant d$, then the coordinates of the points on the ruled surface can be written as

$$X(u,v) = (1-v) \cdot C1_X(t) + v \cdot C2_X(s)$$
$$Y(u,v) = (1-v) \cdot C1_Y(t) + v \cdot C2_Y(s)$$
$$Z(u,v) = (1-v) \cdot C1_Z(t) + v \cdot C2_Z(s)$$

where

$\quad 0 \leqslant u \leqslant 1,$

$\quad 0 \leqslant v \leqslant 1,$

$\quad$ t=a+u*(b-a)

$\quad$ s=c+u*(d-c), if DIRFLG=0

$\quad$ s=d+u*(c-d), if DIRFLG=1

C1(t) and C2(s) are said to be of equal relative parametric value if t and s are evaluated at the same u value.

3.11.2   In case DIRFLG=0, then the first point of curve 1 is joined to the first point of curve 2 and the last point of curve 1 to last point of curve 2. If DIRFLG=1, then the first point of curve 1 is joined to the last point of curve 2, the last point of curve 1 to the first point of curve 2.

3.11.3   If DEVFLG=1, then the surface is a developable surface; if DEVFLG=0, the surface may or may not be a developable surface.

3.11.4   Field 15 of the directory entry accommodates a Form Number. For this entity the options are as follows:

FORM	MEANING
0	Equal relative arc length
1	Equal relative parametric values.

The default is FORM 0.

3.11.5   An example of the Ruled Surface Entity is shown in Figure 3-11. Additional examples are shown in Figure 3-12.
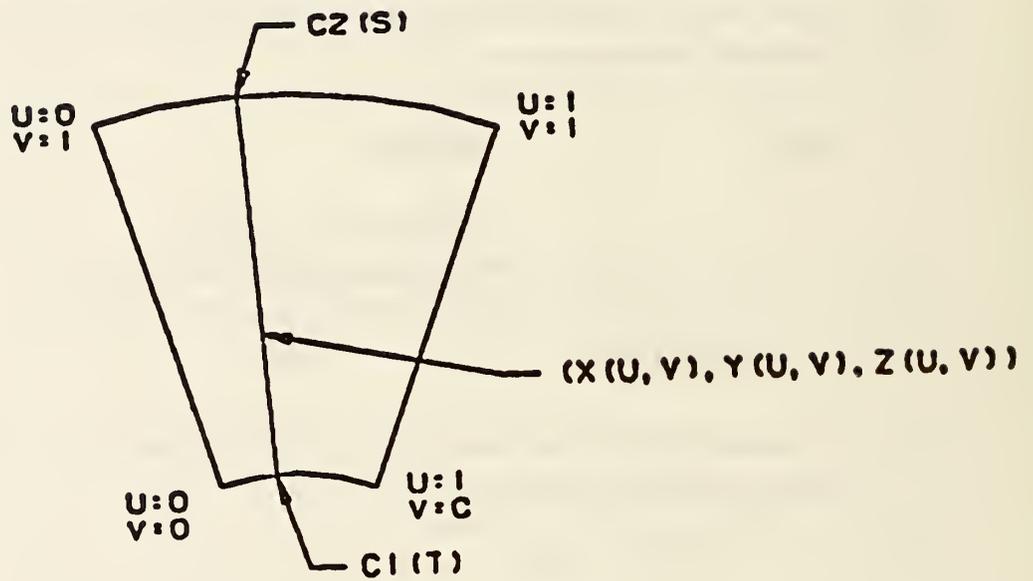
FIG. 3-11   EXAMPLE OF THE RULED SURFACE ENTITY
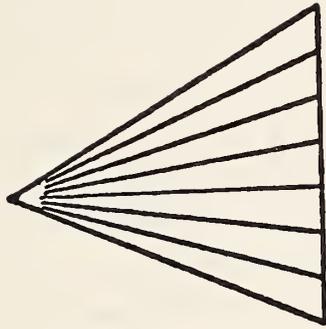
EXAMPLE 3
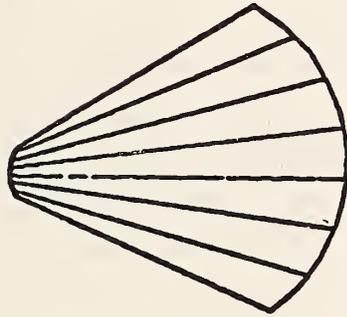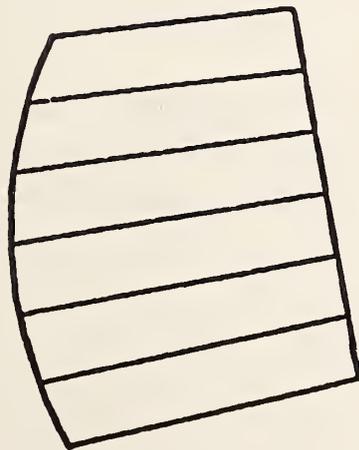
EXAMPLE 2

EXAMPLE 1

FIG. 3-12 EXAMPLES OF THE RULED SURFACE ENTITY

3.11.6   **Directory Data**
ENTITY TYPE NUMBER :   118

3.11.7   **Parameter Data**

Index	Name	Type	Description
1	DE1	Pointer	Pointer to first curve
2	DE2	Pointer	Pointer to second curve
3	DIRFLG	Integer	Direction flag (0=join first to first, last to last 1=join first to last, last to first)
4	DEVFLG	Integer	Developable surface flag (1=Developable, 0=Possibly not)

Additional pointers as required (see 2.2.4.4.2).

3.12   Surface of Revolution Entity

A surface of revolution is defined by an axis of rotation (which must be a line entity), a generatrix, and start and terminate rotation angles. The surface is created by rotating the generatrix about the axis of rotation through the start and terminating angles. Since the axis of rotation is a line entity, it contains in its parameter data section the coordinates of its start point first, followed by the coordinates of its terminate point. The angles of rotation are counterclockwise in the positive direction (See section 3.1.2). The generatrix may be a conic arc, line, circular arc, parametric spline curve, rational B-spline curve, or composite curve.

3.12.1   Examples of surface of revolution entities are shown in Figure 3-13.

3.12.2   The start and terminate angles of the surface can be explained by geometric construction. Refer to Figure 3-14 and the following:

a.   Select a point on the generatrix which does not lie on the axis of rotation; label the point P1.

b.   Construct a line through P1 such that it is perpendicular to the axis of rotation extended; label this line L1.

c.   Construct a plane PN1 containing L1 and perpendicular to the axis of rotation.

d.   All rotations in the plane PN1 about the axis of rotation are applied counterclockwise according to the method described in 3.12.

e.   Rotate counterclockwise the line L1 and the point selected from the generatrix the number of radians indicated in the start angle resulting in $L1_{SA}$. The location is labeled LOC1.

f. Rotate counterclockwise the line L1 and the point P1 selected from the generatrix an additional number of radians given by the terminate angle minus the start angle resulting in $L1_{TA}$. The second location of the point is labeled LOC2.

g. The resulting surface is that generated by rotating the generatrix from LOC1 to LOC2.

3.12.3 **Directory Data**
        ENTITY TYPE NUMBER :   120

3.12.4 **Parameter Data**

Index	Name	Type	Description
1	DE1	Pointer	Pointer to a line entity (axis of revolution)
2	DE2	Pointer	Pointer to generatrix
3	SA	Real	Start angle in radians
4	TA	Real	Terminate angle in radians

Additional pointers as required (see 2.2.4.4.2).
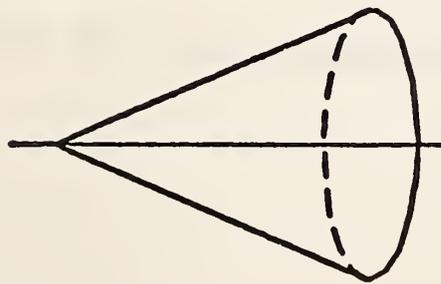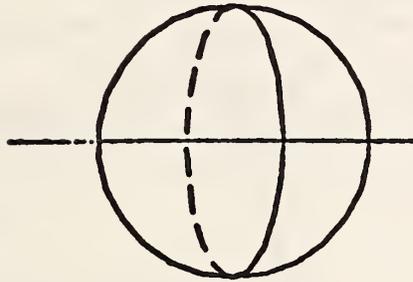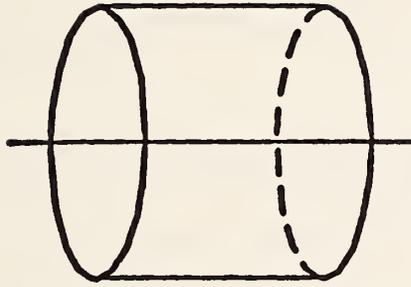
FIG. 3-13 EXAMPLES OF THE SURFACE OF REVOLUTION ENTITY

EXAMPLE 1

EXAMPLE 2

EXAMPLE 3

153
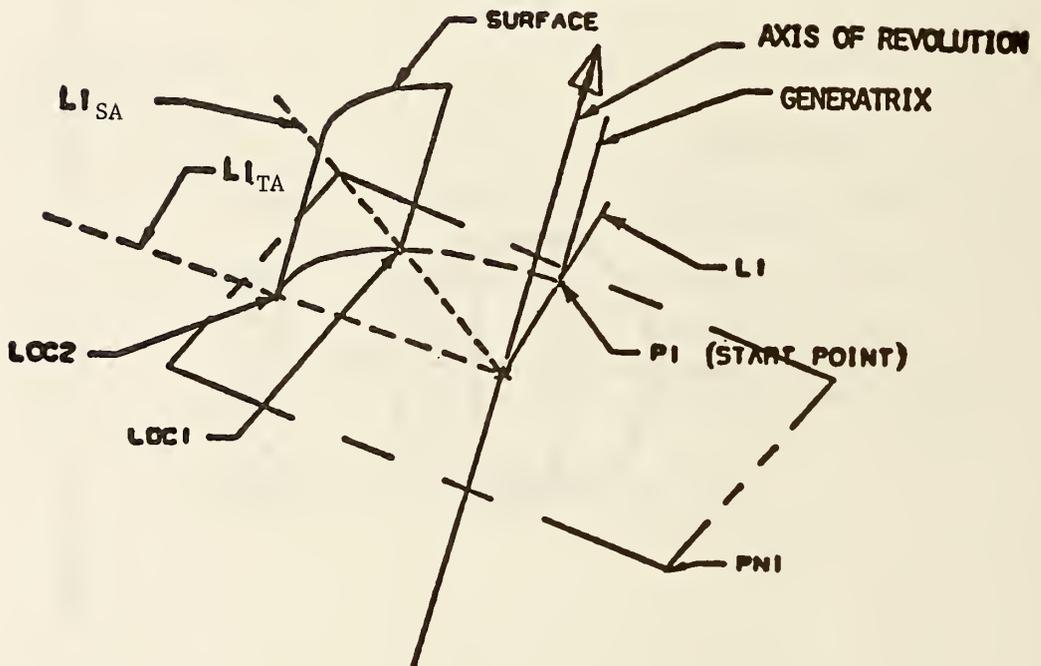
FIG. 3-14 SURFACE OF REVOLUTION START AND TERMINATING ANGLES

3.13   **Tabulated Cylinder Entity**

A tabulated cylinder is a surface formed by moving a line segment called the generatrix parallel to itself along a curve called the directrix. This curve may be a line, circular arc, conic arc, parametric spline curve, rational B-spline curve, or composite curve.

3.13.1   It must be pointed out that different parameterizations of the generating curves will produce different parameterized surfaces, but the underlying point set surface will still be the same. Assuming a parameterization u on the directrix and v on the generatrix, both of which run from 0 to 1, we can express the points on the surface by

$$X(u,v) = CX(u)+v^*(LX-CX(0))$$
$$Y(u,v) = CY(u)+v^*(LY-CY(0))$$
$$Z(u,v) = CZ(u)+v^*(LZ-CZ(0))$$

where $0 \leqslant u \leqslant 1, \ 0 \leqslant v \leqslant 1$

and   CX, CY, CZ represent the X, Y, Z components, respectively, along the directrix curve, while (CX(0), CY(0), CZ(0)) and (LX, LY, LZ) represent the coordinates of the start and terminate points, respectively, of the generatrix.

3.13.2   An example of the tabulated cylinder is shown in Figure 3-15.
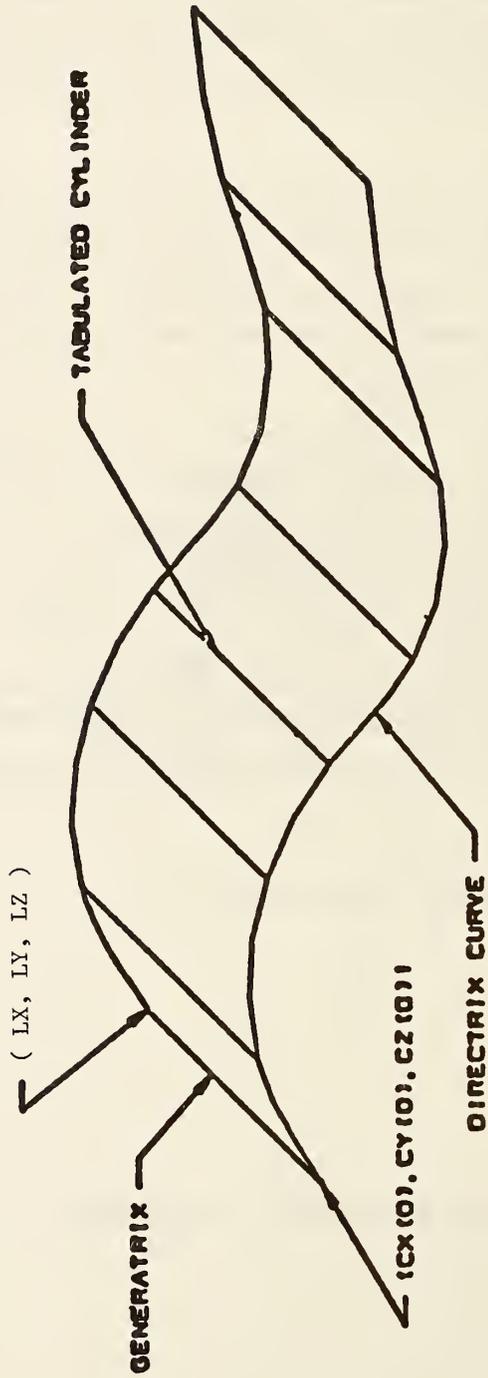
FIG. 3-15 EXAMPLE OF THE TABULATED CYLINDER ENTITY

3.13.3 **Directory Data**
ENTITY TYPE NUMBER : 122

3.13.4 **Parameter Data**

Index	Name	Type	Description
1	DE1	Pointer	Pointer to directrix curve
2	LX	Real	Coordinates of the terminate point of the generatrix. The start point of the generatrix is identical with the start point of the directrix.
3	LY	Real	
4	LZ	Real	

Additional Pointers as required (see 2.2.4.4.2).

3.14   Transformation Matrix Entity

The Transformation Matrix entity transforms three-row column vectors by means of a matrix multiplication and then a vector addition. The notation for this transformation is

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} XINPUT \\ YINPUT \\ ZINPUT \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} = \begin{bmatrix} XOUTPUT \\ YOUTPUT \\ ZOUTPUT \end{bmatrix}$$

Here, $\text{col}\begin{bmatrix} XINPUT, YINPUT, ZINPUT \end{bmatrix}$ (i.e., the column vector) is the vector being transformed, and $\text{col}\begin{bmatrix} XOUTPUT, YOUTPUT, ZOUTPUT \end{bmatrix}$ is the column vector resulting from this transformation. $R = \begin{bmatrix} R_{ij} \end{bmatrix}$ is a 3 row by 3 column matrix of real numbers, and $T = \text{col}\begin{bmatrix} T1, T2, T3 \end{bmatrix}$ is a three-row column vector of real numbers. Thus, 12 real numbers are required for a Transformation Matrix entity. This entity can be considered to be an "operator" entity in that it starts with the input vector, operates on it as described above, and produces the output vector.

3.14.1   Frequently, the input vector lists the coordinates of some point in one coordinate system, and the output vector lists the coordinates of that same point in a second coordinate system. The matrix R and the translation vector T then express a general relationship between the two coordinate systems. By considerating special input vectors such as $\text{col}\begin{bmatrix} 1,0,0 \end{bmatrix}$, $\text{col}\begin{bmatrix} 0,1,0 \end{bmatrix}$, and $\text{col}\begin{bmatrix} 0,0,1 \end{bmatrix}$ and computing the corresponding output results, a geometric appreciation of the spatial relationship between the two coordinate system can be gained.

For example, for

$$R = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

the spatial relationship of the input and output cordinate systems is the following:
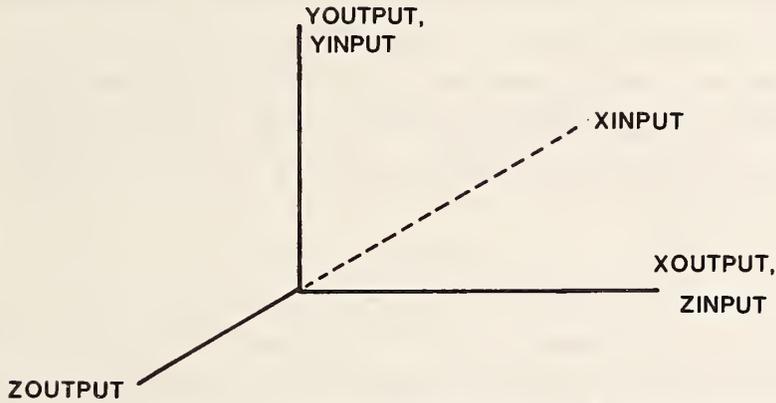
FIGURE 3-16

All coordinate systems are assumed to be orthogonal, cartesian, and right-handed unless specifically noted otherwise.

3.14.2   Following are three specific areas where the Transformation Matrix entity is used to transform coordinates between coordinate systems. Each example area illustrates a specific choice of input and output coordinate system. Other choices of coordinate systems may be appropriate in other application areas.

The usual situation for this type of use of the Transformation Matrix entity is when the input vector refers to the definition space coordinate system for a certain entity, and the output vector refers to the model space coordinate system. (See Sect. 3.1.1) In this case, the matrix R is referred to as the defining matrix, and the Transformation Matrix entity defining R and T is pointed to in field seven (transformation matrix field) of the directory entry of the entity. (See Sect. 2.2.4.3.7) In this use of the Transformation Matrix entity, the matrix R is subject to the restrictions given in Form 0 and Form 1 below.

A second situation is the case when the input vector refers to the model space coordinate system and the output vector refers to a viewing coordinate system. In this case, the matrix R is referred to as a view matrix, and is subject to the restrictions given in Form 0 below. Note that when a planar entity is viewed at true length (i.e., the viewing plane is parallel to the plane containing the entity) then the rotation matrix pointed to by DE Field 7 of the planar entity will be the inverse (=matrix transpose) of the matrix pointed to by DE Field 7 of the View entity. (See Sect. 4.3.11)

A third situation involves finite element modeling applications. Here, it may be the case that an input coordinate system is related to an output coordinate system by a particular R and T, and, in turn, the output coordinate system is then taken as an input coordinate system for a second R and T combination, and so on. These coordinate systems are frequently called local coordinate systems. Model space is frequently called the reference system. For example, the location of a finite element node may be given in one local coordinate system, which may serve as the input coordinate system for a second local coordinate system, which in turn serves as the input coordinate system for the model space coordinate system which is the reference system. Allowable forms of the matrix R for these applications are detailed in Forms 10, 11, and 12 below.

3.14.3   Whenever coordinate systems are related successively to each other as described above, a basic result is that the combined effect of the individual coordinate system changes can be expressed in terms of a single matrix R and a single translation vector T. For example, if the coordinate system change involving the matrix R2 and the translation vector T2 is to be applied following the coordinate system change involving the matrix R1 and the translation vector T1, then the matrix R and the translation vector T expressing the combined changes are R=(R2) (R1) and T = (R2) (T1) + T2.

Here, (R2) (R1) denotes matrix multiplication of 3x3 matrices, where multiplication order is important. The matrix R and the translation vector T are computed similarly whenever more than two coordinate system changes are to be applied successively.

Successive coordinate system changes are specified by allowing a Transformation Matrix entity to reference another Transformation Matrix entity through Field 7 of the Directory Entry. In the example above, the Transformation Matrix entity containing R1 and T1 would contain in its Directory Entry Field 7 a pointer to the Transformation Matrix entity containing R2 and T2. The general rule is that Transformation Matrix entities applied earlier in a succession will reference Transformation Matrix entities applied later. Note that the matrix product (R2) (R1) in the example above does not appear explicitly in the data, but, if needed, must be computed according to the usual rules of matrix multiplication.

A second example of coordinate systems being related successively (or "concatenated", or "stacked"), in addition to the finite element example mentioned above, involves one manner of locating into model space a conic arc that is in standard position in definition space. In this case, R1 and T1 move the conic arc from its standard position to an arbitrary location in any plane in definition space satisfying ZT=constant. (Therefore, $R1_{33}=1.0$, $R1_{31}=R1_{32}=R1_{13}=R1_{23}=0.0$. T1 can be an arbitrary translation vector.) R2 and T2 then position the relocated conic arc into model space. (R2 can be an arbitrary defining matrix and T2 can be an arbitrary translation vector.) Note that for R1 and T1, both the input vector and the output vector refer to the same coordinate system, namely, the definition space for the conic arc.

3.14.4   A 3x3 matrix R is called orthogonal provided its transpose, R^t, about the main diagonal yields a matrix inverse for R. The columns of an orthogonal matrix considered as vectors form an orthogonal collection of unit vectors. As $(R^t)^t=R$, the transpose of an orthogonal matrix is again an orthogonal matrix. The determinant of an orthogonal matrix is equal to either plus one or minus one. In the event R is an orthogonal matrix with determinant equal to positive one, R can be expressed as a rotation about an axis passing through the origin. In this event, R is referred to as a rotation matrix. In the event R is an orthogonal matrix with determinant equal to negative one, R can be expressed as a rotation about an axis passing through the origin followed by a reflection about a plane passing through the origin perpendicular to the axis of rotation.

3.14.5 Allowable Form Numbers. The defining matrix of an entity must use either Form zero or Form one. A defining matrix associated with a view entity must use Form zero.

Form 0: (default) R is an orthogonal matrix with determinant equal to positive one. T is arbitrary. The columns of R taken in order form a right-handed triple in the output coordinate system.

Form 1: R is an orthogonal matrix with determinant equal to negative one. T is arbitrary. The columns of R taken in order form a left-handed triple in the output coordinate system.

3.14.6 Forms 10, 11, 12. These form numbers indicate special matrices used in conjunction with the node entity (type number 134).

Form 10: This form number conveys special information when used in conjunction with the Node entity (type number 134) in Finite Element Applications.

Refer to Fig. 3-17(a) for notation. The matrix R and the vector T are used to transform coordinate data from the u1,u2,u3 coordinate system to the x,y,z local system.

The u1,u2,u3 coordinate system has its origin at an arbitrary fixed point col XOFFSET, YOFFSET, ZOFFSET in the x,y,z coordinate system and is assumed to be displaced parallel to that reference coordinate system. Thus,

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad T = \begin{bmatrix} XOFFSET \\ YOFFSET \\ ZOFFSET \end{bmatrix}$$

(a) Cartesian

(b) Cylindrical

(c) Spherical
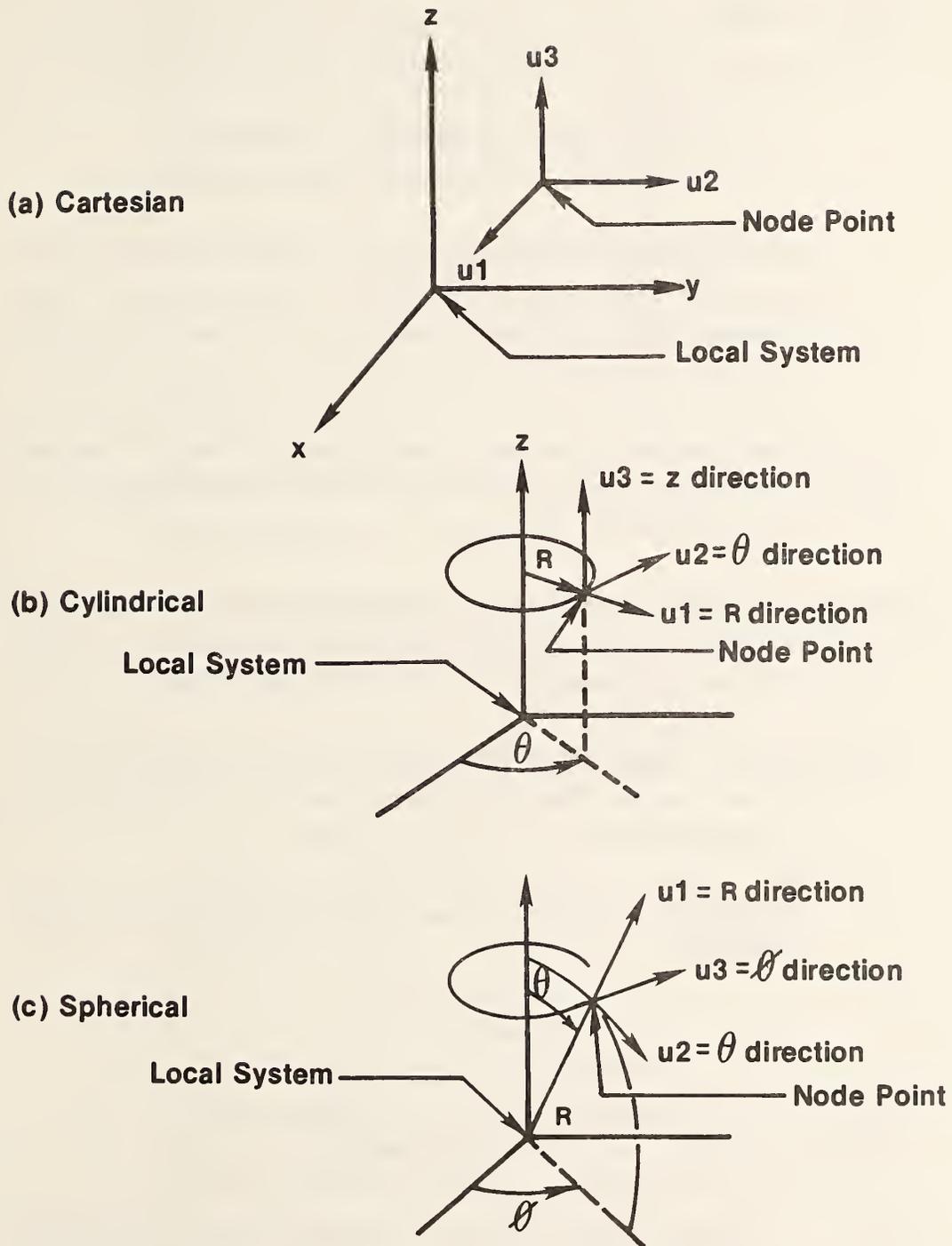
Figure 3-17 Displacement Components

so that

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix} + \begin{bmatrix} XOFFSET \\ YOFFSET \\ ZOFFSET \end{bmatrix} = \begin{bmatrix} XLOCAL \\ YLOCAL \\ ZLOCAL \end{bmatrix}$$

Note that the orientation of the two coordinate systems can be described by saying that the u1,u2,u3 coordinate system is the system obtained by imposing orthogonal curvilinear coordinates onto the x,y,z space and then constructing unit tangent vectors to the three curvilinear coordinate curves at the given fixed point to serve as basis vectors. In this special case of parallel displacement, the curvilinear coordinates imposed are identical to the existing x,y,z coordinates.

Form 11: This form number conveys special information when used in conjunction with the Node entity (type number 134) in Finite Element applications.

Refer to Figure 3-17(b) for notation. The matrix R and the vector T are used to transform coordinate data from the u1, u2, u3 coordinate system to the x,y,z local system.

The u1, u2, u3 coordinate system has its origin at an arbitrary fixed point

$$XOFFSET = r_0 \cos \theta_0 \qquad r_0 \quad 0$$
$$YOFFSET = r_0 \sin \theta_0 \qquad 0 \quad \theta_0 \quad 360^0$$
$$ZOFFSET = z_0 \qquad -\infty < z_0 < \infty$$
$$\text{for } r_0 = 0, \text{ take } \theta = 0^0$$

in the x,y,z coordinate system. The u1,u2,u3 system is the system obtained by imposing orthogonal curvilinear coordinates onto the x,y,z space which are the cylindrical coordinates (r,θ,z) with

$$x = r \cos \theta$$
$$y = r \sin \theta$$
$$z = z,$$

164

and then constructing unit tangent vectors to the three curvilinear coordinate curves at the given fixed point to serve as basis vectors.

Thus, the relationship between the u1, u2, u3 and the x,y,z local coordinate system is given by

$$\begin{bmatrix} \cos\theta_0 & -\sin\theta_0 & 0 \\ \sin\theta_0 & \cos\theta_0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix} + \begin{bmatrix} XOFFSET \\ YOFFSET \\ ZOFFSET \end{bmatrix} = \begin{bmatrix} XLOCAL \\ YLOCAL \\ ZLOCAL \end{bmatrix}$$

Form 12:   This form number conveys special information when used in conjunction with the Node entity (type number 134) in Finite Element applications.

Refer to Fig. 3-17(c) for notation. The matrix R and the vector T are used to transform coordinate data from the u1, u2, u3 coordinate system to the x,y,z local system.

The u1, u2, u3 coordinate system has its origin at an arbitrary fixed point

$$XOFFSET = r_0 \sin\theta_0 \sin\emptyset_0 \qquad\qquad r_0 \geqslant 0$$
$$YOFFSET = r_0 \sin\theta_0 \cos\emptyset_0 \qquad\qquad 0 \leqslant \theta_0 \leqslant 180^\circ$$
$$ZOFFSET = r_0 \cos\theta_0 \qquad\qquad 0 \leqslant \emptyset_0 < 360^\circ$$
$$\text{for } r_0 = 0, \text{ take } \theta_0 = \emptyset_0 = 0^\circ$$
$$\text{for } \theta_0 = 0^\circ \text{ or } 180^\circ, \text{ take } \emptyset_0 = 0^\circ$$

in the x,y,z coordinate system. The u1, u2, u3 system is the system obtained by imposing orthogonal curvilinear coordinates onto the x,y,z space which are the spherical coordinates $(r, \theta, \emptyset)$ with

$$X = r \sin\theta \cos\emptyset$$
$$Y = r \sin\theta \sin\emptyset$$
$$Z = r \cos\theta$$

and then constructing unit tangent vectors to the three curvilinear coordinate curves at the given fixed point to serve as basis vectors.

Thus, the relationship between the u1, u2, u3 and the x,y,z local coordinate systems is given by

$$\begin{bmatrix} \sin\theta_O \cos\phi_O & \cos\theta_O \cos\phi_O & -\sin\phi_O \\ \sin\theta_O \sin\phi_O & \cos\theta_O \sin\phi_O & \cos\phi_O \\ \cos\theta_O & -\sin\theta_O & 0 \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix} + \begin{bmatrix} XOFFSET \\ YOFFSET \\ ZOFFSET \end{bmatrix} = \begin{bmatrix} XLOCAL \\ YLOCAL \\ ZLOCAL \end{bmatrix}$$

See, Kaplan, (KAPL52) or Hildebrand, (HILD76) for a discussion of orthogonal curvilinear coordinate systems.

3.14.7   **Directory Data**
ENTITY TYPE NUMBER: 124

3.14.8   **Parameter Data**

Index	Name	Type	Description
1	R11	Real	Top Row
2	R12	Real	
3	R13	Real	
4	T1	Real	
5	R21	Real	Second Row
6	R22	Real	
7	R23	Real	
8	T2	Real	
9	R31	Real	Third Row
10	R32	Real	
11	R33	Real	
12	T3	Real	

Additional Pointers as required (see 2.2.4.4.2).

3.15   **Flash Entity**

A flash entity is a point in the ZT=0 plane that locates a specific instance of a particular closed area. That closed area can be defined in one of two ways. First, it can be an arbitrary closed area defined by any entity capable of defining a closed area. The points of this entity must all lie in the ZT=0 plane. Second, it can be a member of a predefined set of flash shapes.

In the latter case, parameters 3 through 5 of the flash entity control the final size of the flash. Figure 3-18 indicates the usage of those parameters for the specific flash forms. Parameters 3 through 5 are ignored for form zero.

3.15.1   **Directory Data**
ENTITY TYPE NUMBER: 125

Form	Meaning
0	defined by referenced entity
1	circular
2	rectangle
3	donut
4	canoe

3.15.2   **Parameter Data**

Index	Name	Type	Description
1	X	Real	X reference of flash
2	Y	Real	Y reference of flash
3	DIM1	Real	First flash sizing parameter
4	DIM2	Real	Second flash sizing parameter
5	ROT	Real	Rotation of flash about reference point in radians
6	DE	Pointer	DE of referenced entity or zero

Additional Pointers as required (see 2.2.4.4.2).
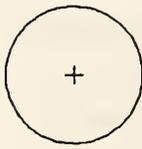
FORM 1 - CIRCULAR

DIMENSION 1 = DIAMETER OF CIRCLE
DIMENSION 2 = NULL OR ZERO
ROTATION = NULL OR ZERO

REFERENCE POINT IS CIRCLE CENTER

FORM 2 - RECTANGLE

DIMENSION 1 = X AXIS LENGTH BEFORE ROTATION
DIMENSION 2 = Y AXIS LENGTH BEFORE ROTATION
ROTATION = ANGLE IN RADIANS COUNTERCLOCKWISE
            FROM X AXIS TO DIMENSION 1

REFERENCE POINT IS CENTER OF RECTANGLE

FORM 3 - DONUT

DIMENSION 1 = DIAMETER OF OUTER CIRCLE
DIMENSION 2 = DIAMETER OF INNER CIRCLE
ROTATION = NULL OR ZERO

REFERENCE POINT IS CIRCLE CENTER
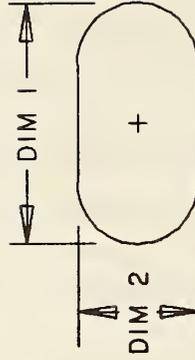
FORM 4 - CANOE

DIMENSION 1 = OVERALL LENGTH
DIMENSION 2 = OVERALL WIDTH
ROTATION = ANGLE IN RADIANS CCW
            FROM X AXIS TO DIMENSION 1

REFERENCE POINT IS CENTER OF CANOE

FIGURE 3-18 FLASH ENTITIES

3.16   <u>Rational B-Spline Curve Entity</u>

The rational B-spline curve may represent analytic curves of general interest. This information is important to both the sending and receiving systems. The directory entry form number parameter is provided to communicate this information. It should be emphasized that use of this curve form should be restricted to communications between systems operating directly on rational B-spline curves and not used as a replacement for the analytic forms for communication. For a brief description of a rational B-spline curves, see Section 4 of Appendix D.

If the rational B-spline curve represents a preferred curve type, the form number corresponds to the most preferred type. The preference order is from 1 through 5 followed by 0. For example, if the curve is a circle or circular arc, the form number is set to 2. If the curve is an ellipse with unequal major and minor axis lengths, the form number is set to 3. If the curve is not one of the preferred types, the form number is set to 0.

If the curve lies entirely within a unique plane, the planar flag (PROP1) is set to 1, otherwise it is set to 0. If it is set to 1, the plane normal (parameters 14+A+4K through 16+A+4K) contain a unit vector normal to the plane containing the curve. These fields exist but are ignored if the curve is non-planar.

If the beginning and ending points on the curve are identical, PROP2 is set to 1. If they are not equal, PROP2 is set to 0.

If the curve is rational (does not have all weights equal), PROP3 is set to 0. If all weights are equal to each other, the curve is polynomial and PROP3 is set to 1. The curve is polynomial since in this case all weights cancel and the denominator sums to one (see Appendix D4).

If the curve is periodic with respect to its parametric variable, set PROP4 to 1, otherwise set PROP4 to 0.

3.16.1 Directory Data

ENTITY TYPE NUMBER: 126

Form	Meaning
0	Form of curve must be determined from the rational B-spline parameters.
1	Line
2	Circular arc
3	Elliptical arc
4	Parabolic arc
5	Hyperbolic arc

3.16.2   Parameter Data

Index	Name	Type	Description
1	K	Integer	Upper index of sum. See Appendix D
2	M	Integer	Degree of basis functions
3	PROP1	Integer	=0 - non-planar =1 - planar
4	PROP2	Integer	=0 - open curve =1 - closed curve
5	PROP3	Integer	=0 - rational =1 - polynomial
6	PROP4	Integer	=0 - non-periodic =1 - periodic

Let $N=K-M+1$ and let $A=N+2M$

7	T(-M)	Real	Knot Sequence
.	.		
.	.		
.	.		
7+A	T(N+M)		
8+A	W(0)	Real	Weights
.	.		
.	.		
.	.		
8+A+K	W(K)		
9+A+K	XO	Real	Control Points
10+A+K	YO		
11+A+K	ZO		
.	.		
.	.		
.	.		
9+A+4K	XK		
10+A+4K	YK		
11+A+4K	ZK		
12+A+4K	V(0)	Real	Starting para-meter value
13+A+4K	V(1)	Real	Ending para-meter value
14+A+4K	XNORM	Real	Unit Normal (if curve is planar)
15+A+4K	YNORM		
16+A+4K	ZNORM		

Additional Pointers as required (see 2.2.4.4.2).

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

3.17   Rational B-Spline Surface Entity

The rational B-spline surface represents various analytical surfaces of general interest. This information is important to both the generating and receiving system. The directory entry Form Number parameter is provided to communicate such information. For a brief description of rational B-spline surfaces, see Section 6 of Appendix D..

If the rational B-spline surface represents a preferred surface type, the form number corresponds to the most preferred type. The preference order is from 1 through 9 followed by 0. For example, if the surface is a right circular cylinder, the form number is set to 2. If the surface is a surface of revolution and also a torus, the form number is set to 5. If the surface is not one of the preferred types, the form number is set to 0.

If, for each fixed value of the second parametric variable the resulting curves which are functions of the first parametric variable are closed, set PROP1 to 1; otherwise set PROP1 to 0. Similarly, if for each fixed value of the first parametric variable the resulting curves which are functions of the second parametric variable are closed, set PROP2 to 1; otherwise set PROP2 to 0.

If the surface is rational (does not have all weights equal) set PROP3 to 0. If all weights are equal to each other, the surface is polynomial and PROP3 is set to 1. The surface is polynomial since in this case all weights cancel and the denominator sums to one (see Appendix D4).

If the surface is periodic with respect to the first parametric variable, set PROP4 to 1; otherwise set PROP4 to 0. If the surface is periodic with respect to the second parametric variable, set PROP5 to 1; otherwise set PROP5 to 0.

3.17.1 Directory Data

ENTITY TYPE NUMBER: 128

Form	Meaning
0	Form of the surface must be determined from the rational B-spline parameters
1	Plane
2	Right circular cylinder
3	Cone
4	Sphere
5	Torus
6	Surface of revolution
7	Tabulated cylinder
8	Ruled surface
9	General quadric surface

3.17.2 Parameter Data

Index	Name	Type	Description
1	K1	Integer	Upper index of first sum. See Appendix D
2	K2	Integer	Upper index of second sum. See Appendix D
3	M1	Integer	Degree of first set of basis functions
4	M2	Integer	Degree of second set of basis functions
5	PROP1	Integer	=1 Closed in first parametric variable direction =0 Not closed
6	PROP2	Integer	=1 Closed in second parametric variable direction =0 Not closed
7	PROP3	Integer	=0 Rational =1 Polynomial
8	PROP4	Integer	=0 Non-periodic in first parametric variable direction =1 Periodic in first parametric variable direction

| 9 | PROP5 | Integer | =0 | Non-periodic in second parametric variable direction |
| | | | =1 | Periodic in second parametric variable direction |

Let $N1 = K1 - M1 + 1$, $N2 = K2 - M2 + 1$,
$A = N1 + 2M1$, $B = N2 + 2M2$, $C = (K1 + 1)(K2 + 1)$

10	S(-M1)	Real	First knot sequence
•	•		
•	•		
•	•		
10+A	S(N1+M1)		
11+A	T(-M2)	Real	Second knot sequence
•	•		
•	•		
•	•		
11+A+B	T(N2+M2)		
12+A+B	W(0,0)	Real	Weights
13+A+B	W(1,0)		
•	•		
•	•		
•	•		
11+A+B+C	W(K1,K2)		
12+A+B+C	X(0,0)	Real	Control Points
13+A+B+C	Y(0,0)		
14+A+B+C	Z(0,0)		
15+A+B+C	X(1,0)		
16+A+B+C	Y(1,0)	Real	Control Points
17+A+B+C	Z(1,0)		
•	•		
•	•		
•	•		
9+A+B+4C	X(K1,K2)		
10+A+B+4C	Y(K1,K2)		
11+A+B+4C	Z(K1,K2)		

Index	Name	Type	Description
12+A+B+4C	U(O)	Real	Starting value for first parametric direction
13+A+B+4C	U(1)	Real	Ending value for first parametric direction
14+A+B+4C	V(O)	Real	Starting value for second parametric direction
15+A+B+4C	V(1)	Real	Ending value for second parametric direction

Additional Pointers as required (see 2.2.4.4.2).

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

3.18   Offset Curve Entity

The Offset Curve entity contains the data necessary to determine the offset of a given curve C. This entity points to the base curve to be offset and contains the offset distance and additional pertinent information. No restriction is placed on the entity types of curves. Any parametric curve may be offset.

It is the intent of this Specification to limit the applicability of offsets to curves which are planar and slope continuous. The offset curve lies in the plane which contains the base curve as follows:

let C denote a curve in definition space which is defined by $r = r(t)$;

let $T(t)$ denote the unit tangent at $r(t)$;
(See FAUX79)

let V be a unit vector normal to the plane which contains C.

Then the offset curve is a curve defined as:
$$O(t) = r(t) + f(s)*(V \times T(t)) \qquad TT1 \leqslant t \leqslant TT2$$

a) if FLAG = 1, a uniform offset distance,
$$f(s) = D1$$

b) if FLAG = 2, an offset distance varying linearly,

$$f(s) = D1 + (D2 - D1)*(s - TD1)/(TD2 - TD1) \text{ with}$$

case (i)   PTYPE = 1
s = arc length along r from $r(TT1)$ to $r(t)$,
$D1$ = the offset at arc length value TD1,
$D2$ = the offset at arc length value TD2
case (ii) PTYPE = 2
S = t,
$D1$ = the offset at parametric value TD1,
$D2$ = the offset at parametric value TD2

c) if FLAG = 3, an offset distance defined by a function,
f(s) is the NDIM-th cordinate function of the curve
pointed to by DE2, with

case (i) PTYPE = 1
    s = arc length along r from r(TT1) to r(t),
case (ii) PTYPE = 2
    s = t

Note that TT1 and TT2 must be chosen to be in the domain of
the base curve r(t).

3.18.1   **Directory Data**
ENTITY TYPE NUMBER: 130

3.18.2   **Parameter Data**

Index	Name	Type	Description
1	DE1	Pointer	Pointer to curve entity to be offset.
2	FLAG	Integer	Offset distance flag: 1 = Single value offset, uniform distance 2 = Offset distance varying linearly 3 = Offset distance as a specified function.
3	DE2	Pointer or 0	Pointer to curve, one coordinate of which describes the offset as a function of its parameter. (0 unless FLAG = 3)
4	NDIM	Integer	Pointer of particular coordinate of DE2 which describes offset as a function of its parameter. (only used if FLAG = 3)

Index	Name	Type	Description
5	PTYPE	Integer	Tapered offset type flag: 1 = Function of arc length 2 = Function of parameter (only used if FLAG=2 or 3)
6	D1	Real	First offset distance. (only used if FLAG=1 or 2)
7	TD1	Real	Arc length or parameter value, depending on PTYPE, of first offset distance. (only used if FLAG=2)
8	D2	Real	Second offset distance.
9	TD2	Real	Arc length or parameter value, depending on PTYPE, of second offset distance. (only used if FLAG=2)
10	VX	Real	X-component of unit vector normal to plane containing curve to be offset.
11	VY	Real	Y-component of unit vector normal to plane containing curve to be offset.
12	VZ	Real	Z-component of unit vector normal to plane containing curve to be offset.
13	TT1	Real	Offset curve starting parameter value.
14	TT2	Real	Offset curve ending parameter value.

Additional Pointers as required (see 2.2.4.4.2).

Parameter data not required for a particular case should be given zero values. For example, if the value of parameter 2 is not 3, then parameters 3 and 4 should be given zero values.

178

3.19   **Connect Point Entity**

A Connect Point Entity describes a point of connection for zero, one or more entities. These entities include those required in piping diagrams, electrical and electronic schematics, and physical designs (e.g., printed wiring board). The Connect Point Entity is referenced from either the Network Subfigure Definition, Network Subfigure Instance, or the Flow Associativity Instance; or it may stand alone in a file. The connect point may be displayed by the receiving system using default display parameters and/or symbol. Also see 2.5.2.

3.19.1   **Directory Data**
ENTITY TYPE NUMBER: 132

3.19.2   **Parameter Data**

Index	Name	Type	Description
1	X	Real	X coordinate of the connection point
2	Y	Real	Y Coordinate of the connection point
3	Z	Real	Z coordinate of the connection point
4	PTR	Pointer	Pointer to directory entry of the display symbol geometry

132 - CONNECT POINT

Index	Name	Type	Description
5	TF	Integer	Type flag 0 = not specified 1 = logical 2 = physical
6	FF	Integer	Function Flag: 0 = not specified 1 = electrical signal 2 = fluid flow path
7	CID	String	Connect Point Function Identifier (e.g., Pin Number or Nozzle Label)
8	PTTCID	Pointer	Pointer to Text Display Template Entity for CID.
9	CFN	String	Connection Point Function Name
10	PTTCFN	Pointer	Pointer to Text Display Template Entity for CFN
11	CPID	Integer	Unique Connect Point Identifier
12	FC	Integer	Connect Point Function Code: 0 = Unspecified (default) 1 = Input 2 = Output 3 = Bidirectional 4 = Power 5 = Ground
13	SF	Integer	Swap Flag 0 = Connect point may be swapped (default) 1 = Connect point may not be swapped
14	PSFI	Pointer	Pointer to "owner" Network Subfigure Instance, or Network Subfigure Definition, or zero.

Additional Pointers as required (see 2.2.4.4.2).

3.20   Node Entity

The node entity is a geometric point used in the definition of a finite element. Directory entry field 7 points to a labeled definition coordinate system Transformation Matrix. The form number of the Transformation Matrix indicates the definition coordinate system type. Coordinate angles for the cylindrical and spherical coordinate systems are specified in degrees.

3.20.1   Every node has a nodal displacement coordinate system associated with it. This is form 10, 11, or 12 of the Transformation Matrix Entity which locates translational and rotational directions for load, restraint and displacement results. Again, the form number of the Transformation Matrix indicates the coordinate system type.

The origin of the nodal displacement coordinate system is always the location of the node. However, the orientation of the nodal displacement axes depends on the location of the node and the type of displacement coordinate system being referenced. Cartesian (Rectangular), cylindrical and spherical are the three possible types.

If the displacement coordinate system is Cartesian, then the nodal displacement axes are parallel to the respective referenced coordinate system. This is illustrated in Figure 3-19(a) Cartesian.

For the cylindrical type displacement coordinate system, the orientation of the nodal displacement axes depends on the coordinate value of the node as defined in the referenced displacement coordinate system. The nodal displacement axes are respectively in the radial, tangential and axial directions as illustrated in Figure 3-19(b) Cylindrical.

Finally, for spherical, the orientation of the nodal displacement axes depend on both the θ and ϕ coordinates of the node as defined in the referenced displacement coordinate system. The nodal displacement axes are respectively in the radial, meridional and azimuthal directions as indicated in Figure 3-19(c) Spherical.

If a node lies on the polar axis of either the cylindrical or spherical coordinate system, the nodal displacement axes are defined parallel to the referenced displacement coordinate system axes. For cylindrical the first axis is the $\theta=0$ axis and the third axis is the z axis. For spherical the first axis is the $\emptyset=0$ axis while the third axis is the $\theta=0$ axis. The remaining axis of both systems is defined by the appropriate cross product of the previously defined axes.

3.20.2 **Directory Data**
ENTITY TYPE NUMBER: 134
    Entity Label: Node Label (Optional)
    Entity Subscript: Node Number (Required)

3.20.3 **Parameter Data**

Index	Name	Type	Description
1	X/R/R	Real	First nodal coordinate
2	Y/θ/θ	Real	Second nodal coordinate
3	Z/Z/$\emptyset$	Real	Third nodal coordinate
4	NDCSP	Pointer	Pointer to the Transformation Matrix form 10, 11 or 12 which defines the Nodal Displacement Coordinate System entity. Default (zero) is Global Cartesian Coordinate System.

Additional Pointers as required (see 2.2.4.4.2).

Figure 3-19 illustrates the definition of a node in the three coordinate systems.
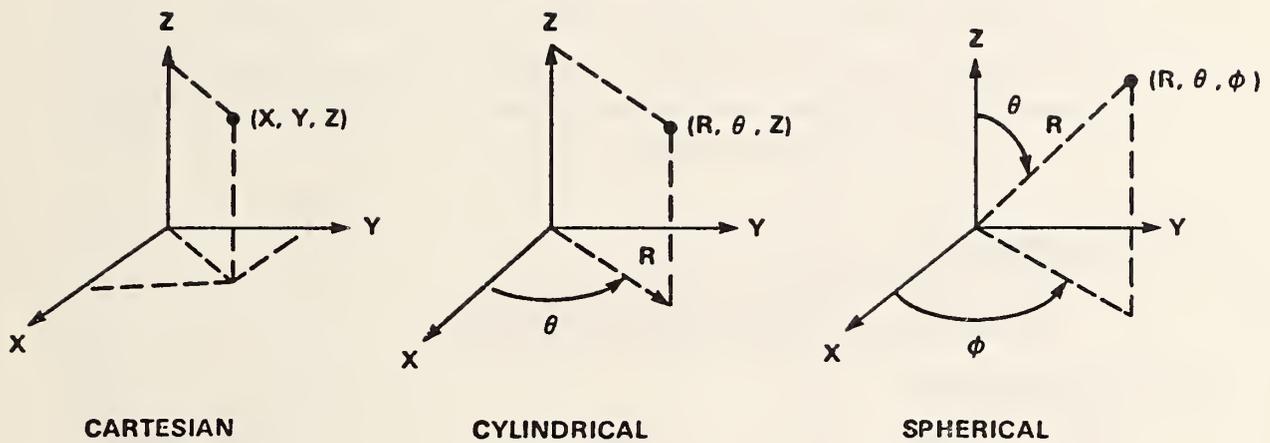
FIGURE 3-19 NODE DEFINITION IN EACH COORDINATE SYSTEM

3.21   **Finite Element Entity**

A finite element is defined by an element topology (i.e., node connectivity) along with physical and material properties.

3.21.1   Table 3-1 lists the data to define the element topology. Figure 3-20 illustrates the node connectivity for each element topology.

3.21.2   In Table 3-1 the element name is an English abbreviation or acronym describing the element. The element topology type is an integer number which will appear as the first parameter of the parameter data. The order is an integer identifying the order of an edge where 0=not applicable, 1=linear, 2=parabolic and 3=cubic. The number of nodes from Table 3-1 will appear as the second parameter of the finite element parameter data. A missing node in the connectivity sequence will have its corresponding pointer value equal to zero.

3.21.3   **Directory Data**
ENTITY TYPE NUMBER: 136
    Entity Label: Element Label (Optional)
    Entity Subscript: Element Number (Required)

3.21.4   **Parameter Data**

Index	Name	Type	Description
1	ITOP	Integer	Topology Type.
2	N1	Integer	Number of Nodes defining Element. See 3.23.2.
3	DE	Pointer	Pointer to first node defining element. See 3.23.2.
.	.	.	
.	.	.	
.	.	.	
N1+2	DE	Pointer	Pointer to last node defining element
N1+3	ETYP	String	Element type name

Additional Pointers as required (see 2.2.4.4.2).

TABLE 3-1 FINITE ELEMENT TOPOLOGY

ELEMENT DATA CHART

Element Data

Element Name	Element Topology Type	Order	Number of Nodes	Number of Edges	Number of Faces
BEAM	1	1	2	1	0
LTRIA	2	1	3	3	1
PTRIA	3	2	6	3	1
CTRIA	4	3	9	3	1
LQUAD	5	1	4	4	1
PQUAD	6	2	8	4	1
CQUAD	7	3	12	4	1
PTSW	8	2	12	9	5
CTSW	9	3	18	9	5
PTS	10	2	16	12	6
CTS	11	3.	24	12	6
LSOT	12	1	4	6	4
PSOT	13	2	10	6	4
LSOW	14	1	6	9	5
PSOW	15	2	15	9	5
CSOW	16	3	24	9	5
LSO	17	1	8	12	6
PSO	18	2	20	12	6
CSO	19	3	32	12	6
ALLIN	20	1	2	1	0
APLIN	21	2	3	1	0
ACLIN	22	3	4	1	0
ALTRIA	23	1	3	3	0
APTRIA	24	2	6	3	0
ALQUAD	25	1	4	4	0
APQUAD	26	2	8	4	0
SPR	27	0	2	0	0
GSPR	28	0	1	0	0
DAMP	29	0	2	0	0
GDAMP	30	0	1	0	0
MASS	31	0	1	0	0
RBDY	32	0	2	0	0
TBEAM	33	1	3	1	0

FIGURE 3-20   FINITE ELEMENT TOPOLOGY SET

1.   BEAM
        E1=1,2
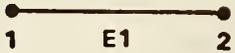
2.   LTRIA - Linear Triangle
        E1=1,2   F1=1,2,3
        E2=2,3
        E3=3,1
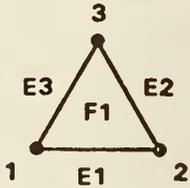
3.   PTRIA - Parabolic Triangle
        E1=1,2,3   F1=1,2,3,4,5,6
        E2=3,4,5
        E3=5,6,1
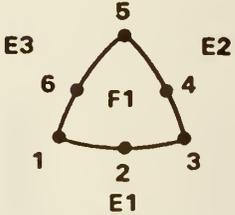
4.   CTRIA - Cubic Triangle
        E1=1,2,3,4   F1=1,2,3,4,5,6,7,8,9
        E2=4,5,6,7
        E3=7,8,9,1
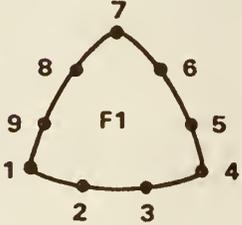
5.   LQUAD - Linear Quadrilateral
        E1=1,2   F1=1,2,3,4
        E2=2,3
        E3=3,4
        E4=4,1
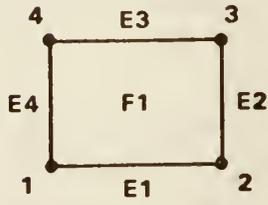
6.   PQUAD - Parabolic Quadrilaterial
        E1=1,2,3   F1=1,2,3,4,5,6,7,8
        E2=3,4,5
        E3=5,6,7
        E4=7,8,1
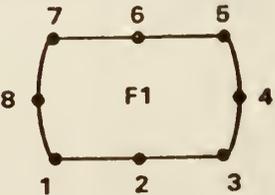
7. CQUAD - Cubic Quadrilateral
   E1=1,2,3,4
   E2=4,5,6,7
   E3=7,8,9,10
   E4=10,11,12,1

   F1=1,2,3,4,5,6,7,8,9,10,11,12
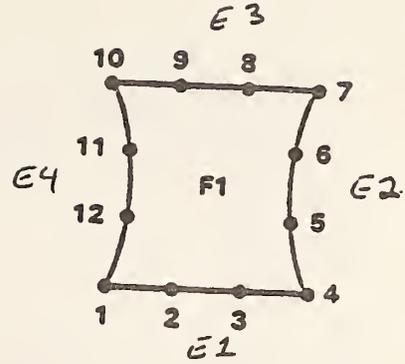
8. PTSW - Parabolic Thick Shell Wedge
E1=1,2,3	E4=7,8,9	E7=1,7
E2=3,4,5	E5=9,10,11	E8=3,9
E3=5,6,1	E6=11,12,7	E9=5,11

   F1=1,2,3,4,5,6
   F2=7,8,9,10,11,12
   F3=1,2,3,9,8,7
   F4=3,4,5,11,10,9
   F5=5,6,1,7,12,11
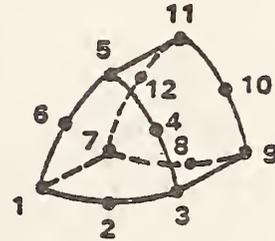
9. CTSW - Cubic Thick Shell Wedge
E1=1,2,3,4	E4=10,11,12,13	E7=1,10
E2=4,5,6,7	E5=13,14,15,16	E8=4,13
E3=7,8,9,1	E6=16,17,18,10	E9=7,16

   F1=1,2,3,4,5,6,7,8,9
   F2=10,11,12,13,14,15,16,17,18
   F3=1,2,3,4,13,12,11,10
   F4=4,5,6,7,16,15,14,13
   F5=7,8,9,1,10,18,17,16
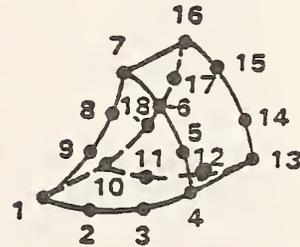
10. PTS - Parabolic Thick Shell
E1=1,2,3	E5=9,10,11	E9=1,9
E2=3,4,5	E6=11,12,13	E10=3,11
E3=5,6,7	E7=13,14,15	E11=5,13
E4-7,8,1	E8=15,16,9	E12=7,15

    F1=1,2,3,4,5,6,7,8
    F2=9,10,11,12,13,14,15,16
    F3=1,2,3,11,10,9   F5=5,6,7,15,14,13
    F4=3,4,5,13,12,11   F6=7,8,1,9,16,15
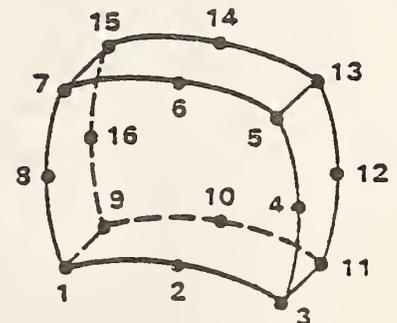
11. CTS - Cubic Thick Shell
E1=1,2,3,4	E5=13,14,15,16	E9=1,13
E2=4,5,6,7	E6=16,17,18,19	E10=4,16
E3=7,8,9,10	E7=19,20,21,22	E11=7,19
E4=10,11,12,1	E8=22,23,24,13	E12=10,22

    F1=1,2,3,4,5,6,7,8,9,10,11,12
    F2=13,14,15,16,17,18,19,20,21,22,23,24
    F3=1,2,3,4,16,15,14,13
    F4=4,5,6,7,19,18,17,16
    F5=7,8,9,10,22,21,20,19
    F6=10,11,12,1,13,24,23,22

187
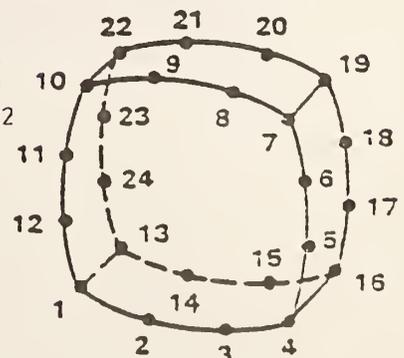
12. LSOT - Linear Solid Tetrahedron

E1=1,2   E4=1,4
E2=2,3   E5=2,4
E3=3,1   E6=3,4
F1=1,2,3
F2=1,2,4
F3=2,3,4
F4=3,1,4
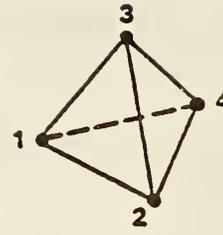
13. PSOT - Parabolic Solid Tetrahedron

E1=1,2,3   E4=1,7,10
E2=3,4,5   E5=3,8,10
E3=5,6,1   E6=5,9,10
F1=1,2,3,4,5,6
F2=1,2,3,8,10,7
F3=3,4,5,9,10,8
F4=5,6,1,7,10,9
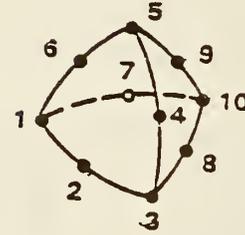
14. LSOW - Linear Solid Wedge

E1=1,2   E4=4,5   E7=1,4
E2=2,3   E5=5,6   E8=2,5
E3=3,1   E6=6,4   E9=3,6
F1=1,2,3
F2=4,5,6
F3=1,2,5,4
F4=2,3,6,5
F5=3,1,4,6
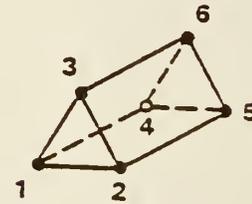
15. PSOW - Parabolic Solid Wedge

E1=1,2,3   E4=10,11,12   E7=1,7,10
E2=3,4,5   E5=12,13,14   E8=3,8,12
E3=5,6,1   E6=14,15,10   E9=5,9,14
F1=1,2,3,4,5,6
F2=10,11,12,13,14,15
F3=1,2,3,8,12,11,10,7
F4=3,4,5,9,14,13,12,8
F5=5,6,1,7,10,15,14,9
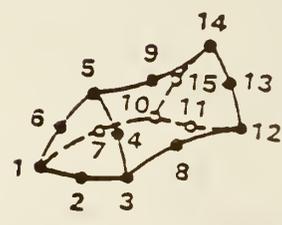
16. CSOW - Cubic Solid Wedge

E1=1,2,3,4   E4=16,17,18,19   E7=1,10,13,16
E2=4,5,6,7   E5=19,20,21,22   E8=4,11,14,19
E3=7,8,9,1   E6=22,23,24,16   E9=7,12,15,22
F1=1,2,3,4,5,6,7,8,9
F2=16,17,18,19,20,21,22,23,24
F3=1,2,3,4,11,14,19,18,17,16,13,10
F4=4,5,6,7,12,15,22,21,20,19,14,11
F5=7,8,9,1,10,13,16,24,23,22,15,12
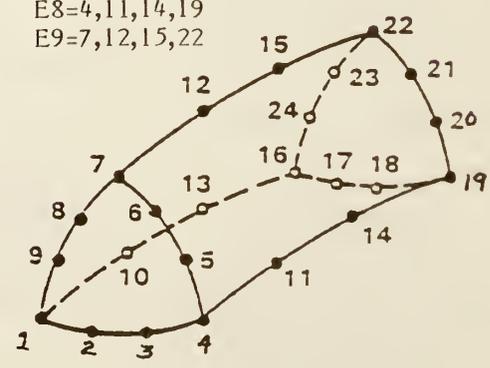
17.   LSO - Linear Solid

E1=1,2	E5=5,6	E9=1,5
E2=2,3	E6=6,7	E10=2,6
E3=3,4	E7=7,8	E11=3,7
E4=4,1	E8=8,5	E12=4,8

F1=1,2,3,4
F2=5,6,7,8
F3=1,2,6,5
F4=2,3,7,6
F5=3,4,8,7
F6=4,1,5,8
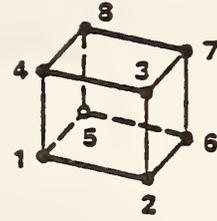
18.   PSO - Parabolic Solid

E1=1,2,3	E7=17,18,19
E2=3,4,5	E8=19,20,13
E3=5,6,7	E9=1,9,13
E4=7,8,1	E10=3,10,15
E5=13,14,15	E11=5,11,17
E6=15,16,17	E12=7,12,19

F1=1,2,3,4,5,6,7,8
F2=13,14,15,16,17,18,19,20
F3=1,2,3,10,15,14,13,9
F4=3,4,5,11,17,16,15,10
F5=5,6,7,12,19,18,17,11
F6=7,8,1,9,13,20,19,12
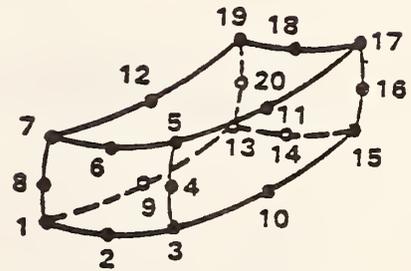
19.   CSO - Cubic Solid

E1=1,2,3,4
E2=4,5,6,7
E3=7,8,9,10
E4=10,11,12,1
E5=21,22,23,24
E6=24,25,26,27
E7=27,28,29,30
E8=30,31,32,21
E9=1,13,17,21
E10=4,14,18,24
E11=7,15, 19, 27
E12=10,16,20,30
F1=1,2,3,4,5,6,7,8,9,10,11,12
F2=21,22,23,24,25,26,27,28,29,30,31,32
F3=1,2,3,4,14,18,24,23,22,21,17,13
F4=4,5,6,7,15,19,27,26,25,24,18,14
F5=7,8,9,10,16,20,30,29,28,27,19,15
F6=10,11,12,1,13,17,21,32,31,30,20,16
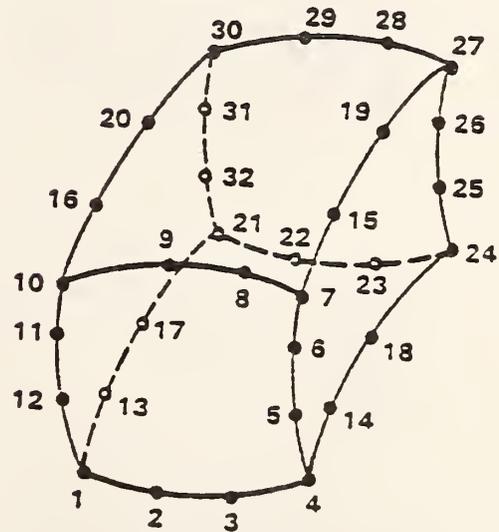
20.   ALLIN - Axisymmetric Linear Line
       E1=1,2   No Faces
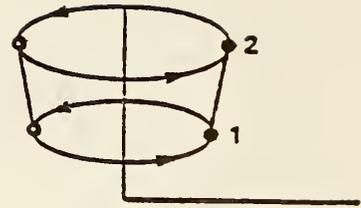
21.   APLIN - Axisymmetric Parabolic Line
       E1=1,2,3   No Faces
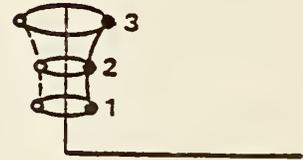
22.   ACLIN - Axisymmetric Cubic Line
       E1=1,2,3,4   No Faces
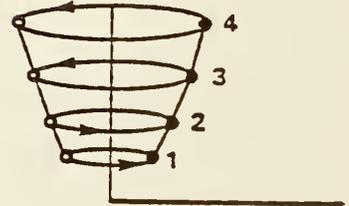
23.   ALTRIA - Axisymmetric Linear Triangle
       E1=1,2
       E2=2,3   No Faces
       E3=3,i
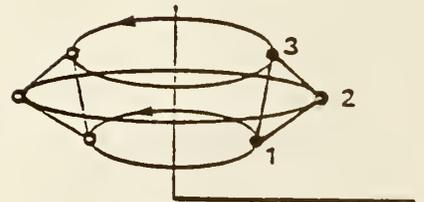
24.   APTRIA - Axisymmetric Parabolic Triangle
       E1=1,2,3
       E2=3,4,5   No Faces
       E3=5,6,1
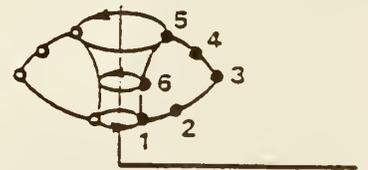
25.   ALQUAD - Axisymmetric Linear Quadrilateral
       E1=1,2
       E2=2,3
       E3=3,4   No Faces
       E4=4,1
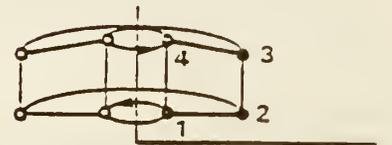
26.   APQUAD - Axisymmetric Parabolic Quadrilateral
       E1=1,2,3
       E2=3,4,5
       E3=5,6,7   No Faces
       E4=7,8,1
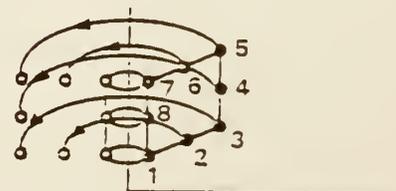
27.   SPR - Spring

No edges or faces

28.   GSPR - Grounded Spring

29.   DAMP - Damper

30.   GDAMP - Grounded damper

31.   MASS - Mass

32.   RBDY - Rigid Body

33.   TBEAM - three noded beam
(no faces)
E1 = 1,2

•3

1 •————————• 2

E1
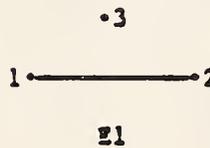
3.22   **Nodal Displacement and Rotation Entity**

The FEM Nodal Displacement and Rotation Entity is used to communicate finite
element post processing data. It contains the incremental displacements and
rotations expressed in radians for each load case and each node in the model. It
also contains a pointer to a general note entity for a description of the load
cases. For each node it contains the node number identifier and the node DE
pointer. The node number identifier is equivalent to the node number in the
directory entry subscript field of the node entity.

3.22.1   **Directory Data**
ENTITY TYPE NUMBER: 138

3.22.2   **Parameter Data**

Index	Name	Type	Description
1	NC	Integer	Number of analysis cases
2	GP1	Pointer	Pointer to general note that describes the first analysis case
.	.	.	.
.	.	.	.
.	.	.	.
1 + NC	GPNC	Pointer	Pointer to general note that describes the last analysis case
2 + NC	NN	Integer	Number of nodes
3 + NC	NO1	Integer	Node number identifier for first node
4 + NC	NP1	Pointer	Pointer to Node Directory Entry

Index	Name	Type	Description
5 + NC	X	Real	X-Incremental translation First Analysis case
6 + NC	Y	Real	Y-Incremental translation
7 + NC	Z	Real	Z-Incremental translation
8 + NC	R_X	Real	R_X-Incremental rotation
9 + NC	R_Y	Real	R_Y-Incremental rotation
10 + NC	R_Z	Real	R_Z-Incremental rotation
. . . 7*NC - 1	. . . X	Real	X-Incremental translation Last analysis case
.	Y	Real	Y-Incremental translation
.	Z	Real	Z-Incremental translation
.	R_X	Real	R_X-Incremental rotation
. . .	. . .		
3 + NC + (NN-1)*(6*NC+2)node NN	NONN	Integer	Node number identifier for NNth node
	NPNN	Pointer	Pointer to Node Directory Entry
	X	Real	X-Incremental translation First analysis case

193

138 - Nodal Displacement and
       Rotation

Index	Name	Type	Description
	Y	Real	Y-Incremental translation First analysis case
	Z	Real	Z-Incremental translation First analysis case
	R_X	Real	R_X-Incremental rotation First analysis case
	R_Y	Real	R_Y-Incremental rotation First analysis case
$10 + NC + (NN-1)*(6*NC+2)$	R_Z	Real	R_Z-Incremental rotation First analysis case
. .	. .		
.	X	Real	X-Incremental translation Last Analysis case
.	Y	Real	Y- " "
.	Z	Real	Z- " "
.	R_X	Real	R_X-Incremental rotation Last analysis case
.	R_Y	Real	R_Y " "
$2+NC + NN(6*NC+2)$	R_Z	Real	R_Z " "

Additional Pointers as required (see 2.2.4.4.2).

3.23   **Offset Surface Entity**

The offset surface is a surface defined in terms of an already existing surface.

1.   Let S=S(u,v) be a regular surface defined by this Specification parametrized and oriented by N(u,v), a differentiable field of unit normal vectors defined on the whole surface, and d a fixed non-zero real number. An offset surface to S is a parametrized surface S(u,v) given by:

$$O(u,v) = S(u,v) + d*N(u,v); \quad u1 \leqslant u \leqslant u2$$
$$v1 \leqslant v \leqslant v2$$

The base surface S(u,v) is referenced by a pointer in the parameter data section, while N(u,v) is found from S(u,v) as defined below. The value of d is provided as a parameter value in the parameter data section.

2.   To determine which one of the two orientations of the orientable regular surface S(u,v) the offset surface will be used to define O, define

$$N(u,v) \;=\; \frac{\dfrac{\partial S}{\partial u} \times \dfrac{\partial S}{\partial v}}{\left\| \dfrac{\partial S}{\partial u} \times \dfrac{\partial S}{\partial v} \right\|}$$

In order to avoid confusion connecting the orientation of the base surface S(u,v), an additional offset indicator is included. That indicator shown in Figure 3-21 consists of the vector (Nx, Ny, Nz) defined by

$$(Nx, Ny, Nz) = N(Um,Vm)/ \left\| N(Um,Vm) \right\|$$

(This is the unit normal vector at the parameter values (Um,Vm))

195

where, if the surface is bounded,

$$Um = \frac{U1 + u2}{2} \quad \text{and} \quad Vm = \frac{v1 + v2}{2}$$

or, if the surface is unbounded,

$$Um = 0.0 \quad \text{and} \quad Vm = 0.0$$

This indicates the direction in which the offset distance, d, is measured positive at (Um,Vm).

CAUTION: the vector (Nx,Ny,Nz) is just an indicator of the direction with respect to the base surface S(u,v) where the offset distance, d, is measured positively; this vector does not participate in the evaluation of the offset surface as is evident from formula for O that defines the offset surface.
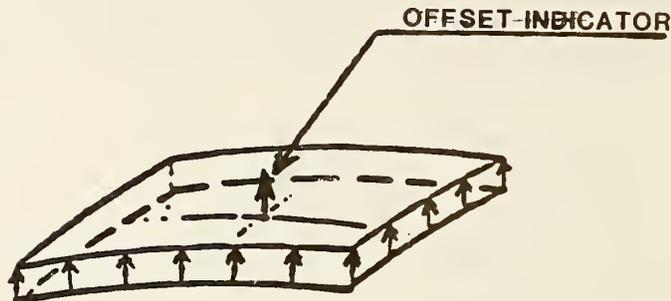
OFFSET-INDICATOR

FIGURE 3-21   OFFSET SURFACE IN 3-D EUCLIDEAN SPACE

3.23.1 <u>Directory Data</u>
ENTITY TYPE NUMBER: 140

3.23.2 <u>Parameter Data</u>

<u>Index</u>	<u>Name</u>	<u>Type</u>	<u>Description</u>
1	Nx	Real	The x-coordinate of the offset indicator N(Um, Vm).
2	Ny	Real	The y-coordinate of the offset indicator N(Um,Vm).
3	Nz	Real	The z-coordinate of the offset indicator N(Um,Vm).
4	d	Real	The distance by which the surface is normally offset on the side of the offset indicator if $d > 0$ and on the opposite side if $d < 0$.
5	DE	Pointer	Pointer to the surface entity to be offset.

Additional Pointers as required (see 2.2.4.4.2).

3.24   **Curve On A Parametric Surface Entity**

The Curve on a Parametric Surface entity associates a given curve with a surface and identifies the curve as lying on the surface.

Let S = S(u,v) = (x(u,v), y(u,v), z(u,v))
be a regular parametrized surface whose domain is a rectangle defined by

$$D = ((u,v) \mid u1 \leqslant u \leqslant u2 \text{ and } v1 \leqslant v \leqslant v2)$$

Let B = B(t) be a curve defined by

$$B(t) = (u(t),v(t)) \quad \text{for } a \leqslant t \leqslant b$$
taking its values in D.

A curve C(t) on the surface S(u,v) is the composition of two mappings, S and B defined as follows:

$$
\begin{aligned}
C(t) \quad &= \text{SoB}(t) \\
&= S(B(t)) \\
&= S(u(t), v(t)) \\
&= (x(u(t),v(t)) , y(u(t),v(t)) , z(u(t),v(t))) \quad a \leqslant t \leqslant b
\end{aligned}
$$

The curve B lies in the two dimensional space which is the domain of the surface S. Therefore, the representation used for B which has been derived from a curve defined in this Specification must be two dimensional: the X and Y coordinates of this curve pointed to by BPTR are used.

The "Entity Use Flag" (DE field 9) of the entity B is set to 05, indicating that B is in the parameter space of the surface. As a consequence of that, B cannot be scaled and if a transformation matrix is to be applied on B it has to map it within the parameter space D in which it resides.

Hence, a curve on a parametric surface is given by:

(a)   the mapping C and an indication that the curve lies on the surface S(u,v),

and

(b)   the mappings B and S whose composition gives the curve C.

A curve on a surface may have been created in one of a number of various ways:

(a)   As the projection on the surface of a given curve in model space in a prescribed way, for example, parallel to a given fixed vector.

(b)   As the intersection of two given surfaces.

(c)   By a prescribed functional relation between the surface parameters "u" and "v".

(d)   By a special curve, such as a geodesic, emanating from a given point in a certain direction, a principal curve (line of curvature) emanating from a certain point, an asymptotic curve emanation from a certain point, an isoparametric curve for a given value, or any other kind of special curve.

The parameter data section contains three pointers:

(a)   A pointer to the curve from which B(t) is derived;
(b)   A pointer to the surface S(u,v);
(c)   A pointer to the mapping C(t).

It also contains:

(d)   A flag to indicate how the curve was created;
(e)   A flag to indicate which of the two alternate representations was preferred by the sending system.

3.24.1 **Directory Data**
ENTITY TYPE NUMBER: 142

3.24.2 **Parameter Data**

Index	Name	Type	Description
1	CRTN	Integer	Indicates the way the curve on the surface has been created: 0 = Unspecified. 1 = Projection of a given curve on the surface. 2 = Intersection of two surfaces. 3 = Isoparametric curve, i.e. either a u-parametric or a v-parametric curve.
2	SPTR	Pointer	Pointer to the surface on which the curve lies.
3	B*PTR	Pointer	Pointer to the entity that contains the definition of the curve C* in the parametric space (u,v) of the surface S.
4	CPTR	Pointer	Pointer to the curve C.
5	PREF	Integer	Indicates preferred representation in the sending system: 0 = Unspecified. 1 = SoB* is preferred. 2 = C is preferred. 3 = C and SoB* are equally preferred.

Additional pointers as required (see 2.2.4.4.2).

3.25 Trimmed (Parametric) Surface Entity

A simple closed curve in the Euclidean plane divides the plane into two disjoint open connected components; one bounded and one unbounded. The bounded one is called the interior region to the curve (herein called, "interior"). The unbounded component is called the exterior region to the curve (herein called, "exterior").

The domain of the trimmed surface is defined as the common region of the interior of the outer boundary and the exterior of each of the inner boundaries and includes the boundary curves. Note that the trimmed surface has the same mapping S(u,v) as the original (untrimmed surface) but different domain. The curves that delineate either the outer or the inner boundary of the trimmed surface are curves on the surface S, and are to be exchanged by means of the Curve on a Parametric Surface (Entity Type 142).

Let S(u,v) be a regular parametrized surface, whose untrimmed domain is a rectangle D consisting of those points (u,v) such that $a \leqslant u \leqslant b$ and $c \leqslant v \leqslant d$ for given constants a, b, c, and d with $a < b$ and $c < d$. Assume that S takes its values in three dimensional Euclidean space so that it can be expressed as

$$S = S(u,v) = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix} \text{ for each ordered pair (u,v) in D.}$$

Also let the mapping S be subject to the following regularity conditions:

- It has continuous normal vector in the interior of D.
- It is one-to-one in D.
- There are no singular points in D, i.e., the vectors of the first partial derivatives of S at any point in D are linearly independent.

Two types of simple closed curves are utilized to define the domain of the trimmed (parametric) surface.

The first type is called the outer boundary. There is exactly one. It lies in D, and in particular, it can be the boundary curve of D.

The second type is called the inner boundary. There can be any number of them including zero. The set of inner boundaries satisfies two criteria:

1. The curves as well as their interiors are mutually disjoint.
2. Each curve lies in the interior of the outer boundary.

If the outer boundary of the surface being defined is the boundary of D and there are no inner boundaries, then the trimmed surface being defined is, in fact, untrimmed.

3.25.1   Directory Data
ENTITY TYPE NUMBER: 144

3.25.2   Parameter Data

Index	Name	Type	Description
1	PTS	Pointer	Pointer to the surface entity that is to be trimmed
2	N1	Integer	=0, if the outer boundary is the boundary of D. =1, otherwise.
3	N2	Integer	This number indicates the number of simple closed curves which constitute the inner boundary of the trimmed surface. In case no inner boundary is introduced, this is set equal to zero.
3+N1	PTO1	Pointer	Pointer to the simple closed curve (Curve on a Parametric Surface Entity), that constitutes the outer boundary of the trimmed surface.

Index	Name	Type	Description
4+N1	PTI1	Pointer	Pointer to the first simple closed inner boundary curve (Curve on a Parametric Surface entity) according to some arbitrary ordering of these entities.
			.
			.
			.
3+N1+N2	PTIN2	Pointer	Pointer to the last simple closed inner boundary curve (Curve on a Parametric Surface entity).

Additional pointers as required (see 2.2.4.4.2)

THIS PAGE LEFT BLANK

4   **NON-GEOMETRY**

4.1   <u>**General**</u>

This section contains capabilities for representing non-geometry and includes:

o   Annotation Entities

o   Structure Entities

Entity numbers from 200 through 499 are reserved for this Section. In addition, some non-geometric entities make use of entity type number 106 (copious data).

4.2   **Annotation Entities**

4.2.1   **Entity Type/Type Number.**

The following entities are defined in this section:

Entity Type Number	Entity Type
106	Copious Data
	Centerline
	Section
	Witness Line
202	Angular Dimension
206	Diameter Dimension
208	Flag Note
210	General Label
212	General Note
214	Leader (Arrow)
216	Linear Dimension
218	Ordinate Dimension
220	Point Dimension
222	Radius Dimension
228	General Symbol
230	Sectioned Area

4.2.2   **Construction.**

Many annotation entities are constructed by using other entities. For example, the dimension entities may have 0, 1, or 2 pointers to witness line entities (a form of copious data), 0, 1, or 2 pointers to leader (arrow) entitites and a pointer to a general note entity.

For some annotation entities, a witness line or leader may not exist. For these cases the parameter data field pointer value can be set zero. If any constructive entity exists, but its display is suppressed, it can be set to blank status or, if allowed, the pointer value can be set to zero.

4.2.3   **Definition Space.**

An annotation entity may be defined in XT, YT, ZT definition space (see the discussion in Section 3.1) or in a two-dimensional space associated with a Drawing Entity (type 404). In the case of XT, YT, ZT definition space, a transformation matrix is applied to locate the annotation entity within model space.

Within the XT, YT, ZT definition space, subordinate entities to an annotation entity may have different ZT displacements. For example, within the linear dimension, a different ZT value may be found in each of: general note, leader, and witness lines (which are pointed to in the linear dimension parameter data). An example showing the use of ZT displacement (DEPTH) is shown in Figure 4-1.

While the option of having dimensions occupy different planes exists, it is expected that only a single plane will be used. The reason for its existence is due to the structure of annotation entities. As each dimension may be comprised of several subordinate entities, each subordinate entity by its definition has the ability to stand alone and may require its own ZT displacement. When used in conjunction with other entities as a subordinate to a primary entity, it is likely, though not necessary, that each ZT is identical.

LENGTH OF LINE = DEPTH
(THE SIGN OF DEPTH INDICATES DIRECTION)
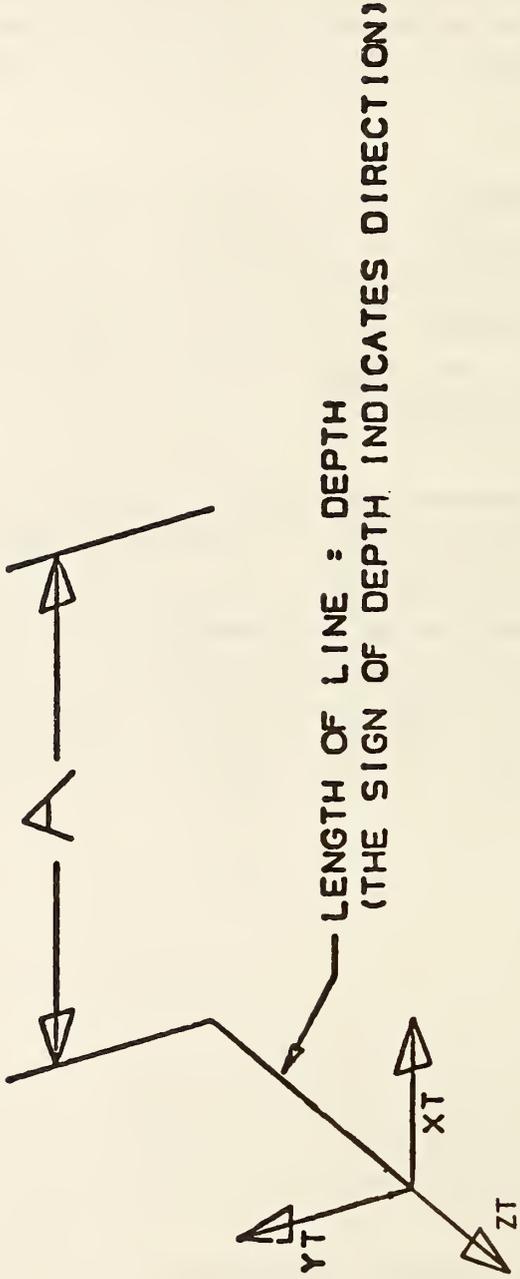
FIG. 4-1 CONSTRUCTION OF ZT. DEPTH OF ANNOTATION ENTITIES

4.2.4   **Angular Dimension Entity.**

An angular dimension entity consists of a general note; zero, one, or two witness lines; two leaders; and an angle vertex point. Figure 4-2 indicates the construction used. Refer to Figure 4-3 for examples of angular dimensions. If two witness lines are used, each is contained in its own copious data entity.

Each leader consists of at least one circular arc segment with an arrowhead at one end. The leader pointers are ordered such that the first circular arc segment of the first leader is defined in a counterclockwise manner from arrowhead to terminate point, and the first circular arc segment of the second leader is defined in a clockwise manner. (Refer to 3.1.2 for information relating to the use of the term counterclockwise).

4.2.4.1   Section 4.2.10 contains a discussion of multi-segment leaders. For those leaders in angular dimension entities consisting of more than one segment, the first two segments are circular arcs with a center at the vertex point. The second circular arc segment is defined in the opposite direction from the first circular arc segment. Remaining segments, if any, are straight lines. Any leader segment in which the start point is the same as the terminate point is to be ignored. This convention arises to facilitate the definition of the second circular arc segment such as in the bottom leader in Figure 4-2. Example 1 in Figure 4-3 illustrates a leader with three segments.

VERTEX

R

START POINT OF LEADER

SECOND AND THIRD POINTS OF LEADER

34.9411

FOURTH POINT OF LEADER
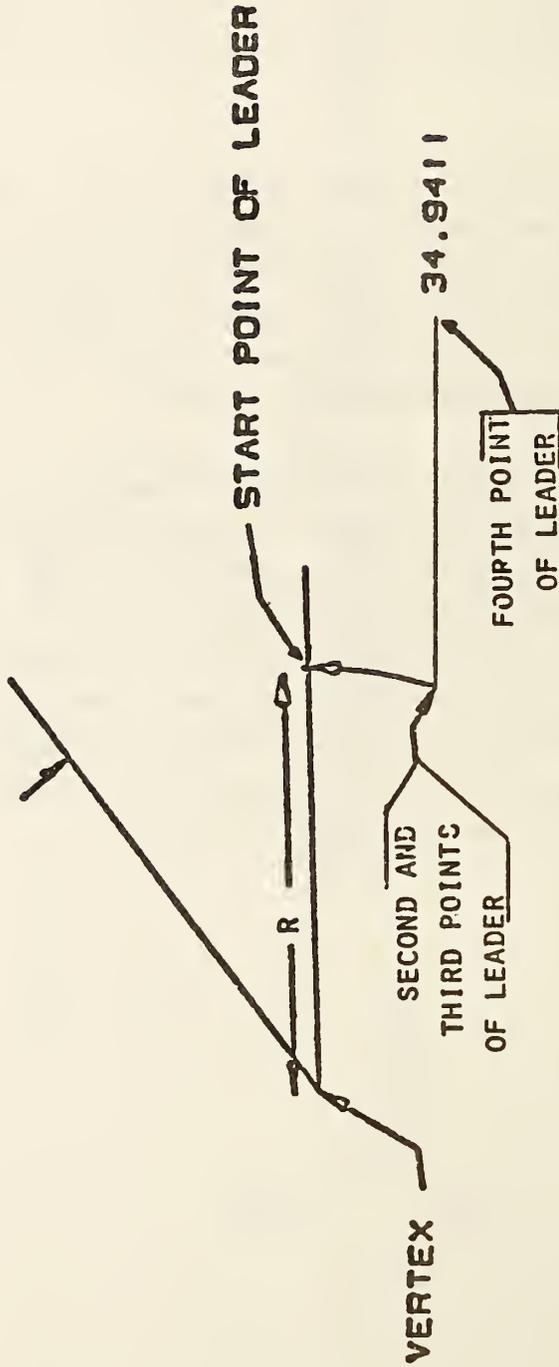
THE RADIUS OF THE ARC IN THE LEADER MUST
BE CALCULATED BETWEEN THE VERTEX POINT
AND THE START POINT OF THE LEADER

FIG. 4-2 ANGULAR DIMENSION: CONSTRUCTION OF ARCS IN
THE ASSOCIATED LEADERS

95.7919°

EXAMPLE 3

80.5627°

EXAMPLE 2

66.6463°
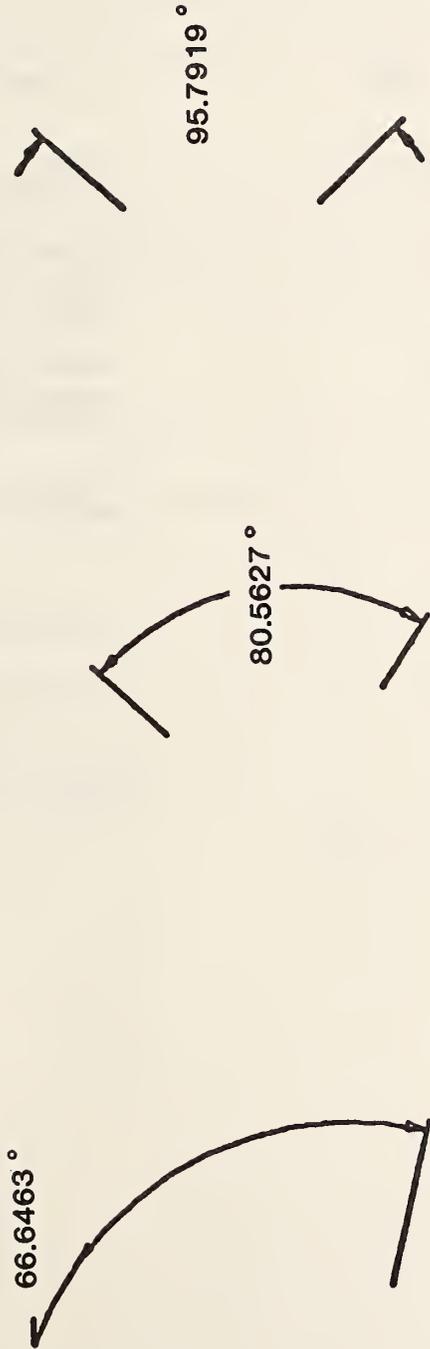
EXAMPLE 1

FIG. 4-3 EXAMPLES OF THE ANGULAR DIMENSION ENTITY

211

4.2.4.2   **Directory Data**
          ENTITY TYPE NUMBER :   202

4.2.4.3   **Parameter Data**

Index	Name	Type	Description
1	GN	Pointer	Pointer to general note directory entry
2	W1	Pointer	Pointer to first witness line directory entry or 0
3	W2	Pointer	Pointer to second witness line directory entry or 0
4	XT	Real	Coordinates of vertex point
5	YT	Real	
6	R	Real	Radius of leader arcs
7	A1	Pointer	Pointer to 1st leader directory entry
8	A2	Pointer	Pointer to 2nd leader directory entry

Additional Pointers as required (see 2.2.4.4.2).

4.2.5   **Centerline Entity**

The centerline entity takes one of two forms. The first, as illustrated in Figure 4-4, Examples 1 and 2, appears as crosshairs and is normally used in conjunction with circles. The second type (Example 3) is a construction between 2 positions.

The Centerline entities are defined as form 20 or 21 of copious data. The associated matrix transforms the XT-YT plane of the centerline into model space. The coordinates of the centerline points describe the centerline display symbol. The display symbol is described by line segments where each line is from

(X_n, Y_n, Z_n), to $(X_{n+1}, Y_{n+1}, Z_{n+1})$ where n = 1,3,5,...,N-1

4.2.5.1   See section 3.5 for parameters of the Centerline Entity.
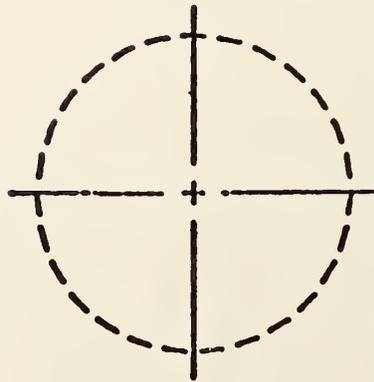
EXAMPLE 3
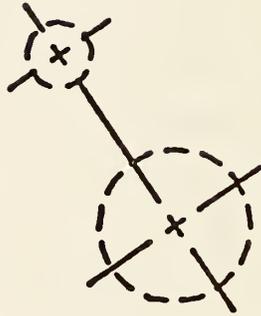
EXAMPLE 2

EXAMPLE 1

FIG. 4-4 EXAMPLES OF THE CENTERLINE ENTITY

4.2.6   Diameter Dimension Entity

A diameter dimension consists of a general note, one or two leaders, and an arc center point. Refer to Figure 4-5 for examples of the diameter dimension entity. The arc center coordinates are used for positioning the diameter dimension line relative to the arc being dimensioned.

4.2.6.1   Directory Data
ENTITY TYPE NUMBER :   206

4.2.6.2   Parameter Data

Index	Name	Type	Description
1	NE	Pointer	Pointer to general note directory entry
2	A1	Pointer	Pointer to first leader directory entry
3	A2	Pointer	Pointer to second leader directory entry or zero
4	XT	Real	Arc center coordinates
5	YT	Real	

Additional Pointers as required (see 2.2.4.4.2).

1.9673 DIA

EXAMPLE 3

1.6818 DIA

EXAMPLE 2

2.4165 DIA
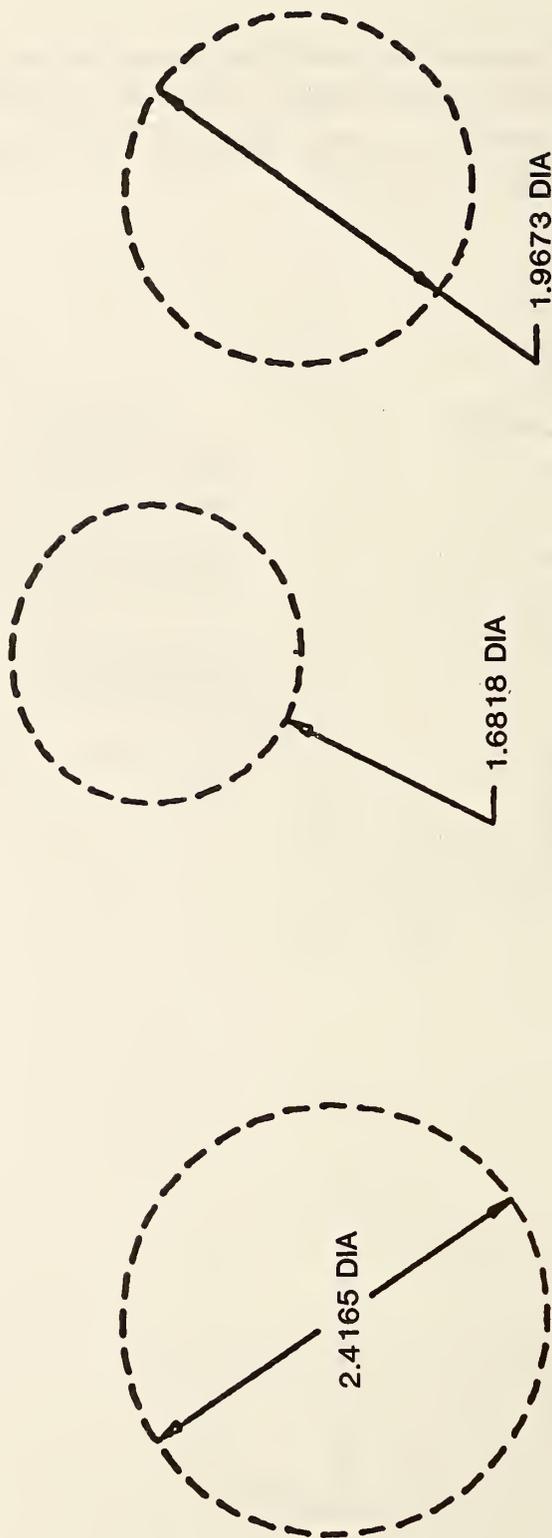
EXAMPLE 1

FIG. 4-5 EXAMPLES OF THE DIAMETER DIMENSION ENTITY

4.2.7   Flag Note Entity

A flag note entity is label information formatted as shown in Figure 4-6. The rotation angle overrides the general note rotation angle and placement. Additional examples of the flag note entity are shown in Figure 4-7.

The flag note entity may be defined with or without associated leaders.
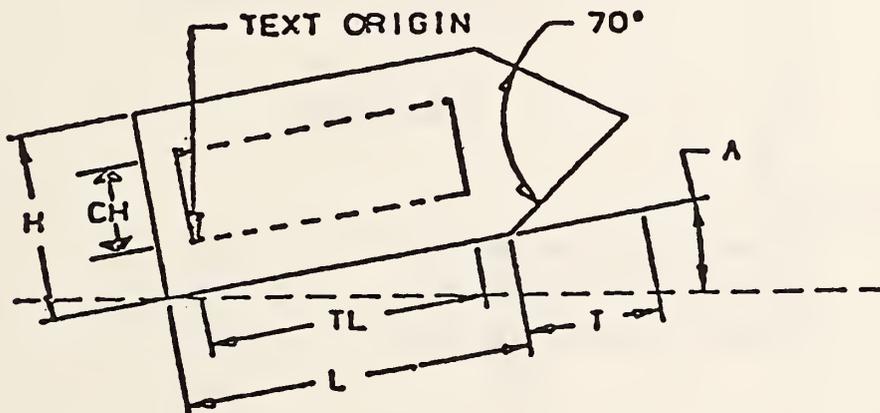
NOTE: Box outlined within flag illustrates bounds of text
and should not be interpreted as a sub-symbol.

Figure 4-6 FLAG NOTE

4.2.7.1   The flag note is constructed from information defined in the general note entity. This data is the character height and number of characters. For this reason, no geometric definition is explicit within the definition of the flag note entity.

The following specifications apply to Figure 4-6.

Variables:

H	=	Height	CH	=	Character height
L	=	Length	NC	=	Number of characters
TL	=	Text Length			(in general note)
T	=	Tip Length	A	=	Rotation angle in radians

Formulas:

TL	=	$(.8)(CH)(NC)+(.4)(CH)(NC-1)$
H	=	$(2)(CH)$
L	=	$(TL)+(.4)(CH)$
T	=	$(.5)(H)/TAN35^{\circ}$

Restrictions:

H shall never be less than .3 in.

L shall never be less than .6 in.

T shall never be less than .214 in.

FIG. 4-7 EXAMPLES OF THE FLAG NOTE ENTITY.
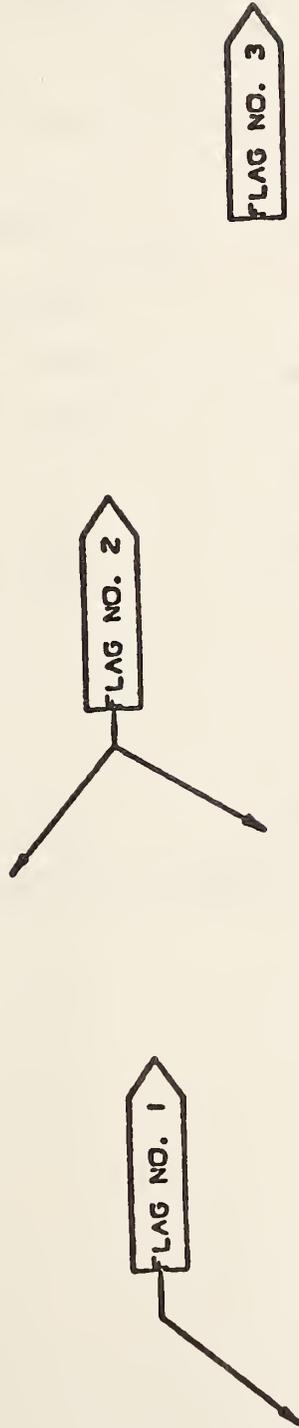
EXAMPLE 1

EXAMPLE 2

EXAMPLE 3

FLAG NO. 1

FLAG NO. 2

FLAG NO. 3

4.2.7.2 **Directory Data**
ENTITY TYPE NUMBER : 208

4.2.7.3 **Parameter Data**

Index	Name	Type	Description
1	XT	Real	Lower left corner coordinate of the flag
2	YT	Real	
3	ZT	Real	
4	A	Real	Rotation angle in radians
5	DENOTE	Pointer	Pointer to general note directory entry
6	N	Integer	Number of arrows (leaders) or zero
7	DE1	Pointer	Pointers to associated leaders
.			
.			
.			
6+N	DE1	Pointer	Pointer to last leader

Additional Pointers as required (see 2.2.4.4.2).

.

4.2.8   **General Label Entity.**

A general label entity consists of a general note with one or more associated leaders.

Examples of general label entities are shown in Figure 4-8

4.2.8.1   **Directory Data**
          ENTITY TYPE NUMBER :   210

4.2.8.2   **Parameter Data**

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to associated general note
2	N	Integer	Number of leaders
3	DE	Pointer	Pointers to associated leaders
.	.	.	
.	.	.	
.	.	.	
N+2	DE	Pointer	Pointer to last leader

Additional Pointers as required (see 2.2.4.4.2).
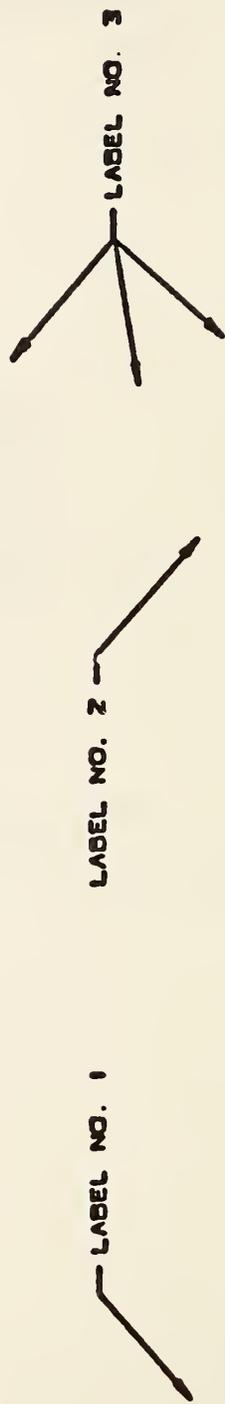
FIG. 4-8 EXAMPLES OF THE GENERAL LABEL ENTITY

4.2.9   General Note Entity.

A general note entity consists of one or more text strings. Each text string contains text, a starting point, text size, and angle of rotation of the text. Examples of general notes are shown in Figure 4-9. The FC value indicates the font number and is an integer. Positive values are pre-defined fonts. Negative values point to user defined fonts or modifications to a pre-defined font.

The following fonts will be defined:

0.	Symbol Font (use no longer recommended)
1.	Standard Block
2.	LeRoy
3.	Futura
4.	Fastfont
5.	Calcomp
6.	Comp 80
7.	Micro-Film Standard
8.	ISO Standard
9.	DIN Standard
10.	Military Standard
11.	Gothic
12.	News Gothic
13.	Lightline Gothic
14.	Simplex Roman
15.	Italic
16.	APL
17.	Century Schoolbook
18.	Helvetica
1001.	Symbol Font 1
1002.	Symbol Font 2

Fonts in the 1000 series display symbols mapped onto ASCII characters as shown in Figures 4 - 10 and 4 - 11. They do not specify a character display font.

A GENERAL NOTE
WITH TWO LINES

EXAMPLE 3

GENERAL NOTE ROTATED 195°

EXAMPLE 2

VERTICAL TEXT

EXAMPLE 5

GENERAL NOTE EXAMPLE

EXAMPLE 1

MIRRORED TEXT

EXAMPLE 4

FIG. 4-9 EXAMPLES OF THE GENERAL NOTE ENTITY

BLANK		0	0	@	©	P	P	`	\	p	℗
!	!	1	1	A	A	Q	Q	a	∠	q	¢
"	"	2	2	B	B	R	R	b	⌗	r	◎
#	#	3	3	C	C	S	S	c	▱	s	Ⓢ
$	$	4	4	D	D	T	T	d	⌢	t	⊓
%	%	5	5	E	E	U	U	e	○	u	⊙
&	&	6	6	F	F	V	V	f	//	v	△
'	'	7	7	G	G	W	W	g	⟋	w	◇
(	(	8	8	H	H	X	X	h	↗	x	⟁
)	)	9	9	I	I	Y	Y	i	≡	y	⟀
*	＊	:	:	J	J	Z	Z	j	⊕	z	Y
+	+	;	;	K	K	[	[	k	⌒	{	{
,	,	<	<	L	L	\	\	l	⊥	|	|
-	-	=	=	M	M	]	]	m	Ⓜ	}	}
.	.	>	>	N	N	^	^	n	∅	~	~
/	/	?	?	O	O	_	_	o	○		

FONT CODE 1001

FIGURE 4-10

225

BLANK		0	0	@	@	P	P	`	`	p	↑
!	!	1	1	A	A	Q	Q	∃	⊠	q	↓
▪	▪▪	2	2	B	B	R	R	b	÷	r	→
✱	±	3	3	C	C	S	S	c	≤	s	←
$	°	4	4	D	D	T	T	d	≥	t	φ
×	%	5	5	E	E	U	U	e	△	u	θ
&	&	6	6	F	F	V	V	f	√	v	Υ
·	◢	7	7	G	G	W	W	g	✕	∨	ψ
(	(	8	8	H	H	X	X	h	≡	×	ω
)	)	9	9	I	I	Y	Y	i	≠	y	λ
✳	✳	:	:	J	J	Z	Z	j	∫	z	α
+	+	;	;	K	K	[	[	k	⊃	(	δ
·	●	<	<	L	L	\	\	l	∨	¦	μ
-	—	●	=	M	M	]	]	m	∧	)	Π
·	●	>	>	N	N	^	^	n	≈	~	—
/	/	?	?	O	O	-	_	o	Σ		

FONT CODE 1002

FIGURE 4-11

Font 1 does not have a defined display. Use of Font 1 implies the receiving system may use any font which displays the appropriate ASCII characters. The intent of this font is for usage when the actual display of the characters is not critical for the application.

Font 0 is an old symbol font and should no longer be used. Figure 4 - 12 is a mapping symbol definition for font 0.

If the FC number is not sufficient to describe the font, a text font definition entity may be used to define the font. If a text font definition is being used, the negative of the pointer value for the directory entry of the text font definition entity is placed in the FC parameter. The use of the values WT, HT, SL, A, and text start point are shown in Figure 4-13.

Within definition space, the parameters for the text block are applied in the following order (See Figure 4-14):

1)   Define the box height (HT) and box width (WT).

      The rotate internal text flag indicates whether the text box is filled with horizontal text or vertical text. The box width is measured from the text start point in the positive XT direction and the box height is measured in the positive YT direction from the text start point, before the rotation angle (A) is applied.

2)   The slant angle is then applied to each individual character. For horizontal text it is measured from the XT axis in a counterclockwise direction. For vertical text the slant angle is measured from the YT axis.

3)   The rotation angle is then applied to the text block. This rotation is applied in a counterclockwise direction about the text start point. The plane of rotation is the XT, YT plane at the depth ZSn (where ZSn is the value given for the text start point).

Octal	Char	Octal	Char	Octal	Char	Octal	Char	Octal	Char	Octal	Char
0	⊠	26	ψ	54	.	102	B	130	X	156	∅
1	÷	27	ω	55	−	103	C	131	Y	157	O
2	≤	30	λ	56	.	104	D	132	Z	160	Ⓟ
3	≥	31	α	57	/	105	E	133	[	161	¢
4	△	32	δ	60	0	106	F	134	\	162	◎
5	√	33	μ	61	1	107	G	135	]	163	Ⓢ
6	×	34	π	62	2	110	H	136	∧	164	⊡
7	≡	35	—	63	3	111	I	137	_	165	⊙
10	≠	36	±	64	4	112	J	140	`	166	◬
11	∫	37	°	65	5	113	K	141	∠	167	◇
12	⊃	40		66	6	114	L	142	中	170	◬
13	∨	41	!	67	7	115	M	143	▱	171	⋈
14	∧	42	"	70	8	116	N	144	⌒	172	Y
15	≈	43	#	71	9	117	O	145	○	173	{
16	Σ	44	$	72	:	120	P	146	//	174	|
17	↑	45	%	73	;	121	Q	147	∅	175	}
20	↓	46	&	74	<	122	R	150	↗	176	~
21	→	47	'	75	=	123	S	151	≡	177	Ƶ
22	←	50	(	76	>	124	T	152	⊕		
23	φ	51	)	77	?	125	U	153	⌒		
24	θ	52	✳	100	©	126	V	154	⊥		
25	ϒ	53	+	101	A	127	W	155	Ⓜ		

FIGURE 4-12 CHARACTER SET & OCTAL CODE FOR
FONT CODE ZERO

4)   The mirror operation is performed next. The value 1 indicates the mirror axis is the (rotated) line perpendicular to the text base line and through the text start point. The value 2 indicates the mirror axis is the (rotated) test base line.

Finally, the Transformation Matrix entity is used to specify the relative position of definition space within model space.

The number of characters (NCn) must always be equal to the character count in its corresponding text string (TEXTn).
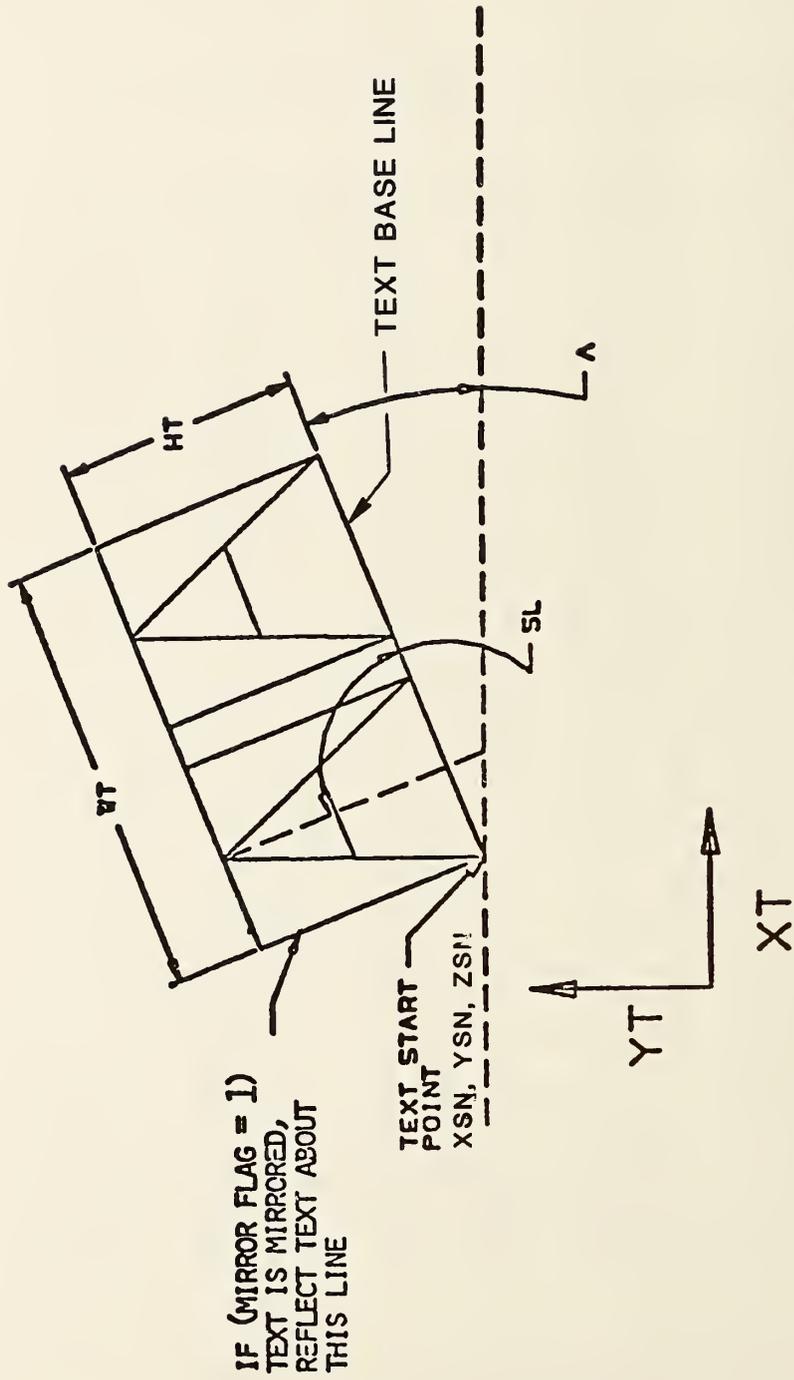
FIGURE 4-13 GENERAL NOTE TEXT CONSTRUCTION

HORIZONTAL TEXT | VERTICAL TEXT

DEFINE BOX

TEXT-START POINT

SLANT ANGLE
(π/2 EXAMPLE)

ROTATE
(3π/2 EXAMPLE)

MIRROR
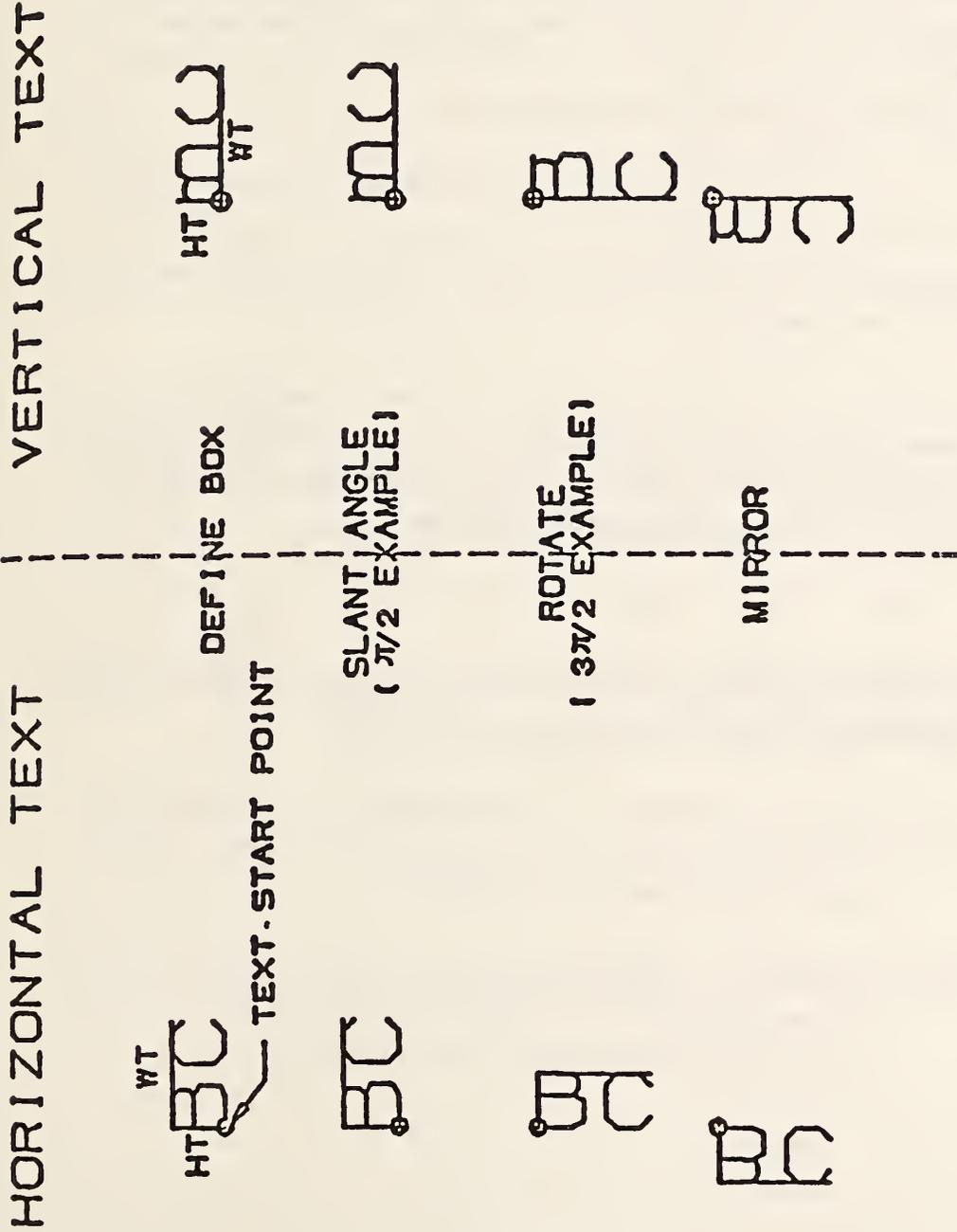
FIG. 4-14 GENERAL NOTE EXAMPLE OF TEXT OPERATIONS

4.2.9.1   The graphical representation and recreation of notes with a special structure are handled by the use of the Form Number in Field 15 of the Directory Entry for this entity. A system to accommodate these notes is outlined below. Any strings after those specified by the form number are considered additional, appended strings that are not related in any particular manner to the previously referenced strings.

4.2.9.2   In the event that a string necessary for the defined structure is not present in the originating system's note, a null string shall be inserted in the general note entity to take the place of the non-existent string to maintain the structure of the data.

4.2.9.3   Notes that contain fractional notation will be represented as mixed numerals. This is done through the use of four consecutive strings representing the whole number, the numerator, the denominator, and the divisor bar. These are examples of the divisor bar string:

    1H/   1H-   2H--   1H__

4.2.9.4   The following form numbers for the general note are used to maintain the graphical representation of the originating system's note:

4.2.9.4.1   Form 0: Simple Note (default) - A general note of one or more strings such that a text string is not related in any manner to another string in the same general note entity.

4.2.9.4.2   Form 1: Dual Stack - A general note of two or more strings where the first two are related in a manner such that they are both left justified and the second string is displayed 'below' the first.

            xxxxxx
            yyyyy

4.2.9.4.3   Form 2: Imbedded Font Change - A general note of two or more strings
                      that is intended as a single string but was divided to accommodate
                      a font change in the string.

                      xxxx yyy xxxx

4.2.9.4.4   Form 3: Superscript - A general note of two or more strings where the
                      second string is a superscript of the first string.

                      yyy^{xxx}

4.2.9.4.5   Form 4: Subscript - A general note of two or more strings where the
                      second string is a subscript of the first string.

                      ^{xxx}yyy

4.2.9.4.6   Form 5: Super-/Sub-script - A general note of three or more strings where
                      the second string is a superscript of the first string and the third
                      string is a subscript of the first string.

                      yyy$^{xxx}_{zzz}$

4.2.9.4.7   Form 6: Multiple Stack/Left Justified - A general note where all strings
                      are left justified to a common margin. These strings originated as
                      a "paragraphed" note.

                      xxxxxxxxxx
                      yyyyyyyyy
                      zzzzzzzzzz

4.2.9.4.8   Form 7: Multiple Stack/Center Justified - A general note where all strings
                      are center justified to a common axis.

                      xxxxxxx
                       yyyy
                      zzzzzzz

4.2.9.4.9   Form 8:   Multiple Stack/Right Justified - A general note where all strings are right justified to a common margin.

xxxxxxxxxx
  yyyyyy
 zzzzzzzz

4.2.9.4.10   Form 100:   Simple Fraction - A general note of four or more strings where the first four strings define a mixed numeral as defined in 4.2.9.3.

$$xx \frac{yyy}{zzz}$$

4.2.9.4.11   Form 101:   Dual Stack Fraction - A general note of eight or more strings which represent two mixed numerals as defined in 4.2.9.3.   These mixed numerals are related such that the fifth through the eighth strings are displayed below the first through the fourth strings respectively.

$$xx \frac{yy}{zz}$$

$$ii \frac{jj}{kk}$$

4.2.9.4.12   Form 102:   Imbedded Font Change/Double Fraction - This general note originated as a single string but was split to accommodate a font change for a special character in the fifth string. This is a general note of nine or more strings where the first and sixth strings represent the whole number string of a mixed numeral as defined in 4.2.9.3. The fifth string is a character (or characters) that was set apart to accommodate the font change.

$$xx \frac{yy}{zz} \ - \ ii \frac{jj}{kk}$$

4.2.9.4.13 Form 105: Super-/Subscript Fraction - A general note of twelve or more strings where the first, fifth, and ninth strings represent the whole number string of a mixed numeral as defined in 4.2.9.3. The second and third mixed numerals are the superscript and subscript respectively of the first mixed numeral.

$$ii\frac{jj}{kk}$$

$$xx\frac{yy}{zz}$$

$$rr\frac{ss}{tt}$$

4.2.9.5 **Directory Data**
ENTITY TYPE NUMBER :   212

4.2.9.6 **Parameter Data**

Index	Name	Type	Description
1	NS	Integer	Number of text strings in general note
2	NC1	Integer	Number of characters in first string (TEXT1) or zero. The number of characters (NCn) must always be equal to the character count of its corresponding text string (TEXTn)
3	WT1	Real	Box width
4	HT1	Real	Box height
5	FC	Integer or Pointer	Font characteristic (Default = 1)
6	SL1	Real	Slant angle of TEXT1 in radians (pi/2 is the value for no slant angle and is the default value)
7	A1	Real	Rotation angle in radians for TEXT1
8	M1	Integer	Mirror flag 0-no mirroring 1 - mirror axis is perpendicular to text base line 2 - mirror axis is text base line
9	VH1	Integer	Rotate internal text flag (0-text horizontal, 1-text vertical)
10	XS1	Real	First text start point
11	YS1	Real	

12	ZS1	Real	Z depth from XT, YT plane
13	TEXT1	String	First text string
14	NC2	Integer	Number of characters in second text string
1+NS*12	TEXTNS	String	Last text string
2+NS*12	NCNS	Integer	Number of characters in last text string

Additional Pointers as required (see 2.2.4.4.2)

4.2.10   **Leader (Arrow) Entity.**

A leader consists of one or more line segments except when the leader is part of an angular dimension (see 4.2.4). The first segment begins with an arrowhead. Remaining segments successively link to a presumed text item. An individual segment is assumed to extend from the end point of its predecessor in the segment list to its defined end point. Examples of leaders are shown in Figure 4-15.

In the use of angular, diameter, and linear dimension, there are instances where the text is exterior to the line or arc lying between the two arrows. In these situations, it remains the case that the appearance of two arrows implies the use of two leaders. These are formed by dividing the line or arc lying between the two arrows into two non-overlapping segments. Refer to Figure 4-16.

Some leaders (for example, the leader involved with the radius dimension in Figure 4-16) give the appearance of locating an arrow interior to a segment. There are two overlapping segments. The first segment begins at the arrow and, in the radius dimension example, ends at the center of the arc or circle being dimensioned. The second segment then retraces the first in the opposite direction and extends it. Leaders of this type for other types of dimensions are constructed similarly. For cases involving angular dimension, the first two segments are arcs.

4.2.10.1   Forms. There are eleven arrowhead types defined (see figure 4-17) and selection is made by entering the form number in directory entry field 15.

EXAMPLE 3

EXAMPLE 2

EXAMPLE 1
(ANGULAR DIMENSION)

FIG. 4-15 EXAMPLES OF THE LEADER ENTITY

239

EXAMPLES OF LEADERS INTERNAL TO A DIMENSION

.6238 R

1.3011 DIA

53.0209°

1.4920
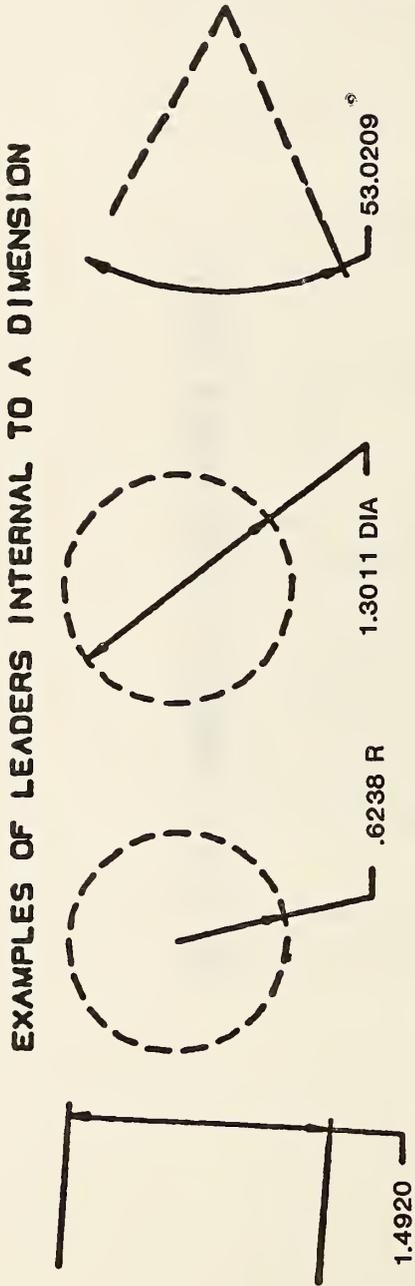
DISPLAY OF LEADER

THE LEADER WILL BE DIVIDED AS SHOWN BELOW

B   B   A   C

A
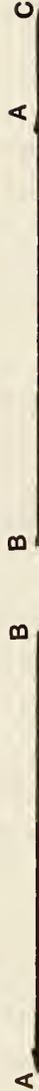
A – FIRST POINT OF INDIVIDUAL LEADERS

B – SECOND POINT (SAME COORDINATES FOR BOTH LEADERS)

C – THIRD POINT OF LEADER (FOLLOWED
    BY OTHER POINTS AS NECESSARY)

FIG. 4-16 STRUCTURE OF LEADERS INTERNAL TO A DIMENSION

FORM 1: WEDGE

FORM 2: TRIANGLE

FORM 3: FILLED TRIANGLE

FORM 4: NO ARROWHEAD

FORM 5: CIRCLE

FORM 6: FILLED CIRCLE

FORM 7: RECTANGLE

FORM 8: FILLED RECTANGLE

FORM 9: SLASH

FORM 10: INTEGRAL SIGN

FORM 11: OPEN TRIANGLE
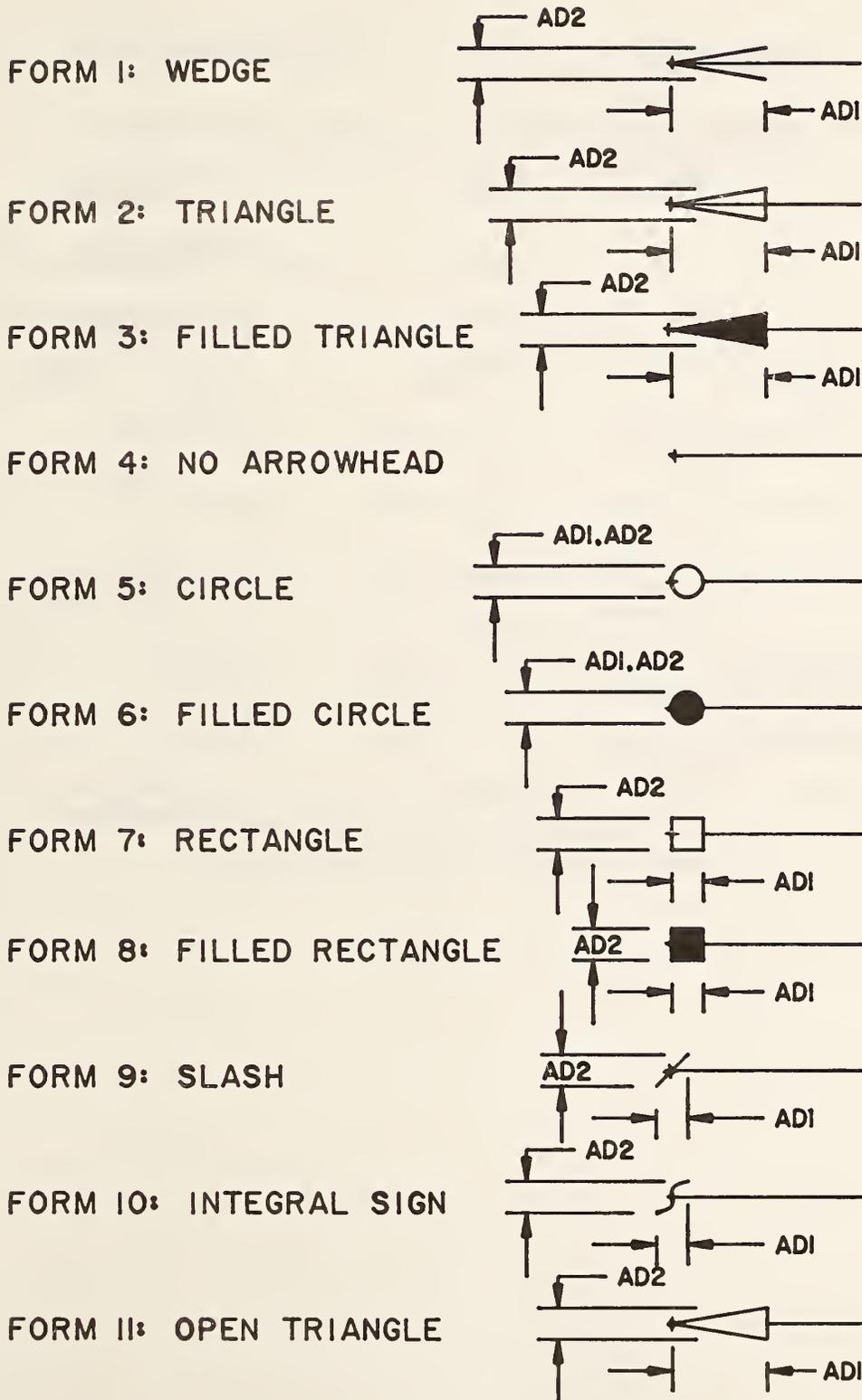
NOTE: • INDICATES ORIGIN (XH, YH) AND IS NOT PART OF ARROWHEAD

FIG. 4-17   ARROWHEAD DEFINITIONS

4.2.10.2   **Directory Data**
ENTITY TYPE NUMBER: 214

4.2.10.3   **Parameter Data**

Index	Name	Type	Description
1	N	Integer	Number of segments
2	AD1	Real	Arrowhead height
3	AD2	Real	Arrowhead width
4	ZT	Real	Z depth
5	XH	Real	Arrowhead coordinates
6	YH	Real	
7	X	Real	Segment tail coordinate pairs
.	Y	Real	
.			
.			
6+2N	Y	Real	Last Segment Coordinate

Additional Pointers as required (see 2.2.4.4.2).

4.2.11   **Linear Dimension Entity.**

A linear dimension consists of a general note; two leaders; and zero, one or two witness lines. Refer to Figure 4-18 for examples of linear dimensions.

4.2.11.1   **Directory Data**
ENTITY TYPE NUMBER :   216

4.2.11.2   **Parameter Data**

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to general note directory entry
2	DEARRW1	Pointer	Pointer to first leader directory entry
3	DEARRW2	Pointer	Pointer to second leader directory entry
4	DEWIT1	Pointer	Pointer to witness line directory entry, 0 if not defined
5	DEWIT2	Pointer	Pointer to witness line directory entry or 0

Additional Pointers as required (see 2.2.4.4.2).

1.5772

EXAMPLE 1

1.5772

EXAMPLE 2

1.1160
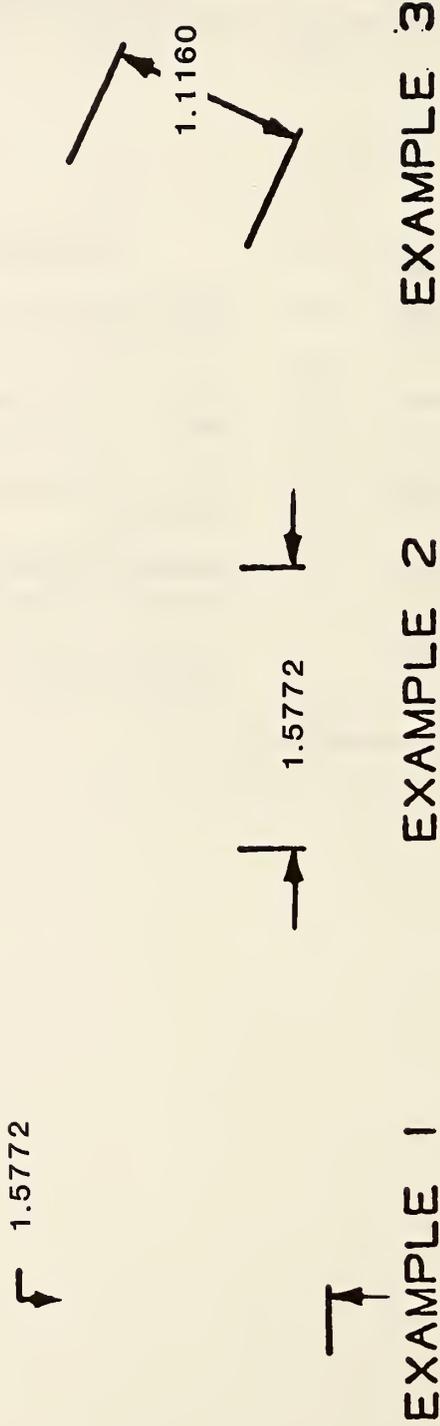
EXAMPLE 3

FIG. 4-18   EXAMPLES OF THE LINEAR DIMENSION ENTITY

4.2.12   **Ordinate Dimension Entity.**

The ordinate dimension entity is used to indicate dimensions from a common base line. Dimensioning is only permitted along the XT or YT axis.

4.2.12.1   An ordinate dimension consists of a general note and a witness line or leader. The values stored are pointers to the directory entry for the associated note and witness line. Examples of ordinate dimensions are shown in Figure 4-19.

4.2.12.2   **Directory Data**
ENTITY TYPE NUMBER :   218

4.2.12.3   **Parameter Data**

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to general note directory entry
2	DEWIT	Pointer	Pointer to witness line directory entry or leader directory entry

Additional Pointers as required (see 2.2.4.4.2).
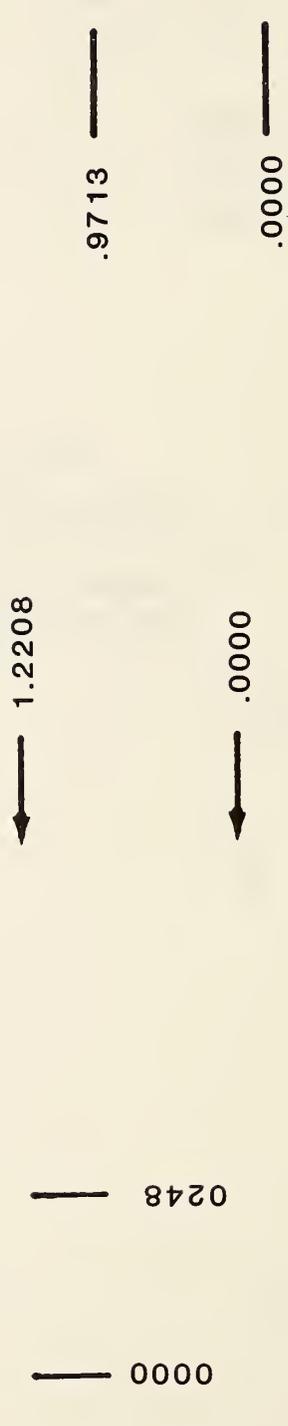
FIG. 4-19   EXAMPLES OF THE ORDINATE DIMENSION ENTITY

4.2.13   **Point Dimension Entity.**

A point dimension consists of a leader, text, and an optional circle or hexagon enclosing the text.

4.2.13.1   The leader will always contain three segments, and its first and last segments are always horizontal or vertical. If a hexagon encloses the text, it will be described by a Composite Curve entity. If a circle or hexagon does not enclose the text, the last segment of the leader will be horizontal and it will underline the text.

4.2.13.2   Examples are shown in Figure 4-20.

4.2.13.3   **Directory Data**
ENTITY TYPE NUMBER :220

4.2.13.4   **Parameter Data**

Index	Name	Type	Description
1	DE1	Pointer	Pointer to general note directory entry
2	DE2	Pointer	Pointer to leader directory entry
3	DE3	Pointer	Pointer to circular arc, composite curve, or 0.

Additional Pointers as required (see 2.2.4.4.2).

EXAMPLE 3

ABCDEFG
5.5228

EXAMPLE 2

BL
2.5901

EXAMPLE 1

WL -.0109
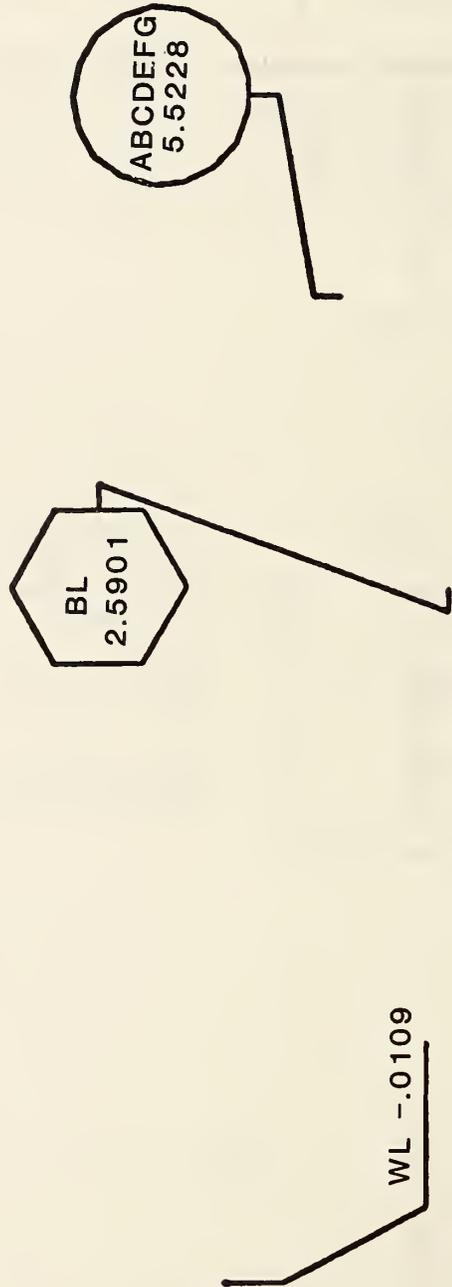
FIG. 4-20 EXAMPLES OF THE POINT DIMENSION ENTITY

248

4.2.14   **Radius Dimension Entity.**

A radius dimension consists of a general note, a leader, and an arc center point, (XT, YT). Refer to Figure 4-21 for examples of radius dimensions.

4.2.14.1   The arc center coordinates are used for positioning the radius dimension line relative to the arc being dimensioned.

4.2.14.2   **Directory Data**
ENTITY TYPE NUMBER: 222

4.2.14.3   **Parameter Data**

Index	Name	Type	Description
1	DEN	Pointer	Pointer to general note directory entry
2	DEP	Pointer	Pointer to leader directory entry
3	XT	Real	Arc center coordinates
4	YT	Real	

Additional Pointers as required (see 2.2.4.4.2).

1.5980 R

EXAMPLE 3

.9474 R

EXAMPLE 2

.9657 R
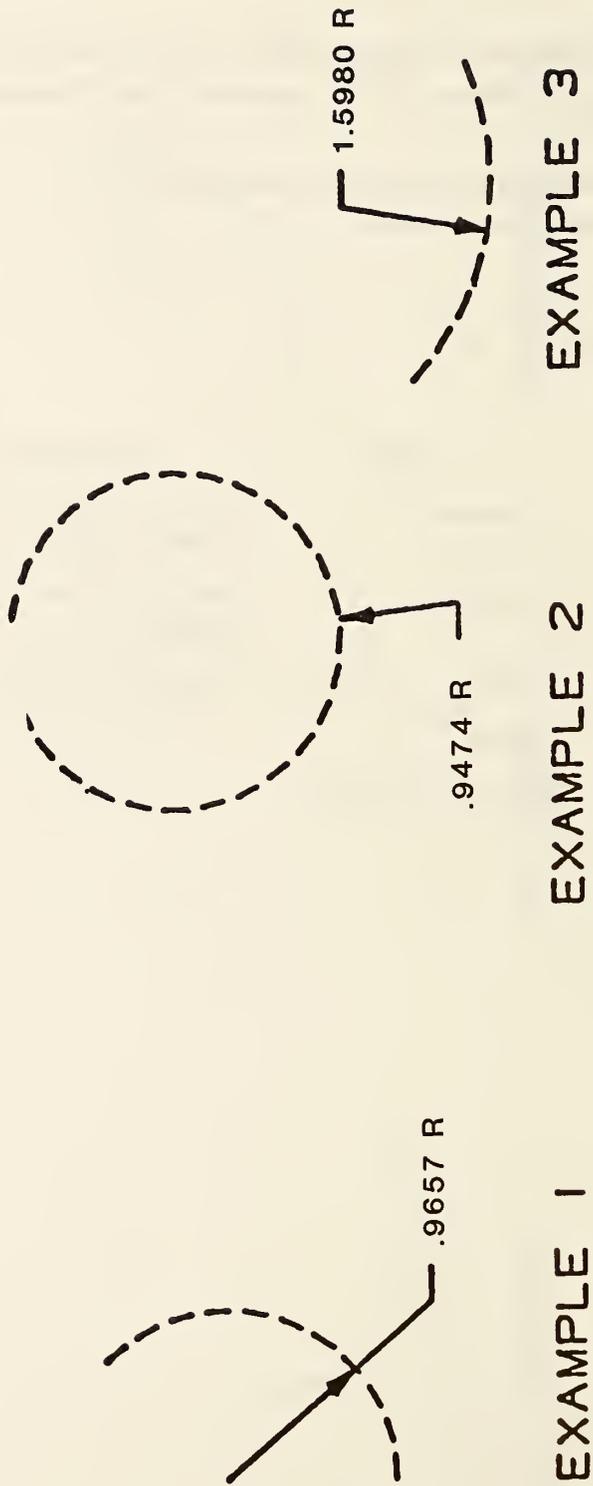
EXAMPLE I

FIG. 4-21 EXAMPLES OF THE RADIUS DIMENSION ENTITY

4.2.15   **Section Entity.**

A section entity is defined as a copious data entity (Type 106 forms 31 to 38). The form number describes how the data are to be interpreted. These descriptions are included for compatability with previous versions of the specification.   The Sectioned Area Entity (Type 230) provides a more compact method for transferring this information.

4.2.15.1   The point data contains a list of points (X_n, Y_n), n=1, 2, . . . , N, (The Z value is constant and N is an even integer.)

The display of the lines consists of solid line segments between the points (X_n, Y_n, Z) and (X_{n+1}, Y_{n+1}, Z) where n = 1,3,5, ... N-1.

A portion of collinear line segments which appear to be a dashed line shall consist of point pairs for each dash.

4.2.15.2   The defined line patterns are described below and illustrated in Figure 4-22. (ANSI79).

Form number	Description
31	Parallel line segments from section edge to edge. (Cast or malleable iron and general use for all materials)
32	Parallel line segments in pairs with a gap between pairs. (Steel)
33	Alternating pattern of a solid line and a set of collinear dash segments. (Bronze, brass, copper, and compositions)
34	Parallel lines in quadruples with a gap between groups. (Rubber, plastic, and electrical insulation)
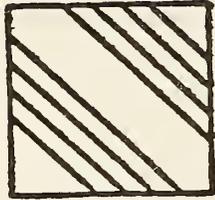
FORM 34
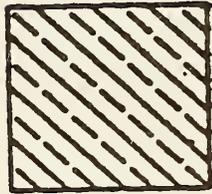
FORM 38
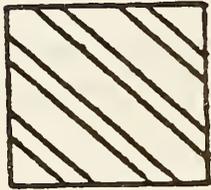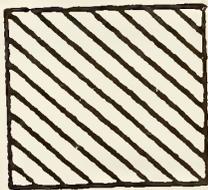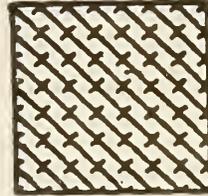
FORM 33
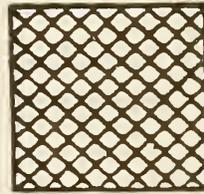
FORM 37
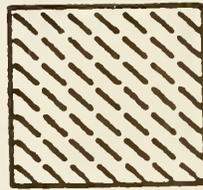
FORM 32
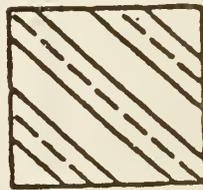
FORM 36

FORM 31

FORM 35

FIGURE 4-22   Examples of the Section Entity

Form number	Description
35	Triples of parallel lines consisting of two solid lines and a set of collinear dash segments between them with a gap between triples. (Titanium and refractory material)
36	Parallel sets of collinear dash segments. (Marble, slate, glass, porcelain)
37	Two perpendicular sets of parallel lines. (White metal, zinc, lead, babbitt, and alloys)
38	Two perpendicular sets of lines with the principal set solid from edge to edge and the second set consisting of collinear dash segments alternating on the solid lines. (Magnesium, aluminum, and aluminum alloys)

4.2.15.3   See Section 3.5 for parameters of the Section Entity.

4.2.16   **General Symbol Entity.**

A general symbol entity consists of a general note; one or more geometric entities which define a symbol; and zero, one or more associated leaders. Examples of general symbol entities are shown in Figure 4-23.

Any geometric entity used to create the symbol will have a subordinate entity switch of 01 and an entity use flag of 01 in field 9 of its Directory Entry section.

4.2.16.1   **Directory Data**
              ENTITY TYPE NUMBER: 228

4.2.16.2   **Parameter Data**

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to associated general note
2	N	Integer	Number of pointers to geometry
3	GPNT1	Pointer	Pointer to defining geometry
.	.	.	
.	.	.	
N+2	GPNTN	Pointer	
N+3	L	Integer	Number of Leaders or zero
N+4	LPNT1	Pointer	Pointers to associated leaders
.	.	.	
.	.	.	
N+3+L	LPNTL	Pointer	

Additional Pointers as required (see 2.2.4.4.2).
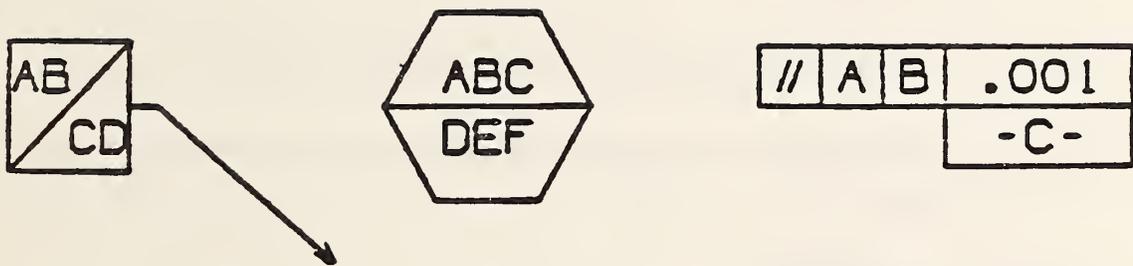
FIGURE 4-23 GENERAL SYMBOL ENTITIES

4.2.17   **Sectioned Area Entity**

A sectioned area is a portion of a design which is to be filled with a pattern of lines. It consists of a pointer to a boundary curve, a specification of the pattern of lines, the coordinates of a point on the lines, the distance between lines, the angle between the lines and the X-axis of the definition space, and the specification of any enclosed boundary curves (islands).

4.2.17.1   The XT and YT coordinates, which may be specified, indicate a location which is on one of the lines. This point allows applications which require specific placements of the lines to constrain them appropriately. If not specified, i.e., indicated by default, the lines need only be within the bounding curve.

4.2.17.2   The angle of the lines has a default value of pi/4, measured in radians.

4.2.17.3   The line pattern is specified according to predefined definitions illustrated in Figure 4-24.

4.2.17.4   **Directory Data**
              ENTITY TYPE NUMBER: 230

4.2.17.5   **Parameter Data**

Index	Name	Type	Description
1	BNDP	Pointer	Pointer to boundary curve-a single closed curve or a composite curve
2	PATRN	Integer	Line pattern code
3	XT	Real	X coordinate through which a line should pass
4	YT	Real	Y coordinate through which a line should pass
5	ZT	Real	Z depth of lines
6	DIST	Real	Normal distance between adjacent lines

7	ANGLE	Real	Angle measured in radians from the XT axis to the lines of the sectioning
8	N	Integer	Number of island curves or zero
9	ISLPT1	Pointer	Pointer to a boundary curve for an island
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
8+N	ISLPTN	Pointer	Pointer to last boundary curve for an island

Additional pointers as required (see 2.2.4.4.2).
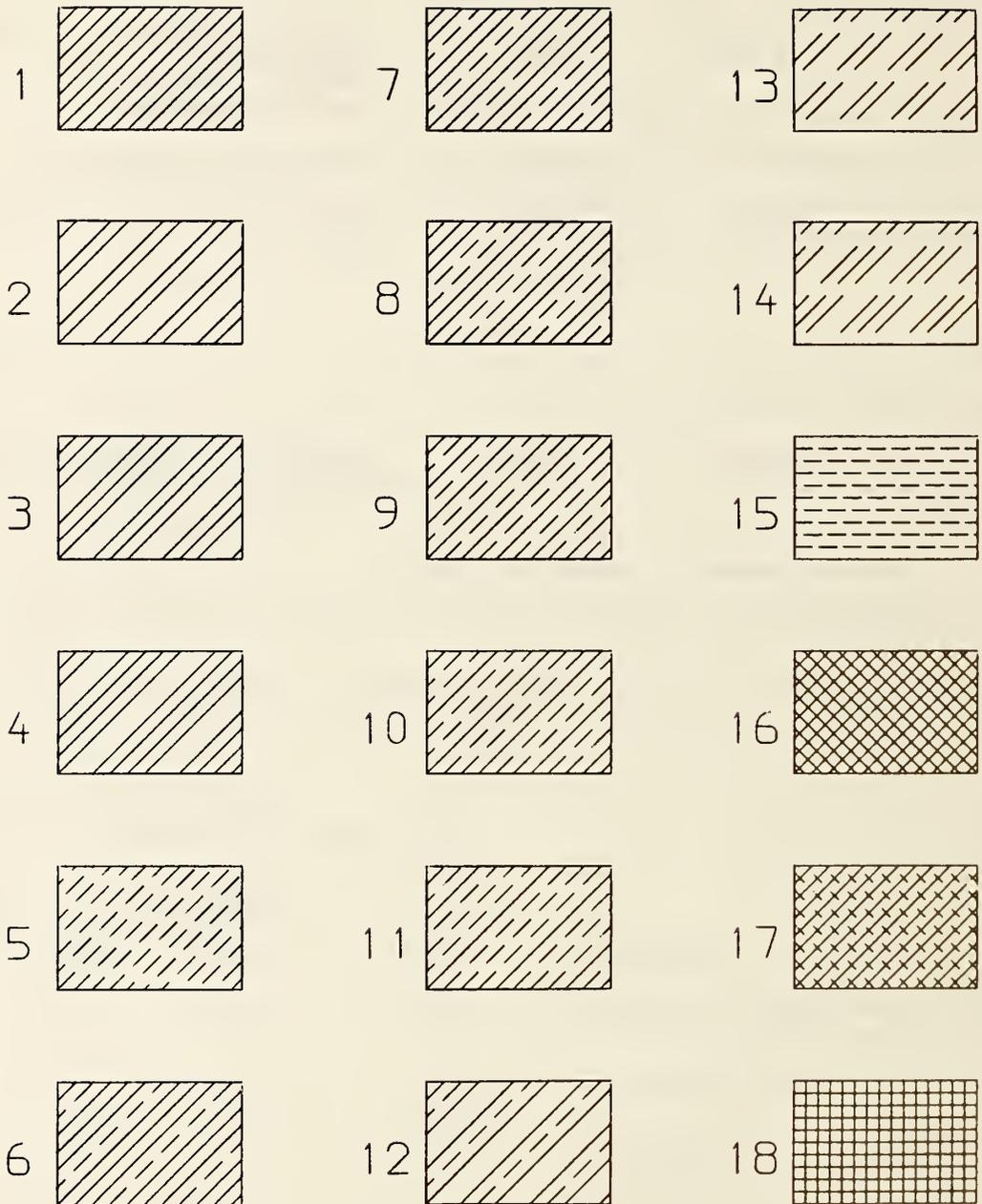
LINE PATTERN CODES

FIGURE 4-24 SECTION LINE PATTERNS

4.2.18   **Witness Line Entity.**

A witness line is a form number 40 of a copious data block that contains one or more straight line segments associated with drafting entities of various types. Each line segment may be visible or invisible. Refer to Figure 4-25 for examples.

4.2.18.1   If the witness line is suppressed, this is indicated by a 0 in the pointer field of the drafting entity pointing to a witness line, or by setting the blank status of the directory entry of the copious data entity for the witness line.

4.2.18.2   Within the copious data, there will be the location from which the witness line gap must be maintained. This point is indicated in the figure as P1. The location will be the first point in the copious data. P1 will be coincident with the geometry being dimensioned or equal to P2 when the location of the geometry is unknown. Note: for those annotation methods that do not allow drafting entities to be displaced from the plane of annotation, coincident with the geometry indicates that a line normal to the plane of annotation connects P1 and the point on the geometry being dimensioned. Note that all points must be collinear, and that the number of points will be odd and at least 3 (3, 5, 7, . . .), with alternating blank and displayed segments. The examples in Figure 4-25 show the blanking of segments and the order of points stored in the copious data.

4.2.18.3   See Section 3.5 for parameters of the Witness Line Entity.
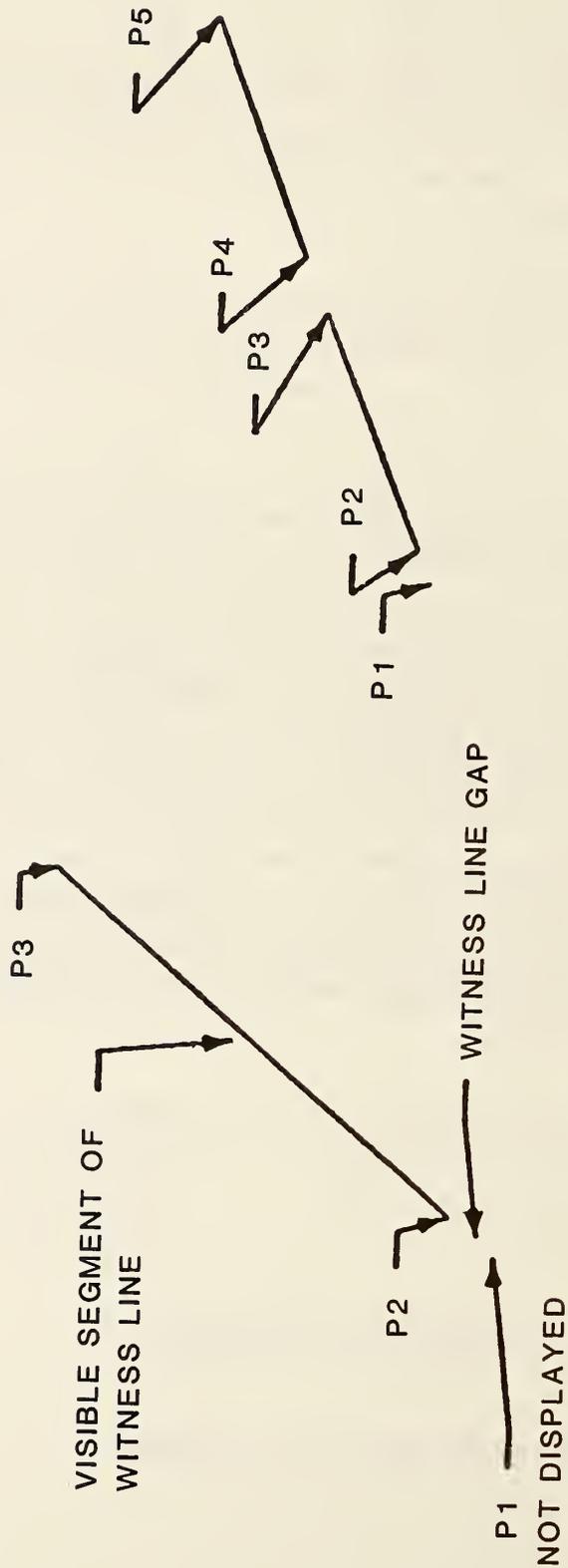
FIGURE 4-25 EXAMPLES OF THE WITNESS LINE ENTITY

4.3   **Structure Entities**

4.3.1   **Entity Type/Type Number**

The following entities are defined in this section:

Entity Type Number	Entity Type
302	Associativity Definition Entity
304	Line Font Definition Entity
306	MACRO Definition Entity
308	Subfigure Definition Entity
310	Text Font Definition Entity
312	Text Display Template
314	Color Definition
320	Network Subfigure Definition
402	Associativity Instance Entity
404	Drawing Entity
406	Property Entity
408	Singular Subfigure Instance Entity
410	View Entity
412	Rectangular Array Subfigure Instance Entity
414	Circular Array Subfigure Instance Entity
416	External Reference
418	Nodal Load/Constraint
420	Network Subfigure Instance
600-699 or 10000-99999 as specified by user	MACRO Instance Entity

4.3.2   **Associativity Definition Entity.**

The associativity definition entity permits the preprocessor to define an associativity schema. That is, by using the associativity definition, the preprocessor defines the type of relationship. It is important to note that this mechanism specifies the syntax of such a relationship and not the semantics.

4.3.2.1   <u>Schema.</u> The definition schema allows the specification of multiple groups of data which are called classes. A class is considered to be a separate list, and the existence of several classes implies an association among the classes as well as among the contents of each class.

For each class, the schema has provision to specify whether or not back pointers are required. A back pointer being required implies that an entity which is a member of this associativity (when it is instanced) has a pointer to the directory entry of the associativity instance in its back pointer parameter section.

The provision in the schema to specify whether or not a class is ordered is used to state whether the order of appearance of entries in the class is significant.

In the schema, "ENTRIES" are the members of the class. However, each entry could be composed of several items. If multiple items are required, they will be ordered. For example, if the entries were locations, each entry might have three items to specify X, Y, and Z values.

The associativity definition will fix the number of classes for an associativity and the number of items per entry in a particular class. Each associativity instance will have a variable number of entries per class. In order to help decode instances of the definition, each item is specified as a pointer (to an entity directory entry) or a data value.

4.3.2.2   Form. Two kinds of associativity are permitted within the file. Pre-defined associativities will have form numbers in the range of 1 to 5000 and are defined in 4.3.3.3. The second kind of associativity is defined in the file by a preprocessor (Form numbers 5001-9999). These definitions appear once in the file for each form of associativity defined, and allow the preprocessor to fill in the definition according to a schema which defines the details of the associativity.

The definition includes the associativity form, the number of class definitions, the number and type of items in each entry, and whether back pointers (from the entity to the associativity) are required. Each set of values (BP, Order, N, and Item type) is considered a class. See 4.3.3.3.1 for a complete example of associativity.

4.3.2.3   **Directory Data**
ENTITY NUMBER :   302

4.3.2.4   **Parameter Data**

Index	Name	Type	Description
1	K	Integer	Number of class definitions
2	BP1	Integer	1- back pointers required 2- back pointers not required
3	OR1	Integer	1-ordered class 2-unordered class
4	N1	Integer	Number of items per entry
5	IT1(1)	Integer	1-pointer to a directory entry 2-value 3-parameter is a value or a pointer
.		if parameter $\geq$ 0 it is a value if parameter $<$ 0 it is a pointer	
.			

4+N1

The items in parameters 2 through 4+N1 are repeated for each of the K classes.

4.3.3   **Associativity Instance Entity.**

Each time an associativity relation is needed in the file an associativity instance entity is used.

The form number of the associativity instance will identify the meaning of the entity. If the form number is between 1 and 5000, the definition is specified as described in 4.3.2.2. If the form number is between 5001 and 9999, an associativity definition will occur in the file and the structure field of the instance (DE field 3) will contain a pointer to the directory entry of the associativity definition.

Each entity that is a member of an associativity instance can contain a back pointer to the associativity instance (see 2.2.4.4.2).

The parameters K and N1, N2, ...NK are specified in the associativity definition. (see 4.3.2.4)

4.3.3.1   **Directory Data**
ENTITY TYPE NUMBER :402

4.3.3.2   **Parameter Data**

Index	Name	Type	Description
1	NE1	Integer	Number of class one entries
2	NE2	Integer	Number of class two entries
.			
.			
.			
K	NEK	Integer	Number of class K entries

264

For K classes with (NE1,...,NEK) entries
with (N1, . . . , NK) items per entry

Index	Name	Type	Description
K + 1	I1,1,1	Variable	Class 1, Entry 1, Item 1
		.	Item 2
		.	. .
		.	. .
		.	. .
		Variable	Item N1
		Variable	Entry 2, Item 1
		.	.
		.	.
		.	.
		Variable	Item N1
		.	.
		.	.
		.	.
		Variable	Entry NE1, Item 1
		.	.
		.	.
		.	.
		.	Item N1
	I2,1,1	Variable	Class 2, Entry 1, Item 1
		.	.
		.	.
		.	.
		Variable	Item N2
		Variable	Entry 2, Item 1
		.	.
		.	.
		.	.
		.	Item N2
		.	.
		.	.
		.	.
		Variable	Entry NE2, Item N2
	.	.	
	.	.	
	.	.	
	IK,1,1	Variable	Class K, Entry 1, Item 1
X	IK,NEK,NK	Variable	Class K, Entry NEK, Item NK

Additional Pointers as required (see 2.2.4.4.2).

4.3.3.3   <u>Pre-defined Associativities.</u> As defined in 4.3.2.2, the associativity definition entity will only occur for Form Numbers 5001 through 9999. The following paragraphs contain the definitions of the pre-defined associativities as they would appear if they were defined by a user. Also included in this Section are the descriptions of each associativity's parameters in a manner similar to other entities in this Specification.

4.3.3.3.1   FORM NUMBER: 1   <u>Group</u>

The Group Associativity allows a collection of a set of entities to be maintained as a single, logical entity. Figure 4-26 is an example.

There are four form numbers which specify Group associativities:

Form Number	Description
1	Unordered group with backpointers required
7	Unordered group with backpointers not required
14	Ordered group with backpointers required
15	Ordered group with backpointers not required

The first (Form=1) is defined here; the others are defined in subsections 4.3.3.3.6 (Form=7), 4.3.3.3.10 (Form=14), and 4.3.3.3.11 (Form=15), respectively.

<u>DEFINITION</u>

Index	Set Value	Meaning
1	1	One class
2	1	Back pointers required
3	2	Unordered
4	1	One item per entry
5	1	The item is a pointer

DESCRIPTION

Directory Data
ENTITY NUMBER: 402
FORM NUMBER: 1

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE	Pointer	Pointer to entity 1
3	DE	Pointer	Pointer to entity 2
.	.	.	
.	.	.	
N+1	DE	Pointer	Pointer to entity N

Additional pointers as required (see. 2.2.4.4.2).
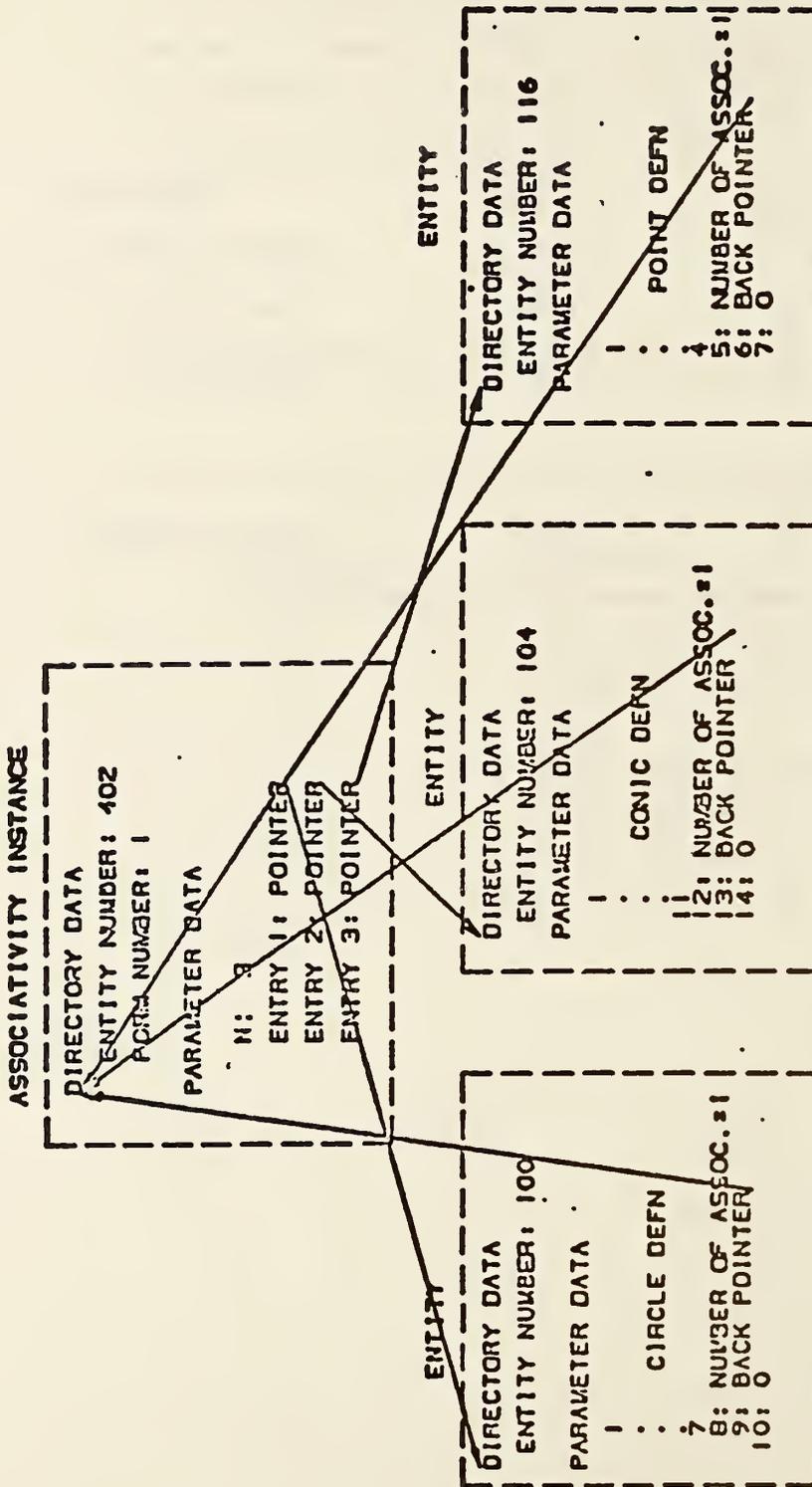
FIGURE 4-26 ASSOCIATIVITY INSTANCE AND ENTITIES

4.3.3.3.2   FORM NUMBER: 2   **External Logical Reference File Index**

The External Logical Reference File Index appears in one file which contains references from another file. It contains a list of the symbolic names used by the referencing files and the DE pointers to the corresponding definitions within the referenced file. See section 2.5.4 and the External Reference Entity (type 416) for more detail.

DEFINITION

Index	Set Value	Meaning
1	1	One class (externally referenced entities)
2	2	Backpointers not required
3	2	Unordered list of entries in a class
4	2	Number of items in an entry
5	2	First item is a value (External Reference Entity Symbolic Name)
6	1	Second item is a pointer (Internal Entity DE Pointer)

DESCRIPTION

Directory Data
ENTITY NUMBER: 402
FORM NUMBER: 2

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of Index Entries
2	NAME1	String	External Reference Entity Symbolic Name
3	PTR1	Pointer	Internal Entity DE Pointer
.	.	.	
.	.	.	

269

| 2N | NAMEN | String | Last External Reference Entity Symbolic Name |
| 2N+1 | PTRN | Pointer | Last Internal Entity DE Pointer |

Additional Pointers as Required (see 2.2.4.4.2)

4.3.3.3.3   FORM NUMBER :   3   **Views Visible**

When an entity is to be displayed in a single view, a pointer to that view entity is entered in parameter 6 of the entity's DE.

If an entity is to be displayed in more than one view, parameter 6 of its DE contains a pointer to an instance of a Form 3 associativity. This form of the associativity contains two classes of information. The first class contains the number of views visible followed by pointers to each of the view entities visible in the specific associativity instance. The second class contains the number of entities whose display is specified by this instance, followed by pointers to each of the entities.

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1	
2	1	Back pointers required
3	2	Unordered
4	1	One item per entry
5	1	Item is a pointer (to view entity)
	Class 2	
6	2	Back pointers not required
7	2	Unordered
8	1	One item per entry
9	1	Item is a pointer (to other entity)

270

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 3

Parameter Data

Index	Name	Type	Description
1	N1	Integer	Number of views visible
2	N2	Integer	Number of entities displayed in these views
3	DEV1	Pointer	Pointer to view entity
.	.	.	
.	.	.	
N1+2	DEVN1	Pointer	
N1+3	DE1	Pointer	Pointer to entity whose display is being specified by this associativity instance
.	.	.	
.	.	.	
N1+N2+2	DEN2	Pointer	Pointer to entity N2

Additional pointers as required (see 2.2.4.4.2).

4.3.3.3.4   FORM NUMBER: 4   **Views Visible, Color, Line Weight**

This associativity is an extension of Form Number 3. For those entities that are visible in multiple views, but must have a different line font, color number, or line weight in each view, there will be an occurrence of the associativity instance Form Number 4.

In the parameter data portion of the associativity instance, the parameter N1 will indicate the number of blocks containing the view visible, line font, color number, and line weight specifications. Each block will contain a pointer to the view entity, a line font value or 0, a pointer to a line font definition entity if the line font value was 0, a color value or pointer to a color definition entity, and a line weight value. Parameter N2 will contain the number of entities which are members of this associativity (i.e., entities which have this particular display characteristic).

Note that N2 may often be 1. If more than one entity appears in Class 2 the complete set of display characteristics in Class 1 apply to <u>each</u> entity in Class 2.

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1 (View)	
2	1	Back pointers required
3	2	Unordered
4	5	Five items per entry
	(Entry template)	
5	1	Pointer to view directory entry
6	2	Line font value
7	1	Pointer to line font directory entry
8	3	Color number (value) or pointer

272

9	2	Line weight (value)
	Class 2 (entity)	
10	2	Back pointers not required
11	2	Unordered
12	1	One item per entry
13	1	Item is a pointer (to entity)

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 4

Parameter Data

Index	Name	Type	Description
1	N1	Integer	Number of blocks containing the view visible, line font, color number, and line weight information
2	N2	Integer	Number of entities which have this particular set of display characteristics
3	DEV1	Pointer	Pointer to view entity 1
4	LF1	Integer	Line font value or 0
5	DEF1	Pointer	If parameter 4 = 0, pointer to a line font definition. Otherwise = 0.
6	CN1	Integer or Pointer	Color number value 1
7	LW1	Integer	Line weight value 1
8	DEV2	Pointer	Pointer to view entity 2
.	.	.	
.	.	.	

5*N1+2	LWN1	Integer	Line weight value N1
5*N1+3	DE1	Pointer	Pointer to entity 1
.	.	.	
.	.	.	
5*N1+N2+2	DEN2	Pointer	Pointer to entity N2

Additional pointers as required (see 2.2.4.4.2).

4.3.3.3.5   FORM NUMBER: 5   Entity Label Display

Some entities may have one or more possible displays for their entity labels, depending on the view in which they are being displayed. For those entities, there will be an occurrence of the associativity instance Form Number 5.

In the parameter data portion of the associativity instance, the parameter N will indicate the number of blocks containing label placement information. Each block will contain a pointer to a view entity which specifies the view of visibility.   The remaining information (text location, leader, and level number) applies to the label for that view.

DEFINITION

Index	Set Value	Meaning
1	1	One class
2	2	Back pointers not required
3	1	Ordered
4	7	Seven items per entry
5	1	Pointer to view directory entry
6	2	XT of text location
7	2	YT of text location
8	2	ZT of text location
9	1	Pointer to leader directory entry
10	2	Entity label level number
11	1	Pointer to entity

402, Form 5 - ENTITY LABEL DISPLAY

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 5

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of label placements
2	DEV1	Pointer	Pointer to a first view entity
3	XT1	Real	XT coordinate of text location in first view
4	YT1	Real	YT coordinate of text location in first view
5	ZT1	Real	ZT coordinate of text location in first view
6	DEL1	Pointer	Pointer to leader in first view
7	LLN1	Integer	Entity label level number in first view
8	DE1	Pointer	Pointer to first entity being displayed
9	DEV2	Pointer	Pointer to second view entity
.	.	.	.
.	.	.	.
.	.	.	.
15	DE2	Pointer	Pointer to second entity being displayed
.	.	.	.
.	.	.	.
.	.	.	.
7*N-5	DEVN	Pointer	Pointer to N1th view entity
.	XTN	Real	XT coordinate of text location in Nth view
.	YTN	Real	YT coordinate of text location in Nth view

Index	Name	Type	Description
.	ZTN	Real	ZT coordinate of text location in Nth view
.	DELN	Pointer	Pointer to leader in Nth view
.	LLNN	Integer	Entity label level number in Nth view
7*N+1	DEN	Pointer	Pointer to Nth entity being displayed

Additional pointers as required (see 2.2.4.4.2)

4.3.3.3.6   FORM NUMBER: 7   **Group Without Back Pointers**

DEFINITION

Index	Set Value	Meaning
1	1	One class
2	2	Back pointers not required
3	2	Unordered
4	1	One item per entry
5	1	The item is a pointer

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 7

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE1	Pointer	Pointer to entity 1
.	.	.	
.	.	.	
N+1	DEN	Pointer	Pointer to entity N

Additional Pointers as required (see 2.2.4.4.2).

4.3.3.3.7   FORM NUMBER: 9   **Single Parent Associativity**

This associativity defines a logical structure of one independent (parent) entity and one or more subordinate (children) entities.

Both parent and child entities require back pointers to this instance.   Any necessary display parameters are governed by the parent entity.

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1 (parent)	
2	1	Back pointers required
3	2	Unordered
4	1	One item per entry
5	1	Item is pointer to parent entity
	Class 2 (children)	
6	1	Back pointers required
7	1	Ordered
8	1	One item per entry
9	1	Item is pointer to child entity

402, Form 9 - SINGLE PARENT
ASSOCIATIVITY

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 9

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of Parent Entities (NP=1)
2	NC	Integer	Number of Children
3	DE	Pointer	Pointer to Parent Entity
4	DE1	Pointer	Pointer to Child Entity 1
.	.	.	
.	.	.	
.	.	.	
2+NC	DENC	Pointer	Pointer to Child Entity NC

Additional Pointers as required (see 2.2.4.4.2).

4.3.3.3.8   FORM NUMBER: 12   **External Reference File Index**

The External Reference File Index appears in one file which contains
definitions referenced by another file. It contains a list of the symbolic
names used by the referencing files and the DE pointers to the corresponding
definitions within the referenced file. See section 2.5.4 and the External
Reference Entity (Type 416) for more detail.

DEFINITION

Index	Set Value	Meaning
1	1	One class (externally referenced entities)
2	2	Backpointers not required
3	2	Unordered list of entries in a class
4	2	Number of items in an entry
5	2	First item is a value (External Reference Entity Symbolic Name)
6	1	Second item is a pointer (Internal Entity DE Pointer)

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 12

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of Index Entries
2	NAME1	String	External Reference Entity Symbolic Name
3	PTR1	Pointer	Internal Entity DE Pointer
.	.	.	
.	.	.	
2N	NAMEN	String	Last External Reference Entity Symbolic Name
2N+1	PTRN	Pointer	Last Internal Entity DE Pointer

Additional Pointers as required (see 2.2.4.4.2)

4.3.3.3.9   FORM NUMBER: 13   <u>Dimensioned Geometry Associativity</u>

This associativity links a dimension entity with the geometry entities it is dimensioning. The pointers to the entities being dimensioned have interpretations related to the type of dimension entity. See Figure 4-27.

DEFINITION

<u>Index</u>	<u>Set Value</u>	<u>Meaning</u>
1	2	Two classes
Class 1 (Dimension Entity)		
2	1	Back pointers required
3	2	Unordered
4	1	One item (pointer to dimension)
5	1	Item is pointer
Class 2 (Related Geometry)		
6	2	Back pointers not required
7	2	Unordered
8	1	One item per entry (pointers to geometry)
9	1	Item is pointer
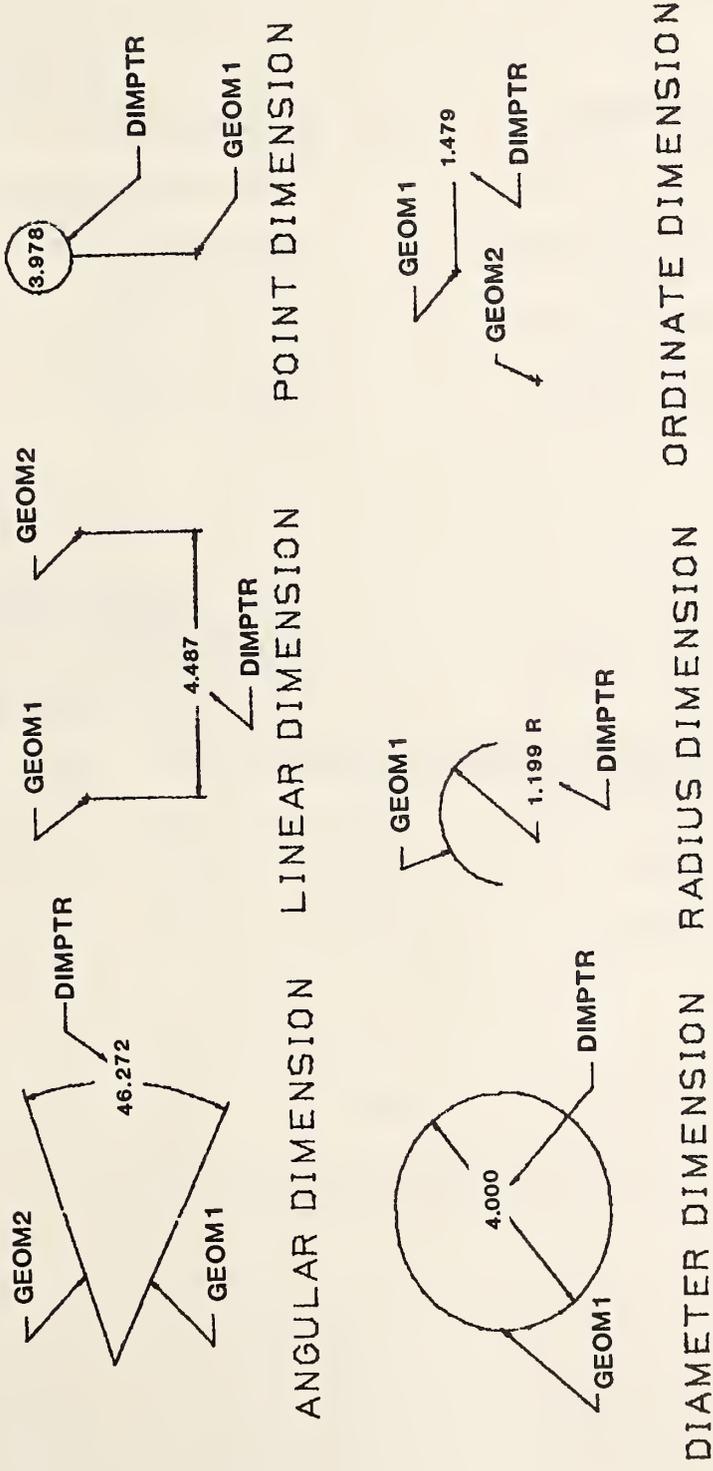
FIGURE 4-27   DIMENSIONED GEOMETRY ASSOCIATIVITY

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 13

Parameter Data

Index	Name	Type	Description
1	ND	Integer	Number of dimensions (ND=1)
2	NG	Integer	Number of associated geometry entities
3	DIMPTR	Pointer	Pointer to dimension
4	GEOM1	Pointer	Pointer to geometry entity 1
.	.	.	.
.	.	.	.
3+NG	GEOMNG	Pointer	Pointer to geometry entity NG

Additional Pointers as required (see 2.2.4.4.2)

4.3.3.3.10   FORM NUMBER: 14   **Ordered Group with Backpointers**

DEFINITION

Index	Set Value	Meaning
1	1	One class
2	1	Back pointers required
3	1	Ordered
4	1	One item per entry
5	1	The item is a pointer

DESCRIPTION

Directory Data
ENTITY NUMBER: 402
FORM NUMBER: 14

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE	Pointer	Pointer to entity 1
3	DE	Pointer	Pointer to entity 2
.	.	.	
.	.	.	
N+1	DE	Pointer	Pointer to entity N

Additional Pointers as required (see 2.2.4.4.2).

4.3.3.3.11   FORM NUMBER: 15   **Ordered Group without Backpointers**

DEFINITION

Index	Set Value	Meaning
1	1	One class
2	2	Back pointers not required
3	1	Ordered
4	1	One item per entry
5	1	The item is a pointer

DESCRIPTION

Directory Data
ENTITY NUMBER: 402
FORM NUMBER: 15

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE	Pointer	Pointer to entity 1
3	DE	Pointer	Pointer to entity 2
.	.	.	
.	.	.	
N+1	DE	Pointer	Pointer to entity N

Additional Pointers as required (see 2.2.4.4.2).

4.3.3.3.12   FORM NUMBER: 16   **Planar Associativity**

This associativity is used to indicate that a collection of entities is coplanar. They may be geometric, annotative, and/or structural. In the case of an entity containing subordinate entities, these must also be coplanar.

The first class contains the pointer to the transformation matrix indicating the plane the entities have been moved to. The plane in question is the image under this transformation of the XY plane. As noted in the description for DE field 7, the value 0 may be used to indicate the identity transformation matrix. This matrix is informational only for the associativity; the constituent entities must properly position themselves in model space.

The second class contains the pointers to the coplanar entities.

402, Form 16 - PLANAR ASSOCIATIVITY

DEFINITION

Index	Set Value	Meaning
1	2	Two classes

Class 1 (Transformation Matrix)

2	2	No back pointers
3	1	Ordered class
4	1	Number of items per entry
5	1	Pointer to directory entry

Class 2 (Coplanar Entities)

6	1	Back pointers required
7	2	Unordered class
8	1	Number of items per entry
9	1	Pointer to directory entry

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 16

Parameter Data

Index	Name	Type	Description
1	1	Integer	One transformation matrix
2	N	Integer	Number of entities in this plane pointed to by this associativity
3	DETR	Pointer	Pointer to transformation matrix moving data from XY plane into space or zero
4	DE1	Pointer	Pointer to first entity on plane specified
.	.	.	.
.	.	.	.
.	.	.	.
N+3	DEN	Pointer	Pointer to last entity on plane on specified

Additional Pointers as required (see 2.2.4.4.2)

4.3.3.3.13   FORM NUMBER: 18   **Flow**

The Flow Associativity represents a single signal or a single fluid flow path. The associativity contains seven classes.

Class one contains the type and function flags:

Type Flag	Meaning
0	Not specified (Default)
1	Logical
2	Physical

The use of the Type Flag is mandatory when both the logical (e.g., schematic) and physical (e.g., printed board) product definitions are in the same file. In such a file, the type flag shall not be zero.

The function flag differentiates between a fluid path and an electrical conductor:

Function Flag	Meaning
0	Not specified (Default)
1	Electrical signal
2	Fluid flow path

A fluid flow path is a single path from a starting connect point entity. The path may include additional intermediate connect points but separate flow entities will be required to describe the branch flow paths. Class four, the Join, lists the elements of the path. For fluid flow paths the elements are ordered as they occur along the flow path and the Connect Points (class three) lists the connect points in the same sense as the order of the Join. That is, the start of the fluid flow path is the one listed first; the end of the fluid flow path is listed last.

Class two contains pointers to other associated flow associativities. These other associativities may implement alternative flow representations. The obvious example of this is the file containing both the schematic and physical product definitions.   The corresponding Flow associativities of each type would be paired.

Class three is the Link, which contains the list of pointers to the Connect Point entities involved in the signal/flow.

Class four is the Join, which contains the list of pointers to the entities representing the graphical implementation of the signal/flow.

Class five contains the flow names which are associated with the signal/flow.

Class six contains a list of pointers to the Text Display Template entities which are to be used to display the first flow name listed in class five. The Text Display Templates provide the locations and attributes for the signal name display.

The text string for display is obtained from the first flow name listed in class five.

Class seven contains a list of pointers to flow paths which branch from the current flow path. This is an ordered list and the "main" continuation of the path, if any, is always listed last. A null pointer is used if there is no continuation of the main path.

DEFINITION

Index	Set Value	Meaning
1	7	Seven classes
	Class 1 (Context Flag)	
2	2	Back pointers not required
3	1	Ordered
4	1	One item per entry
5	2	Item is value
	Class 2 (Associated Flows)	
6	2	Back pointers not required
7	2	Unordered
8	1	One item per entry
9	1	Pointer to Flow Associativity
	Class 3 (Connect Points (Link))	
10	1	Back pointers required
11	1	Ordered
12	1	One item per entry
13	1	Pointer to Connect Point Entity
	Class 4 (Join)	
14	1	Back pointers required
15	1	Ordered
16	1	One item per entry
17	1	Pointer to geometry or Subfigure instance
	Class 5 (Flow Name)	
18	2	Back pointers not required
19	2	Unordered
20	1	One item per entry
21	2	Item is value
	Class 6 (Flow Name Display)	
22	2	Back pointers not required
23	2	Unordered
24	1	One item per entry
25	1	Pointer to Text Display Template Entity
	Class 7 (Flow Continuations)	
26	1	Back pointers required
27	1	Ordered
28	1	One item per entry
29	1	Item is a pointer

The Status Flag values should be set as follows:

Subordinate	To be set by sending system
Entity Use	03
Hierarchy	00

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 18

Parameter Data

Index	Name	Type	Description
1	NCF	Integer	Count of context flags (=2)
2	NF	Integer	Count of associated Flow associativities
3	NC	Integer	Count of Connect Point entities
4	NJ	Integer	Count of Join entities (geometry of subfigure)
5	NN	Integer	Count of Flow names
6	NT	Integer	Count of Text Display Templates for Flow name display
7	NP	Integer	Count of continuation flow associativities
8	TF	Integer	Type of Flow 0 = not specified (Default) 1 = logical 2 = physical . . other as required .
9	FF	Integer	Function flag: 0 = not specified (Default) 1 = electrical signal 2 = fluid flow path . . other as required .

Index	Name	Type	Description
10	SPTR1	Pointer	Pointer to directory entry for Flow Associativity 1
...	...	...	
NF+9	SPTRNF	Pointer	Pointer to directory entry for Flow Associativity NF
NF+10	CPTR1	Pointer	Pointer to directory entry for Connect Point 1
...	...	...	
NF+NC+9	CPTRNC	Pointer	Pointer to directory entry for Connect Point NC
NF+NC+10	JPTR1	Pointer	Pointer to Join entity 1
...	...	...	
NF+NC+NJ+9	JPTRNJ	Pointer	Pointer to Join entity NJ
NF+NC+NJ+10	NAME1	String	Flow name 1
...	...	...	
NF+NC+NJ+NN+9	NAMENN	String	Flow Name NN
NF+NC+NJ+NN+10	GPTR1	Pointer	Pointer to directory entry for Text Template 1
...	...	...	
NF+NC+NJ+NN+NT+9	GPTRNT	Pointer	Pointer to directory entry for Text Template NT
NF+NC+NJ+NN+NT+10	CFPTR1	Pointer	Pointer to continuation flow associativity entity 1
NF+NC+NJ+NN+NT+NP+9	CFPTRNP	Pointer	Pointer to continuation flow associativity entity NP (the "main" continuation)

Additional Pointers as required (see 2.2.4.4.2)

THIS PAGE INTENTIONALLY LEFT BLANK

4.3.3.3.14   **Obsolete Associativity Forms**

The addition of new entities and associativities which greatly increase the capability for transfer of specific data constructs has given cause to deprecate associativity forms published in previous versions of this Specification. The parameter lists for these forms are included herein to provide for interpretation of files created under an earlier version. The new associativity forms or entities which are valid for the previous forms are as follows:

OBSOLETE FORM #		VALID FORM # OR ENTITY TYPE #	
6	View List	3	Views Visible
8	Signal String	18	Flow
10	Text Node	312	Text Template
11	Connect Node	132	Connect Point

The parameter lists for these following associativities may have additional parameters as specified in section 2.2.4.4.2:

4.3.3.3.14.1   FORM NUMBER: 6   **View List**

DEFINITION

Index	Set Value	Meaning
1	2	Two classes

Class 1 (View)

Index	Set Value	Meaning
2	1	Back pointers required
3	2	Unordered
4	1	One item per entry
5	1	Pointer to view directory entry

Class 2 (Entities)

Index	Set Value	Meaning
6	1	Back pointers required
7	2	Unordered
8	1	One item per entry
9	1	Pointer to directory entry of entity visible in view

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 6

Parameter Data

Index	Name	Type	Description
1	1	Integer	Single entry in first class
2	N1	Integer	Number of entities in second class
3	DEV	Pointer	Pointer to view entity
4	DE1	Pointer	Pointers to entities visible in view specified in parameter 3
.	.	.	.
.	.	.	.
.	.	.	.
N1+3	DEN1	Pointer	

4.3.3.3.14.2   FORM NUMBER: 8   **Signal String**

DEFINITION

Index	Set Value	Meaning
1	4	Four classes
	Class 1 (Signal Names)	
2	2	Back pointers not required
3	1	Ordered
4	1	One item per entry
5	2	Item is value
	Class 2 (Connections)	
6	1	Back pointers required
7	2	Unordered
8	1	One item per entry
9	1	Pointer to Connect Node
	Class 3 (Schematic)	
10	1	Back pointers required
11	1	Ordered
12	1	One item per entry
13	1	Pointer to geometry
	Class 4 (Physical Layout)	
14	1	Back pointers required
15	1	Ordered
16	1	One item per entry
17	1	Pointer to geometry

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 8

Parameter Data

Index	Name	Type	Description
1	NS	Integer	Number of signal names
2	N1	Integer	Number of Connection Nodes
3	N2	Integer	Number of Entities in schematic signal string
4	N3	Integer	Number of Entities in physical signal string
5	SIG1	String	Signal name
.	.		
NS+4	SIGNS		

Index	Name	Type	Description
NS+5	PC1	Pointer	Pointer to Connect Nodes
.	.	.	
NS+N1+4	PCN1	Pointer	
NS+N1+5	PS1	Pointer	Pointer to Entity in schematic logical signal string
NS+N2+N1+4	PSN2	Pointer	
NS+N2+N1+5	PP1	Pointer	Pointer to entity in physical signal string
.	.	.	
NS+N3+N2+N1+4	PPN3	Pointer	

4.3.3.3.14.3   FORM NUMBER: 10   **Text Node**

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1 (Geometry Pointers)	
2	1	Back pointers required
3	1	Ordered class
4	1	One item per entry
5	1	Item is pointer (to point entity)
	Class 2 (Text Description)	
6	2	Back pointers not required
7	1	Ordered class
8	7	Seven items/entry
9	2	Box length
10	2	Box height
11	3	Font characteristic
12	2	Slant angle
13	2	Rotation angle
14	2	Mirror flag
15	2	Rotate internal flag

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 10

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of geometry pointers
2	1	Integer	One Text Description
3	GP1	Pointer	Pointer to point (original location)
4	GP2	Pointer	Pointer to instance point (first instance)
.	.	.	
.	.	.	
NP+2	GPNP	Pointer	Pointer to instance point (NP-1 instance)
NP+3	WT	Real	Box length
NP+4	HT	Real	Box height
NP+5	FC	Integer or pointer	Font characteristic (default = 1)
NP+6	SL	Real	Slant angle of text in radians. pi/2 is the value for no slant angle and is the default value.
NP+7	A	Real	Rotation angle in radians for text.
NP+8	M	Integer	Mirror flag (0=no mirror, 1=YT mirror axis, 2=XT mirror axis.)
NP+9	VH	Integer	Rotate internal text flag (0=text horizontal, 1=text vertical)

4.3.3.3.14.4   FORM NUMBER: 11   **Connect Node**

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
Class 1	(Geometry pointers)	
2	1	Back pointers required
3	1	Ordered class
4	1	One item per entry
5	1	Pointer (to point entity)
Class 2	(Connection entities)	
6	2	Back pointers not required
7	2	Unordered class
8	1	One item per entry
9	2	Item is value

DESCRIPTION

Directory Data
ENTITY TYPE NUMBER: 402
FORM NUMBER: 11

Parameter Data

Index	Name	Type	Description
1	NC	Integer	Number of pointers (to points)
2	NP	Integer	Number of entries in second class
3	PT1	Pointer	Pointer to defining point (original location)
4	PT2	Pointer	Pointer to instance point (first instance)
.	.	.	
.	.	.	
NC+2	PTNC	Pointer	Pointer to last instance point (NC-1 instance)
NC+3	DT1	Data	First data entry
.	.	.	
.	.	.	
NC+NP+2	DTNP	Data	Last data entry

4.3.4   **Drawing Entity.**

The Drawing entity specifies a drawing as a collection of annotation entities (i.e., any entity with use flag set to 01) defined in drawing space, and views (i.e., projections of model space data in view space) which, together, constitute a single representation of a part, in the sense that an engineering drawing constitutes a single representation of a part in standard drafting practice. Views are specified by referring to the View Entity. If desired, multiple drawings can be included in a single file, referring to the same model space.

Drawings are located in drawing space as illustrated in Figure 4-28, with sides coincident with the drawing coordinate system axes, and with the lower left corner at the origin (0,0). The drawing space coordinate system (XD, YD) is a special 2-dimensional coordinate system used for view origin locations in the Drawing entity and for annotation entities referenced by the Drawing entity. Any Z coordinates are ignored in the referenced annotation entities and any transformation matrix from definition space to drawing space must be 2-dimensional (i.e., in entity 124, $T_3 = R_{13} = R_{31} = R_{32} = R_{23} = 0.0$ and $R_{33} = 1.0$).

Annotation entities can be defined in drawing space and be referenced by the Drawing entity directly, or can be defined in model space and appear in individual views. When defined in drawing space, the subordinate entity switch should be set to physically dependent (01). The subordinate entity switch for a View entity referenced by the Drawing entity should be set to logically dependent (02).

The transformation of a view from view space to drawing space is controlled by the view scale factor S, specified in the View entity, and the view origin drawing locations, specified in the Drawing entity. In the case of orthographic parallel projection the transformation is:

$$
\begin{bmatrix} XD \\ YD \end{bmatrix} = \begin{bmatrix} S & 0 & 0 \\ 0 & S & 0 \end{bmatrix} \begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} + \begin{bmatrix} XORIGIN \\ YORIGIN \end{bmatrix}
$$

where $\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix}$ denotes the view space coordinates

and $\begin{bmatrix} XORIGIN \\ YORIGIN \end{bmatrix}$ denotes the drawing space coordinates of the origin of the transformed view (see 4.3.11).

If known, the name of the drawing may be provided by using the Name Property (406, form 15). If not provided, a receiving system may default the name of the drawing.

The size of the drawing may be communicated by using the Drawing Size Property (406, form 16).

The units for drawing space may be different from the model space units specified in the Global section. This is accomplished by use of the Drawing Units Property (406, form 17). When this property is not provided, the drawing units will be the same as the model units.

The following values are given in drawing units:

- view origin drawing locations
- drawing size
- coordinates of annotation entities referenced directly

Refer to Figures 4-28 and 4-29 for examples of the use of the drawing entity.
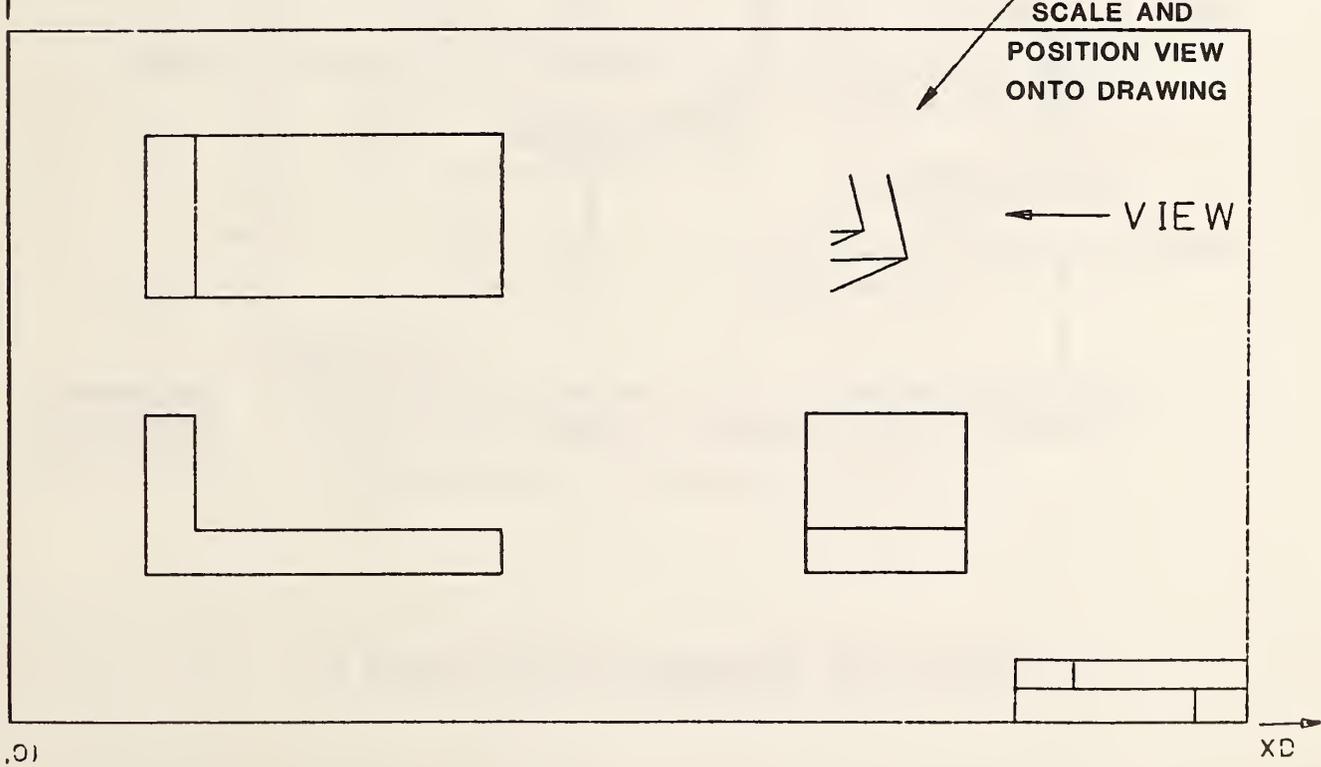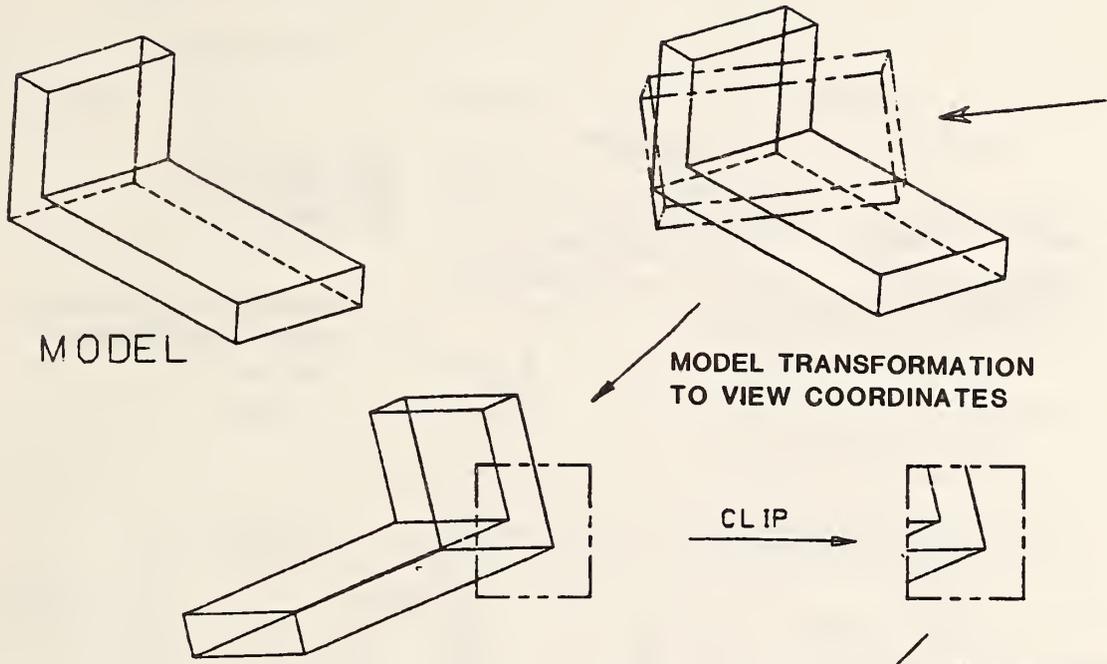
FIGURE 4-28 DRAWING ENTITY EXAMPLE 1

VIEW 1 ORIGIN

(ORIGIN OF VIEW 1
COORDINATE SYSTEM)

THIS LINE
NOT DISPLAYED
IN VIEW 1

VIEW SPACE TO DRAWING
SPACE TRANSFORMATION
FOR VIEW 1

DRAWING

VIEW 2

VIEW 1

DRAWING LOCATION
FOR VIEW 1 ORIGIN

VIEW 3

ANNOTATIONS
REFERENCED BY
DRAWING ENTITY

ORIGIN OF DRAWING COORDINATE SYSTEM
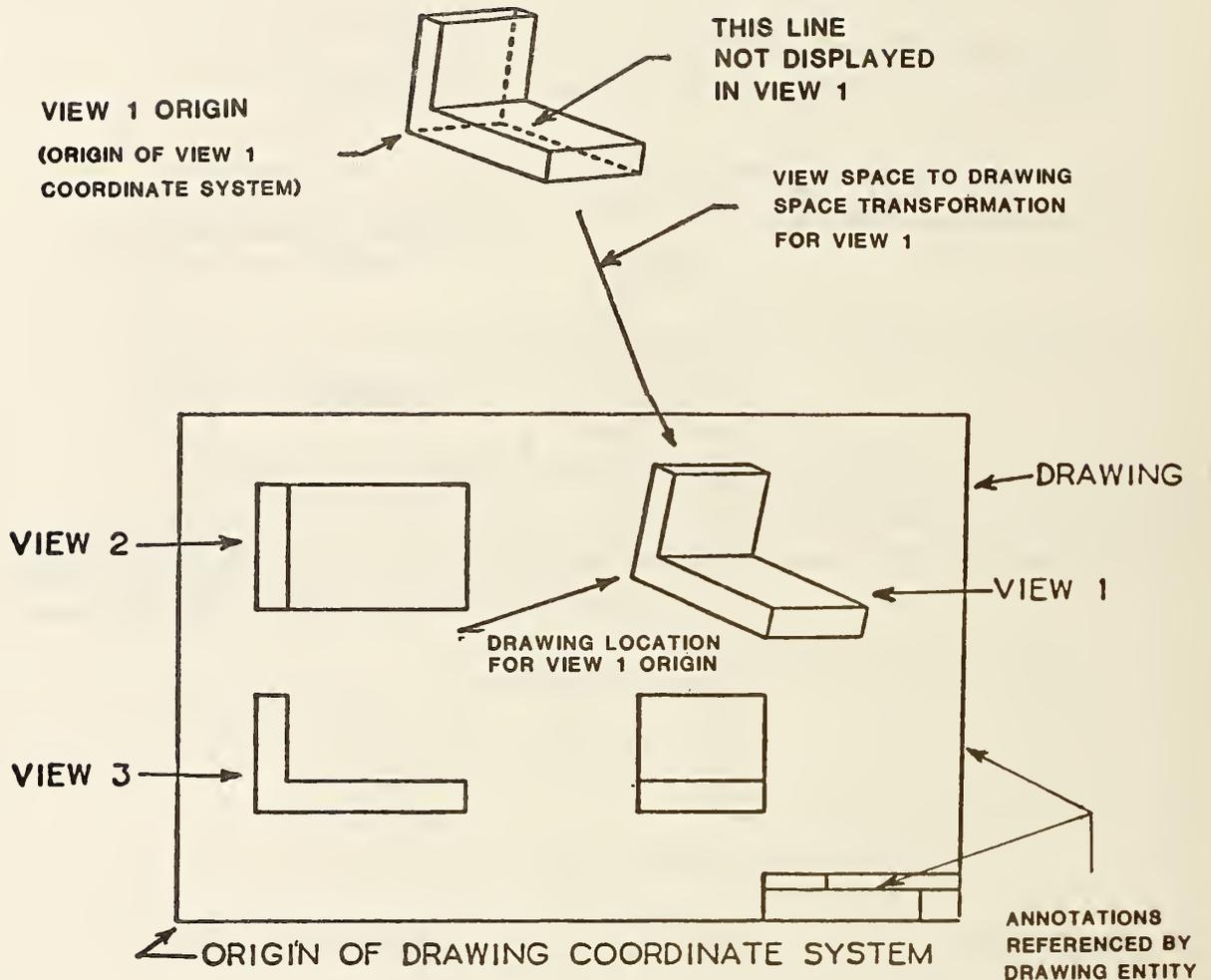
FIGURE 4-29 DRAWING ENTITY EXAMPLE 2

4.3.4.1   **Directory Data**
           ENTITY TYPE NUMBER :   404

4.3.4.2   **Parameter Data**

Parameter	Value	Format	Comment
1	N	Integer	Number of view pointers
2	VPTR1	Pointer	Pointer to directory entry of first view entity
3	XORIGIN1	Real	Drawing space coordinate of the origin of the first transformed view
4	YORIGIN1	Real	Drawing space coordinate of the origin of the first transformed view
5	VPTR2	Pointer	Pointer to directory entry of second view entity
.	.		
.	.		
3N+2	M	Integer	Number of annotation entities (may be zero)
3N+3	DPTR1	Pointer	Pointer to first annotation entity in this drawing
.	.		
.	.		
.	.		
3N+M+2	DPTRM	Pointer	Pointer to Mth annotation entity in this drawing

Additional Pointers as required (see 2.2.4.4.2)

4.3.5   **Line Font Definition Entity.**

Two types of line fonts may be defined. One type considers a line font as a repetition of a basic pattern of visible-blanked (or, on-off) segments superimposed on a line or a curve. The line or curve is then displayed according to the basic pattern. The other type considers a line font as a repetition of a template figure that is displayed at regularly spaced locations along a planar anchoring curve. The anchoring curve itself has no visual purpose.

Any line or curve geometry entity type may reference a Line Font Definition entity by inserting a pointer to that entity in its Directory Entry field 4, the line font pattern field. The type of line font being specified is then indicated by a Form Number in the Line Font Definition entity.

Setting the Form Number to 1 in a Line Font Definition entity specifies that the line font type is to be a repetition of template figure displays along the referencing anchoring curve. The template figure is specified as a subfigure definition entity. In this case, four Index values specify the entity as follows:

The first parameter specifies the orientation of the template displays. This may remain constant, or it may vary with the direction of the anchoring curve at the point of each template figure display location. The second parameter is a pointer to the subfigure definition entity containing the template display. The third parameter specifies display locations on the anchoring curve by giving the common arc length distance between corresponding points on successive template figure displays. The fourth parameter gives a scale factor to be applied to the template subfigure at each display location.

Figure 4-30 illustrates two examples of a line font with Form Number equal 1. In each case, the anchoring curve is a straight line.
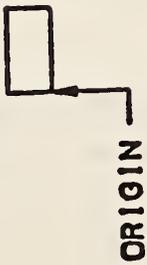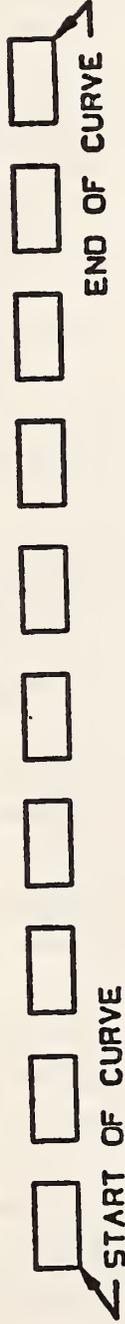
FIGURE 4-30 USE OF SUBFIGURE WITH LINE FONTS

309

Setting the Form Number to 2 in a Line Font Definition entity specifies that the line font type is to be a repetition of a basic visible-blank pattern superimposed on the referencing line or curve. An arbitrary number M of segments may be used in the basic pattern. When the basic pattern is laid out horizontally, the "first" segment is the leftmost one, the "Mth" segment is the rightmost one. The length (in the units of the curve on which the pattern is being superimposed) of each segment of the pattern may be specified individually. This allows the visible-blank sequence of the pattern to alternate between "visible" and "blank" regardless of the lengths of the segments, but does not prohibit adjacent segments from being either both visible or both blanked when unequal lengths are employed. Another option for some patterns is to hold the length constant across segments, and achieve variation in the lengths of the visible and blanked segments by making the "visible" and/or the "blank" segments be adjacent as required.

For example, a basic pattern whose left two-thirds is visible, and whose right third is blanked may be described either by the sequence "visible-blank" with the length of the first segment twice that of the second, or else by the sequence "visible-visible-blank", with the lengths of all three segments equal.

The visible-blank sequence is specified by correlating it with the rightmost M bits in the binary representation of a string of hexadecimal digits, the Mth segment being associated with the units bit of the binary representation of the rightmost hexadecimal digit. A "0" represents a "blank", or "off" segment, a "1" represents a "visible", or "on" segment.

For this line font type, the first parameter is the positive integer M giving the number of segments in the basic pattern. Then, parameter values 2 through M+2 specify the entity as follows:

Parameter values 2 through M+1 give the lengths of the M segments. Parameter value M+2 is the minimal string of hexadecimal digits whose significance has been described above.

Figure 4-31 shows an example of the Form Number 2. Shown is a font made up of five segments of unequal length. Two repetitions of the basic font are illustrated.

Form	Meaning
1	Line font specified by a repeating template subfigure
2	Line font specified by a repeating visible-blanked pattern

4.3.5.1 **Directory Data**
ENTITY TYPE NUMBER: 304
FORM NUMBER: 1

4.3.5.2 **Parameter Data**

Index	Name	Type	Description
1	M	Integer	M=0: Each template display is oriented by aligning the axes of the subfigure definition coordinate system with the axes of the definition space of the anchoring curve. M=1: Each template display is oriented by aligning the X axis of the subfigure definition coordinate system with the tangent vector of the anchoring curve at the point of incidence of the curve and the origin of the subfigure, and the Z axis of the subfigure definition coordinate system with the Z axis of the definition space of the anchoring curve.
2	L1	Pointer	Pointer to the subfigure definition for the template displays.
3	L2	Real	Common arc length distance between corresponding points on successive template figure displays.
4	L3	Real	Scale factor to be applied to the subfigure.

Additional Pointers as Required (see 2.2.4.4.2)

311

4.3.5.3 **Directory Data**
ENTITY TYPE NUMBER: 304
FORM NUMBER: 2

4.3.5.4 **Parameter Data**

Index	Name	Type	Description
1	M	Integer	Number of segments in the basic pattern of visible-blanked segments.
2	L1	Real	Length of the first segment of the basic pattern.
3	L2	Real	Length of the second segment of the basic pattern.
4	L3	Real	Length of the third segment of the basic pattern.
.	.	.	.
.	.	.	.
.	.	.	.
M+1	LM	Real	Length of the Mth segment of the basic pattern.
M+2	B	String	$\left\lfloor \left(\dfrac{M-1}{4}\right) + 1 \right\rfloor$ hexadecimal digits indicating which segments of the basic pattern are visible and which are blanked, where $\lfloor\ \ \rfloor$ represents the greatest integer result. E.g., "5" indicates that segments 1 and 3 are visible. Bits are right justified.

Additional Pointers as Required (see 2.2.4.4.2)

M = 5

L1 = L3 = L4 = L5 = 2.0

L2 = 1.0

BIT PATTERN = 10110

HEXADECIMAL STRING = 2H16

RESULTING LINE FONT:

(2 CYCLES)
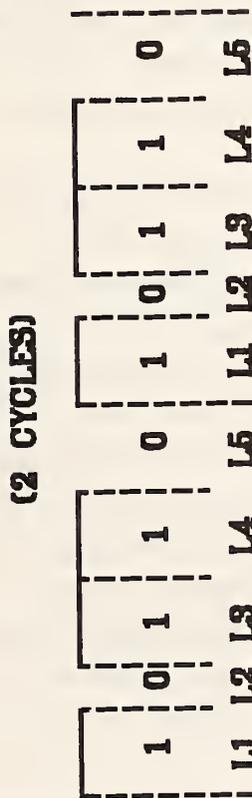
FIGURE 4-31 CONSTRUCTION OF LINE FONTS

313

4.3.6   **MACRO Capability**

4.3.6.1   <u>General</u>.   This Specification provides a means for communicating 3-dimensional and 2-dimensional geometric models and drawings.   The Specification, however, does not provide a format for every geometric or drafting entity available on all currently used CAD/CAM systems, and is thus a common subset of such entities.   To allow exchange of a larger subset of entities - a subset containing some of the entities not defined in this Specification but which can be defined in terms of the basic entities, the MACRO capability is provided.   This capability allows the use of the Specification to be extended beyond the common entity subset, utilizing a formal mechanism which is a part of the Specification.

The MACRO capability provides for the definition of a "new" entity in terms of other entities.   The "new" entity schema is provided in a MACRO definition which occurs once for every "new" entity in the file.   Instances of these "new" entities are replaced during the MACRO processing by the constituent entities specified in the corresponding MACRO definition.

A MACRO definition is written using the MACRO definition entity.   The parameter section of the entity contains the MACRO body.   In the MACRO body, eleven types of language statements are usable.   The statements are LET, SET, REPEAT, CONTINUE, BREAK,   IF, LABEL, GOTO, MACRO, ENDM, MREF.   The details of the MACRO syntax are given in 4.3.6.2. Each of the statements in a MACRO definition entity is terminated by a record delimiter.

In order to use a "new" entity defined by the MACRO definition, a MACRO instance is placed in the file.   The directory entry portion of an instance specifies the new entity type number in field 1 and 11 of the directory entry record and refers to the definition by a pointer in the structure field (DE field 3).   The parameters for the instance are placed in the parameter record of the instance.

The directory entry record of a MACRO definition has a standard form.

The attributes 4 through 9, 12, 13, 15, 18, and 19 have no significance. The default values for these attributes are taken from the directory entry record of the MACRO instance (described in 4.3.6.4).

The parameter data records of a MACRO definition consist of MACRO language statements. The statements are not in Hollerith form, i.e., they have no preceding "H" specification. The statements are free format and may branch over record boundaries (see 2.2.3). Every statement is terminated by a record delimiter.

4.3.6.2   MACRO Syntax

Constants. Constants may be integer, real, double precision real, pointer or string (See 2.2.2).

Variables. Variable names may be from one to six characters in length. The first character must be one of the characters listed below. This character determines the variable type. It is not possible to override the conventions. The six character limitation includes the first character. Upper and lower case letters are recognized as distinct, i.e., X is different from x. Variable names longer than six characters may be used; however, only the first six characters will be significant. Variable names may contain imbedded blanks. These blanks are NOT taken as part of the name; therefore "A B" is equivalent to "AB." Except for the first character, as outlined below, all characters must be alphabetic (A-Z or a-z), or numeric (0-9).

Variable type	First character
Integer	I-N, i-n
Real	A-H, O-Z a-h, o-z
Double precision real	!
String	$
Pointer	#
Label	&

Examples of valid variable names are:

Integer:	IJK	ICOUNT	K101	NTIMES	max
Real:	XYZ	X1	y2	QrsTul	

Double precision real:

	!h	!xi	!Y2	!12341
String:	$str	$TITLE	$label	
Pointer:	#line	#note	#REF	#XYZ1

Some invalid variable names:

$$$$	($ not permitted after first character)
1X43B	(1 may not be first character)
A.BC	(. is illegal)

Note that there are no "reserved" words. Thus a variable name such as "MACRO", which is identical to a statement keyword (described below), will not confuse the interpreter, although it may confuse the user of such a MACRO. It is suggested that these words be avoided.

Functions. Functions similar to FORTRAN library functions are provided. The rules for mixed mode have been relaxed so that it is not necessary to use SQRT(2.) instead of SQRT(2). While this assists the preprocessor writer in preparing MACROs, it places a responsibility on the writer of a processor for the MACRO language in handling the mixed mode. While the arguments can be mixed mode, the functions do have a specific type of value that they return, i.e., integer, real, double precision real, or string. The functions are listed here by the type of value returned. The type of argument usually used is also noted for clarity. For example, either IDINT(!d) or INT(!d) will work

equally well, although the meaning might be a little clearer with IDINT(!d). Functions are only recognized in upper case.

FUNCTIONS RETURNING INTEGER VALUES:

IABS(i)

      Returns the absolute value of i.

IDINT(!d)

      Returns integer part of !d.

IFIX(x) or INT(x)

      Returns the integer part of x.

ISIGN(i)

      Returns 1 if i is positive, 0 if it is zero, or -1 if it is negative.

FUNCTIONS RETURNING REAL VALUES:

ABS(x)

      Returns absolute value of x.

AINT(x)

      Returns integer part of x, in real form.

ALOG(x)

      Returns natural logarithm of x.

ALOG10(x)

      Returns common (base 10) logarithm of x.

ATAN(x)

      Returns arctangent of x; angle returned in radians.

COS(x)

      Returns cosine of angle x; angle in radians.

EXP(x)

      Returns natural anti-logarithm of x ("e to the x").

FLOAT(i)

      Returns a real (floated) value for i, e.g., FLOAT(2) returns "2."

SIGN(x)

      Returns 1. if x is positive, 0. if x is zero, and -1. if x is negative.

SIN(x)

> Returns sine of angle x; angle in radians.

SNGL(!d)

> Returns single precision (real) value of double precision variable !d. As many significant digits of !d as possible are used in the returned value.

SQRT(x)

> Returns square root of x.

TAN(x)

> Returns tangent of angle x; angle in radians.

FUNCTIONS RETURNING DOUBLE PRECISION REAL VALUES:

DABS(!d)

> Returns absolute value of !d.

DATAN(!d)

> Returns arctangent of !d; value returned in radians.

DBLE(x)

> Returns double precision real value of x. Note that this is merely a conversion, not an extension. Thus, DBL(.333333333) will return .333333333D0, but not .33333333333333333333333333D0. Thus, DBLE(1./3.) is not necessarily equal to 1D0/3D0.

DCOS(!d)

> Returns cosine of angle !d; angle in radians.

DEXP(!d)

> Returns natural anti-logarithm of !d(i.e., e to the !d).

DLOG(!d)

> Returns natural logarithm of !d.

DLOG10(!d)

> Returns common (base 10) logarithm of !d.

DSIGN(!d)

> Returns 1D0 if !d is positive, 0D0 if zero, -1D0 if negative.

DSIN(!d)

> Returns sine of angle !d; angle in radians.

DSQRT(!d)

> Returns square root of !d.

DTAN(!d)
>   Returns tangent of angle !d; angle in radians.

FUNCTIONS RETURNING STRING VALUES:

STRING (expression, format)
>   Returns the character representation of the argument "expression". See
>   section 4.3.6.3.3 for a description of the argument "format".

Expressions. Expressions may be formed using the above functions, variables
and constants, and the following operators:

Function	Symbol
addition	+
Subtraction	-
multiplication	*
division	/
exponentiation	**

The operators are evaluated in normal algebraic order, i.e., first
exponentiation, then unary negation, then multiplication or division, then
addition or subtraction. Within any one hierarchy, operators evaluate left to
right. Parentheses may be used to override the normal evaluation order, as in
the expression "A*(B+C)," which is different from "A*B+C." Extra
parentheses do not alter the value of the expression; it is a good idea to use
them, even if not truly necessary. Examples of expressions include:

X+1.0
-B+SQRT(B**2 - 4*A*C)
I + 1
3.14159/2.
-X
!DEL*(!ALPHA-!BETA)

Except for the ** operator, it is never permissable to have two operators next to each other, i.e., not 2*-2, but -2*2 or 2*(-2). Multiplication may not be implied by parentheses, e.g., (A+B)(C+D) is invalid, and AB does not imply A*B, but rather the separate variable AB.

Mode of expression evaluation. Mixed mode (integer mixed with real, etc.) is permitted. Whenever two different types are to be operated upon, the calculation is performed in the "higher" type. Integer is the lowest type, real is next, and double precision real is the highest. Note, however, that this decision is made for each operation, not once for the entire expression. Thus 1/3 + 1.0 evaluates to 1.0, because the "1/3" is done first, and it is done in integer mode. Integer mode truncates fractions, and does not round. Therefore, the expression "2/3+2/3+2/3" has a value of zero.

Conditional expressions

Conditional expressions may be formed using functions, variables and constants and the following six standard relational operators:

Function		Symbol
less than or equal	($\leq$)	.LE.
less than	($<$)	.LT.
equal	($=$)	.EQ.
greater than or equal	($\geq$)	.GE.
greater than	($>$)	.GT.
not equal	($\neq$)	.NE.

Examples of conditional expressions include:

X.GT.3
SQRT(A+B).NE.I+1
(!A-!B).GE.3.14159

4.3.6.3   MACRO Definition Entity. The MACRO definition entity specifies the action of a specific MACRO. After having been specified in a definition entity, the MACRO can be used as often as necessary by means of the MACRO instance entity.

The MACRO definition entity differs from other entity structures in this Specification by consisting of only language statements in the parameter data. The character strings constituting the language statements in the MACRO definition are not set off by means of the nH structure of string constants but rather consist of only the actual character string terminated by a record delimiter (see 2.2.2.5).

4.3.6.3.1   Directory Data
ENTITY TYPE NUMBER: 306

4.3.6.3.2   Parameter Data

Index	Name	Type	Description
1	ID	Literal	MACRO
2	NE	Integer	Entity Type ID
3	TEXT	Language statement	First statement
4	TEXT	Language statement	Second Statement
.	.	.	.
.	.	.	.
.	.	.	.
N+2	TEXT	Language statement	Last statement
N+3	T	Literal	ENDM

Additional parameters as required (see 2.2.4.4.2).

4.3.6.3.3   There are eleven language statements that can be used. They are:

LET
SET
REPEAT
CONTINUE
BREAK
IF
LABEL
GOTO
MACRO
ENDM
MREF

These "keywords" are recognized only in upper case, and every statement must begin with one of these keywords. Statements are free format; blanks and tabs are ignored except within strings. Statements may extend over several lines, or more than one statement may be present on a line. All statements are terminated by a record delimiter which must be present.

LET statement (Arithmetic)

This is the assignment statement and is equivalent to the LET statement of BASIC. The format of a LET statement is:

LET variable = expression;

The expression and the variable may be integer, real, or double precision; they need not be of the same type. Note that this is an assignment statement and not an algebraic equality. All of the variables on the right hand side of the expression must have been previously defined; it cannot be assumed that variables will default to zero if they are undefined. Some examples of valid LET statements:

LET HYPOT = SQRT(A**2+B**2);
LET X = X + 1;
LET ROOT1 = -B + SQRT(B*B - 4*A*C);
LET I = i;
LET !XYZ = I * 2;
LET START = 0;

LET Statement (String)

String variables allow characters to be manipulated. String variables may be used in statements almost anywhere that any other variable type may be used; exceptions are noted below.

String variables may be used in LET statements. Note that they shall not be mixed with any other type of variable in a LET statement. Also note that arithmetic operations (i.e., +,-, *, /,**) are not possible with string variables. Two forms of LET statements for string variables are possible:

> LET $str = 23Hstring of 23 characters;
> or
> LET $str1 = $str2;

In the first case, the 23 characters following the H are assigned to the string variable $str. In the second case, the string "$str2" is copied into "$str1." Examples of these statements include:

> LET $title = 3HBox;
> LET $subti = 6HBottom;
> LET $x = $subti;

Note that if a string variable appears on the right hand side of the statement, it must have been previously defined. Spaces are not ignored within a string constant; they become part of the string. Any printable ASCII character may be part of a string.

There is one other form for setting up a string. It involves the STRING function. The STRING function may only appear in this form. Specifically, it shall not appear in SET statement argument lists, MACRO statements, or MREF statements. However, string constants, such as "6Hstring," and variables, such as "$x" , may appear in SET statements and MACRO statements.

The form of a LET statement including string function is:

LET $str = STRING(expression, format);

where "expression" is any normal integer, real or double precision real expression, "$str" is a string variable name, and "format" is a format specification as used in a FORTRAN FORMAT statement. The allowable format specifications are:

    Iw
    Fw.d
    Ew.d
    Dw.d

The effect of this statement is to convert the numeric value of the expression into characters, e.g., the statement:

    LET $PI = STRING(3.14159,F7.5);
    will result in the same thing as
    LET $PI = 7H3.14159;

Of course, the usefulness of the STRING function is that expressions can be converted, rather than constants. Thus:

    LET x = 1;
    LET y = 2;
    LET $xyz = STRING(x+y+1,F5.0);
    will result in the same thing as
    LET $xyz = 5H   4.;

The rules for the format specifications follow the standard FORTRAN convention. "Iw" causes integer conversion, resulting in "w" characters. "Fw.d" causes real conversion, resulting in "w" characters, with "d" characters after the decimal point. "Ew.d" results in real conversion, but using an exponent form. "Dw.d" is the same as "E" but for double precision real values. Note that this is one place where mixed mode is not allowed. The type of format specification and the type of the expression's result must be identical.

LET statements (Attributes)

Attributes (those appearing in the directory entry record for the MACRO instance) may be set using the LET statement. The format is:

> LET /attribute name = expression;
> or
> LET /attribute name = /HDR;

The first form allows an attribute to be set to any constant value, including numeric expressions. Attributes may also be set to string constants or string variables, but not to the result of a STRING function.

Examples:

> LET /LEV = 1;
> LET /VIEW = 3;
> LET /LABEL = 6HBottom;
> LET /LABEL = $X;

The second form allows restoring an attribute to its default value.

Examples:

> LET /LEV = /HDR;
> LET /LABEL = /HDR;

The word "/HDR" is the only non-constant that is allowed on the right side of an attribute assignment statement. The effect is to restore the value of the attribute to what it was in the directory entry for the instance or, in some cases, to a specified default value. The defaults are described below.

Attributes may not be mixed with any other variable type nor may they appear anywhere but in the above two forms of LET statements.

The allowable attribute names and their defaults are given here. A default of /HDR indicates that the attribute defaults to the value in the directory entry of the instance.

Attribute	Name	Default
Line font pattern	/LFP	/HDR
Level	/LEV	/HDR
View	/VIEW	/HDR
Transformation matrix	/MTX	/HDR
Label display associativity	/CE	0
Blank status	/BS	/HDR
Subordinate entity	/SE	/HDR
Entity use	/ET	/HDR
Hierarchy	/HF	/HDR
Line weight	/LW	/HDR
Color	/PN	/HDR
Form	/FORM	0
Entity label	/LABEL	blanks
Entity subscript	/SUB	0

SET statement

The SET statement establishes directory and parameter data entries for the specified entity. The form is:

      SET #ptr = entity type number, argument list;

"#ptr" is a pointer variable, such as "#XYZ"; "entity type number" is an IGES entity type number, such as "110"; and "argument list" is a group of variables which is the parameter data of the entity.

Examples of this type of SET statement:

      SET #LINE = 110,X1,Y1,Z,X2,Y2,Z,0,0;
      SET #ABC = 828,Z,A+B/C,Y1,X2,Y2+1,0,0;
      SET #qwe = 864,15Hstrings allowed, X,Y,$this2;

The argument list may contain expressions and may spread over more than one line. At least one argument must be present, i.e., the argument list may not be null. The entity type number may not be an expression; i.e., it must be an integer constant. The pointer variable will be assigned a value corresponding to the sequence number of the directory entry of the entity created.

"Forward referencing" of pointers is valid in the argument list of a SET statement. A pointer may appear in the argument list of a SET statement that comes before the SET statement defining the pointer. The only restriction is that any pointer so referenced must appear on the left hand side of one SET statement.

Pointers which appear on the left hand side of more than one SET statement or those which are located inside of REPEAT loops should not be forward referenced.

Note that the STRING function is not allowable in a SET statement -- use a separate LET statement with a string variable instead.

REPEAT statement

The REPEAT statement causes a group of statements terminated by a CONTINUE statement to be repeated a specified number of times. The form of a REPEAT statement is:

REPEAT expression;

The expression is evaluated, and the resulting value is the number of times the statements will be repeated. The expression may be of integer, real or double precision real type; in the case of real or double precision real expressions, the result is truncated to determine the repeat count. If the repeat count is zero or negative, the group of statements is still executed one time.

Examples of REPEAT statements:

```
REPEAT 3;
REPEAT N+1;
REPEAT 0;    (will execute the loop one time)
REPEAT X+Y;
```

REPEAT statements may be nested only to a depth of ten.

After a REPEAT statement, such as REPEAT N, it is valid to alter the value of N. This does not affect the repeat count. Also note that REPEAT is unlike a FORTRAN DO statement because there is no variable being incremented on every pass.

CONTINUE statement

The CONTINUE statement marks the end of a REPEAT group. The form of a CONTINUE statement is:

```
CONTINUE;
```

When a CONTINUE statement is encountered, the repeat count is decremented by one and checked to see if it is greater than zero. If it is, the interpreter goes back to the first statement after the most recent REPEAT. If not, then the next statement is processed. The number of REPEAT statements and CONTINUE statements in a MACRO shall be the same. A CONTINUE statement(s) is not implied by ENDM.

BREAK statement

A BREAK statement may be used within a REPEAT construct to terminate
the processing of statements of the REPEAT construct before the completion
of the specified number of loops, such as upon detection of a condition during
the processing. The form of a BREAK statement is:

        BREAK;
          or
        IF conditional expression, BREAK;

When a BREAK statement is encountered, processing of MACRO statements
resumes with the statement immediately following the CONTINUE statement
marking the end of the affected REPEAT construct.

IF statement

The IF statement causes a single language statement to be executed if a
certain condition is true.

The form of an IF statement is:

    IF conditional expression, language statement;

where "conditional expression" is a conditional expression as described in
section 4.3.6.2. "language statement" can be any statement allowed in a
MACRO except:

        MACRO
        ENDM
        IF
        LABEL

Examples of IF statements:

        IF A.LT.3,LET A=3;
        IF B.EQ.0,SET   #LIN1=110,...;
        IF SWITCH.EQ.1, GOTO &A;

LABEL statement

The LABEL statement is used to mark a position in a MACRO where the execution control is transferred to using a GOTO statement.

The form of a LABEL statement is:

    LABEL label name;

where "label name" is any character string beginning with an ampersand (&). It should be from one to six characters long (including the &). Within a single MACRO definition, all label names must be unique.

Examples:

        LABEL &loop;
        LABEL &end;

GOTO statement

This statement is used to transfer the execution control to a particular point which is marked by a LABEL statement.

The form of a GOTO statement is:

        GOTO label name;

where 'label name' is any label name specified in a LABEL statement.

The GOTO statement can be used to jump both forward and backward, but both the GOTO statement and the target LABEL must be at the same nesting level and within the same REPEAT construct.

Examples:

        GOTO &start;
        GOTO &end;

MACRO statement

The MACRO statement is used to signify the start of a MACRO definition. The first statement in every MACRO definition must be a MACRO statement.

The form of a MACRO statement is:

306,MACRO, entity type number, argument list;

where "entity type number" is the assigned entity number of the MACRO, and "argument list" is a list of parameters that are to be assigned values at execution time. Entity type numbers in the range of 600 to 699 and 10000 to 99999 will be used.

The argument list may not be null. The parameters in the argument list take the form of the variables described in Section 4.3.6.2. Note that the argument list may not contain expressions, only symbolic variable names.

One additional type of variable, the "class variable" can be used in an argument list. The class variable takes the form:

size_variable (class_var_1, class_var_2, ..., class_var_n)

where size_variable precedes the occurrence of the class variable in the argument list and class_var_i members are referenced in the MACRO body by means of a subscript:

class_var_i (J)

In the MACRO statement, the size_variable is used to identify the class variable collection being defined. In the MACRO body, the size_variable indicates the number of sets of class variable members that are included (i.e. the number of times the class variable collection is to be repeated).

A simple class variable example is:

N(ITEM1, ITEM2, ITEM3, ITEM4)

which specifies a class variable collection with four members. For an instance of the MACRO, using the example class variable with N equal to 3, three sets of class variable data for the collection are available to the macro body statements. The parameter list for the associated MACRO instance is:

ITEM1(1), ITEM2(1), ITEM3(1), ITEM4(1),...,ITEM3(3), ITEM4(3)

Each value for each member of the class variable may be referenced individually:

ITEM1(1), ITEM1(2), ITEM1(3), ITEM1(4), etc.

in any order, or implicitly, by using an index variable, e.g., J:

ITEM3 (J) with J ranging from 1 to 3

Use of the class variable to represent a MACRO with a parameter list identical to the Views Visible Associativity, Form 4, would be specified as:

306,MACRO,681,N1,N2,N1(#DEV,LF,#DEF,IPN,LW),
       N2(#DE),N,N(#DEA),M,M(#DEP);

where   N1(#DEV,LF,#DEF,ICN,LW) indicates the blocks of views visible, line font, color number, and line weight information;
N2(#DE) contains the pointers to the entities included in the view;
N(#DEA) contains the back pointers/text pointers; and
M(#DEP) contains the pointers to properties.

332

Note that zero is a valid value for a size_variable in a MACRO instance.

ENDM statement

ENDM signifies the end of a MACRO. The form of an ENDM statement is:

ENDM;

All MACROs must have an ENDM statement as their last statement. ENDM is not implied by the end of the parameter data section.

MREF statement

The MREF statement is used to reference another MACRO from inside a MACRO definition. The form of an MREF statement is:

MREF, ptr, entity type number, argument list;

where "ptr" may be either a pointer variable or an integer expression. The value is the sequence number of the directory entry record of the MACRO definition being referenced. "Entity type number" is the assigned entity number of the MACRO being referenced. "Argument list" is exactly like that of a SET statement. The effect of the argument list is to replace the symbolic names found in the MACRO definition with the values of the expressions contained in the MREF statement, so that execution of the referenced MACRO will start with the appropriate values. Note that MREF does not start expansion of the referenced MACRO. Rather it creates an entity entry which may later be expanded. It is thus not possible for a MACRO being referenced to have access to any of those values except for those in the argument list. (All variables not in the argument list are treated as local variables.) Even then, it is not possible for the occurrence of a MREF statement to alter any of those values.

Examples of MREF statements:

MREF,#mac1,600,X1,Y1,Z1,X2,Y2,3.1;
MREF,33,621,A,B,3+X/W+1,6*W,3.,0,6Hstring,$x;

It is not strictly necessary for the values in a MREF statement to be of the same type as the values in the definition MACRO, within certain limitations. Integer, real, and double precision real values may be freely mixed, although it might be considered a good idea not to do so. String values may only appear where string variables appear in the definition.

4.3.6.4   MACRO Instance Entity.   A MACRO instance entity is used to invoke a MACRO. The parameter data records of the instance contain values for the arguments to the MACRO. This is similar to a standard entity entry.

The directory entry for a MACRO instance entity contains the attribute values that are to be used as the default values during the expansion of the MACRO. The only special field is the structure field (directory entry field 3), which contains a pointer to the directory entry of the MACRO definition. The pointer is preceded by a minus sign.

4.3.6.4.1   Directory Data

ENTITY TYPE
NUMBER:

As defined for each MACRO in the range 600 to 699 or 10000 to 99999.

Structure field:

Pointer to directory entry of MACRO definition, preceded by a minus sign.

Other attributes:

Default values to be used during expansion of the MACRO. Attributes listed as defaulting to /HDR obtain their values from here. (See discussion of LET statement (attributes) in Section 4.3.6.3.3.)

335

4.3.6.4.2   Parameter Data

The parameter data section for an instance has the following form:
(Note that, as with all parameter data entities, the first record begins with the entity type number as defined in the MACRO.)

Index	Name	Type	Description
1,...K	As appropriate for the particular MACRO		The values for the arguments to the MACRO. They must agree in format and number with the arguments in the MACRO statement of the definition.

Additional pointers as required (See 2.2.4.4.2)

4.3.6.5   Examples of MACRO usage.

Five examples are given to illustrate some of the capabilities of a MACRO.
1. Isosceles Triangle
2. Repeated Parallelograms
3. Concentric Circles
4. Ground Symbol
5. Useful Features

Example 1:

The following MACRO definition creates an isosceles triangle, given a vertex point, a base width, a height and a scale.

Directory Data
ENTITY TYPE NUMBER: 306

Parameter Data

306, MACRO, 621, X1, Y1, A1, A2, K;

LET Z = 0;

SET #Line1   =   110,X1,Y1,Z,X1+(K*A1),Y1+(K*A2/2.),Z,0,0;

SET #Line2   =   110,X1+(K*A1),Y1+(K*A2/2.),Z,
                   X1+(K*A1),Y1-(K*A2/2.),Z,0,0;

SET #Line3   =   110,X1+(K*A1),Y1-(K*A2/2.),Z
                   X1,Y1,Z,0,0;

ENDM;

The MACRO can be used to create a triangle by using a MACRO instance which supplies the needed values for X1, Y1, A1, A2, and K. The parameter data section for the MACRO instance would have the following format:

Index	Name	Type	Description
1	X	Real	X coordinate of vertex
2	Y	Real	Y coordinate of vertex
3	A1	Real	Height of triangle
4	A2	Real	Base of triangle
5	K	Integer	Scaling factor

In particular, to create a triangle shown in Figure 4-32 with a base of 5. and a height of 17., a vertex at (0,0), and a scale factor 1, the following instance could be placed into the file:

Directory Data:

ENTITY TYPE NUMBER: 621

Structure: -nnn, where "nnn" is the sequence number of the directory entry of the definition.

Other attributes: As desired for default values during MACRO expansion.

Parameter data:

621, 0., 0., 17., 5., 1;
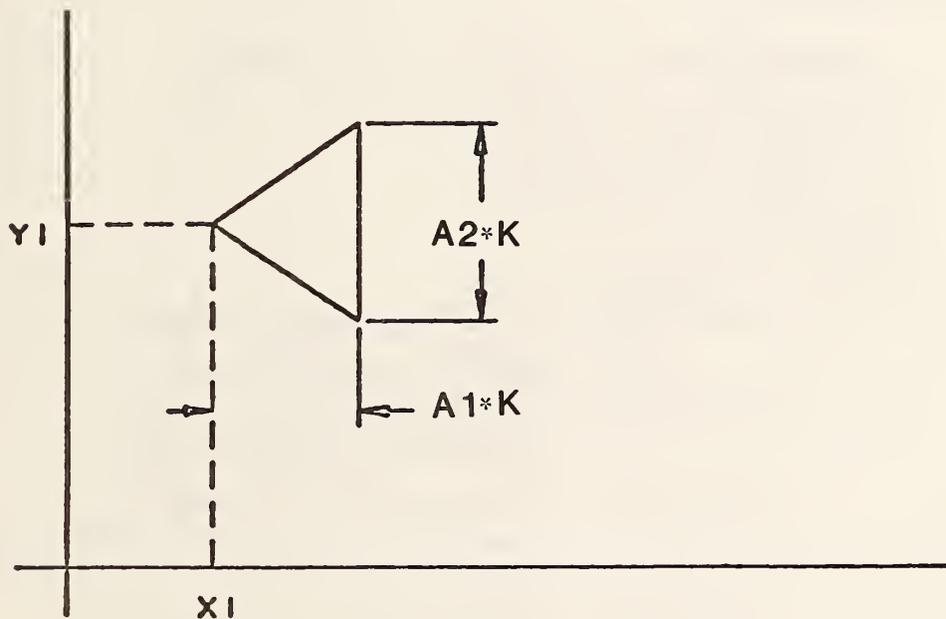
FIGURE 4-32 EXAMPLE OF TRIANGLE MACRO

Example 2: Repeated parallelograms

The following MACRO takes the coordinates of three points and a repetition number as arguments and creates a pattern of repeated parallelograms as shown in Figure 4-33. The three points represent the vertices of the initial parallelogram. The repetition number argument (NTANG) controls how many additional parallelograms will be drawn offset in the positive X and Y direction from the initial one. For simplicity, the points have been constrained to all lie in a plane parallel to the X-Y plane.

Directory Data
ENTITY TYPE NUMBER: 306

Parameter Data

```
        306,MACRO, 600, X1, Y1, X2, Y2, X3, Y3, Z, NTANG;
        IF NTANG.EQ.0, GOTO &END;
        LET XDEL = (X2-X1)/NTANG;
        LET YDEL = (Y3-Y1)/NTANG;
        LET K = 0;
        REPEAT NTANG +1;
                SET #VLINE =      110,X1+K*XDEL,Y1+K*YDEL,Z,
                                  X2+K*XDEL,Y2+K*YDEL,Z,0,0;
                SET #HLINE =    110, X1+K*XDEL,Y1+K*YDEL,Z,
                                  X3+K*XDEL,Y3+K*YDEL,Z,0,0;
                SET #VLINE =    110,X3+K*XDEL,Y3+K*YDEL, Z,
                                  X3+(X2-X1)+K*XDEL,Y2+(Y3-Y1)+K*YDEL,
                                  Z, 0,0;
                SET #HLINE =    110,X2+K*XDEL,Y2+K*YDEL, Z,
                                  X3+(X2-X1)+K*XDEL,Y2+(Y3-Y1)+K*YDEL,
                                  Z, 0,0;
                LET K  =  K + 1;
        CONTINUE;
        LABEL &END;
        ENDM;
```

Parameter Data for an instance of this MACRO looks like this:

600, 1., 1., 2., 5., 5., 2., 1., 3;
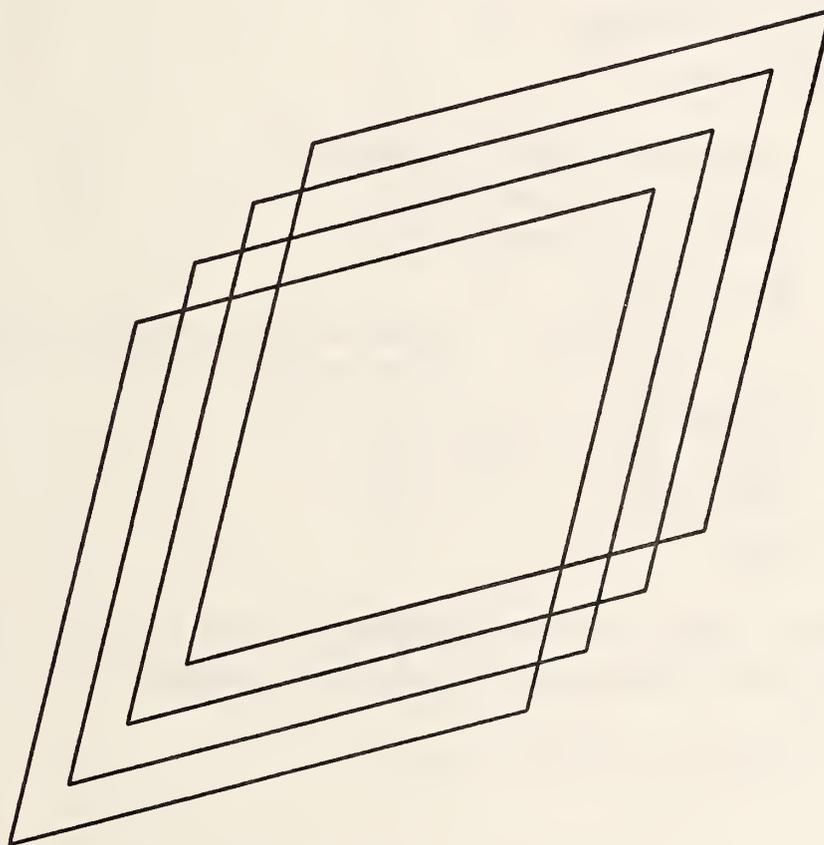
FIGURE 4-33 REPEATED PARALLELOGRAMS

Example 3: Concentric circles

The following MACRO, given a coordinate, a radius, and a number, creates concentric circles out to the radius. A point is put into the center. Figure 4-34 shows the result.

Directory Data
ENTITY TYPE NUMBER: 306
Parameter Data

```
306,MACRO,601,XC,YC,ZC,R,NCIRC;
IF NCIRC .EQ. 0, GOTO &END;
LET DELTR = R/NCIRC;
REPEAT NCIRC;
        SET #CIR =      100,ZC,XC,YC,XC,YC+R,XC,YC+R,0,0;
        LET R =         R - DELTR;
CONTINUE;
SET #PT = 116, XC, YC, ZC, 0, 0, 0;
LABEL &END;
ENDM;
```

Parameter Data for an instance of the MACRO which would create four concentric circles around the origin out to a radius of 20 looks like this:

```
601, 0., 0., 0., 20., 4;
```
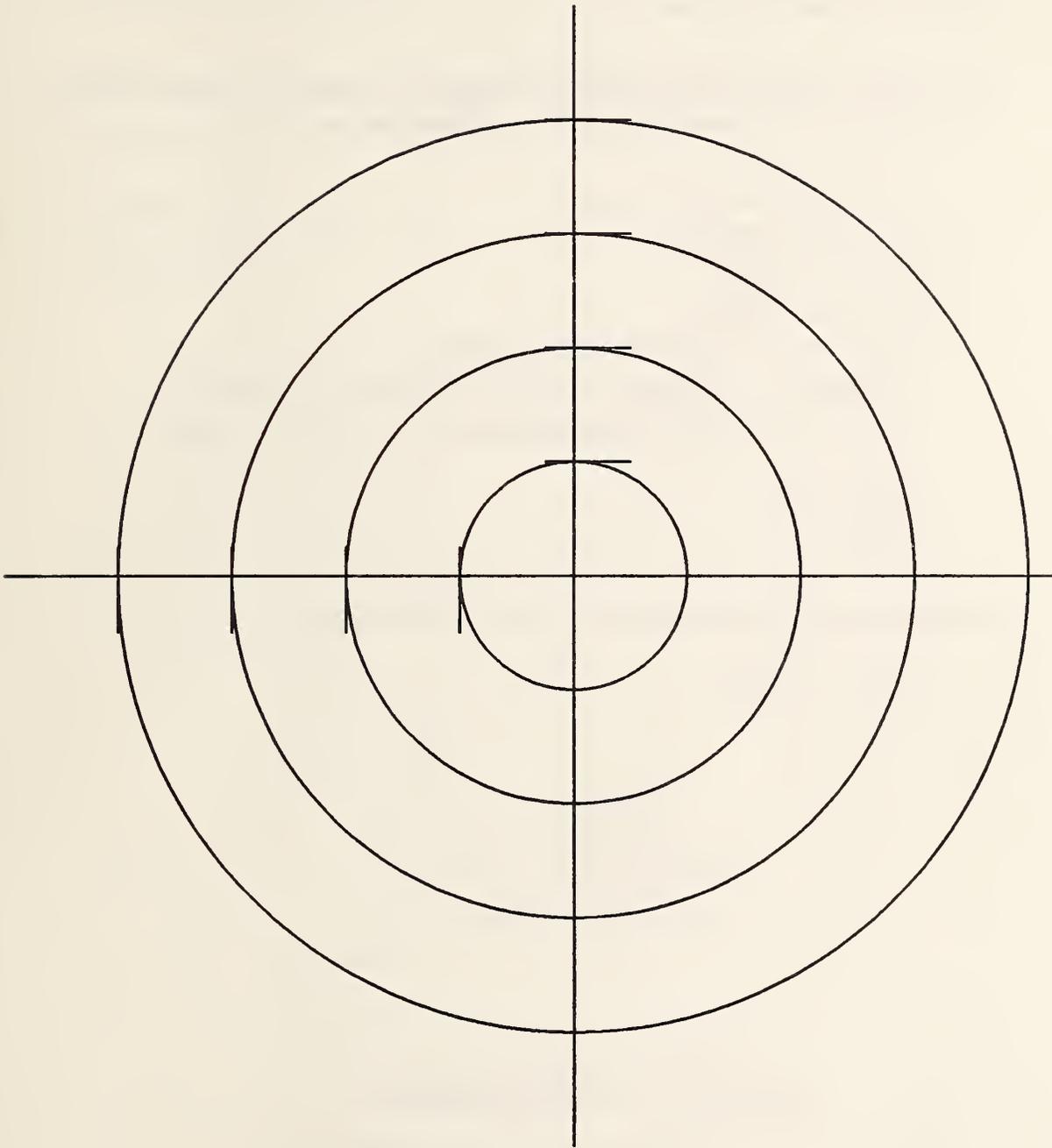
FIGURE 4-34 CONCENTRIC CIRCLES

Example 4: Electrical ground symbol

This MACRO takes a point and a base length and constructs a ground symbol (horizontally) at that point. Figure 4-35 shows the result.

```
306,MACRO, 602, X, Y, Z, B;
IF B.EQ.0, GOTO &A;
LET DELY = B/6;
LET DELX = DELY;
SET #LINE1 =   110, X, Y, Z, X+B, Y, Z, 0, 0;
SET #LINE2 =   110, X+DELX, Y-DELY, Z, X+B-DELX, Y-DELY, Z, 0, 0;
SET #LINE3 =   110,X+2*DELX,Y-2*DELY,Z,X+B-2*DELX,Y-2*DELY,Z,
 0, 0;
LABEL &A;
ENDM;
```

Parameter Data for an instance of this MACRO looks like this:

```
602, 1., 6., 2., 1.3;
```
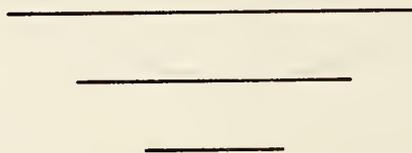
FIGURE 4-35 GROUND SYMBOL

Example 5: Useful features

This last example demonstrates the use of various MACRO features. It is not meant
as an example of a "useful" MACRO.

Directory Data
ENTITY TYPE NUMBER: 306
Parameter Data

```
306,MACRO,613,NROW,NCOL,VDIST,HDIST,!ANGLE;
LET/LABEL = 6HPOINTS;
LET !SIN=DSIN(!ANGLE); LET !COS=DCOS(!ANGLE);
LET YHD=HDIST*!SIN;
LET XHD=HDIST*!COS;
LET YVD=VDIST*!COS;
LET XVD=VDIST*(-!SIN);
LET IRC=0; LET ICC=0;
REPEAT NROW;
        LET XCOL=IRC*XVD;
        LET YCOL=IRC*YVD;
        REPEAT NCOL;
                LET X = XCOL + ICC*XHD;
                LET Y = YCOL + ICC*YHD;
                SET #PT = 116, X, Y, 0., 0,0,0;
                LET ICC = ICC + 1;
        CONTINUE;
        LET IRC = ICC +1;
        CONTINUE;
        LET $NPTS = STRING(NROW*NCOL,I7);
        LET/LABEL = $NPTS;
        SET #LINE = 110, 0., 0., 0., 10., 0., 0.;
        SET #CIRC = 100, 0., 0., 0., 10., 0., 10., 0.;
        MREF, 22, 601, 0., 0., 0., 10., 5;
        ENDM;
```

Parameter Data for an instance of this MACRO looks like this:

```
613, 4, 5, 0. 2, 0.1, 7.85398D-01;
```

4.3.7 Property Entity.

The property entity contains numerical or textual data. It also has a form number to indicate its meaning. Certain generic Property form numbers are described in the following sections and are expected to be augmented by others in future versions of this Specification. Form numbers in the range 5001 - 9999 are left undefined for users.

Note that properties can also point to other properties, as well as participate in associativities, point to related general notes, or display text by pointing to a Text Display Template.

Property instances are usually referenced by the presence of a pointer to the instance in the second group of additional pointers as described in section 2.2.4.4.2; however, as stated in section 1.5.1, when an instance is independent it applies to all entities on the same leel as the instance.

4.3.7.1 Directory Data
ENTITY TYPE NUMBER :   406

4.3.7.2 Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of property values
2		Variable	Property values
.			
.			
.			
1+N			

Additional Pointers as required (see 2.2.4.4.2).

4.3.7.3   Defined Properties

4.3.7.3.1   FORM NUMBER: 1   **Definition levels**

DESCRIPTION

For one or more entities in the file that are defined on a set of multiple levels, there will be an occurrence of the property instance (Form 1). In the parameter data portion of the property instance, the first parameter, N1, will contain the number of multiple levels followed by a list of those levels. Each entity that is defined on this set of levels will contain a pointer (in the level field of the directory entry) to this property instance. A different set of multiple levels will result in a different property instance.

Index	Name	Type	Description
1	N1	Integer	Number of property values
2	L	Integer	Level number
.	.	.	
.	.	.	
.	.	.	
1+N1	.	.	

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.2   FORM NUMBER: 2   **Region Restriction**

DESCRIPTION

This property allows entities that can define regions to set an applications restriction over that region.   The restrictions will indicate whether a given applications item must lie completely within regions with this property or completely outside such regions.   The restriction applies to all points of entities used to represent the applications item and to all points within the effect of the item when all properties, such as line widening, are applied.

Each of the property values in this property will have one of three values indicating the region restriction relevant to the application's item.

Property Value	Description
0	No Restriction
1	Item must be inside region
2	Item must be outside region

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=3)
2	EVR	Integer	Electrical vias restriction (EVR=0,1 or 2)
3	ECPR	Integer	Electrical components restriction (ECPR=0,1 or 2)
4	ECRR	Integer	Electrical circuitry restriction (ECRR=0,1 or 2)

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.3   FORM NUMBER: 3   Level Function

DESCRIPTION

This property is used to transfer the meaning or intended use of a level in the sending system. An instance of this property shall apply to all entities in the same file with the same DE level value (field 5), without the requirement of a pointer to it (see section 1.5.1). Parameter 2 is used to record an integer code number when the sending system uses a level-use index or table. Parameter 3 is used to record the level-use text, whether such text is obtained from the index which provided parameter 2, or exists independently. Either parameter 2 or parameter 3 may have a default value. This property may be readily added to a file (by edit or data merge) when level-use information is required by the receiving system or archive. The parameter (2 and 3) values of an instance of this property shall apply to multiple levels if the instance's level value is a pointer to an instance of Property Form 1. Note that parameter 3 was an integer value for "source level" in Version 2. The source level (for this Version) shall be the level value for the instance of this property.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=2)
2	FC	Integer	Function description code (Default = 0)
3	FD	String	Function Description (Default = null string)

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.4   FORM NUMBER: 4   **Region Fill Property**

DESCRIPTION

This property helps define the functional value of any closed region. It classifies the region as to its "filled" status. It will be used most often to identify which region-defining entities are defining a functional region (or a gap in that region) and which have other purposes. The actual function of the region will likely be determined in conjunction with level or subfigure membership.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=2)
2	FC	Integer	Fill code: 0 solid fill 1 unfill (i.e., a gap in solid fill) 2 meshed fill (indicates that an associativity is used to link the fill area with its fill mesh description). Using the associativity will allow the implementation of this obsolete method. The recommended method of mesh fill is to use Sectioned Area entity (230).
3	O	Pointer	Obsolete. Note: a previous implementation of this parameter was as a pointer to the DE of a section entity defining linear segments of meshed fill. This previous implementation would be indicated by a non-zero value.

Additional pointers as required (see 2.2.4.4.2).

348

4.3.7.3.5   FORM NUMBER: 5   **Line Widening**

DESCRIPTION

This property defines the characteristics of entities when they are used to define the location of items such as strips of metalization on printed wiring boards.

The justification flag terminology is interpreted as follows: right justified means that a defining line segment forms the right edge of the widened line in the direction from first defining point to second. Left justified is the opposite while center justified indicates that the defining line segment splits the widening exactly in half. Figure 4-36 indicates the measurement of the property values.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=5)
2	WM	Real	Width of metalization
3	CC	Integer	Cornering codes 0 rounded 1 squared
4	EF	Integer	Extension flag 0 No extension 1 One-half width extension 2 Extension set by parameter
5	JF	Integer	Justification flag 0 center justified 1 left justified 2 right justified
6	E	Real	Extension value parameter 4=2 (Note: this value may be negative)

Additional Pointers as required (see 2.2.4.4.2)
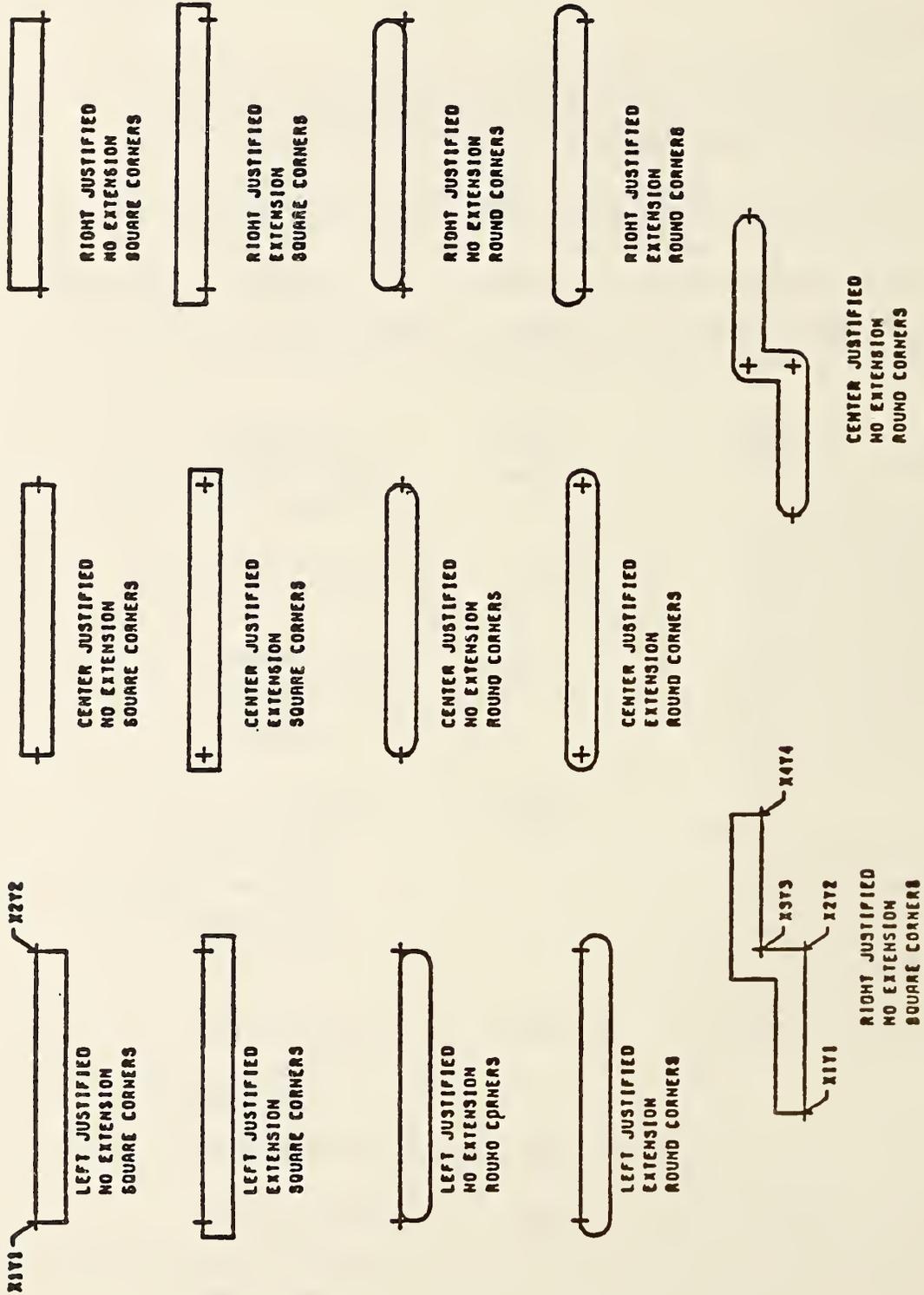
FIGURE 4-36 LINE WIDENING EXAMPLES

4.3.7.3.6   FORM NUMBER: 6   **Drilled Hole**

DESCRIPTION

The Drilled Hole property identifies an entity representing a drilled hole through a printed circuit board. The parameters of the property define the characteristics of the hole necessary for actual machining. The layer range indicated by parameters 5 and 6 refers to physical layers of the assembled printed circuit board.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=5)
2	DDS	Real	Drill diameter size
3	FDS	Real	Finish diameter size
4	PF	Integer	Plating indication flag (0=no, 1=yes)
5	LNL	Integer	Lower numbered layer
6	HNL	Integer	Higher numbered layer

Additional Pointers as required (2.2.4.4.2)

4.3.7.3.7   FORM NUMBER: 7   <u>Reference Designator</u>

DESCRIPTION

The Reference Designator property attaches a text string containing the value of a component reference designator to an entity being used to represent a component. This property is not to be used for the primary reference designator when a component is represented by a Network Subfigure Instance as that value is included in the subfigure parameters.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=1)
2	RD	String	Reference designator text

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.8   FORM NUMBER: 8   **Pin Number**

DESCRIPTION

The Pin Number property attaches a text string representing a component pin number to an entity being used to represent an electrical component's pin. This property is not to be used when a pin is represented by a Connect Point entity as the pin number is included in one of the Connect Point parameters.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=1)
2	PN	String	Pin Number Value

     Additional Pointers as required (see 2.2.4.4.2)

353

4.3.7.3.9   FORM NUMBER: 9   **Part Number**

DESCRIPTION

The Part Number property attaches a set of text strings that define the common part numbers to an entity being used to represent a physical component. Null text values in any parameter will imply that the missing value is not relevant to the transferred data.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=4)
2	GPN	String	Generic part number or name
3	MPN	String	MIL-STD part number
4	VPN	String	Vendor part number or name
5	IPN	String	Internal part number

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.10   FORM NUMBER: 10   Hierarchy

DESCRIPTION

The hierarchy property provides the ability to control the hierarchy of each directory entry attribute. This property is referenced when the directory entry status digits 7 and 8 are 02.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=6)
2	LF	Integer	Line font
3	VU	Integer	View
4	LAB	Integer	Entity level
5	BL	Integer	Blank status
6	LW	Integer	Line weight
7	CO	Integer	Color number

Additional Pointers as required (see. 2.2.3.4.2).

Acceptable values for parameters 2 through 7 are 0 and 1.
See definition in section 2.2.4.3.9.4.

355

4.3.7.3.11   FORM NUMBER: 11   **Tabular Data**

DESCRIPTION

The tabular data property provides a structure to accommodate point form data. The basic structure is a two dimensional array organized in column row order. In the simplified form this structure can contain a single list of values. The more complex forms contain multiple lists of independent and dependent variables.

The PROPERTY TYPE is the key used to define the dependent variable data values.

PROPERTY TYPES 1 to 5000 are reserved for defining finite element material properties.

The units used in these properties are specified in the MKSA (Meters, Kilograms, Seconds, Ampere) system according to the following table:

Length = meters
Mass   = kilograms
Time   = seconds
Current = amperes
Energy = joules
Force   = newtons

DE Form Number   11 = Tabular Data Property

Index	Name	Type	Description
1	N	Integer	# of Property values
2	PTYPE	Integer	Property Type 1 = Young's Modulus ND=3 2 = Poisson's Ratio ND=3

3 = Shear Modulus
ND=3

4 = Material Matrix
ND=21

5 = Mass Density
ND=1

6 = Thermal Expansion
Coefficient
ND=3

7 = Laminate Material
Stiffness Matrix
ND=6

8 = Bending Material
Stiffness Matrix
ND=6

9 = Transverse Shear
Material Stiffness
Matrix
ND=3

10 = Bending Coupling
Material Stiffness
Matrix
ND=6

11 = Material Coordinate
System
ND=3

12 = Nodal Load/Constraint
Data ND=# of Degrees
of Freedom

13 = Sectional Properties
for Beam Elements
ND=8 if the properties
are the same at both
ends of the beam
other wise,
ND=16

14 = Beam End Releases
ND=12

			15 = Offsets ND=9 or 18
			16 = Stress Recovery Information ND=12 or 24
			17 = Element Thickness ND=1 or n
			18 = Non-Structural Mass ND=1
			19 = Thermal Conductivity ND=3
			20 = Heat Capacity ND=1
			21 = Convective Film Coefficient ND=1
			22 = Radiation Parameters ND=4 . . . n = Open Ended
3	ND	Integer	# of dependent variables
4	NI	Integer	# of independent variables
5	TYP_1	Integer	Type of first independent variable 1 = Temperature 2 = Pressure 3 = Relative humidity 4 = Rate of Strain 5 = Velocity 6 = Acceleration 7 = Time 8 = Strain . . n = Open Ended
4+NI	TYPNI	Integer	Type of last independent variable

Index	Name	Type	Description
5+NI	$NVALI_1$	Integer	Number of different values of the <u>first</u> independent variable
.	.		
.	.		
.	.		
4+2*NI	$NVALI_{NI}$	Integer	Number of different values of the <u>last</u> independent variable
5+2*NI	$VALI_1$	Real	First value of <u>first</u> independent variable
	$VALI_{NI}$	Real	Last value of <u>last</u> independent variable
M+1	VALD (j,k)	Real	Value of the dependent variable where

$$M + ND \cdot \underbrace{\prod_{i=1}^{NI} NVALI_i}_{a} = \underbrace{M + ND \cdot a}_{N+1} \qquad J=1...ND; \; k=1...\underbrace{\prod_{i=1}^{NI} NVALI_i}_{a}$$

$$VALD \,(1,1) = f_1(VALI_1(1), VALI_2(1), \ldots\ldots\ldots\ldots\ldots VALI_{NI}(1))$$
.
.
.
$$VALD \,(ND,1) = f_{ND}(VALI_1(1), \ldots\ldots\ldots\ldots\ldots VALI_{NI}(1))$$

$$VALD \,(1,2) = f_1(VALI_1(2), \ldots\ldots\ldots\ldots\ldots VALI_{NI}(1))$$
.
.
.
$$VALD \,(ND,2) = f_{ND}(VALI_1(2), \ldots\ldots\ldots\ldots\ldots VALI_{NI}(1))$$
.
.
.
$$VALD \,(1,a) = f_1(VALI_1(NVALI_1) \ldots\ldots\ldots\ldots\ldots VALI_{NI} \,(NVALI_{NI}))$$
.
.
.
$$VALD(ND,a) = f_{ND}(VALI_1(NVALI_1) \ldots\ldots\ldots\ldots\ldots VALI_{NI} \,(NVALI_{NI}))$$

Additional Pointers as required (see 2.2.4.4.2).

MATERIAL PROPERTY TYPE DEFINITIONS:

PTYPE = 1   Young's Modulus

Young's modulus relates stress to strain in materials. In the simple case:

$$\bar{\sigma} = E \, \bar{\epsilon}$$

Where $\bar{\sigma}$ = Stress Vector (Row of Elements σ_x, σ_y, σ_z)
       $\bar{\epsilon}$ = Strain Vector (Row of Elements ϵ_x, ϵ_y, ϵ_z)
       E = Young's Modulus Matrix of Diagonal Elements
               E_{xx}, E_{yy}, and E_{zz}

The modulus is a vector with three principal values E_{xx}, E_{yy}, and E_{zz}. This implies that ND (Number of Dependent Variables) is equal to three.

In matrix form:

$$
\begin{vmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{vmatrix} =
\begin{vmatrix} E_{xx} & 0 & 0 \\ 0 & E_{yy} & 0 \\ 0 & 0 & E_{zz} \end{vmatrix}
\begin{vmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \end{vmatrix}
$$

PTYPE = 2   Poisson's Ratio

Poisson's ratio is the ratio of transverse strain in the j-direction when stressed in the i-direction, i.e.,

$$\nu_{ij} = \frac{\epsilon_j}{\epsilon_i}$$

Where ν = Poisson's Ratio
         ϵ = Strain
         i = One Orthogonal Direction
         j = Another Orthogonal Direction

The Poisson's Ratio is a vector consisting of matrix elements with the three principal values, ν_{XY}, ν_{XZ}, ν_{YZ}

The other off diagonal matrix values are reciprocals of the principal values.

i.e., $\nu_{xy} = \dfrac{1}{\nu_{yx}}$

This implies that ND (Number of Dependent Variables) is equal to three.

In matrix form for an orthotropic material:

$$
\begin{vmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \end{vmatrix} =
\begin{vmatrix}
\dfrac{1}{E_{xx}} & -\dfrac{\nu_{yx}}{E_{yy}} & -\dfrac{\nu_{zx}}{E_{zz}} \\[2ex]
-\dfrac{\nu_{xy}}{E_{xx}} & \dfrac{1}{E_{yy}} & -\dfrac{\nu_{zy}}{E_{zz}} \\[2ex]
-\dfrac{\nu_{xz}}{E_{xx}} & -\dfrac{\nu_{yz}}{E_{yz}} & \dfrac{1}{E_{zz}}
\end{vmatrix}
\begin{vmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{vmatrix}
$$

361

PTYPE = 3   Shear Modulus

Shear Modulus - The ratio of shear stress to shear strain.

$$G = \frac{\tau_s}{\gamma}$$

Where G is the Shear Modulus,
   τ_s is the Shear Stress, and
   γ is the Shear Strain

The Shear Modulus is a vector with three princpal values:

G_{xy}, G_{yx}, and G_{xz}.

This implies that the ND (Number of Dependent Variables) is equal to three.

In matrix form for orthotropic materials:

$$
\begin{vmatrix} \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{vmatrix}
=
\begin{vmatrix} \dfrac{1}{G_{xy}} & 0 & 0 \\ 0 & \dfrac{1}{G_{yz}} & 0 \\ 0 & 0 & \dfrac{1}{G_{zx}} \end{vmatrix}
\begin{vmatrix} \tau_{s_{xy}} \\ \tau_{s_{yz}} \\ \tau_{s_{zx}} \end{vmatrix}
$$

PTYPE = 4   Material Matrix

Material matrix defines the tensor qualitites of the material. For example:

$$\bar{\sigma} = |C|\bar{\epsilon}$$

Where $\vec{\sigma}$ is the Stress Vector,

         $\bar{\epsilon}$ is the Strain Vector, and

         $|C|$ is the Material Matrix

Because of symmetry, the elements $C_{ji.} = C_{ij}$. Therefore, 21 elements define the material matrix.

C_{11}	C_{44}	C_{46}
C_{12}	C_{15}	C_{56}
C_{22}	C_{25}	C_{66}
C_{13}	C_{35}	
C_{23}	C_{45}	
C_{33}	C_{55}	
C_{14}	C_{16}	
C_{24}	C_{26}	
C_{34}	C_{36}	

This implies that ND = 21.

In Matrix form:

$$
\begin{vmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{vmatrix}
=
\begin{vmatrix}
C_{11} & C_{12} & C_{13} & C_{14} & C_{15} & C_{16} \\
C_{21} & C_{22} & C_{23} & C_{24} & C_{25} & C_{26} \\
C_{31} & C_{32} & C_{33} & C_{34} & C_{35} & C_{36} \\
C_{41} & C_{42} & C_{43} & C_{44} & C_{45} & C_{46} \\
C_{51} & C_{52} & C_{53} & C_{54} & C_{55} & C_{56} \\
C_{61} & C_{62} & C_{63} & C_{64} & C_{65} & C_{66}
\end{vmatrix}
\begin{vmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{vmatrix}
$$

PTYPE = 5   Mass Density

Mass density is the Mass/Unit Volume.

This implies that ND = 1.

Mass Density = ρ

PTYPE = 6   Thermal Expansion Coefficient

THERMAL EXPANSION COEFFICIENT is a material property that computes the strain given a temperature differential, i.e.,

$$\epsilon = \alpha \, \Delta \, T \quad \text{or} \quad \alpha = \frac{\epsilon}{\Delta T}$$

Where ϵ = strain

α = thermal expansion coefficient

ΔT = the temperature differential

The THERMAL EXPANSION COEFFICIENT may be represented as a vector with three principal values:

a_{xx}, a_{yy} and a_{zz}.

This implies that ND (number of Dependent Variables) is equal to three.

PTYPES 7 - 11   Composite Materials

Composite materials will be represented with linkages to Tabular Data Property 11 as described in Figure 4-37. PTYPES required are:

PTYPE	DESCRIPTION
7.	Laminate material stiffness matrix
8.	Bending material stiffness matrix
9.	Transverse shear material stiffness matrix.
10.	Bending coupling material stiffness matrix.
11.	Material Coordinate System.
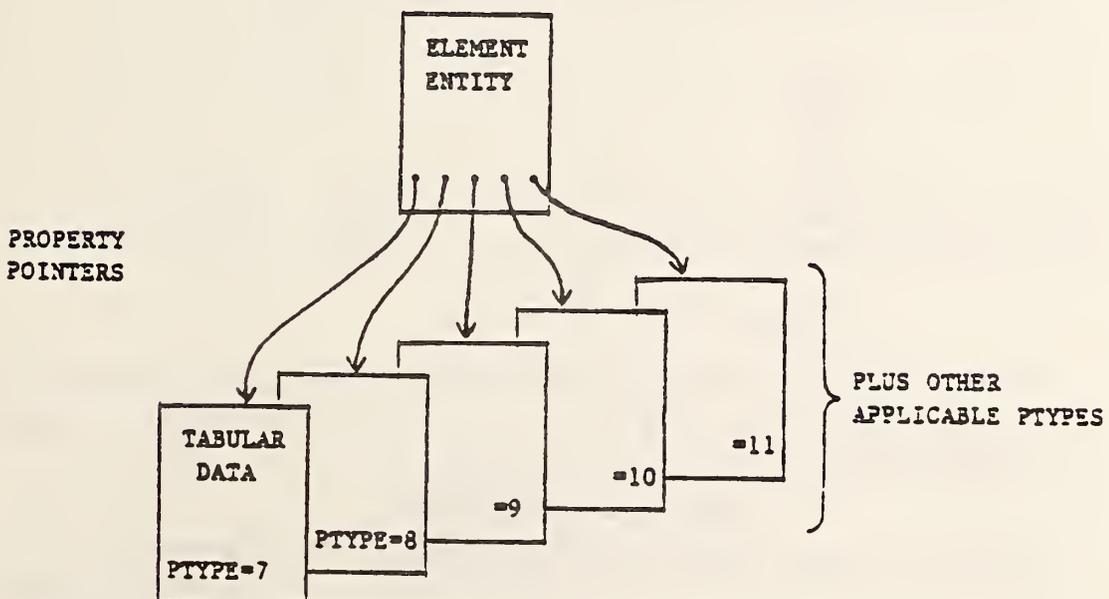
FIGURE 4-37 COMPOSITE MATERIAL CONFIGURATION

PTYPE = 7   Laminate Material Stiffness Matrix

The membrane material stiffness matrix defines anisotropic material properties for shell =membrane action. For example:

$$\bar{f} = t \ [M] \ \bar{\epsilon}$$

Where $\bar{f}$ = Forces per unit length row of elements (f_x, f_y, f_{xy})

$\bar{\epsilon}$ = Midplane strains row of elements $(\epsilon_x, \epsilon_y, \epsilon_{xy})$

t = Shell thickness — see element property

$[M]$ = Membrane material stiffness matrix

$\bar{F}$ and $\bar{\epsilon}$ Defined in shell material coordinate system, PTYPE = 11

Because of symmetry, the elements Mji = Mij. Therefore, six elements define the membrane material stiffness matrix. This implies ND = 6.

M_{11}
M_{12}
M_{13}
M_{22}
M_{23}
M_{33}

The matrix $[M]$ is a laminate material stiffness matrix which is calculated from lamina stress strain matrices $[G]_n$. One method for calculating $[M]$ for a laminate containing m plys is:

$$[Mij] = \frac{1}{t} \ \sum_{n=1}^{m} \ [Gij]_n \ \Delta t_n$$

Δt_n is thickness of nth

t is total thickness of laminate

Where $[G]_n$ is stress strain matrix for n^{th} ply of laminate, defined in the material coordinate system

PTYPE = 8 Bending Material Stiffness Matrix

The bending material stiffness matrix defines the anisotropic material properties for shell bending. For example:

$$\overline{m} = \frac{t^3}{12} [B] \, \overline{\chi}$$

Where

$\overline{m}$ = Shell bending moments per unit length (Row of elements M_x, M_y, M_{xy}

$\overline{\chi}$ = Shell curvature (Row of elements (χ_x, χ_y, χ_{xy})

t = Shell thickness — see element property

$[B]$ = Bending material stiffness matrix $\overline{M}$ and $\overline{\chi}$ are defined in shell material coordinate system, PTYPE = 11

Because of symmetry the elements Bij = Bji. Therefore, six elements define the bending stress strain matrix. This implies ND = 6

B_{11}

B_{12}

B_{13}

B_{22}

B_{23}

B_{33}

The matrix $[B]$ is a laminate matrix for bending, which is calculated from lamina matrices $[G]_n$. One method for calculating $[B]$ for a laminate containing m plys is:

$$[Bij] = \frac{12}{t^3} \sum_{n=1}^{m} (Z_n^2 [Gij]_n \, \Delta t_n)$$

Where

$[G]_n$ is stress strain matrix of lamina for n^{th} ply of laminate, defined in the shell material coordinate system

Δt_n is thickness of nth ply

Z_n is the normal distance from midplane of shell to centroid of ply

t is total thickness of shell

367

PTYPE = 9   Transverse Shear Material Stiffness Matrix

The transverse shear material stiffness matrix defines anisotropic material properties for transverse shear flexibility in shell structure. For example:

$$\overline{V} = t_s \, [S] \, \overline{\gamma}$$

Where $\overline{V}$ = transverse shear force per unit

length (row of elements V_x, V_y)

$\overline{\gamma}$ = transverse shear strains, dimensionless

(Row of elements (γ_x, γ_y)

$t_s = \dfrac{5}{6}$, effective transverse shear thickness

$[S]$ = transverse shear material stiffness matrix

$\overline{V}$ and $\overline{\gamma}$ are defined in material coordinate system

Because of symmetry Sij = Sji. Therefore, three elements define the transverse shear material stiffness matrix. This implies ND = 3.

$$S_{11}$$
$$S_{12}$$
$$S_{22}$$

The matrix $[S]$ is a laminate material stiffness matrix for transverse shear flexibility. If the matrix is not defined, deflections normal to the shell do not include contributions from transverse shear strain.

PTYPE = 10   Bending Coupling Material Stiffness Matrix

The membrane - bending coupling material stiffness matrix defines the anisotropic material properties for shell structure with the neutral axis for bending offset from the midplane of the shell. For example:

$$\bar{f} = t^2 [C] \bar{\chi}$$

and

$$\bar{m} = t^2 [C]^T \bar{\epsilon}$$

Where $\bar{f}$ =   Forces per unit length (row of elements f_x, f_y, f_{xy})

$\bar{m}$ =   Bending moments per unit length (row of elements m_x, m_y, m_{xy})

$\bar{\chi}$ =   Curvature (measurements of bending strain), units (meter)$^{-1}$ (Row of elements X_x, X_y, X_{xy})

$\bar{\epsilon}$ =   Midplane strains (row of elements ϵ_x, ϵ_y, ϵ_{xy})

t =   Shell thickness

$\bar{f}$, $\bar{m}$, $\bar{\chi}$ and $\bar{\epsilon}$ are defined in the shell material coordinate system

Because of symmetry, the elements $C_{ij} = C_{ji}$. Therefore, six elements define the membrane bending coupling material matrix. This implies ND = 6.

C_{11}
C_{12}
C_{13}
C_{22}
C_{23}
C_{33}

PTYPE = 10   (Continued)

The matrix $[C]$ is a laminate matrix for membrane bending coupling, which is calculated from lamina stress strain matrices $[G]_n$. One method for calculating $[C]$ for a laminate containing m plys is:

$$[Cij] = \frac{1}{t^2} \sum_{n=1}^{m} (Z_n [Gij]_n \, \Delta t_n)$$

Where $[G]_n$ is stress strain matrix for n^{th} ply, defined in shell
                  material coordinate system
        Δt_n is thickness of nth ply
        Z_n is the normal distance from midplane of shell to centroid of ply
        t is thickness of shell

PTYPE = 11   Material Coordinate System

The orientation of the element material coordinate system is specified by a set of direction cosines defining a vector D. The use of the vector D depends upon the element type.

For element topology type 1 and 33 of entity type 136, the following Figure 4-38 illustrates the use of the vector D to define the element material coordinate system. For topology type 33 the vector D is defined by the reference node 3.
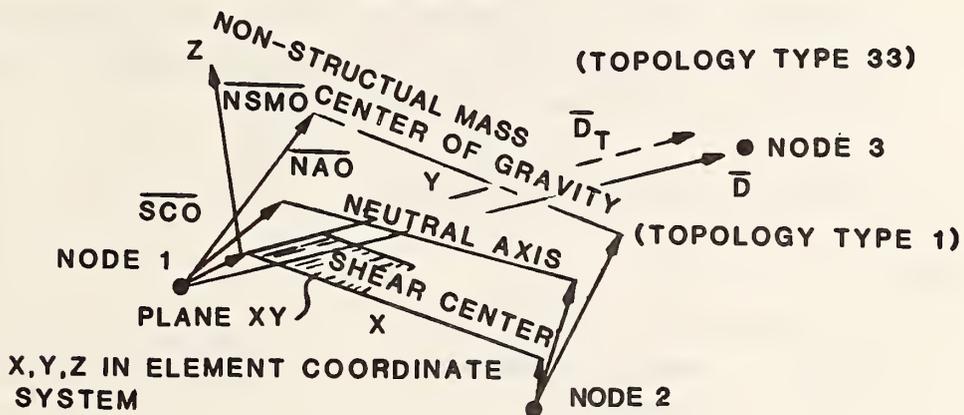
FIGURE 4-38 Elemental and Material Coordinate System

The cosines for vector D are translated to the location of the shear center offset to establish the reference planes for material property definition (vector DT).

For element topology types 2 through 26 of the entity type 136, the following paragraphs discuss the use of vector D to define the material coordinate system.

The projection of D on the plane of the element face (F1) (outward normal N) defines a vector in the X direction of the material coordinate system.

$$\bar{x} = \bar{N} \times \bar{D} \times \bar{N}$$

$\bar{N}$ = positive outward normal of the element face (F1), and is defined by nodal connection of the element, i.e.,

$$\bar{N} = \bar{S_1} \times \bar{S_2}$$

$\bar{S}_1$ = vector from first to second corner of element face (F1)

$\bar{S}_2$ = vector from second to third corner of element face (F1)

PTYPE = 11   (Continued)

Three direction cosines are required to define the Vector D in the global coordinate system. Therefore, ND = 3.

$$D_1$$
$$D_2$$
$$D_3$$

The vectors $\overline{D}$ and $\overline{N}$ define the element material X and Z axes, respectively. The internal load and strain sign conventions must be described to ensure consistent definition of material types 7-10. See Figure 4-39.

INTERNAL LOAD SIGN CONVENTION:

where:

x, y, z   are material coordinate system axes

u, v, w   are displacements of a point in the material coordinate system.
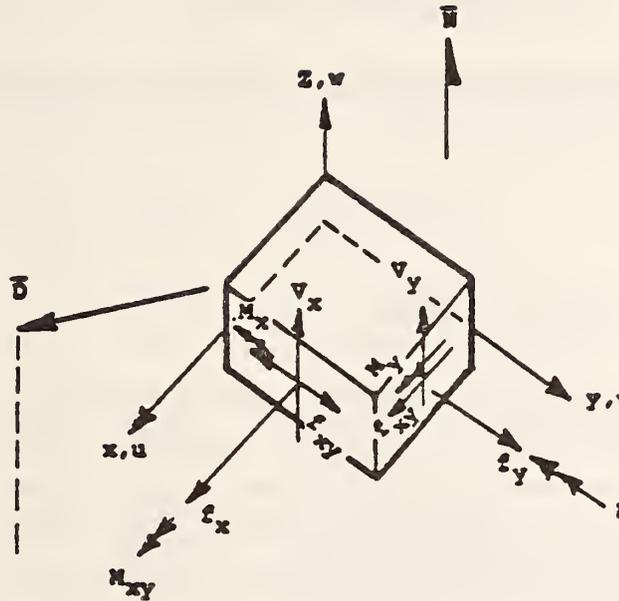
$$\left\{\begin{matrix} f_x \\ f_y \\ f_{xy} \end{matrix}\right\} = \bar{f} \quad \text{forces per unit length}$$

$$\left\{\begin{matrix} M_x \\ M_y \\ M_{xy} \end{matrix}\right\} = \bar{M} \quad \text{moments per unit length}$$

$$\left\{\begin{matrix} V_x \\ V_y \end{matrix}\right\} = \bar{V} \quad \begin{matrix}\text{transverse shear forces} \\ \text{per unit length}\end{matrix}$$

$$\left\{\begin{matrix} \epsilon_x \\ \epsilon_y \\ \epsilon_{xy} \end{matrix}\right\} = \epsilon \quad \text{midplane strains}$$

$$\left\{\begin{matrix} \chi_x \\ \chi_y \\ \chi_{xy} \end{matrix}\right\} = \epsilon \quad \text{bending curvatures}$$

$$\left\{\begin{matrix} \gamma_x \\ \gamma_y \end{matrix}\right\} = \gamma \quad \text{transverse shear strains}$$

STRAIN DISPLACEMENT RELATIONSHIPS

$$\epsilon_x = \frac{\partial u}{\partial x}, \qquad \epsilon_y = \frac{\partial v}{\partial y}, \qquad \epsilon_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}$$

$$\chi_x = \frac{\partial^2 u}{\partial x^2}, \qquad \chi_y = \frac{\partial^2 w}{\partial^2 y}, \qquad \chi_{xy} = 2\frac{\partial^2 w}{\partial x \partial y}$$

$$\gamma_x \approx \frac{\partial w}{\partial x}, \qquad \gamma_y \approx \frac{\partial w}{\partial y}$$

FIGURE 4-39 INTERNAL LOAD SIGN

PTYPE = 12   Nodal Loads/Constraints Data

The nodal load/constraint data will be stored in the tabular data property form number in the following manner:

PTYPE = 12

ND   = # of degrees of freedom

For example, if the load vector has an X, Y, Z M_x, M_y, and M_z components; the ND=6. (Note M_x, M_y, M_z refer to moments). If the load vector has an X and Y component, the ND=2. If the load vector has only a Z-component, then ND=3. In other words the X-component is the 1st degree, the Y-component is the 2nd degree, and the Z-component is the 3rd degree. Other components are treated in a similar manner starting in the 4th degree for the rotation X component.

The constraint vector has an X, Y, Z, M_x, M_y, and M_z constraints. These constraints are represented by a 0 = No Constraint and 1 = Constraint. ND = 6 always for constraints.

PTYPE = 13   Sectional Properties for Beam Elements

Sectional properties for beam elements define the structural characteristics of the beam. These properties are:

PROPERTY	UNITS	DESCRIPTION
Area	M^2	Area of section
I_x	M^4	Area moment of inertia about the element x-axis
I_y	M^4	Area moment of inertia about the element y-axis
I_{xy}	M^4	Product of inertia
J	M^4	Torsional stiffness parameter
SR_{xy}	Unitless	Shear stiffness ratio
SR_{xz}	Unitless	Shear stiffness ratio
WC	M^6	Warping coefficient

If the properties are the same at both ends of the beam then ND=8, otherwise ND=16. If ND=16, two sets of section properties are specified. They are stated in order of the topology set grid number scheme.

PTYPE = 14   Beam End Releases

Beam end releases specify whether the ends of the beam are constrained or free to move. If free to move then both translation and rotational freedoms of the end are considered.

	PROPERTY	DESCRIPTION
FOR EACH END	X	X direction translation freedom/constraint
	Y	Y direction translation freedom/constraint
	Z	Z direction translation freedom/constraint
	M_x	X direction rotational freedom/constraint
	M_y	Y direction rotational freedom/constraint
	M_z	Z direction rotational freedom/constraint

The value for each property X, Y, Z, M_x, M_y, M_z, is either a 0=unconstrained; +1=constrained to the global coordinate system; or -1=constrained to the element coordinate system.

The beam release must be specified at both ends. Therefore, ND=12.

The beam ends are defined by the Topology Set grid number scheme.

PTYPE = 15   Offsets

Offsets are global x,y,z values used to define the location of the shear center axis, neutral axis, and non-structural center of mass relative to the element end nodes.

Figure 4-38 shows   SCO - "Shear Center Offset";
                      NAO - "Neutral Axis Offset"; and
                      NSMO - "Non-Structural Mass Offset".

These offsets are vectors in the global coordinate system (model space) relative to the end of the beam.

PROPERTY	DESCRIPTION
SCO_x	Shear Center Offset in global x direction
SCO_y	Shear Center Offset in global y direction
SCO_z	Shear Center Offset in global z direction
NAO_x	Neutral Axis Offset in global x direction
NAO_y	Neutral Axis Offset in global y direction
NAO_z	Neutral Axis Offset in global z direction
$NSMO_x$	Non-Structural Mass Offset in global x direction
$NSMO_y$	Non-Structural Mass Offset in global y direction
$NSMO_z$	Non-Structural Mass Offset in global z direction

FOR

EACH

END

ND=9 or ND=18 depending on whether or not both ends must be specified. They are stated in order of the topology set grid number scheme.

PTYPE = 16   Stress Recovery Information

Stress Recovery Information is used to define up to four offset points at each beam end at which stress levels will be recovered from the finite element analysis program.

These offset points are described as global x, y, z offsets from each end node.

All offset points are in a plane which is normal to thebeam element axis.

These points occur in pairs from one end of the beam to the other.

PTYPE = 16   (Continued)

		PROPERTY	DESCRIPTION

1st Pair of Offsets

END1

SRI_{1x1} — Stress Recovery Information beam end 1 x-direction pair 1

SRI_{1y1} — Stress Recovery Information beam end 1 y-direction pair 1

SRI_{1z1} — Stress Recovery Information beam end 1 z-direction pair 1

END2

SRI_{2x1} — Stress Recovery Information beam end 2 x-direction pair 1

SRI_{2y1} — Stress Recovery Information beam end 2 y-direction pair 1

SRI_{2z1} — Stress Recovery Information beam end 2 z-direction pair 1

.
.
.

4th Pair of Offsets

END1

SRI_{1x4} — Stress Recovery Information beam end 1 x-direction pair 4

SRI_{1y4} — Stress Recovery Information beam end 1 y-direction pair 4

SRI_{1z4} — Stress Recovery Information beam end 1 z-direction pair 4

END2

SRI_{2x4} — Stress Recovery Information beam end 2 x-direction pair 4

SRI_{2y4} — Stress Recovery Information beam end 2 y-direction pair 4

SRI_{2z4} — Stress Recovery Information beam end 2 z-direction pair 4

379

PTYPE = 17   Element Thickness

Element Thickness defines the net section thickness for a homogeneous element or individual plate thickness for a laminate or sandwich plate. ND=1 or n and is defined as follows:

ND=1   Scalar thickness of element

ND=n   Thickness of the nth plate of a sandwich or laminate

The thickness is in the order from 1 to n,

The thicknesses are measured in the positive Z direction in order of increasing z in local element coordinate system.

PROPERTY   DESCRIPTION

t_1 Thickness for homogeneous plate or the thickness for the first lamina of the plate

.

.

.

t_n Thickness for the nth lamina of the plate.

PTYPE = 18   Non-Structural Mass

Non-Structural Mass is defined as the mass not accounted for in volume and density information for the structural elements. ND=1.

PROPERTY   DESCRIPTION

NSM   Mass/unit length or
                  Mass/unit area or
                  Mass/unit volume

                  this depends on type of element.

PTYPE = 19   Thermal Conductivity

Thermal Conductivity relates heat flow across a surface as a function of temperature. The heat balance equation shows this relationship:

$$\frac{\partial}{\partial x}\left[k_x \frac{\partial T}{\partial x}\right] + \frac{\partial}{\partial y}\left[k_y \frac{\partial T}{\partial y}\right]$$

$$+ \frac{\partial}{\partial z}\left[k_z \frac{\partial T}{\partial z}\right] = \rho\, C_p \frac{\partial T}{\partial t} - Q_I$$

Where

k_n =   Thermal conductivity coefficient where $n = x, y, z$

C_p =   Heat capacity at constant pressure

ρ =   Material Density

T =   Temperature

t =   Time

Q_I =   Energy converted to heat internally

x,y,z are defined in the local element coordinate system.

If we consider k (thermal conductivity coefficient) as independent of direction, then k can be represented as constant in the x, y, z directions:

i.e.,

$$\frac{\partial}{\partial x}\left[k_x \frac{\partial T}{\partial x}\right] = \frac{\partial k_x}{\partial x}\frac{\partial T}{\partial x} + k_x \frac{\partial^2 T}{\partial x^2}$$

Now assuming a constancy in the x direction:

$$\frac{\partial}{\partial x}\left[k_x \frac{\partial T}{\partial x}\right] = k_x \frac{\partial^2 T}{\partial x^2}$$

Therefore the thermal conductivity is a vector with three principal values, k_x, k_y, and k_z. This implies that ND (Number of Dependent Variables) is equal to three.

PTYPE = 19   Thermal Conductivity (Continued)

In matrix form for constraint conductivity:

$$k_x \; k_y \; k_z \begin{vmatrix} \dfrac{\partial^2 T}{\partial x^2} \\[2ex] \dfrac{\partial^2 T}{\partial y^2} \\[2ex] \dfrac{\partial^2 T}{\partial z^2} \end{vmatrix} = \rho \; C_p \frac{\partial T}{\partial t} - Q_I$$

Now assuming no internal heat sources Q_I and the steady state where $\dfrac{\partial T}{\partial t} = 0$

Consequently, integrating in one direction

$$Q = k_x \; A \; \frac{\partial T}{\partial x}$$

Where Q is the heat flux in the x direction across a surface of area A normal to the x-direction. Likewise solutions may be found in the y and z direction.

PROPERTY	DESCRIPTION
K_x	Thermal Conductivity coefficient x direction
K_y	Thermal Conductivity coefficient y direction
K_z	Thermal Conductivity coefficient z direction

PTYPE = 20   Heat Capacity

Heat Capacity is a material's ability to store heat. The heat balance equation shows the relationship of heat capacity to the spatial variation and time variation of temperature.

$$\frac{\partial}{\partial x}\left[k_x \frac{\partial T}{\partial x}\right] + \frac{\partial}{\partial y}\left[k_y \frac{\partial T}{\partial y}\right] + \frac{\partial}{\partial z}\left[k_z \frac{\partial T}{\partial z}\right] = \rho\, C_p\, \frac{\partial T}{\partial t} - Q_I$$

Where

k_n = Thermal conductivity coefficient where $n=x,y,z$

C_p = Heat capacity at constant pressure

ρ = Material Density

T = Temperature

t = Time

Q_I = Energy converted to heat internally

If we consider constant pressure, the heat capacity can be considered a constant.

Consequently, ND=1 to define the constant.

PROPERTY	DESCRIPTION
Cp	Heat capacity at constant pressure

PTYPE = 21   Convective Film Coefficient

Convective film coefficient relates to the amount of heat flux that is conveted to adjacent materials at the interface boundary of a heat source.

$$Q = -h_c A \, \Delta T$$

Where Q = the heat flux

h_c = the convective film coefficient

A = the surface area through which the heat flows

ΔT = the Temperature differential between the materials.

The convective film coefficient may be represented as a constant. This implies that ND=1.

PROPERTY	DESCRIPTION
h_c	Convective Film Coefficient

PTYPE = 22   Electromagnetic Radiation Parameters

Properties for Absorptivity, Transmissivity, Reflectivity, and Emissivity are defined for structural elements using four values. ND=4.

PROPERTY	DESCRIPTION
a	Absorptivity Constant
t	Transmissivity Constant
R	Reflectivity Constants
E	Emissivity Constant

Examples:

Consider the representation of the mass density as a function of pressure:

INDEX	NAME	RECORDED VALUE
1	N	9
2	PTYPE	5
3	ND	1
4	NI	1
5	TYPI	2
6	NVALI	2
7	$VALI_1$	50
8	$VALI_2$	25
9	VALD(1,1)	33
10	VALD(1,2)	46

Additional pointers as required (see 2.2.4.4.2)

Examples:

Consider the representation of Young's modulus for a linear, static, independent case:

INDEX	NAME	RECORDED VALUE
1	N	6
2	PTYPE	1
3	ND	3
4	NI	0
$4 + 2 * NI + 1 = 5$	E_{xx}	30×10^6
6	E_{yy}	30×10^6
7	E_{zz}	30×10^6
.		
.		
.		

Additional pointers as required (see 2.2.4.4.2)

4.3.7.3.12   FORM NUMBER: 12   **External Reference File List**

DESCRIPTION

The External Reference File List appears in an IGES file which references definitions that reside in another IGES file. It contains a list of the names of the IGES files directly referenced by entities within this IGES file. See section 2.5.4 and the External Reference Entity (Type 416) for more detail.

Index	Name	Type	Description
1	N	Integer	Number of List Entries
2	NAME1	String	External Reference File Name
... N+1	NAMEN	String	Last External Reference File Name

Additional pointers as required (see 2.2.4.4.2).

4.3.7.3.13   FORM NUMBER: 13   **Nominal Size**

DESCRIPTION

A nominal size consists of a value, a name and optionally a reference to an engineering standard. The nominal size value is a real value in the units appropriate for the specified name. The name is a string constant, but the following values are predefined:

Nominal Size Name	Meaning
3HAWG	American Wire Gauge
3HIPS	Iron Pipe Size
2HOD	Outside Diameter schedule, i.e., tubing

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=2 or 3)
2	SZ	Real	Nominal size value
3	NM	String constant	Nominal size name
4	SP	String constant	Name of relevant engineering standard (optional)

Additional pointers as required (see 2.2.4.4.2)

4.3.7.3.14   FORM NUMBER: 14 **Flow Line Specification**

DESCRIPTION

The flow line specification property attaches one or more text strings to entities being used to represent a flow line.

Index	Name	Type	Description
1	NP	Integer	Number of property values
2	L1	String constant	Primary flow line specification name
3	L2	String constant	Modifier (optional)
.	.	.	.
.	.	.	.
.	.	.	.
NP+1	LN	String constant	Modifier (optional)

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.15   FORM NUMBER: 15   **Name**

DESCRIPTION

This property contains a string which specifies a user-defined name. It can be used
for any entity that does not have a name explicity specified in the parameter data
for the entity.

Index	Name	Type	Description
1	NP	Integer	No. of property Values (NP=1)
2	NAME	String	Entity Name

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.16   FORM NUMBER: 16   **Drawing Size**

DESCRIPTION

This property specifies the size of the drawing in drawing units. The origin of the drawing is defined to be (0,0) in drawing space.

Index	Name	Type	Description
1	NP	Integer	No. of property Values (NP=2)
2	XS	Real	X Size (Extent of Drawing along positive XD axis)
3	YS	Real	Y Size (Extent of Drawing along positive YD AXIS)

Additional Pointers as required (see 2.2.4.4.2)

4.3.7.3.17   FORM NUMBER: 17   **Drawing Units**

DESCRIPTION

This property specifies the drawing space units as outlined in the drawing entity (Type 404). The drawing units are given in the same form as the model space units in the global section (see 2.2.4.2.14).

Index	Name	Type	Description
1	NP	Integer	No. of Property Values (NP=2)
2	FLAG	Integer	Units Flag
3	UNIT	String	Units Name

Additional Pointers as required (see 2.2.4.4.2).

4.3.8   **Subfigure Definitions**

The following definition entities are used to provide a "template" for the instances of subfigures (see 2.5.1).

4.3.8.1   Subfigure Definition Entity.

The subfigure definition entity is designed to support the concept of a subpicture (if one equates drawing creation with graphics picture processing). This entity permits a single definition of a detail to be utilized in multiple instances in the creation of the whole picture. The contents of the subfigure include a set of pointers to any combination of entities and other subfigures. DEPTH indicates the actual nesting of the subfigures. If DEPTH=0, the subfigure has no references to any subfigure instances. A subfigure cannot reference a subfigure instance that has equal or greater depth. A DEPTH=N indicates there is a reference to an instance of a subfigure definition with DEPTH N-1.

4.3.8.1.1   **Directory Data**
               ENTITY TYPE NUMBER :   308

4.3.8.1.2   **Parameter Data**

Index	Name	Type	Description
1	DEPTH	Integer	Depth of subfigure (indicating the amount of nesting)
2	NAME	String	Subfigure name
3	N	Integer	Number of entities in the subfigure
4	DE	Pointer	Pointers to the directory entries for the associated entities
.	.	.	
.	.	.	
N+3	.	Pointer	Pointer to the last directory entry

Additional Pointers as required (see 2.2.4.4.2).

4.3.8.2   Network Subfigure Definition Entity

The Network Subfigure Definition entity permits a single definition of a detail to be used in many instances in the file, similar to the Subfigure Definition entity (Type 308).   It differs from the ordinary Subfigure Definition in that it defines a specialized subfigure, one whose instances may participate in networks. To participate in a network, points of connection (Connect Point entity Type 132) must be defined (see indices NA+7 and NA+NC+7) and instanced along with the subfigure. Often, products which contain networks are designed first as schematics (showing the logical connections or relationships), which are then converted into the designs of the physical products. Whenever both a logical design and a physical design are present in the same file, the processor needs a way to determine which entities belong in which design. The Type Flag field (index NA+4) implements this distinction. Other fields, such as NAME and DEPTH, function in exactly the same manner as the Subfigure Definition (308).

Note: the depth of the subfigure is inclusive of both Network Subfigures entity (Type 320) and the ordinary subfigure entity (Type 308). Thus, the two may be nested but must indicate that in the depth parameter.

4.3.8.2.1   **Directory Data**
              ENTITY TYPE NUMBER: 320

4.3.8.2.2   **Parameter Data**

Index	Name	Type	Description
1	DEPTH	Integer	Depth of subfigure (indicating the amount of nesting)
2	NAME	String	Subfigure name

Index	Name	Type	Description
3	NA	Integer	Number of associated (child) entities in the subfigure exclusive of Primary Reference Designator and connect points.
4	APTR1	Pointer	Pointer to the directory entry of the first associated entity.
. . .			
NA+3	APTRNA	Pointer	Pointer to the directory entry for the associated entity NA.
NA+4	TF	Integer	Type Flag: 0 = not specified 1 = logical 2 = physical
NA+5	PRD	String	Primary reference designator
NA+6	DPTR	Pointer	Pointer to the directory entry of the primary reference designator Text Display Template
NA+7	NC	Integer	Number of associated (child) Connect Point entities
NA+8	CPTR1	Pointer	Pointer to the directory entry for the associated Connect Point 1 Connect
. .	. .	. .	
NA+NC+7	CPTRNC	Pointer	Pointer to the directory entry for the associated Connect Point NC

Additional Pointers as required (see 2.2.4.4.2).

4.3.9 **Subfigure Instances**

4.3.9.1 Singular Subfigure Instance Entity

This entity defines the occurrence of a single instance of the defined subfigure (Type 308). See Figure 4-40 and Section 2.5.1.

4.3.9.1.1 <u>Directory Data</u>
<u>ENTITY TYPE NUMBER</u>: 408

4.3.9.1.2 <u>Parameter Data</u>

<u>Index</u>	<u>Name</u>	<u>Type</u>	<u>Description</u>
1	DE	Pointer	Pointer to subfigure definition entry
2	X	Real	Translation data relative to either model space or to the definition space of a referring entity
3	Y	Real	"
4	Z	Real	"
5	S	Real	Scale factor (default = 1.0)

Additional Pointers as required (see 2.2.4.4.2).

4.3.9.2 Rectangular Array Subfigure Instance Entity.

The rectangular array produces copies of an object called the base entity, arranging them in equally spaced rows and columns. The following types of entities are valid for use as a base entity: group associativity, point, line, circular arc, conic arc, parametric spline curve, rational B-spline curve, any annotation entity, rectangular array instance, circular array instance, or subfigure definition. The number of columns and rows of the rectangular array, together with their respective horizontal and vertical displacements are given. Also, the coordinates of the lower left hand corner for the entire array is indicated. This is where the first entity in the reproduction process is placed and is called position No. 1. The successive positions are counted vertically up the first column, then vertically up the second column to the right, and so on.

The array of instance locations for the base entity is rotated about the line through the point (X,Y), parallel to the ZT-axis. The angle of rotation is specified in radians counterclockwise from the positive XT-axis. The instances of the base entity are not rotated from their original orientation.

A DO-DON'T flag enables one to display only a portion of the array. If the DO flag is chosen, half or fewer of the elements of the rectangular array are to be defined. If the DON'T flag is chosen, half or more of the elements of the rectangular array are to be defined.

4.3.9.2.1   **Directory Data**
ENTITY TYPE NUMBER: 412

4.3.9.2.2   **Parameter Data**

Index	Name	Type	Description
1	DE	Pointer	Pointer to base entity
2	S	Real	Scale factor (default = 1.0)
3	X	Real	Coordinates of point
4	Y	Real	to be used as lower
5	Z	Real	left hand corner of array
6	NC	Integer	Number of columns
7	NR	Integer	Number of rows
8	DX	Real	Horizontal distance between columns
9	DY	Real	Vertical distance between rows
10	AX	Real	Rotation angle in radians
11	LC	Integer	DO-DON'T list count =L. (L=0 indicates all to be displayed.)

Index	Name	Type	Description
12	DDF	Integer	DO-DON'T flag (DO=0; DON'T=1)
13	N1	Integer	Number of first position to be processed (DO), or to be processed (DON'T)
12+LC	NLC	Integer	Number of last position

Additional pointers as required (see 2.2.4.4.2).

RESULTING PLACEMENT
OF SUBFIGURE ENTITIES

(X,Y,Z) SUBFIGURE PLACEMENT

SUBFIGURE PLACEMENT

SUBFIGURE

MODEL SPACE ORIGIN IN
RELATION TO SUBFIGURE
ENTITIES

SUBFIGURE DEFINITION
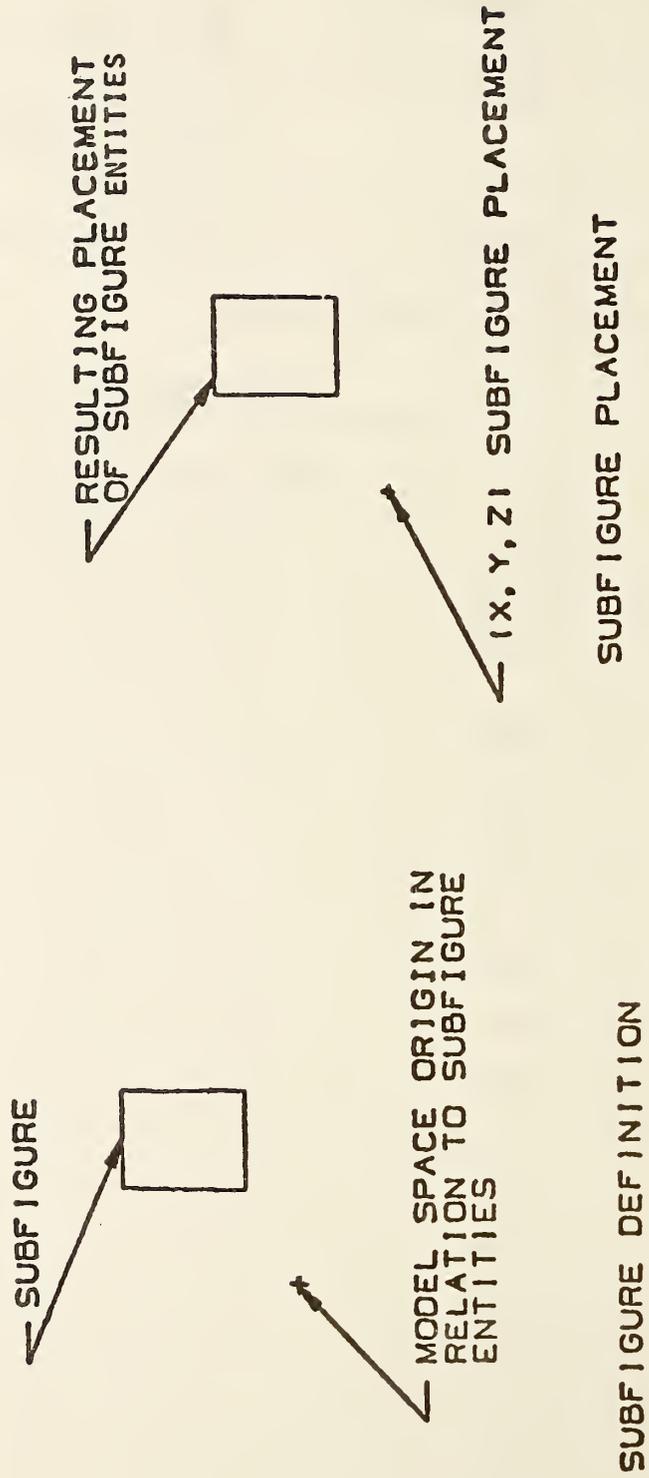
FIGURE 4-40   SUBFIGURE ORIGIN

4.3.9.3   Circular Array Subfigure Instance Entity

The circular array produces copies of an object called the base entity, arranging them around the edge of an imaginary circle whose center and radius are specified. The following types of entities are valid for use as a base entity: group associativity, point, line, circular arc, conic arc, parametric spline curve, rational B-spline curve, any annotation entity, rectangular array instance, circular array instance, or subfigure definition. The number of possible instance locations for the base entity is specified and the location of the first instance position is specified in terms of a radius and a start angle measured positive, counterclockwise in radians from the line through the point (X,Y), parallel to the ZT-axis. The successive positions follow a counterclockwise direction around the imaginary circle and are distributed according to a given delta angle.

A DO-DON'T flag enables one to display only a portion of the array. If the DO-flag is chosen, half or fewer of the elements of the circular array are to be defined. If the DON'T-flag is chosen, half or more of the elements of the circular array are to be defined.

4.3.9.3.1   **Directory Data**
               ENTITY TYPE NUMBER: 414

4.3.9.3.2   **Parameter Data**

Index	Name	Type	Description
1	DE	Pointer	Pointer to base entity
2	NE	Integer	Total number of possible instance locations
3	X	Real	Coordinates of center
4	Y	Real	of imaginary circle
5	Z	Real	
6	R	Real	Radius of imaginary circle
7	AS	Real	Start angle in radians
8	AD	Real	Delta angle in radians

414 – CIRCULAR ARRAY
      SUBFIGURE INSTANCE

Index	Name	Type	Description
9	LC	Integer	DO-DON'T list count = L. (L=0 indicates all replicated entities to be displayed)
10	DDF	Integer	DO-DON'T Flag (DO=0; DON'T=1)
11	N1		Number of first position to be processed (DO), or to be not processed (DON'T)
.	.	.	.
.	.	.	.
.	.	.	.
10+LC	NLC	Integer	Number of last position

Additional Pointers as required (see 2.2.4.4.2).

4.3.9.4   Network Subfigure Instance Entity

Each instance of a Network Subfigure Definition Entity is specified by a Network Subfigure Instance Entity. It functions and may be used as described in section 2.5.1.

In addition though, the points of connection (connect Point Entity Type 132) specified by the Network Subfigure Definition Entity must be instanced and associated with each Network Subfigure Instance (see indices 11 and 12).

The Type Flag field (index 8) implements the distinction between logical design and physical design data if both may be present in the file.

The Network Subfigure Instance Entity allows different scale factor in the x, y, and z axes. This scaling is performed before the translation from X, Y, and Z and before the Transformation matrix entity pointed to in the directory entry (if any) is applied. The scaling does not apply to the model space placement coordinates (X, Y, Z).

4.3.9.4.1   **Directory Data**
ENTITY TYPE NUMBER: 420

4.3.9.4.2   **Parameter Data**

Index	Name	Type	Description
1	DE	Pointer	Pointer to network definition entry
2	X	Real	Translation data relative to either model space or to the definition space of a referring entity
3	Y	Real	"
4	Z	Real	"

Index	Name	Type	Description
5	XS	Real	Scale factor in definition space x axis (default 1.0)
6	YS	Real	Scale factor in definition space y axis (default XS)
7	ZS	Real	Scale factor in definition space z axis (default XS)
8	TF	Integer	Type flag: 0 - not specified 1 - logical 2 - physical
9	PDR	String	Primary reference designator
10	DPTR	Pointer	Pointer to the directory entry of the primary reference designator Text Display Template
11	NC	Integer	Number of associated (child) Connect Point entities
12 . .	CPTR1 . .	Pointer . .	Pointer to the directory entry for associated Connect Point 1
NC+11	CPTRNC	Pointer	Pointer to the directory entry for associated Connect Point NC

Additional Pointers as required (see 2.2.4.4.2)

4.3.10   **Text Font Definition Entity.**

This entity defines the appearance of characters in a text font. The data describing the appearance of a character may be located by the Font Code (FC) and the ASCII character code. This entity may describe any or all the characters in a character set. Thus, this entity may be used to describe a complete font or a modification to a subset of characters in another font. Font Number and Font Name are the number and name used to reference the font on the originating system. When this entity is a modification to another font, the Supersedes Font value (Parameter 3) indicates which font the entity modifies. This value is an integer which indicates the font number to be modified or the negative of the pointer value to the directory entry of another text font definition entity. When this entity modifies another font, i.e., Parameter 3 references another font, the definitions in this entity supersede the definition in the original font. For example, a complete set of characters may have their font definition specified by this entity. Another text font definition entity could reference the first definition and modify a subset of the characters.

Each character is defined by overlaying an equally spaced square grid over the character. The character is decomposed into straight line segments which connect grid points. Grid points are referenced by standard Cartesian coordinates. The position of the character relative to the grid is defined by two points. The character's origin point is placed at the origin (0,0) of the grid and defines the position of the character relative to the text origin of that character. The second point defines the origin point of the character following the character being defined. This allows the spacing between characters to be specified. Construction of text strings consists of placing the character origin of the first character at the text string origin and placing subsequent character origins at the location specified in the previous character as the location of the next character's origin.

The parameterization of the character appearance is described by the motion of an imaginary pen moving between grid points. Commands to move the pen reference the grid location to which the pen is to move. The pen may be "lifted" such that its movement is not displayed. The representation of the movement of the pen is a sequence of pen commands and grid locations. Each movement of the pen is represented by a pen updown flag and a pair of integer grid coordinates. The pen up/down flag defaults to pen down. A flag value of 1 means the pen is to be lifted (i.e., display off) and moved to the next location in the sequence. Upon arrival at this location the pen is returned to a "down" position (i.e., display on)

The grid size is related to the text height through the scale parameter. This parameter defines how many grid units equal one text height unit.

4.3.10.1   **Directory Data**
ENTITY TYPE NUMBER :   310

4.3.10.2   **Parameter Data**

Index	Name	Type	Description
1	FC	Integer	Font Number
2	FNAME	String	Font Name
3	SF	Integer	Number of the font which this definition supersedes
4	SCALE	Integer	Number of grid units which equal one text height unit
5	N	Integer	Number of characters in this definition
6	AC1	Integer	ASCII code for first character
7	NX1	Integer	Grid location of the next character's origin

Index	Name	Type	Description
8	NY1	Integer	
9	NM1	Integer	Number of pen motions for first character
10	$PF1_1$	Integer	Pen up flag 0 = Down, 1 = Up
11	$X1_1$	Integer	Grid location to which the pen is to move
12	$Y1_1$	Integer	
.	.	.	
.	.	.	
.	.	.	
9+NM1*3	AC2	Integer	ASCII code for second character
10+NM1*3	NX2	Integer	Grid location of the next character origin
11+NM1*3	NY2	Integer	"
12+NM1*3	NM2	Integer	Number of pen motions for second character
.	.	.	
.	.	.	
$5+4N+\sum_{i=1}^{N}3*NMi$	.	Integer	Last grid location of last character

Additional Pointers as required (see 2.2.4.4.2).

An example of this entity using the character in Figure 4-41 is

FC	1
FNAME	8H STANDARD
SF	
SCALE	8
N	60
AC1	65
NX1	11
NY1	0
NM1	4
PF1	0
X1	4
Y1	8
PF2	0
X2	8
Y2	0
PF3	1
X3	2
Y3	4
PF4	0
X4	6
Y4	4
.	
.	
.	

In the parameter section of the IGES file it would look like:
1,8HSTANDARD,,8,60,65,11,0,4,,4,8,,8,0,1,2,4,,6,4....

Figure 4-42 provides another example.
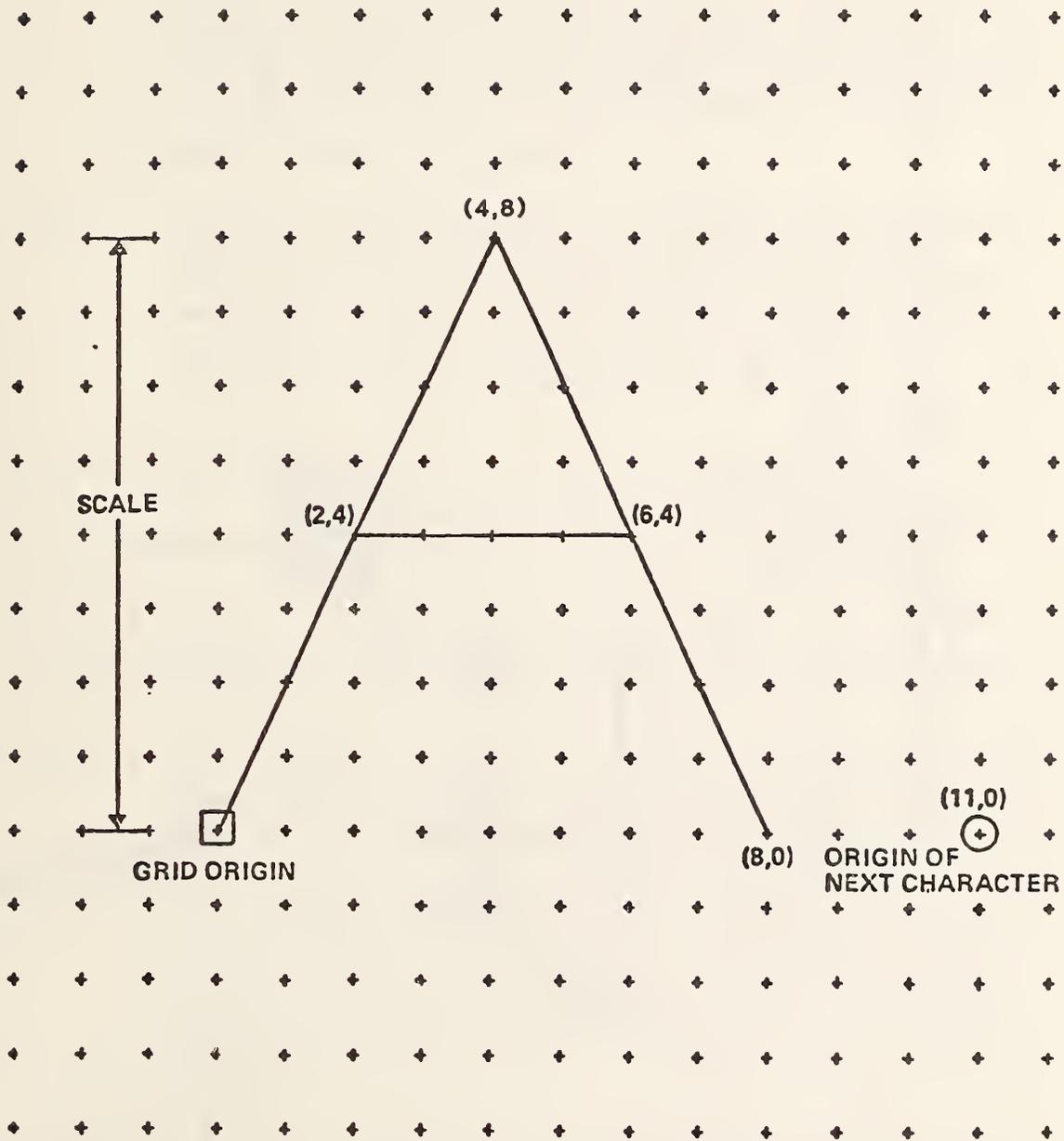
FIGURE 4-41 EXAMPLE OF A CHARACTER DEFINITION
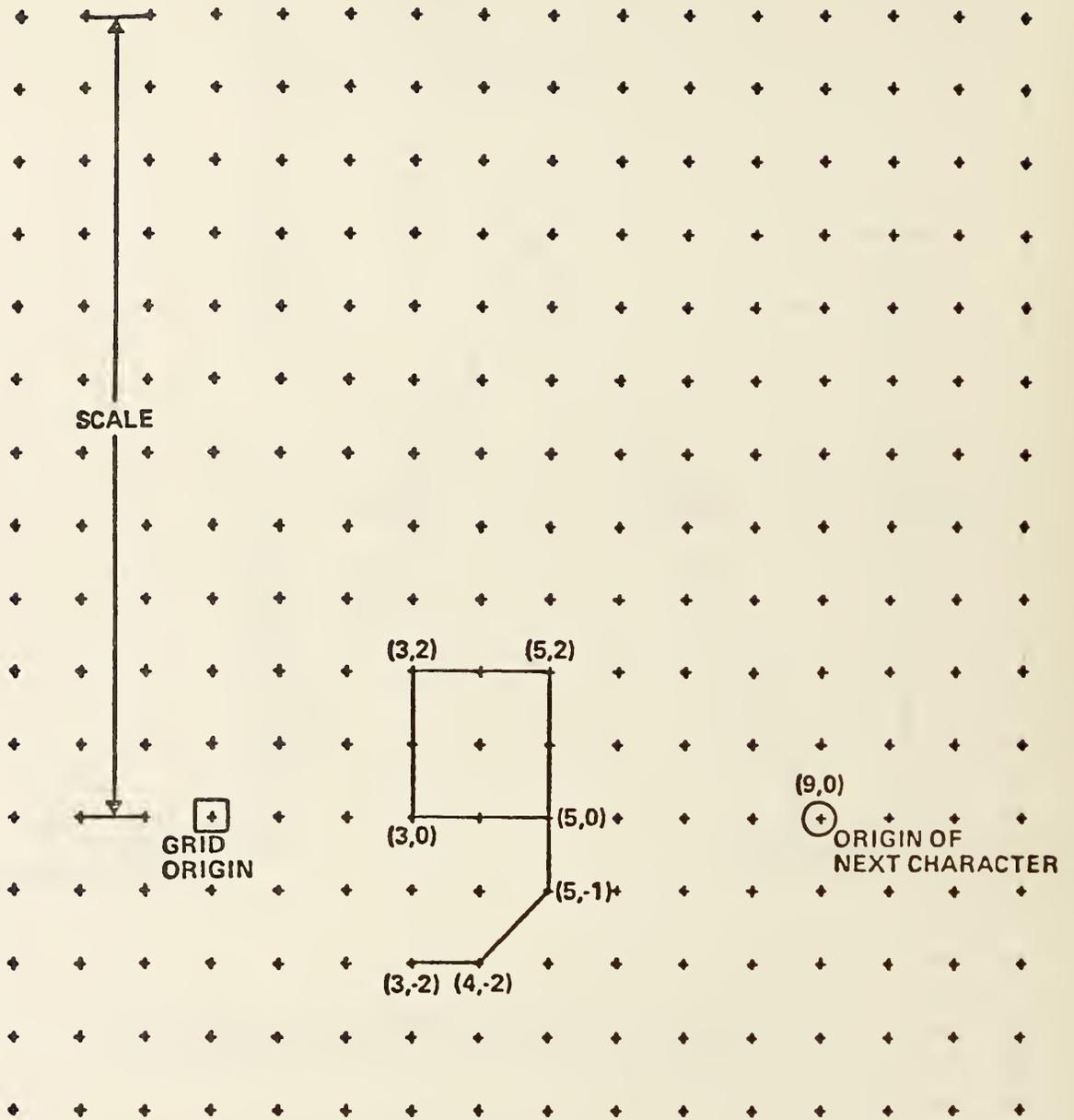
FIGURE 4-42 SECOND CHARACTER DEFINITION EXAMPLE

4.3.11 View Entity.

The View Entity defines a framework for specifying a viewing orientation of an object in three dimensional model space (X,Y,Z). The framework is also used to support the projection of all or part of model space onto a view plane. One type of projection, an orthographic parallel projection, can be specified.

4.3.11.1

Orthographic Parallel Projection. An orthographic parallel projection onto a view plane of an object in model space is formed by passing rays normal to the view plane through each point of the object and finding the intersection with the view plane as shown in Figure 4-43.
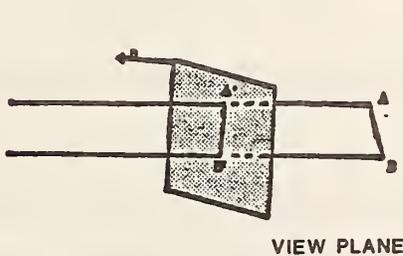
VIEW PLANE

Figure 4-43 Orthographic Parallel Projection
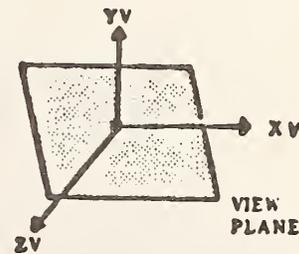
Figure 4-44 View Origin

4.3.11.2

View Coordinate System. The view plane can be described by introducing a right-handed view coordinate system, (XV, YV, ZV) into model space. The view plane is the XV, YV plane, i.e., the plane ZV=0. The view direction is along the positive ZV axis toward the view plane, i.e., in the direction of the vector (0,0,-1). The positive YV axis points in the "up" direction in the resulting view. The point (0,0,0) in the view coordinate system as shown in Figure 4-44 is called the view origin. Thus a complete viewing orientation is specified by a view coordinate system.

4.3.11.3

View Coordinates Obtained from Model Coordinates. View coordinates are obtained from model coordinates through translation and rotation. There are several ways that systems use to specify the data required to transfer from model to view coordinates. However, in each case, the data can be recorded using Form 0 of the Transformation Matrix entity such that the model coordinates are taken as input and the view coordinates are produced as output, as follows, where R denotes the rotation matrix and T the translation vector (see 3.14).

411

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} R \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T \end{bmatrix}$$

In this situation, R is called the view matrix.

The View entity specifies the view matrix and the translation vector by use of a pointer to a Transformation Matrix entity in field 7 of the Directory Entry. In the special case when the view matrix is the identity matrix and there is zero translation, a zero value in field 7 may be used.

Example 1: (View coordinates obtained from model coordinates by a translation and then a rotation.)

The system defines a viewing orientation by specifying a view origin (XO, YO, ZO) in model space and a rotation matrix so that

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} 3x3\ rotation \\ Matrix \end{bmatrix} \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} XO \\ YO \\ ZO \end{bmatrix} \right)$$

or,

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} 3x3\ rotation \\ Matrix \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} 3x3\ rotation \\ Matrix \end{bmatrix} \begin{bmatrix} XO \\ YO \\ ZO \end{bmatrix} .$$

Therefore, the rotation matrix is the view matrix, and in the Transformation Matrix entity,

$$R = \begin{bmatrix} 3x3\ rotation \\ Matrix \end{bmatrix} , \quad T = - \begin{bmatrix} 3x3\ rotation \\ Matrix \end{bmatrix} \begin{bmatrix} XO \\ YO \\ ZO \end{bmatrix}$$

Example 2: (View coordinates obtained from model coordinates by a rotation and then a translation.)

The system defines a viewing orientation by specifying a rotation matrix and a translate or pan vector (XL, YL, ZL) expressed in the rotated coordinate system, so that

$$
\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} 3x3 \text{ rotation} \\ \text{Matrix} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} XL \\ YL \\ ZL \end{bmatrix} \cdot
$$

Therefore, the rotation matrix is the view matrix, and in the Transformation Matrix entity,

$$
R = \begin{bmatrix} 3x3 \text{ rotation} \\ \text{Matrix} \end{bmatrix} , \qquad T = - \begin{bmatrix} XL \\ YL \\ ZL \end{bmatrix} \cdot
$$

4.3.11.4 Simple Form of the View Entity. The View entity provides a view number for the purpose of identifying differing view orientations. However, no standard indexing scheme is presumed to exist.

In its simplest form, the View entity consists of a pointer to the Transformation Matrix entity (in field 7 or the Directory Entry), and a view number. The Transformation Matrix entity specifies a view matrix R and a translation vector T as given in the preceeding section.

4.3.11.5 Projection of a View Volume. In some cases, a view volume and a scale factor may be required to control the projection of the view into a two dimensional drawing space specified by a Drawing entity (see 4.3.4).

The view volume bounds that portion of the data which will be projected after clipping is performed. The view volume is a rectangular parallelepiped with limits specified by Plane entities defined in the model coordinate system. The absence of a clipping in a particular direction may be included by setting the pointer for the appropriate Plane entity equal to zero.

The Plane entities used to define the view volume shall not be artibrary planar definitions. After transformation from model coordinates to view coordinates each plane must result in a plane perpendicular to the appropriate view coordinate system axis (e.g., the left side of the view volume must transform into a plane XV=constant). In addition, only the coefficients of the Plane entity are required, i.e., the bounding curve and display symbol are not used. See Figure 4-45.
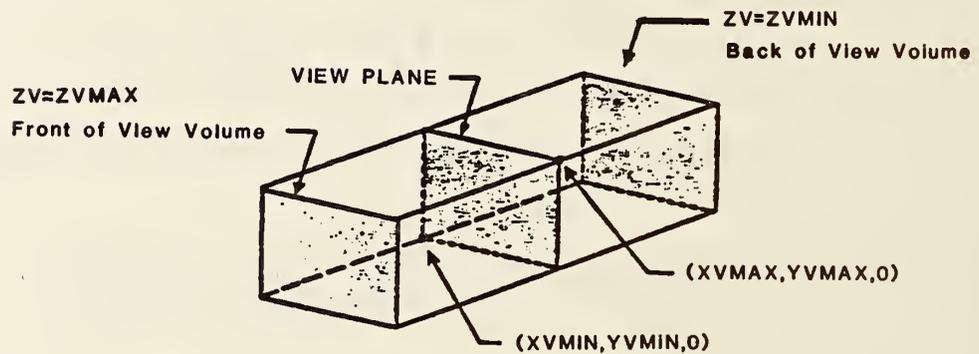
FIGURE 4-45 VIEW VOLUME

4.3.11.6   Projection Operations. The order of operations for the view entity is as follows:

1.   Transform from model to view space.
2.   Perform clipping (if included).
3.   Perform projection onto the view plane.
4.   Transform from view space to drawing space.

The projection onto the view plane and the transform from view space to drawing space can be controlled by the following equation in the case of orthographic parallel projection, where S is the scale factor and XORIGIN and YORIGIN are defined in the Drawing Entity (see 4.3.4).

$$\begin{bmatrix} XD \\ YD \end{bmatrix} = \begin{bmatrix} S & O & O \\ O & S & O \end{bmatrix} \begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} + \begin{bmatrix} XORIGIN \\ YORIGIN \end{bmatrix}$$

4.3.11.7   Entity Display. The display of an entity in a particular view is controlled by the use of the view value in field 6 of the Directory Entry for the entity. If this value is zero or undefined, the entity is displayed with its own characteristics in all views unless display is controlled by other parameters (e.g., pointed to by another entity such as Subfigure Definition or Drawing). If this value is a pointer to a View entity, the entity is displayed with its own characteristics in only the one view.

The selection of multiple views and/or display characteristics for an entity may be made by using one of the Views Visible associativity entities (type number 402, form 3 or 4). The view value for the entity then is a pointer to this associativity instead of to a View entity.

4.3.11.8   **Directory Data**
ENTITY TYPE NUMBER: 410

4.3.11.9   **Parameter Data**

Index	Name	Type	Description
1	VNO	Integer	View number
2	SCALE	Real	Scale factor (Default = 1.0)
3	XVMINP	Pointer	Pointer to left side of view volume (XVMIN plane) or zero
4	YVMAXP	Pointer	Pointer to top of view volume (YVMAX plane) or zero
5	XVMAXP	Pointer	Pointer to right side of view volume (XVMAX plane) or zero
6	YVMINP	Pointer	Pointer to bottom of view volume (YVMIN plane) or zero

| 7 | ZVMINP | Pointer | Pointer to back of view volume (ZVMIN plane) or zero |
| 8 | ZVMAXP | Pointer | Pointer to front of view volume (ZVMAX plane) or zero |

Additional Pointers as required (see 2.2.4.4.2)

4.3.12   External Reference Entity

The External Reference Entity provides a link between an entity in a referencing file and the definition or a logically related entity in referenced file. Further, in keeping with the concept of treating this entity as the definition which it replaces the subordinate entity switch should be set as it would be on the definition replaced. See section 2.5.4 for the entities used in the linkage.

Three forms of the External Reference Entity are defined. Two of these forms are used to reference a definition and one form is a logical reference. Form 0 is used when a single definition from the referenced file is desired. This would be the case where the referenced file contained a collection of definitions. Form 1 is used when the entire file is to be instanced. This would be the case where the referenced file contained a complete sub-assembly. Form 2 is used for external logical references where an entity in one file relates to an entity in a separate file (e.g., when each sheet of a drawing is a separate file, and a flange on one sheet is also depicted on, or mates with, a flange on another sheet). Forms 0 and 2 require an entity-unique symbolic name. The following entities and the parameter which supplies the symbolic name are identified for use:

Entity	Name	Parameter Index	Description
132	Connect Point	9	CP Function Name (unique)
302	Associativity Definition	( )	Implementor assigned
304	Line Font Definition	( )	Implementor assigned
306	MACRO Definition	2	Entity Type Identification
308	Subfigure Definition	2	Subfigure name (unique)
310	Text Font Definition	2	Font name
312	Text Display Template	( )	Implementor assigned
314	Color Definition	( )	Implementor assigned
320	Network Subfigure Definition	2	Subfigure name (unique)

Possible alternatives for the entity-unique symbolic name for those entities marked "Implementor assigned" could be: (1) the property REFERENCE DESIGNATOR (Type 406, Form 7), or (2) the ENTITY LABEL, ENTITY SUBSCRIPT (directory entry fields 18 and 19.)

4.3.12.1   **Directory Data**
ENTITY TYPE NUMBER: 416

4.3.12.2   **Parameter Data** (Form 0 and 2)

Index	Name	Type	Description
1	EXTFID	String	External Reference File Identifier (contained as Global Parameter Number 4 in the referenced file)
2	EXTNAM	String	External Reference Entity Symbolic Name

Additional Pointers as required (see 2.2.4.4.2)

4.3.12.3   **Parameter Data** (Form 1)

Index	Name	Type	Description
1	EXTFID	String	External Reference File Identifier (contained as Global Parameter Number 4 in the referenced file)

Additional Pointers as required (see 2.2.4.4.2)

4.3.13   Nodal Load/Constraint Entity

This entity relates loads and/or constraints to specific nodes in the
Finite Element Model.   This is accomplished by creating a relation
between node entities and the tabular data property that contains the
load or constraint data.   Each load and constraint case will require a
nodal load/constraint entity and a tabular data property Form 11 with
PTYPE=12.

Figure 4-45 shows the relationship or linkage between the nodal
load/constraint entity and the tabular data which carries the load or
constraint vector.   The relationship or linkage is also shown on the
general note entity which described the load/constraint test case being
performed.   There is a one to one correspondence between the load case
description and the general note entity and the pointer to the tabular
data containing the load case magnitudes (see also section 2.5.6).
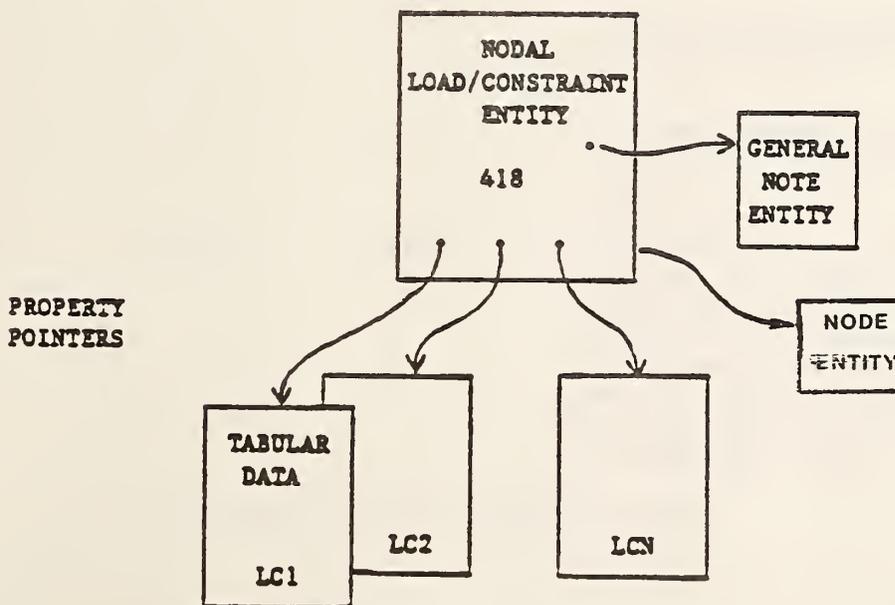
FIGURE 4-46   NODAL LOAD / CONSTRAINT

4.3.13.1   **Directory Data**
            ENTITY TYPE NUMBER: 418

4.3.13.2   **Parameter Data**

Index	Name	Type	Description
1	NC	Integer	Total number of cases
2	TYPE	Integer	1 = Loads 2 = Constraints
3	DE	Pointer	Node Pointer
4	PTR1	Pointer	Pointer to definition of each load case stored in general note entity
.	.	.	.
.	.	.	.
.	.	.	.
NC+3	PTRNC	Pointer	

Additional Pointers as required (see 2.2.4.4.2).

4.3.14   **Color Definition Entity**

The Color Definition Entity is used to communicate the relationship of
the primary (red, green, and blue) colors to the intensity level of the
respective graphics devices as a percent of the full intensity range.

These red, green, blue coordinates (RGB) can be readily transformed to
cyan, magenta, yellow (CMY) and to hue, lightness, saturation (HLS)
using transformations that are given in Appendix F.

4.3.14.1   <u>Directory Data</u>
                ENTITY TYPE NUMBER: 314

4.3.14.2   **Parameter Data**

Index	Name	Type	Description
1	CC1	Real	First color coordinate (red) as a percent of full intensity (range 0. to 100.)
2	CC2	Real	Second color coordinate (green) as a percent of full intensity (range 0. to 100.)
3	CC3	Real	Third color coordinate (blue) as a percent of full intensity (range 0. to 100.)
4	CNAME	String	Color name; this is an optional character string which may contain some verbal description of the color. If the color name is not provided and additional pointers are required, a placekeeper must be supplied between CC3 and the first additional pointer.

Additional pointers as required (see 2.2.4.4.2).

4.3.15 Text Display Template Entity

The Text Display Template Entity is used to display self-contained information such as numbers and/or text strings according to the template's display parameters. There are two forms of the Text Display Template; Absolute and Incremental.

Note: For a more detailed description of the parameters, see the General Note Entity (type 212)

4.3.15.1 Absolute Text Display Template (Form 0)

This form of the Text Display Template specifies the parameters for the text block at the specified starting point. The text to be displayed are taken from the entity pointing to this instance of the Text Display Template. If the text string to be displayed is not explicitly pointed to then the first text string specified in the pointing entity will be displayed.

4.3.15.2 Directory Data
ENTITY TYPE NUMBER: 312
FORM NUMBER: 0

4.3.15.3 Parameter Data

Index	Name	Type	Description
1	CBW	Real	Character box width
2	CBH	Real	Character box height
3	FC	Integer or Pointer	Font characteristic (or pointer if negative) (Default = 1)
4	SL	Real	Slant angle (Radians) (Default = pi/2)
5	A	Real	Rotation angle (Radians)

6	M	Integer	Mirror Flag
7	VH	Integer	Rotate internal text flag
8	XS	Real	coordinates of lower left corner of first character box; see form for coordinate type
9	YS	Real	
10	ZS	Real	

Additional pointers as required (see paragraph 2.2.4.4.2).

4.3.15.4 Incremental Text Display Template (Form 1)

This form of the Text Display Template specifies the parameters for the text block at a starting point specified incrementally from one taken from the entity point to the given instance of the Text Display Template. The text to be displayed is taken from the entity pointing to this instance of the Text Display Template. If the text string to be displayed is not explicitly pointed to, then the first string specified in the pointing entity will be displayed.

4.3.15.5 **Directory Data**
ENTITY TYPE NUMBER: 312
FORM NUMBER: 1

4.3.15.6 **Parameter Data**

Index	Name	Type	Description
1	CBW	Real	Character box width
2	CBH	Real	Character box height
3	FC	Integer or Pointer	Font characteristic (or pointer if negative) (Default = 1).
4	SL	Real	Slant angle (Radians) (Default = pi/2)
5	A	Real	Rotation angle (Radians)
6	M	Integer	Mirror Flag
7	VH	Integer	Rotate internal text flag
8	DXS	Real	Increment in X from X coordinate found in parent entity

| 9 | DYS | Real | Increment in Y from Y coordinate found in parent entity |
| 10 | DZS | Real | Increment in Z from Z coordinate found in parent entity |

Additional pointers as required (see 2.2.4.4.2).

APPENDIX A

PART FILE EXAMPLES

This appendix contains three sample parts encoded in the Specification ASCII format. These files are included herein to provide a guide to the usage of IGES and, as such, do not represent all design application uses. The files are a two-dimensional application using structure entities, a three-dimensional mechanical part with dimensioning, and a three-dimensional part with 2-D drawing views defined.

Example file 1 is an integrated circuit (IC) cell. The IC application was selected because of the predominance of 2-D geometry used in electrical designs. The geometry used in the cell in Figure A-1 consists of simple closed area, linear path entities and line widening property. The structure entities are nested subfigures using a network subfigure definition, array subfigure instance. A connect point is included to identify the signal port. The geometry is on five different levels, each representing a process mask. The entity label field of each directory entry record has optional text included to describe the entity's use. The entities in this file would be typical of those used in an IC application to transfer either cell libraries or a complete design between design systems. The file of a design prepared for pattern generation, with subfigures resolved and the geometry fractured, would use the flash entity exclusively. The cell file was adapted from a cell library in A Guide to LSI Implementation with kind permission from author Carlos Sequin.

Example file 2 is a three-dimensional mechanical part containing geometry entities and annotation entities typically found on engineering drawings. Included as geometry are points, lines, circular arcs and conics. For annotation the file includes linear dimensions, angular dimensions, radius dimensions, ordinate dimensions, a general label and general notes. Figure A-2 shows the mechanical part which was used during one of the early public demonstrations of intersystem data exchange.

Example file 3 is included to show the use of view entities and drawing entities in conjunction with a three-dimensional part model to convey a drawing to the receiving system. Figure A-3 shows the example drawing. In this way model geometry and viewing parameters are logically separate. A 3-D model as well as the drawing is received enabling additional views to be created if necessary, and changes to the part model are reflected in all views.
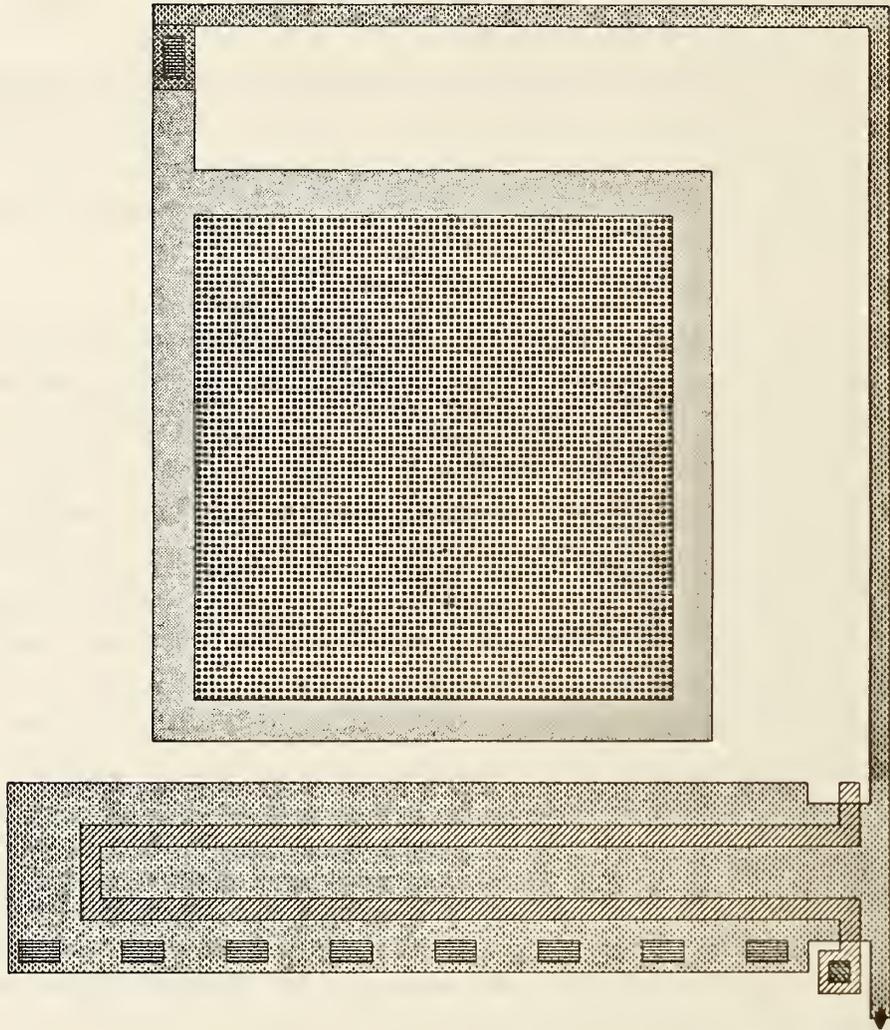
FIGURE   A - 1   ELECTRICAL PART EXAMPLE

EXAMPLE 1 ELECTRICAL PART

```
INTEGRATED CIRCUIT SEMICUSTOM CELL (ONE PART OF A LIBRARY FILE)          S       1
1H,,1H;,10H5MICRONLIB,5HPADIN,9HEXAMPLE 1,4HIGES,16,38,06,38,13,         G       1
10HIC.LIBRARY,1.0,9,2HUM,1,,13H851128.090000,0.01,265.0,9HC.H.PARKS,     G       2
23HGENERAL DYNAMICS/POMONA,4,0;                                          G       3
        308      01            1       0          0        00020201D     01
        308       0            1                      SUBFIG1      D     02
        106      02            1       1          0        00020200D     03
        106       0      4      1      63             VDDPORT      D     04
        106      03            1       1          0        00020200D     05
        106       0      4      1      63             GNDPORT      D     06
        106      04            1       1          0        00020200D     07
        106       0      4      1      63             BONDPAD      D     08
        106      05            1       7          0        00020200D     09
        106       0      5      1      63            GLASSBOX      D     10
        320      06            1       0          0        00010201D     11
        320       0            1                      CELLFIG      D     12
        408      07            1       0          0        00030200D     13
        408       0            1                        INST1      D     14
        106      08            1       3          0        00020200D     15
        106       0      3      2      63              ACTBOX      D     16
        106      10            1       3          0        00020200D     17
        106       0      3      1      63              ACTBOX      D     18
        106      11            1       3          0        00020200D     19
        106       0      3      1      11              ACTSTG      D     20
        106      12            1       3          0        00020200D     21
        106       0      3      2      63              ACTBOX      D     22
        132      14            1       3          0        00020400D     23
        132       0            1                      SIGPORT      D     24
        106      15            1       6          0        00020200D     25
        106       0      8      2      63                 CUT      D     26
        106      17            1       6          0        00020200D     27
        106       0      8      2      63                 CUT      D     28
        308      19            1       0          0        00020201D     29
        308       0            1                      SUBFIG2      D     30
        106      20            1       6          0        01030200D     31
        106       0      8      1      63              CUTDEF      D     32
        412      21            1       0          0        00030201D     33
        412       0            1                      CUTARR      D     34
        106      22            1       2          0        00020200D     35
        106       0      2      2      11             GATESTG      D     36
        106      24            1       2          0        00020200D     37
        106       0      2      2      63             GATEBOX      D     38
        106      26            1       1          0        00020200D     39
        106       0      4      1      63             GATEBOX      D     40
        406      27            1       0          0        00010200D     41
        406       0            1       5              LINWIDTH      D     42
308,0,6HPADBLK,4,03,05,07,09;                                          01P     01
106,1,5,0.,0.,0.,265.,0.,265.,-20.,0.,-20.,0.,0.;                      03P     02
106,1,5,0.,30.,-245.,245.,-245.,245.,-265.,30.,-265.,30.,-245.;        05P     03
106,1,5,0.,65.,-65.,200.,-65.,200.,-200.,65.,-200.,65.,-65.;           07P     04
106,1,5,0.,75.,-75.,190.,-75.,190.,-190.,75.,-190.,75.,-75.;           09P     05
320,1,5HPADIN,12,13,15,17,19,21,23,25,27,33,35,37,39;                  11P     06
408,01,0.,0.,0.;                                                       13P     07
106,1,5,0.,30.,-210.,222.5,-210.,222.5,-255.,30.,-255.,30.,            15P     08
-210.;                                                                 15P     09
106,1,5,0.,65.,-25.,75.,-25.,75.,-45.,65.,-45.,65.,-25.;               17P     10
106,1,3,0.,77.5,-27.5,240.,-27.5,240.,-262.5,0,1,41;                   19P     11
106,1,5,0.,222.5,-215.,237.5,-215.,237.5,-247.5,222.5,-247.5,          21P     12
222.5,-215.;                                                           21P     13
132,240.,-265.,0.,,2,1,,,,,01,2,1,11;                                  23P     14
```

```
106,1,5,0.,67.5,-32.5,72.5,-32.5,72.5,-42.5,67.5,-42.5,67.5,                        25P      15
-32.5;                                                                              25P      16
106,1,5,0.,227.5,-252.5,232.5,-252.5,232.5,-257.5,227.5,-257.5,                     27P      17
227.5,-252.5;                                                                       27P      18
308,0,7HCONTACT,1,31;                                                               29P      19
106,1,5,0.,-5.,2.5,5.,2.5,5.,-2.5,-5.,-2.5,-5.,2.5;                                 31P      20
412,29,1.0,37.5,-250.,0.,8,1,25.,0.,0.,0;                                           33P      21
106,1,6,0.,232.5,-212.5,232.5,-222.5,50.,-222.5,50.,-240.,                          35P      22
232.5,-240.,232.5,-247.5,0,1,41;                                                    35P      23
106,1,5,0.,225.,-250.,235.,-250.,235.,-260.,225.,-260.,225.,                        37P      24
-250.;                                                                              37P      25
106,1,5,0.,65.,-30.,75.,-30.,75.,-65.,65.,-65.,65.,-30.;                            39P      26
406,5,5.0,1,1,0,0;                                                                  41P      27
S       1G       3D       42P       27                                               T       1
```
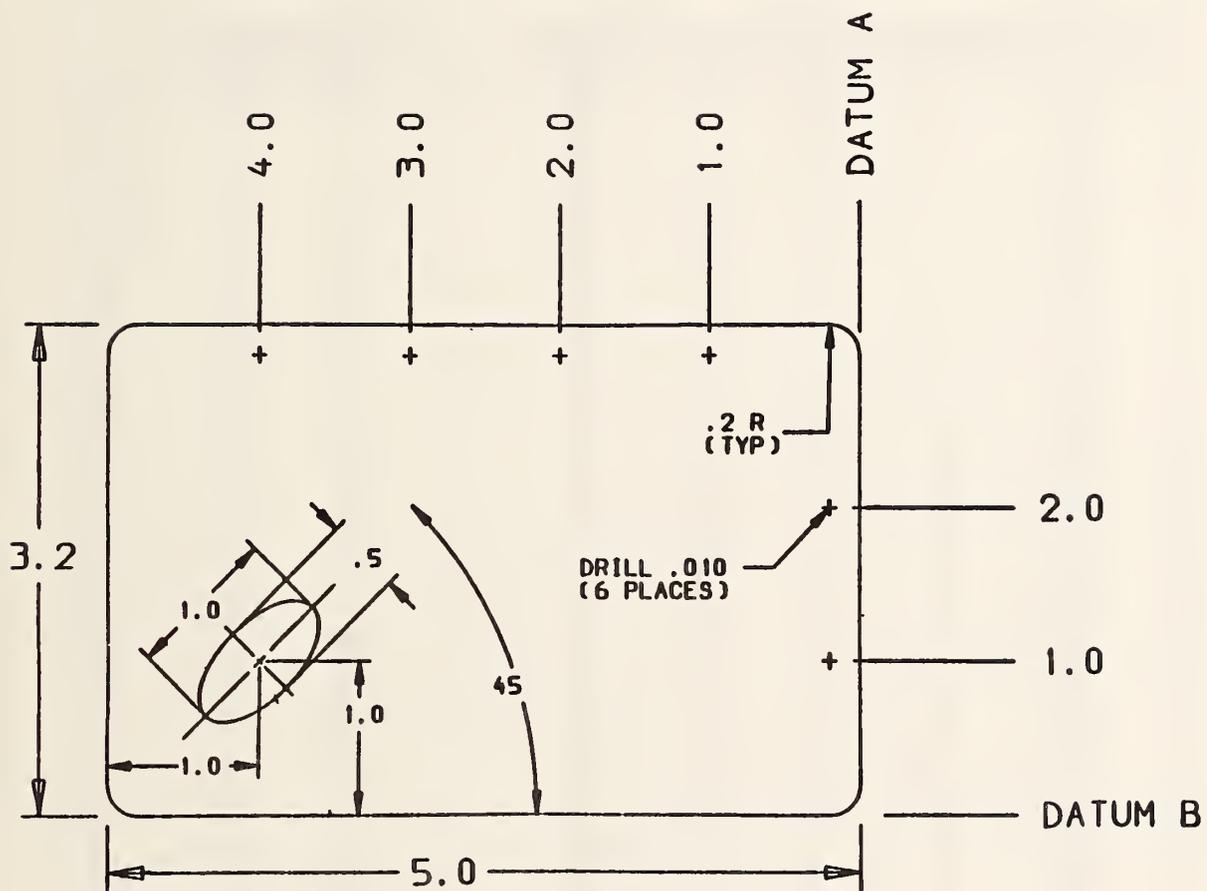
FIGURE A - 2 MECHANICAL PART EXAMPLE

EXAMPLE 2 MECHANICAL PART

```
THIS IS A SAMPLE PART FOR USE IN THE AUTOFACT 82 IGES DEMONSTRATION    S     1
,,8HPANEL123,10HPANEL.IGES,4HEX 2,4HIGES,16,38,7,38,14,8HPANEL123,1.0,   G     1
1,4HINCH,1,0.028,13H850816.144004,0.0005,100.0,9HD. BRIGGS,6HBOEING,4,0;G     2
      124        1       1       1       0       0       0       0      0D     1
      124        0       0       1       0                              0D     2
      212        2       1       1       5       0       1       0  10100D     3
      212        0       0       3       0                              0D     4
      214        5       1       1       5       0       1       0  10100D     5
      214        0       0       1       2                              0D     6
      210        6       1       1       5       0       1       0    100D     7
      210        0       0       1       0                              0D     8
      110        7       1       1       1       0       0       0      0D     9
      110        0       0       1       0                              0D    10
      110        8       1       1       1       0       0       0      0D    11
      110        0       0       1       0                              0D    12
      110        9       1       1       1       0       0       0      0D    13
      110        0       0       1       0                              0D    14
      110       10       1       1       1       0       0       0      0D    15
      110        0       0       1       0                              0D    16
      100       11       1       1       1       0       1       0      0D    17
      100        0       0       1       0                              0D    18
      100       12       1       1       1       0       1       0      0D    19
      100        0       0       1       0                              0D    20
      100       13       1       1       1       0       1       0      0D    21
      100        0       0       1       0                              0D    22
      100       14       1       1       1       0       1       0      0D    23
      100        0       0       1       0                              0D    24
      116       15       1       1       2       0       0       0      0D    25
      116        0       0       1       0                              0D    26
      116       16       1       1       2       0       0       0      0D    27
      116        0       0       1       0                              0D    28
      116       17       1       1       2       0       0       0      0D    29
      116        0       0       1       0                              0D    30
      116       18       1       1       2       0       0       0      0D    31
      116        0       0       1       0                              0D    32
      104       19       1       1       3       0       1       0      0D    33
      104        0       0       2       1                              0D    34
      116       21       1       1       2       0       0       0      0D    35
      116        0       0       1       0                              0D    36
      116       22       1       1       2       0       0       0      0D    37
      116        0       0       1       0                              0D    38
      212       23       1       1       4       0       1       0  10100D    39
      212        0       0       1       0                              0D    40
      214       24       1       1       4       0       1       0  10100D    41
      214        0       0       1       2                              0D    42
      214       25       1       1       4       0       1       0  10100D    43
      214        0       0       1       2                              0D    44
      106       26       1       1       4       0       1       0  10100D    45
      106        0       0       1      40                              0D    46
      106       27       1       1       4       0       1       0  10100D    47
      106        0       0       1      40                              0D    48
      216       28       1       1       4       0       1       0    100D    49
      216        0       0       1       0                              0D    50
      212       29       1       1       4       0       1       0  10100D    51
      212        0       0       1       0                              0D    52
      214       30       1       1       4       0       1       0  10100D    53
      214        0       0       1       2                              0D    54
      214       31       1       1       4       0       1       0  10100D    55
      214        0       0       1       2                              0D    56
      106       32       1       1       4       0       1       0  10100D    57
```

106	0	0	1	40				0D	58
106	33	1	1	4	0	1	0	10100D	59
106	0	0	1	40				0D	60
216	34	1	1	4	0	1	0	100D	61
216	0	0	1	0				0D	62
212	35	1	1	5	0	1	0	10100D	63
212	0	0	1	0				0D	64
106	36	1	1	5	0	1	0	10100D	65
106	0	0	1	40				0D	66
218	37	1	1	5	0	1	0	100D	67
218	0	0	1	0				0D	68
212	38	1	1	5	0	1	0	10100D	69
212	0	0	1	0				0D	70
106	39	1	1	5	0	1	0	10100D	71
106	0	0	1	40				0D	72
218	40	1	1	5	0	1	0	100D	73
218	0	0	1	0				0D	74
212	41	1	1	5	0	1	0	10100D	75
212	0	0	1	0				0D	76
106	42	1	1	5	0	1	0	10100D	77
106	0	0	1	40				0D	78
218	43	1	1	5	0	1	0	100D	79
218	0	0	1	0				0D	80
212	44	1	1	5	0	1	0	10100D	81
212	0	0	1	0				0D	82
106	45	1	1	5	0	1	0	10100D	83
106	0	0	1	40				0D	84
218	46	1	1	5	0	1	0	100D	85
218	0	0	1	0				0D	86
212	47	1	1	5	0	1	0	10100D	87
212	0	0	1	0				0D	88
106	48	1	1	5	0	1	0	10100D	89
106	0	0	1	40				0D	90
218	49	1	1	5	0	1	0	100D	91
218	0	0	1	0				0D	92
212	50	1	1	5	0	1	0	10100D	93
212	0	0	1	0				0D	94
106	51	1	1	5	0	1	0	10100D	95
106	0	0	1	40				0D	96
218	52	1	1	5	0	1	0	100D	97
218	0	0	1	0				0D	98
212	53	1	1	5	0	1	0	10100D	99
212	0	0	2	0				0D	100
106	55	1	1	5	0	1	0	10100D	101
106	0	0	1	40				0D	102
218	56	1	1	5	0	1	0	100D	103
218	0	0	1	0				0D	104
212	57	1	1	5	0	1	0	10100D	105
212	0	0	2	0				0D	106
106	59	1	1	5	0	1	0	10100D	107
106	0	0	1	40				0D	108
218	60	1	1	5	0	1	0	100D	109
218	0	0	1	0				0D	110
212	61	1	1	5	0	1	0	10100D	111
212	0	0	2	0				0D	112
214	63	1	1	5	0	1	0	10100D	113
214	0	0	1	2				0D	114
222	64	1	1	5	0	1	0	100D	115
222	0	0	1	0				0D	116
212	65	1	1	6	0	1	0	10100D	117

431

212	0	0	1	0				0D	118
214	66	1	1	6	0	1	0	10100D	119
214	0	0	1	2				0D	120
214	67	1	1	6	0	1	0	10100D	121
214	0	0	1	2				0D	122
106	68	1	1	6	0	1	0	1010100D	123
106	0	0	1	40				0D	124
106	69	1	1	6	0	1	0	10100D	125
106	0	0	1	40				0D	126
216	70	1	1	6	0	1	0	100D	127
216	0	0	1	0				0D	128
212	71	1	1	6	0	1	0	10100D	129
212	0	0	1	0				0D	130
214	72	1	1	6	0	1	0	10100D	131
214	0	0	1	2				0D	132
214	73	1	1	6	0	1	0	10100D	133
214	0	0	1	2				0D	134
106	74	1	1	6	0	1	0	10100D	135
106	0	0	1	40				0D	136
106	75	1	1	6	0	1	0	1010100D	137
106	0	0	1	40				0D	138
216	76	1	1	6	0	1	0	100D	139
216	0	0	1	0				0D	140
212	77	1	1	6	0	1	0	10100D	141
212	0	0	1	0				0D	142
214	78	1	1	6	0	1	0	10100D	143
214	0	0	1	2				0D	144
214	79	1	1	6	0	1	0	10100D	145
214	0	0	1	2				0D	146
106	80	1	1	6	0	1	0	10100D	147
106	0	0	1	40				0D	148
106	81	1	1	6	0	1	0	10100D	149
106	0	0	1	40				0D	150
216	82	1	1	6	0	1	0	100D	151
216	0	0	1	0				0D	152
212	83	1	1	6	0	1	0	10100D	153
212	0	0	1	0				0D	154
214	84	1	1	6	0	1	0	10100D	155
214	0	0	1	2				0D	156
214	85	1	1	6	0	1	0	10100D	157
214	0	0	1	2				0D	158
106	86	1	1	6	0	1	0	10100D	159
106	0	0	1	40				0D	160
106	87	1	1	6	0	1	0	10100D	161
106	0	0	1	40				0D	162
216	88	1	1	6	0	1	0	100D	163
216	0	0	1	0				0D	164
110	89	1	4	6	0	0	0	100D	165
110	0	0	1	0				0D	166
110	90	1	4	6	0	0	0	100D	167
110	0	0	1	0				0D	168
212	91	1	1	6	0	1	0	10100D	169
212	0	0	1	0				0D	170
214	92	1	1	6	0	1	0	10100D	171
214	0	0	1	2				0D	172
214	93	1	1	6	0	1	0	10100D	173
214	0	0	1	2				0D	174
106	94	1	1	6	0	1	0	1010100D	175
106	0	0	1	40				0D	176
106	95	1	1	6	0	1	0	1010100D	177

```
      106         0         0         1        40                                    0D   178
      202        96         1         1         6         0         1         0     100D   179
      202         0         0         1         0                                    0D   180
124,1.000,0.0,0.0,0.0,0.0,1.000,0.0,0.0,0.0,0.0,1.000,0.0,0,0;                        1P     1
212,2,10,0.98,0.10,1,1.571,0.0,0,0,3.210,1.656,0.0,10HDRILL .010                      3P     2
,10,1.02,0.100,1,1.571,0.0,0,0,3.210,1.506,0.0,10H(6 PLACES),0,                       3P     3
0;                                                                                    3P     4
214,2,0.150,0.050,0.0,4.800,2.000,4.562;1.546,4.262,1.546,0,0;                        5P     5
210,3,1,5,0,0;                                                                        7P     6
110,0.0,0.200,0.0,0.0,3.000,0.0,0,0;                                                  9P     7
110,0.200,0.0,0.0,4.800,0.0,0.0,0,0;                                                 11P     8
110,5.000,0.200,0.0,5.000,3.000,0.0,0,0;                                             13P     9
110,0.200,3.200,0.0,4.800,3.200,0.0,0,0;                                             15P    10
100,0.0,4.800,3.000,5.000,3.000,4.800,3.200,0,0;                                     17P    11
100,0.0,0.200,3.000,0.200,3.200,0.0,3.000,0,0;                                       19P    12
100,0.0,0.200,0.200,0.0,0.200,0.200,0.00,0,0;                                        21P    13
100,0.0,4.800,0.200,4.800,0.0,5.000,0.200,0,0;                                       23P    14
116,4.000,3.000,0.0,0,0;                                                             25P    15
116,3.000,3.000,0.0,0,0;                                                             27P    16
116,2.000,3.000,0.0,0,0;                                                             29P    17
116,1.000,3.000,0.0,0,0;                                                             31P    18
104,10.000,-12.000,10.000,-8.000,-8.000,7.000,0.0,1.3538,1.3538,                     33P    19
1.3538,1.3538,0,0;                                                                   33P    20
116,4.800,1.000,0.0,0,0;                                                             35P    21
116,4.800,2.000,0.0,0,0;                                                             37P    22
212,1,3,0.421,0.156,1,1.571,0.0,0,0,-0.639,1.614,0.0,3H3.2,0,0;                      39P    23
214,1,0.150,0.050,0.0,-0.454,3.200,-0.454,1.870,0,0;                                 41P    24
214,1,0.150,0.050,0.0,-0.454,0.0,-0.454,1.514,0,0;                                   43P    25
106,1,3,0.0,0.0,3.200,-0.094,3.200,-0.579,3.200,0,0;                                 45P    26
106,1,3,0.0,0.0,0.0,-0.094,0.0,-0.579,0.0,0,0;                                       47P    27
216,39,41,43,45,47,0,0;                                                              49P    28
212,1,3,0.437,0.156,1,1.571,0.0,0,0,2.032,-0.447,0.0,3H5.0,0,0;                      51P    29
214,1,0.150,0.050,0.0,0.0,-0.369,1.932,-0.369,0,0;                                   53P    30
214,1,0.150,0.050,0.0,5.000,-0.369,2.519,-0.369,0,0;                                 55P    31
106,1,3,0.0,0.0,0.0,0.0,-0.094,0.0,-0.494,0,0;                                       57P    32
106,1,3,0.0,5.000,0.0,5.000,-0.094,5.000,-0.494,0,0;                                 59P    33
216,51,53,55,57,59,0,0;                                                              61P    34
212,1,3,0.374,0.156,1,1.571,1.571,0,0,4.078,4.181,0.0,3H1.0,0,0;                     63P    35
106,1,3,0.0,4.000,3.094,4.000,3.188,4.000,3.993,0,0;                                 65P    36
218,63,65,0,0;                                                                       67P    37
212,1,3,0.421,0.156,1,1.571,1.571,0,0,3.078,4.183,0.0,3H2.0,0,0;                     69P    38
106,1,3,0.0,3.000,3.094,3.000,3.188,3.000,3.996,0,0;                                 71P    39
218,69,71,0,0;                                                                       73P    40
212,1,3,0.437,0.156,1,1.571,1.571,0,0,2.078,4.177,0.0,3H3.0,0,0;                     75P    41
106,1,3,0.0,2.000,3.094,2.000,3.188,2.000,3.989,0,0;                                 77P    42
218,75,77,0,0;                                                                       79P    43
212,1,3,0.437,0.156,1,1.571,1.571,0,0,1.078,4.177,0.0,3H4.0,0,0;                     81P    44
106,1,3,0.0,1.000,3.094,1.000,3.188,1.000,3.989,0,0;                                 83P    45
218,81,83,0,0;                                                                       85P    46
212,1,3,0.374,0.156,1,1.571,0.0,0,0,6.211,0.922,0.0,3H1.0,0,0;                       87P    47
106,1,3,0.0,4.894,1.000,4.988,1.000,6.024,1.000,0,0;                                 89P    48
218,87,89,0,0;                                                                       91P    49
212,1,3,0.421,0.156,1,1.571,0.0,0,0,6.211,1.922,0.0,3H2.0,0,0;                       93P    50
106,1,3,0.0,4.894,2.000,4.988,2.000,6.024,2.000,0,0;                                 95P    51
218,93,95,0,0;                                                                       97P    52
212,1,7,1.248,0.156,1,1.571,0.,0,0,6.23,-0.078,0.0,7HDATUM B,0,                      99P    53
0;                                                                                   99P    54
106,1,3,0.0,5.094,0.0,5.188,0.0,6.042,0.0,0,0;                                      101P    55
218,99,101,0,0;                                                                     103P    56
212,1,7,1.232,0.156,1,1.571,1.571,0,0,5.078,4.193,0.,7HDATUM A,                     105P    57
```

433

```
0,0;                                                                      105P    58
106,1,3,0.0,5.000,3.094,5.000,3.187,5.000,4.006,0,0;                      107P    59
218,105,107,0,0;                                                          109P    60
212,2,4,0.35,0.100,1,1.571,0.,0,0,4.029,2.611,0.,4H.2 R,5,0.500,          111P    61
0.100,1,1.571,0.0,0,0,4.029,2.461,0.0,5H(TYP),0,0;                        111P    62
214,2,0.150,0.050,0.0,4.877,3.185,4.862,2.511,4.562,2.511,0,0;            113P    63
222,111,113,4.800,3.000,0,0;                                              115P    64
212,1,3,0.240,0.100,1,1.571,0.0,0,0,1.559,0.602,0.0,3H1.0,0,0;            117P    65
214,1,0.150,0.050,0.0,1.663,0.0,1.663,0.538,0,0;                          119P    66
214,1,0.150,0.050,0.0,1.663,1.000,1.663,0.766,0,0;                        121P    67
106,1,3,0.0,0.200,0.0,0.294,0.0,1.788,0.0,0,0;                            123P    68
106,1,3,0.0,1.000,1.000,1.094,1.000,1.788,1.000,0,0;                      125P    69
216,117,119,121,123,125,0,0;                                              127P    70
212,1,3,0.240,0.100,1,1.571,0.0,0,0,0.537,0.275,0.0,3H1.0,0,0;            129P    71
214,1,0.150,0.050,0.0,1.000,0.325,0.809,0.325,0,0;                        131P    72
214,1,0.150,0.050,0.0,0.0,0.325,0.473,0.325,0,0;                          133P    73
106,1,3,0.0,1.000,1.000,1.000,0.906,1.000,0.200,0,0;                      135P    74
106,1,3,0.0,0.0,0.012,0.0,0.106,0.0,0.200,0,0;                            137P    75
216,129,131,133,135,137,0,0;                                              139P    76
212,1,3,0.240,0.100,1,1.571,0.0,0,0,0.470,1.289,0.0,3H1.0,0,0;            141P    77
214,1,0.150,0.050,0.0,0.971,1.736,0.688,1.453,0,0;                        143P    78
214,1,0.150,0.050,0.0,0.264,1.029,0.459,1.225,0,0;                        145P    79
106,1,3,0.0,1.354,1.354,1.287,1.420,0.882,1.825,0,0;                      147P    80
106,1,3,0.0,0.646,0.646,0.580,0.713,0.175,1.118,0,0;                      149P    81
216,141,143,145,147,149,0,0;                                              151P    82
212,1,2,0.170,0.100,1,1.571,0.0,0,0,1.631,1.622,0.0,2H.5,0,0;             153P    83
214,1,0.150,0.050,0.0,1.863,1.509,2.146,1.226,0,0;                        155P    84
214,1,0.150,0.050,0.0,1.509,1.863,1.226,2.146,0,0;                        157P    85
106,1,3,0.0,1.177,0.823,1.243,0.890,1.951,1.598,0,0;                      159P    86
106,1,3,0.0,0.823,1.177,0.890,1.243,1.598,1.951,0,0;                      161P    87
216,153,155,157,159,161,0,0;                                              163P    88
110,0.500,0.500,0.0,1.500,1.500,0.0,0,0;                                  165P    89
110,1.225,0.775,0.0,0.775,1.225,0.0,0,0;                                  167P    90
212,1,2,0.220,0.100,1,1.571,0.0,0,0,2.566,0.786,0.0,2H45,0,0;             169P    91
214,1,0.150,0.050,0.0,2.847,0.0,2.754,0.722,0,0;                          171P    92
214,1,0.150,0.050,0.0,2.013,2.013,2.683,0.951,0,0;                        173P    93
106,1,3,0.0,4.988,0.0,4.894,0.0,2.972,0.0,0,0;                            175P    94
106,1,3,0.0,2.000,2.000,2.066,2.066,2.101,2.101,0,0;                      177P    95
202,169,175,177,0.0,0.0,2.847,171,173,0,0;                                179P    96
S      1G      2D      180P      96                                         T     1
```
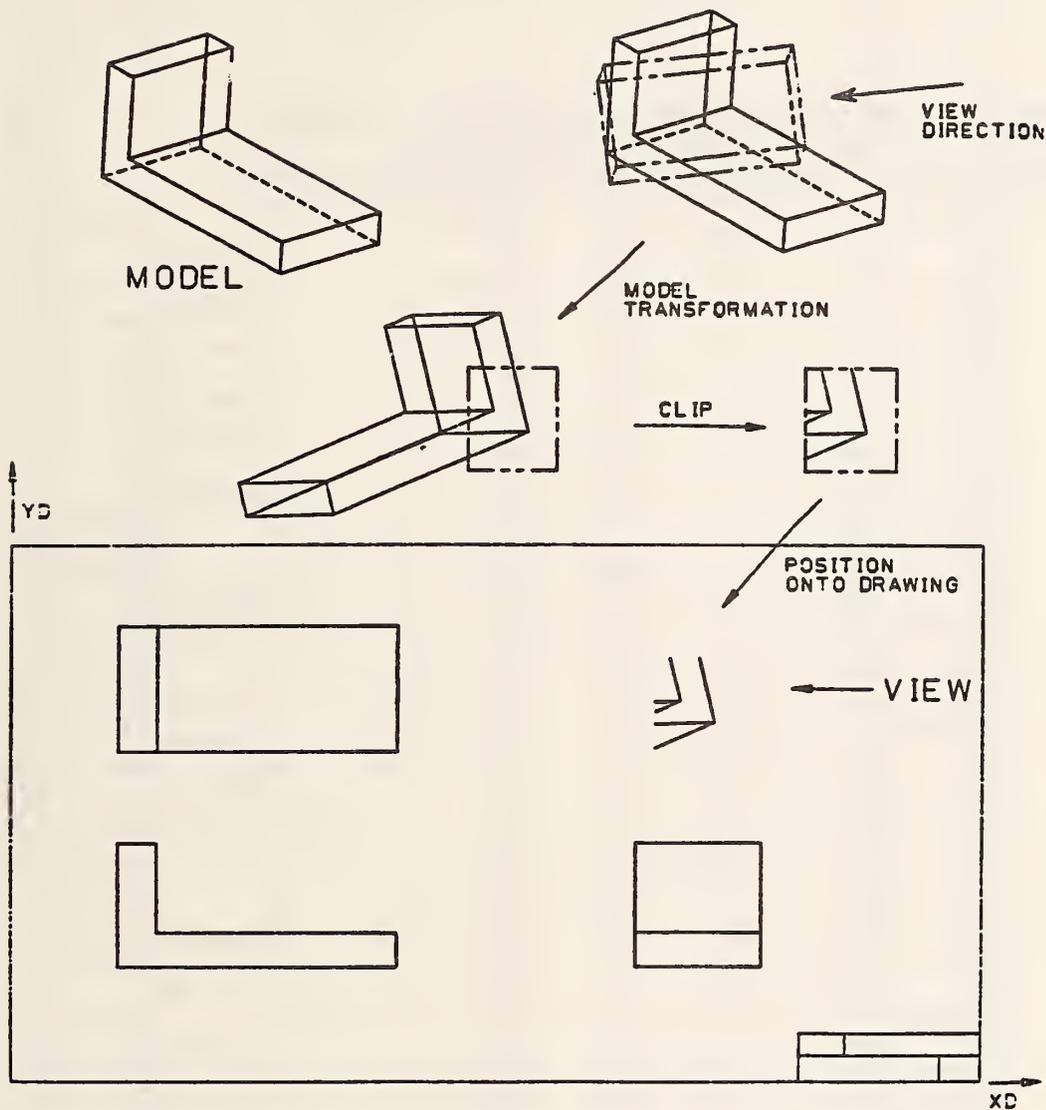
FIGURE A - 3   DRAWING AND VIEW EXAMPLE

435

EXAMPLE 3 VIEW AND DRAWING PART

```
EXAMPLE OF MODEL WITH VIEW AND DRAWING ENTITIES -----------------        S       1
1H,,1H;,19HPROD ID FROM SENDER,11HMANUAL.IGS ,9HEXAMPLE 3,                 G       1
4HIGES,32,38,8,153,15,20HPROD ID FOR RECEIVER,                            G       2
1.00000000E+00,1,4HINCH,8,1.00000000E+00,13H851203.163006,                G       3
1.00000000E-06,1.00000000E+03,6HAUTHOR,12HORGANIZATION,4,0;               G       4
      406        1        1        0        0        0        0    000010300D       1
      406        0        0        1       15               NAME_PRP      1D       2
      124        2        1        0        0        0        0    000010300D       3
      124        0        0        2        0               ROT MAT       1D       4
      108        4        1        0        0        0        0    000010201D       5
      108        0        0        2        1               VIEW_PLN      1D       6
      108        6        1        0        0        0        0    000010201D       7
      108        0        0        2        1               VIEW_PLN      1D       8
      108        8        1        0        0        0        0    000010201D       9
      108        0        0        2        1               VIEW_PLN      1D      10
      108       10        1        0        0        0        0    000010201D      11
      108        0        0        2        1               VIEW_PLN      1D      12
      410       12        1        0        0        0        3    000000201D      13
      410        0        0        1        0                   VIEW      1D      14
      406       13        1        0        0        0        0    000010300D      15
      406        0        0        1       15               NAME_PRP      2D      16
      124       14        1        0        0        0        0    000010300D      17
      124        0        0        1        0               ROT MAT       2D      18
      108       15        1        0        0        0        0    000010201D      19
      108        0        0        1        1               VIEW_PLN      1D      20
      108       16        1        0        0        0        0    000010201D      21
      108        0        0        1        1               VIEW_PLN      1D      22
      108       17        1        0        0        0        0    000010201D      23
      108        0        0        1        1               VIEW_PLN      1D      24
      108       18        1        0        0        0        0    000010201D      25
      108        0        0        1        1               VIEW_PLN      1D      26
      410       19        1        0        0        0       17    000000201D      27
      410        0        0        1        0                   VIEW      2D      28
      406       20        1        0        0        0        0    000010300D      29
      406        0        0        1       15               NAME_PRP      3D      30
      124       21        1        0        0        0        0    000010300D      31
      124        0        0        1        0               ROT MAT       3D      32
      108       22        1        0        0        0        0    000010201D      33
      108        0        0        1        1               VIEW_PLN      1D      34
      108       23        1        0        0        0        0    000010201D      35
      108        0        0        1        1               VIEW_PLN      1D      36
      108       24        1        0        0        0        0    000010201D      37
      108        0        0        1        1               VIEW_PLN      1D      38
      108       25        1        0        0        0        0    000010201D      39
      108        0        0        1        1               VIEW_PLN      1D      40
      410       26        1        0        0        0       31    000000201D      41
      410        0        0        1        0                   VIEW      3D      42
      406       27        1        0        0        0        0    000010300D      43
      406        0        0        1       15               NAME_PRP      4D      44
      108       28        1        0        0        0        0    000010201D      45
      108        0        0        1        1               VIEW_PLN      1D      46
      108       29        1        0        0        0        0    000010201D      47
      108        0        0        1        1               VIEW_PLN      1D      48
      108       30        1        0        0        0        0    000010201D      49
      108        0        0        1        1               VIEW_PLN      1D      50
      108       31        1        0        0        0        0    000010201D      51
      108        0        0        1        1               VIEW_PLN      1D      52
      410       32        1        0        0        0        0    000000201D      53
      410        0        0        1        0                   VIEW      4D      54
      110       33        1        1    10002        0        0    000010100D      55
```

110	1	4	1	0			LINE	1D	56
110	34	1	1	10002	0	0	000010100D		57
110	1	4	1	0			LINE	2D	58
110	35	1	1	10002	0	0	000010100D		59
110	1	4	1	0			LINE	3D	60
110	36	1	1	10002	0	0	000010100D		61
110	1	4	1	0			LINE	4D	62
110	37	1	1	10002	0	0	000010100D		63
110	1	4	1	0			LINE	5D	64
110	38	1	1	10004	0	0	000010100D		65
110	1	4	1	0			LINE	6D	66
110	39	1	1	10004	0	0	000010100D		67
110	1	4	1	0			LINE	7D	68
110	40	1	1	10004	0	0	000010100D		69
110	1	4	1	0			LINE	8D	70
110	41	1	1	10004	0	0	000010100D		71
110	1	4	1	0			LINE	9D	72
406	42	1	1	0	0	0	000010300D		73
406	0	0	1	15			NAME_PRP	5D	74
404	43	1	1	0	0	0	000000201D		75
404	0	0	2	0			DRAWNG	1D	76
110	45	1	1	10002	0	0	000000000D		77
110	1	2	1	0			LINE	10D	78
110	46	1	1	10002	0	0	000000000D		79
110	1	2	1	0			LINE	11D	80
110	47	1	1	10002	0	0	000000000D		81
110	1	2	1	0			LINE	12D	82
110	48	1	1	10002	0	0	000000000D		83
110	1	2	1	0			LINE	13D	84
110	49	1	1	10002	0	0	000000000D		85
110	1	2	1	0			LINE	14D	86
110	50	1	1	10002	0	0	000000000D		87
110	1	2	1	0			LINE	15D	88
110	51	1	1	10002	0	0	000000000D		89
110	1	2	1	0			LINE	16D	90
110	52	1	1	10002	0	0	000000000D		91
110	1	2	1	0			LINE	17D	92
110	53	1	1	10002	0	0	000000000D		93
110	1	2	1	0			LINE	18D	94
110	54	1	1	10002	0	0	000000000D		95
110	1	2	1	0			LINE	19D	96
110	55	1	1	10002	0	0	000000000D		97
110	1	2	1	0			LINE	20D	98
110	56	1	1	10002	0	0	000000000D		99
110	1	2	1	0			LINE	21D	100
110	57	1	1	10002	0	0	000000000D		101
110	1	2	1	0			LINE	22D	102
110	58	1	1	10002	0	0	000000000D		103
110	1	2	1	0			LINE	23D	104
110	59	1	1	10002	0	0	000000000D		105
110	1	2	1	0			LINE	24D	106
110	60	1	1	10002	0	0	000000000D		107
110	1	2	1	0			LINE	25D	108
110	61	1	1	10002	0	0	000000000D		109
110	1	2	1	0			LINE	26D	110
110	62	1	1	10002	0	0	000000000D		111
110	1	2	1	0			LINE	27D	112

```
406,1,7HCLIPPED;                                              00000001P      1
124,-0.67499,-0.171,0.71774,1.00728,-0.24401,0.96977,0.00157, 00000003P      2
-0.821391,-0.69631,-0.17408,-0.69631,4.613057,0,0;            00000003P      3
```

```
108,0.67499,0.171,-0.71774,5.0055,0,6.390347,2.455541,-0.379235,00000005P      4
0.,0,0;                                                              00000005P      5
108,-0.24401,0.96977,0.00157,3.55308,0,3.025032,4.420965,           00000007P      6
2.494731,0.,0,0;                                                     00000007P      7
108,-0.67499,-0.171,0.71774,2.99094,0,0.992841,1.088152,            00000009P      8
5.360119,0.,0,0;                                                     00000009P      9
108,0.24401,-0.96977,-0.00157,1.9103,0,4.358156,-0.877273,          00000011P     10
2.486153,0.,0,0;                                                     00000011P     11
410,4,1.,5,7,9,11,0,0,0,1,1;                                         00000013P     12
406,1,5HRIGHT;                                                       00000015P     13
124,0.,0.,-1.,0.,0.,1.,0.,0.,1.,0.,0.,0.,0,0;                        00000017P     14
108,0.,0.,1.,36.,0,0.,0.,36.,0.,0,0;                                 00000019P     15
108,0.,1.,0.,24.59627,0,0.,24.59627, 0.,0.,0,0;                      00000021P     16
108,0.,0.,-1.,36.,0,0.,0.,-36.,0.,0,0;                               00000023P     17
108,0.,-1.,0.,24.59627,0,0.,-24.59627, 0.,0.,0,0;                    00000025P     18
410,3,1.,19,21,23,25,0,0,0,1,15;                                     00000027P     19
406,1,3HTOP;                                                         00000029P     20
124,1.,0.,0.,0.,0.,0.,-1.,0.,0.,1.,0.,0.,0,0;                        00000031P     21
108,-1.,0.,0.,36.,0,-36.,0.,0.,0.,0,0;                               00000033P     22
108,0.,0.,-1.,24.59627,0,0.,0.,-24.59627, 0.,0,0;                    00000035P     23
108,1.,0.,0.,36.,0,36.,0.,0.,0.,0,0;                                 00000037P     24
108,0.,0.,1.,24.59627,0,0.,0.,24.59627, 0.,0,0;                      00000039P     25
410,2,1.,33,35,37,39,0,0,0,1,29;                                     00000041P     26
406,1,5HFRONT;                                                       00000043P     27
108,-1.,0.,0.,36.,0,-36.,0.,0.,0.,0,0;                               00000045P     28
108,0.,1.,0.,24.59627,0,0.,24.59627, 0.,0.,0,0;                      00000047P     29
108,1.,0.,0.,36.,0,36.,0.,0.,0.,0,0;                                 00000049P     30
108,0.,-1.,0.,24.59627,0,0.,-24.59627, 0.,0.,0,0;                    00000051P     31
410,1,1.,45,47,49,51,0,0,0,1,43;                                     00000053P     32
110,67.,0.,0.,67.,2.,0.,0,0;                                         00000055P     33
110,60.,4.,0.,60.,2.,0.,0,0;                                         00000057P     34
110,57.,4.,0.,57.,0.,0.,0,0;                                         00000059P     35
110,70.,4.,0.,57.,4.,0.,0,0;                                         00000061P     36
110,70.,2.,0.,57.,2.,0.,0,0;                                         00000063P     37
110,0.,40.,0.,0.,0.,0.,0,0;                                          00000065P     38
110,70.,40.,0.,0.,40.,0.,0,0;                                        00000067P     39
110,70.,0.,0.,70.,40.,0.,0,0;                                        00000069P     40
110,0.,0.,0.,70.,0.,0.,0,0;                                          00000071P     41
406,1,7HDRAWING;                                                     00000073P     42
404,4,13,51.,29.,27,46.,8.,41,7.,24.,53,7.,8.,9,55,57,59,61,63,      00000075P     43
65,67,69,71,0,1,73;                                                 00000075P     44
110,0.,9.,0.,0.,9.,-9.,0,0;                                          00000077P     45
110,2.5,9.,0.,2.5,9.,-9.,0,0;                                        00000079P     46
110,2.5,2.5,0.,2.5,2.5,-9.,0,0;                                      00000081P     47
110,20.,2.5,0.,20.,2.5,-9.,0,0;                                      00000083P     48
110,20.,0.,0.,20.,0.,-9.,0,0;                                        00000085P     49
110,0.,0.,0.,0.,0.,-9.,0,0;                                          00000087P     50
110,0.,9.,-9.,0.,0.,-9.,0,0;                                         00000089P     51
110,2.5,9.,-9.,0.,9.,-9.,0,0;                                        00000091P     52
110,2.5,2.5,-9.,2.5,9.,-9.,0,0;                                      00000093P     53
110,20.,2.5,-9.,2.5,2.5,-9.,0,0;                                     00000095P     54
110,20.,0.,-9.,20.,2.5,-9.,0,0;                                      00000097P     55
110,0.,0.,-9.,20.,0.,-9.,0,0;                                        00000099P     56
110,0.,9.,0.,0.,0.,0.,0,0;                                           00000101P     57
110,2.5,9.,0.,0.,9.,0.,0,0;                                          00000103P     58
110,2.5,2.5,0.,2.5,9.,0.,0,0;                                        00000105P     59
110,20.,2.5,0.,2.5,2.5,0.,0,0;                                       00000107P     60
110,20.,0.,0.,20.,2.5,0.,0,0;                                        00000109P     61
110,0.,0.,0.,20.,0.,0.,0,0;                                          00000111P     62
S      2G        4D      112P        62                                       T      1
```

APPENDIX B

ELECTRICAL/ELECTRONIC PRODUCT REPRESENTATION

INTRODUCTION

Purpose

The purpose of this appendix is to provide IGES implementors and users with a roadmap of the IGES representation of electrical/electronic product designs. The topics of discussion will include (but are not limited to) design, engineering, manufacturing, testing, and inspection.

Assumptions

The reader should have a basic understanding of electrical/electronic product design using CAD/CAM/CAE tools, including (but not limited to) connectivity, component descriptions, placement and routing, and the manufacturing interface. Each topic will be discussed in general, but these discussions are not intended to provide a tutorial in the subject.

CONNECTIVITY

In General

Forming a connection between two or more items requires the ability to represent the following:

(1)   the exact location of each connection point
(2)   the signal formed and its identification (if any)
(3)   the physical connection between the items (if any)

The term "connect node" will refer to a database entity which represents the exact location of connection. The term "link" will refer to the representation of the signal formed, and "signal name" will refer to the signal identifier. The

term "join" will refer to the database entity or entities which represent the physical connection between the items.

Each item to be connected requires a connect node to represent each possible connection point of the item. A signal may be formed between any such items by a link which references the connect nodes to be connected. This creates an associativity between the connect nodes, and thus the connected items. The signal name may be used to uniquely identify the particular signal formed. The join may be used to provide a graphical representation of the signal. In electrical applications the join will most often be represented by a line (schematic) or a widened line (printed wiring board).

In electrical applications, the items to be connected are typically electrical components (i.e., resistor, 16-pin DIP, etc.). Most often, these components are represented by subfigures which are defined once, then referenced (instanced) in the database for each occurrence of the component. Each pin of the component is a potential connection point in a signal; thus each subfigure has a connect node defined for each pin. When such a subfigure is instanced, its connect nodes must also be instanced. This allows each connect node to participate in the unique signal to which it belongs. An instanced connect node, when added to a signal, is different from its definition which is not a member of any signal.

These subfigures, representing electrical components, often contain text describing the component and its pins. In some cases (e.g., part number), this text is fixed and unchanging. In other cases (e.g., reference designator), the text may be variable, and thus may not be filled in until the subfigure is instanced. This text (sometimes called a "text node"), like the connect node, is instanced along with its parent subfigure. In some cases, a connect node and a text node are related (e.g., the connect node represents a component pin and the text node represents the pin number).

In IGES

In IGES, the connect node is represented by the Connect Point entity. The text node is represented by the Text Display Template entity. The Flow Associativity entity represents a signal and contains the link, signal name, and pointers to the join entities. The Network Subfigure entities (definition and instance) represent electrical components which participate in signals. A number of property entities will also be used, as mentioned below.

Network Subfigure Construction

A component is constructed using the Network Subfigure Definition entity. The graphics representing the component geometry are referenced as for the Subfigure Definition entity. In addition, a separate set of pointers to defining Connect Point entities is provided. These Connect Point entities define the locations and characteristics of the component's pins. Properties, for example the Part Name property, may be attached to the Network Subfigure Definition entity.

Connect Points

A component pin (or surface mounted device pad, IC I/0 port, lead frame, schematic symbol lead, etc.) is represented by the Connect Point entity. The Connect Point entity is used in both logical and physical product designs. The exact location in model space is specified, along with several characteristic flags (connection type, function type, I/0 direction). There is a pointer to the parent Network Subfigure entity (definition or instance), which provides a much-needed association for signal processing. An additional Subfigure Instance pointer is provided for Connect Point display. This allows a graphical symbol to be displayed, representing the Connect Point. The pin number is provided in the Function Connect Point Identifier field, along with a pointer to a Text Display Template for pin number display. A pin function name is provided in the Connect Point Function Name field, along with a pointer to a Text Display Template for its display.

Signal Construction

A signal, representing one set of electrically common Connect Points, is constructed using the Flow Associativity entity. It contains pointers to other associated Flow Associativity entities, the Connect Point entities participating in the signal (this is the Link), and the Join entities representing the geometry of the signal (either logical or physical). Also contained is a list of signal names which may be used to identify the signal, along with a set of pointers to Text Display Template entities which may be used to display the first signal name in a number of locations. Two characteristic flags determine the signal type (logical or physical), and the function type (fluid flow or electrical signal).

A signal, then, is represented by one Flow Associativity entity pointing to a set of electrically common Connect Points. This is the Link. The Join entities represent the physical display geometry of the signal. For a schematic (logical), a line without width is typically used. For a printed board (physical), a line with the Line Widening property is typically used. A number of signal names may be associated with the signal. Multiple displays of the first, or primary, name are possible.

The components participating in a signal are represented by the Network Subfigure Instance entity. Note that the Connect Point entities "belonging" to a component are instanced along with the subfigure. This is necessary to allow a subfigure to participate in a number of different signals, while retaining unique component/pin identification. Each component is usually identified by a reference designator. This is supplied by the Primary Reference Designator field of the Network Subfigure. Any alternate reference designators may be designated with the Reference Designator property, attached through the normal property pointer mechanism.

Information Display

Throughout the above discussion, references to the Text Display Template entity have been made. This entity allows text, embedded in an entity, to be displayed without the redundant specification of the text string. There are

two reasons for this feature. First, it eliminates a possible source of error by allowing the information to be specified in only one place. Second, it reduces the file size overhead. This entity exists in two forms, absolute and incremental. The absolute form provides an exact location for display of the information, as in the display of a reference designator. The incremental form provides an offset to be applied to the parent entity's location to provide the exact location for display of information like pin numbers. When a direct pointer for information display is provided, the base location is readily determined from the parent entity's location such as when displaying a pin number. In the case of property value display, the base location must be determined by "remembering" the location of the property entity's parent entity. This would occur when displaying the Part Name. Also in this case, the pointer to the Text Display Template entity is located in the additional pointers section of the property entity parameters:

Additional Considerations

The situation is exactly the same for both logical and physical representations. The only differences arise in the subfigure and Join entities used. In fact, a file may contain representations for both the schematic and the board. The Flow Associativity entity contains a type flag to indicate the connection type (logical or physical). In this case, one Flow Associativity would represent the logical connection and a second the physical connection. The two associativities would be related by the pointers provided in the Flow Associativity.

Figures

The following figures illustrate certain aspects of the above discussions. Figure B-1 illustrates the basic entity relationships. Figure B-2 and Table B-1 illustrate the usage of the Text Display Template. Figure B-3 illustrates an actual implementaiton. Figure B-4 shows an example of logical and physical signals and their relationships in the same file.

IGES ELECTRICAL ENTITY DESCRIPTIONS

The following entities (in entity number order) are the subset of IGES entities which have particular meanings when used for electrical product data.

100   Circular Arc Entity

The electrical use of this entity is in the geometric representation of component parts and their symbolic representations. In such usage, it is generally part of a subfigure. It may be used as a join entity. Its use may be defined by a Level Function property or DE Level field.

102   Composite Curve Entity

The electrical use of this entity is in the geometric representation of component parts and their symbolic representations. In such usuage, it is generally part of a subfigure. It may be used as a join entity. Its use may be defined by a Level Function property or DE Level field.

106   Copious Data Entity

Forms 11, 12, and 13 of this entity may be used to implement the electrical join (i.e., schematic or wiring diagram circuit connection lines). Any of these forms may point to a Line Widening property. Examples of this entity-property combination are circuit paths on a PC board or IC metalization, or as a bus in a schematic. Form 63, Simple Closed Area, may be used to define an auto-router restriction area or a PC (or IC) defined area having special attributes.

108   Plane Entity

Certain layers of PC design are designated as "ground", "power", or "heat sink", and as such are large conductive areas. These layers, as well as larger curved or rounded conductive areas on other layers, are best defined by the Plane entity. Note that the form number indicates whether the bounded region is positive or void (i.e., copper clad area or cutout).

110   Line Entity

The Line entity has several important uses in the electrical application. It may be used to construct component outlines, and to represent both logical and physical connections (as a join entity). As a physical join entity, the Line Widening property will most often be attached, giving the width of metalization to be etched on the board. As a logical join entity, the line will most typically be used without the Line Widening property.

116   Point Entity

The point entity is used to locate features that do not participate in electrical connectivity, for example, a mounting hole.

124   Transformation Matrix Entity

A Transformation Matrix may be used to rotate subfigures to other than normal (top up) positions. Generally, rotations are about the Z axis for PC and IC constructs, but may be about any axis for 3D cabling files.

125   Flash Entity

The Flash entity may be used to represent a repetitive artwork feature which is usually produced by photo-optical means.   Examples include PC pads, targets, clearances, and IC features.

132   Connect Point Entity

The Connect Point entity is used to represent a point of connection. A subfigure defining an electrical component typically uses the Connect Point entity to represent a pin of the logical or physical component or symbol. A Connect Point may also be used in a "stand-alone" mode, representing a via hole, for example.

212   General Note Entity

A General Note is used to display constant text. Design notes would require a
General Note, for example.

302   Associativity Definition Entity

When the originating system provides for a relationship not included among the
IGES predefined associativities, this entity is required. Possible uses are to
relate subfigures to entities in other databases (e.g., circuit analysis or text
requirements) or for back-annotation purposes.

312   Text Display Template Entity
      Form 0: absolute, Form 1: incremental

The Text Display Template may be used to display text which may be unique in
each instance of the defined entity (A pin number, for example).

320   Network Subfigure Definition Entity

For PC and Cable usage, a subfigure usually represents a component and its
required PC constructs. This entity is normally a library physical or logical
figure in the originating system.

402   Associativity Instance Entity

This entity relates other entities within a file to provide a "set" with a
common meaning. Those associativities which are predefined by IGES are
identified by IGES form numbers (e.g., form 18: Flow). The user defined
associativities are defined by an entity 302 and its form number.

406   Property Entity

The use of a property to indicate the meaning or purpose of a geometric entity
applies to electrical constructs as well as general graphics. A Connect Point

entity may point to the Drilled Hole property. A Plane entity or Simple Closed Area entity may point to the Region Restriction property. Any property, however, may point to the Text Display Template, with the text string specified in the property. In this case, the Text Display Template locates the text display. This entity is an open-ended list allowing for expansion to address future needs such as simulation, testing, inspection applications, and extensions into electrical/electronic systems hierarchical design.

412   Rectangular array Subfigure Instance Entity
414   Circular Array Subfigure Instance Entity

These entities may be used to instance multiple Network Subfigure Instances, but must not be used for instancing Connect Points.

420   Network Subfigure Instance Entity

This entity allows an electrical component to be instanced in a number of unique locations. Note that "owned" Connect Point entities must be instanced with this entity.
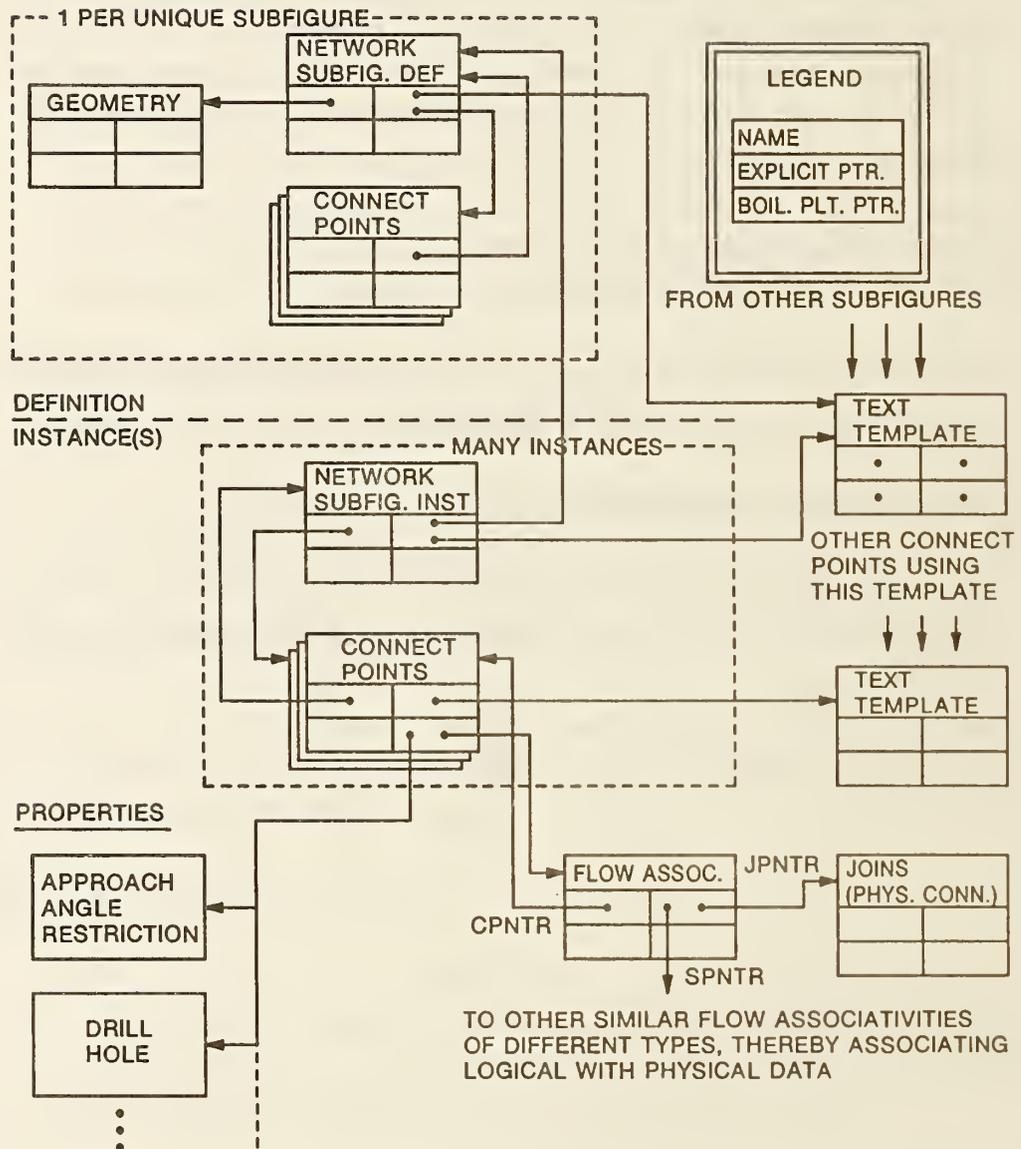
Figure B-1 General Pointer and Entity Model

TABLE 1. TEXT TEMPLATE VALUES FOR SAMPLE SCHEMATIC

TEMPLATE	TLEFT	TRIGHT	TTOP	TBOT	
WIDTH	.10	.10	.10	.10	
HEIGHT	.13	.1S	.13	.13	
SLANT ANG.	0.	0.	0.	0.	
ROTN. ANG.	0.	0.	0.	0.	
MIRR. FLG	0	0	0	0	
VRT/HORZ	0	0	0	0	
DE FORM NO.	21	21	21	21	
X (DX)	−.09	−.03	+.03	+.03	
Y (DY)	+.03	+.03	−.15	+.1	
Z (DZ)	0.	0.	0.	0.	
U1	P1–P8	P9–P16	-----	-----	PINS USING
U2	P1–P8	P9–P16	-----	-----	THE TEXT
U3	P1–P4	P9–P12	P13–P16	P5–P8	TEMPLATES
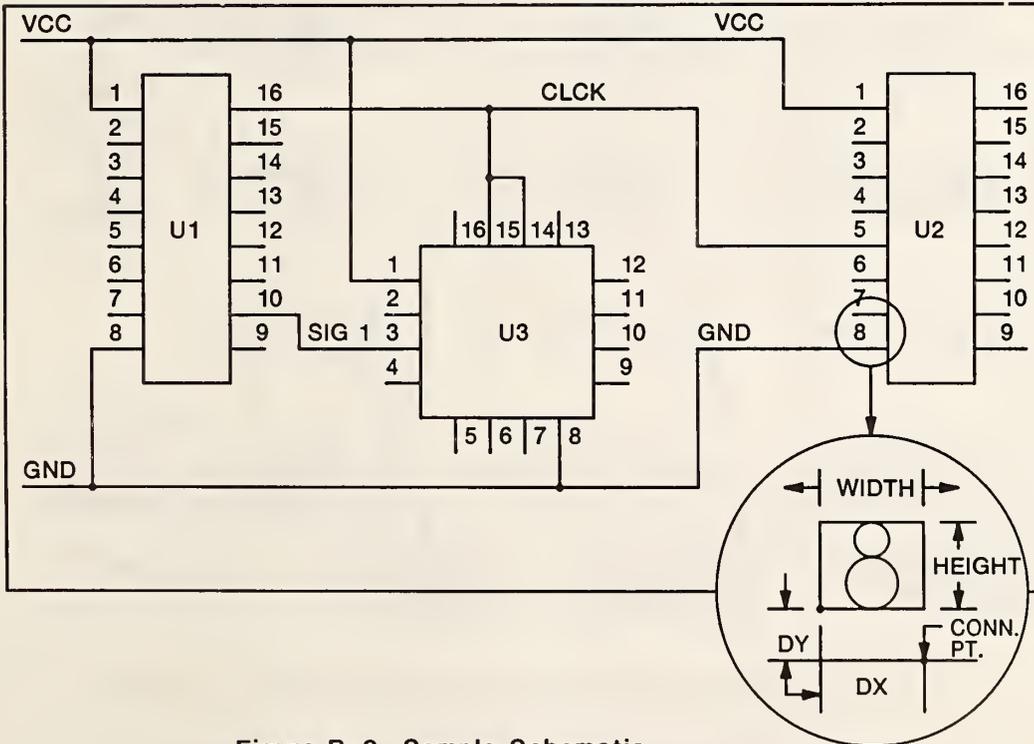
Figure B-2 Sample Schematic
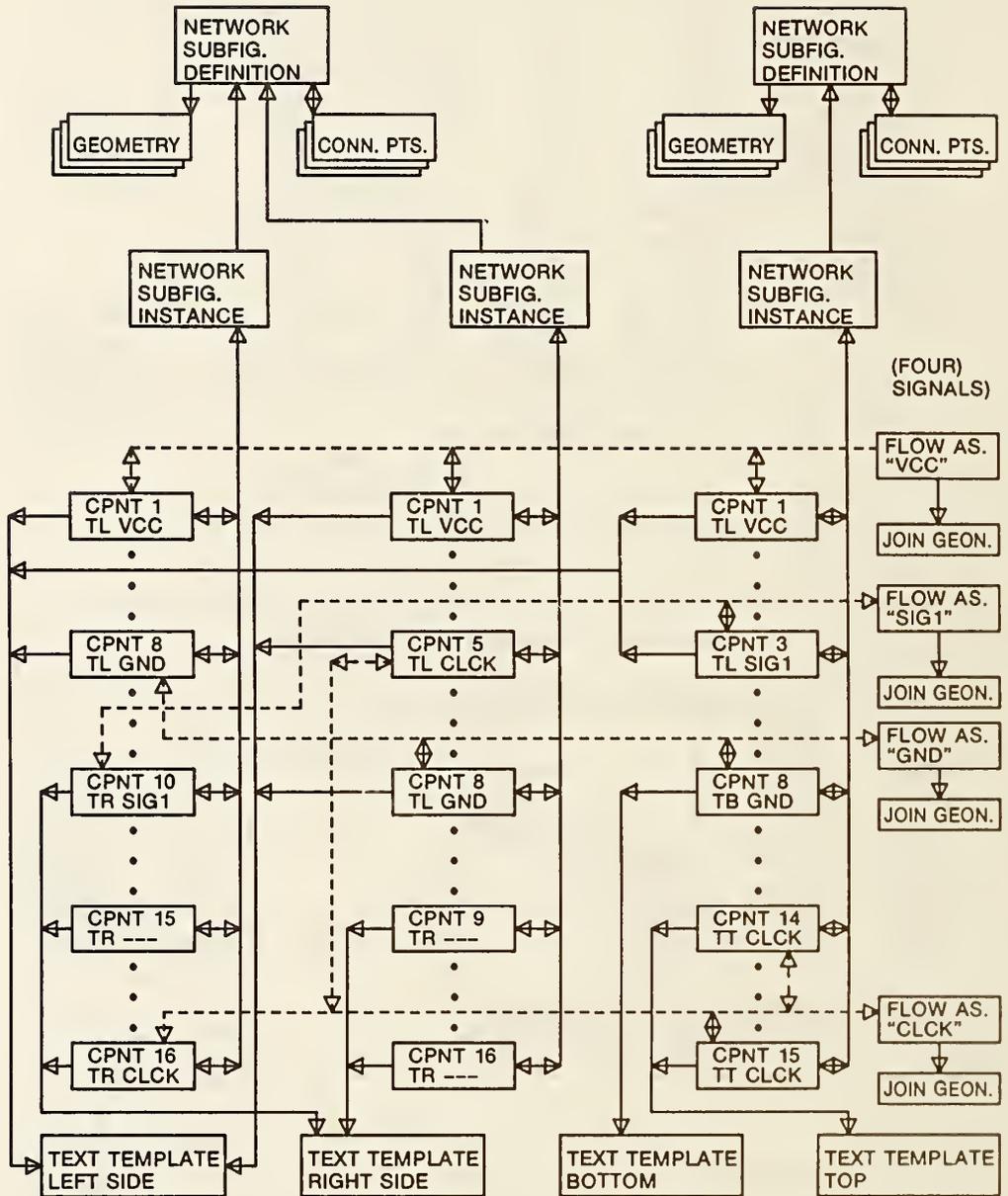
Figure B-3 Entity Relations Chart for Sample Schematic
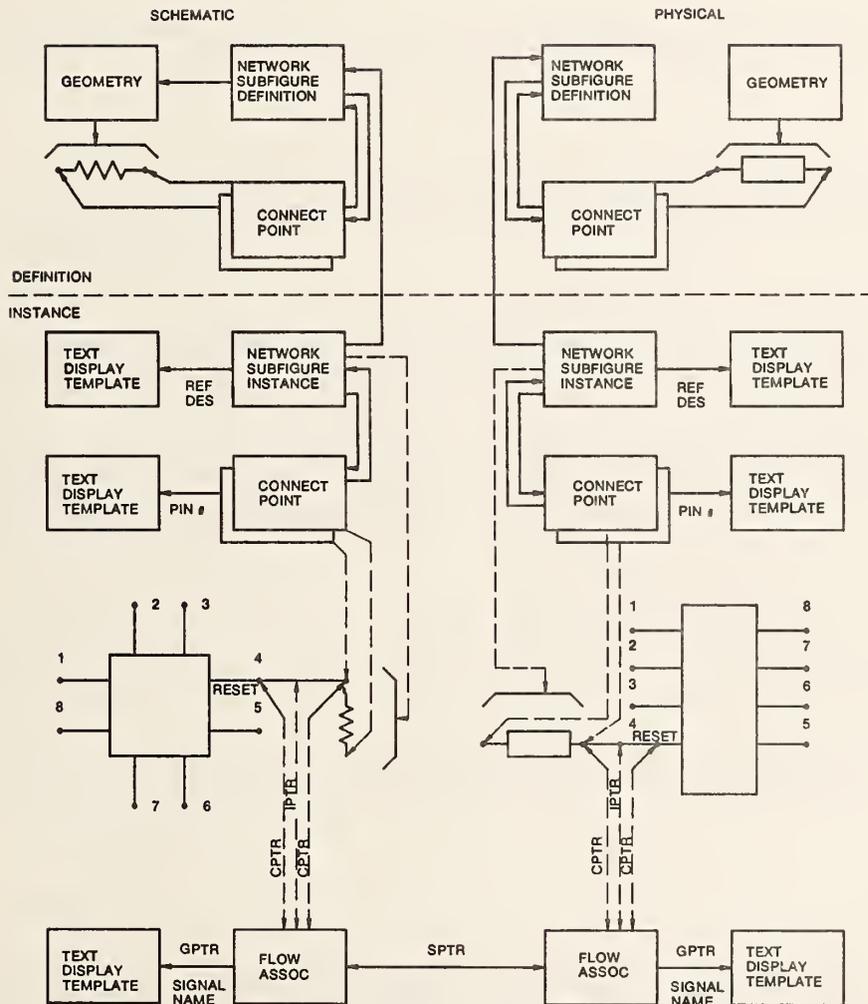
Figure B-4   Schematic/Physical Diagram for Sample Schematic

THIS PAGE LEFT BLANK

APPENDIX C
PLANT FLOW SHEET REPRESENTATION

Flowsheet Characteristics

Process flowsheets carry several important types of information which determine what data structures must be present. They must show:

1.   The identity of the process streams or lines flowing through the plant. Process and instrumentation diagrams typically show line designation, size, spec, and stream direction.

2.   The identity of "nozzles" on equipment to which line connections are made, and into and out of which process streams flow.

3.   The identity of equipment (i.e., valves, columns, pumps) which control, mix, pump and process the stream.

4.   The association of "nozzles" with the equipment to which they belong.

5.   The association of the beginning and end of each stream with unique nozzles.

6.   Indication of where a line is continued if it is too large for a single sheet and is continued elsewhere.

Support for Flowsheets

The graphics of a piping and instrument diagram will typically be composed of lines, arcs, and text. Components will consist of symbols (subfigure defintions) which are defined once by a set of lines and arcs and then referenced (as subfigure instances) as many times as needed.

A piping or instrument line is described by a path associativity entity which lists the from and to nozzles (points to their connect point entities) and then lists the in-line components. Properties can be attached to any entity and Pipe Nominal Size and Line Spec Name Properties would be attached to pipe line path entities.

Plant Flow Sheet Example

A simple flowsheet is shown in Figure C-1. The way the flowsheet would appear on paper is shown in Figure C-1a. The recommended assignment of connection points is shown in Figure C-1b.

An exploded view of the flowsheet showing the conceptual entities to be used to describe the flowsheet is given in Figure C-2a. These include two subfigures, "tank" and "valve", and four connect points belonging to them. Subfigures are defined at one time in the file for multiple use. When subfigures are used, each use is an "instance" of previously defined subfigure definitions. Two additional connect points are also shown to permit flow paths to begin and end properly. Five line entities appear to visually indicate how things are connected.

Logically the process designer has assigned these entities to pipelines or streams depending on what is being described. The logical flow paths for two process streams "A" and "B" are shown in Figure C-2b. Pipelines and streams are ordered sequences of equipment. When the line or stream reaches a point where two continuing paths are possible it must be told how to proceed. In some practical cases, both continuing paths may belong logically to the line or stream. A mechanism to indicate these continuations is provided by the flow path entity.

Describing how to encode a flowsheet involves:

(1) How to define the subfigure,
(2) How to instance the subfigure and the connect points it owns,
(3) How to define graphic lines which give pipelines and streams their appearance, and
(4) How to define the pipelines and streams themselves.

The entities which encode this flowsheet are given in Table C-1. The entity numbers given are the DE pointers to the example file shown in Table C-2.
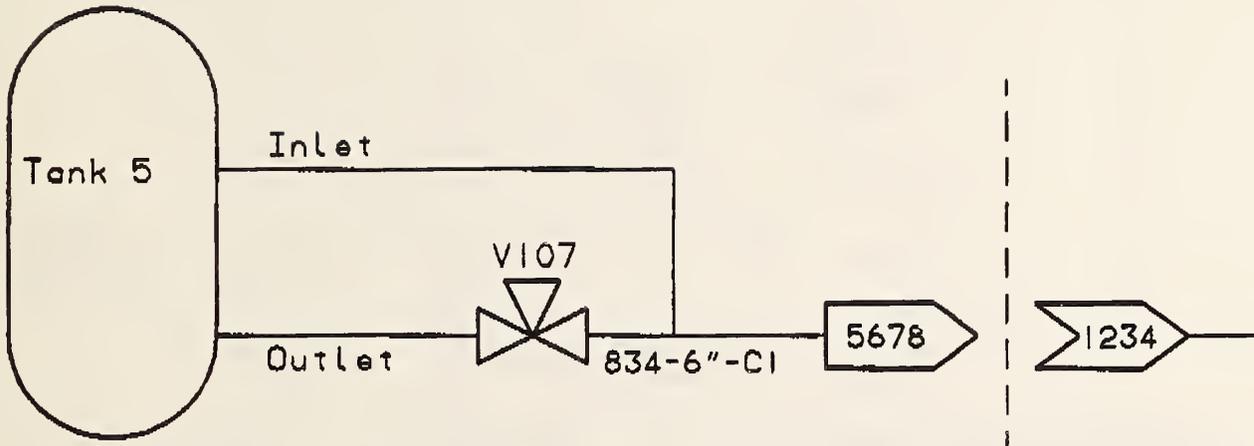
FIGURE C-1a FLOWSHEET APPEARANCE
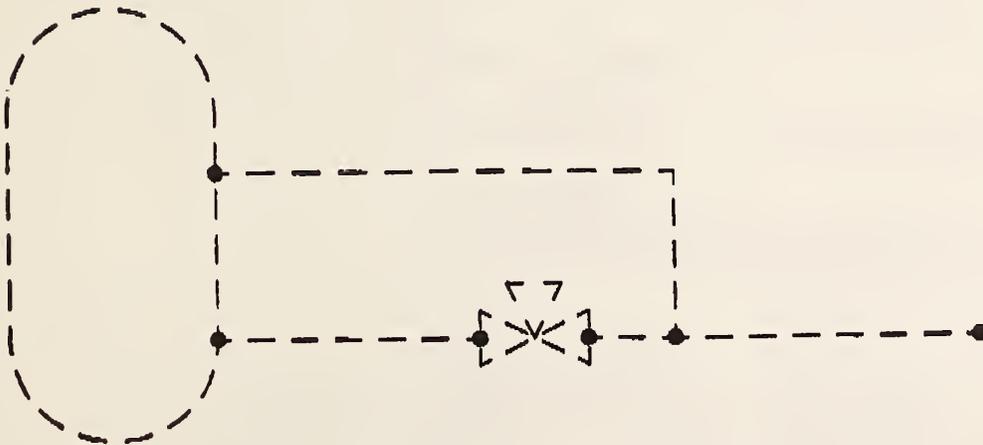
FIGURE C-1b REQUIRED CONNECTION POINTS
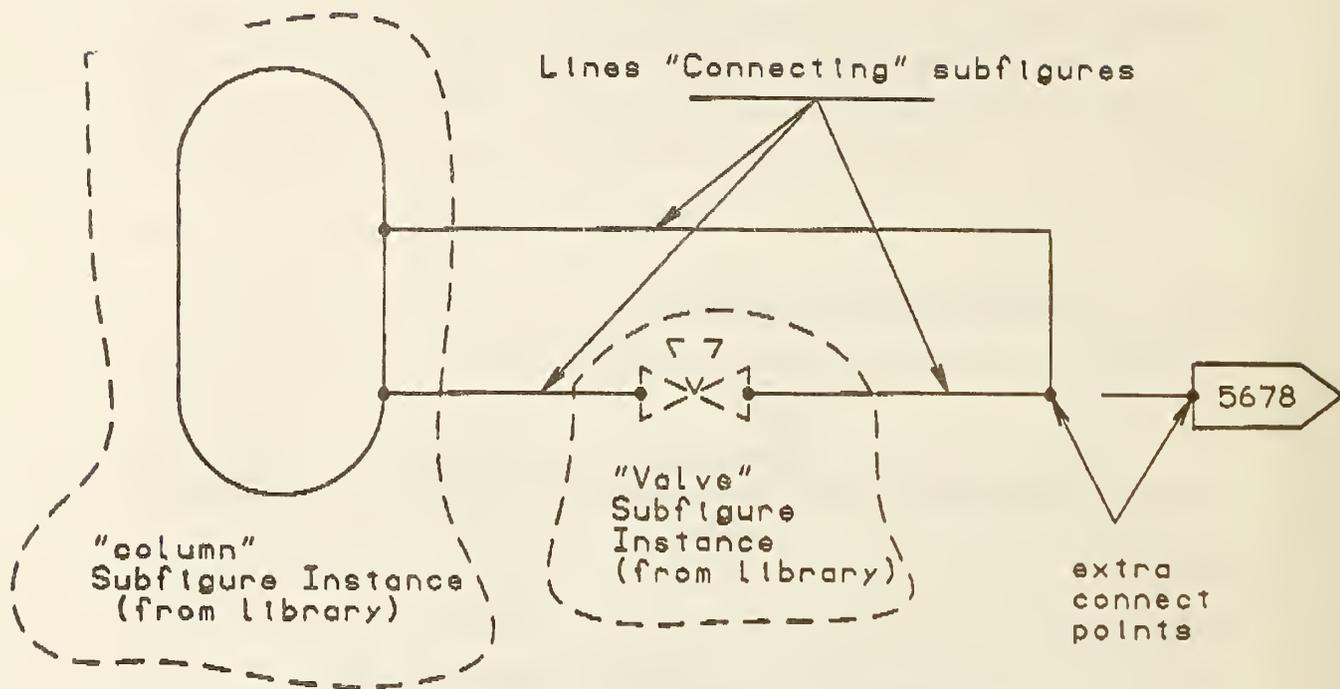
FIGURE C-2a CONCEPTUAL FLOWSHEET ENTITIES
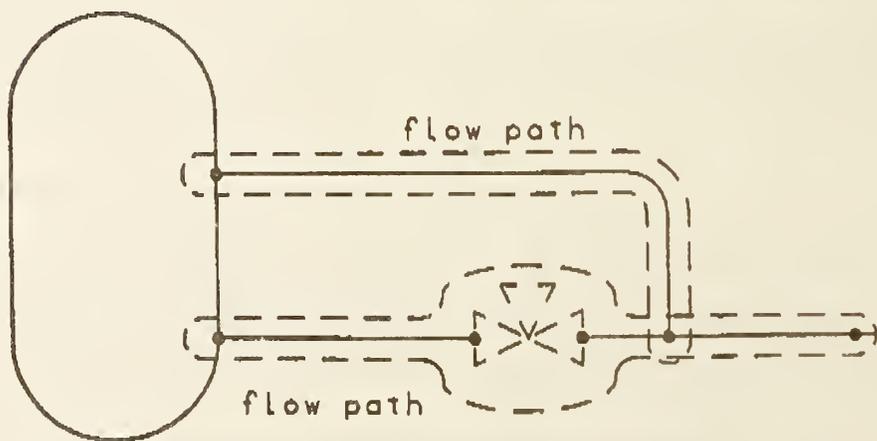
FIGURE C-2b LOGICAL FLOW PATH FOR STREAMS

TABLE C-1
ENTITIES IN PI & D TEST CASE

Entity #	Type	Remarks
1	Network subfigure def	Column
3	Network subfigure def	Valve
5	Line	Part of column
7	Circular arc	"
9	Line	"
11	Circular arc	"
13, 15	Connect point	Valve nozzle
17-29	Lines	Part of valve
31	General note	Miscellaneous drawing text
33	Text display template	Flow path name display
35	Network subfigure inst.	Instance of column
37	Network subfigure inst.	Instance of valve
39, 41	Connect point	Instance of tank nozzles
43, 45	Connect point	Instance of valve nozzles
47	Connect point	Branch point
49	Connect Point	Flow path end
51-59	Lines	Geometry of connecting path
61, 63, 65	Nominal size property	
67	Part number property	
69	External reference	Continuation of flow path
71, 73	Flow associativity	
75, 77	Text display template	Templates for subfigure name reference designator display
79	Text display template	Template for display of nozzle names
81-89	Line	Continuation symbol
91	Line spec property	A100
93	External reference file list	
95	External reference file index	

Network Subfigure Definitions

The definitions of the tank and valve subfigures are shown in Figures C-3a and C-3b. Each subfigure consists of the Network Subfigure Definition entity, geometry entities, and connect point entities. The tank is defined by entities 1,5,7,9 and 11. The Network Subfigure Definition entity #1 contains lists of the geometry entities (#5,7,9, and 11). This is merely a definition which is "instanced" when actually needed.

The valve has a similar form. The valve Network Subfigure Definition entity #3 points to geometric entities #17 through 29, and to Connect Point entities 13 and 15.

Network Subfigure Instances

When the tank and valve network subfigures are instanced, the subfigure's connect points will be duplicated and associated with the instance. The geometric entities will **not** be duplicated. Figure C-4a shows the subfigures as instanced in the flowsheet with instance entities #35 ("tank") and #37 ("valve"). Note that the subfigures may be scaled when they are instanced. In the "valve" example different scale factors are used in the x and y axes. Independent x and y axis scaling is often used by CAD vendors to "fit" their symbols to the drawing. Network subfigure instance #35 points to subfigure definition #1 and to the new duplicated connect points #39 and #41. Similarly, Network Subfigure Instance #37 points to Subfigure Definition #3 and to duplicate connect points #43 and 45.

Geometric Entities for Pipeline or Stream

Lines will appear on the drawing to indicate graphically how symbols are connected into pipelines and streams. These geometric entities alone do not logically connect symbols. This is done by Flow Path Associativity instance entities. The geometric entities are lines #51 through #59.
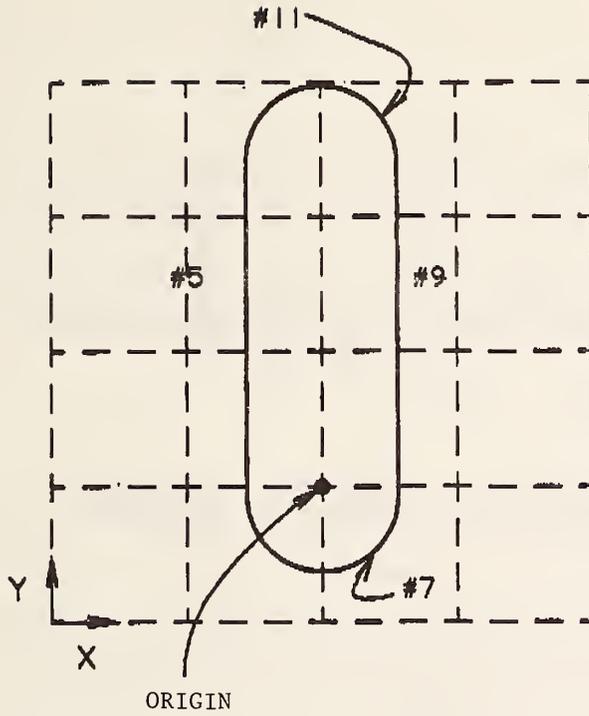
FIGURE C-3a ENTITY #1 TANK
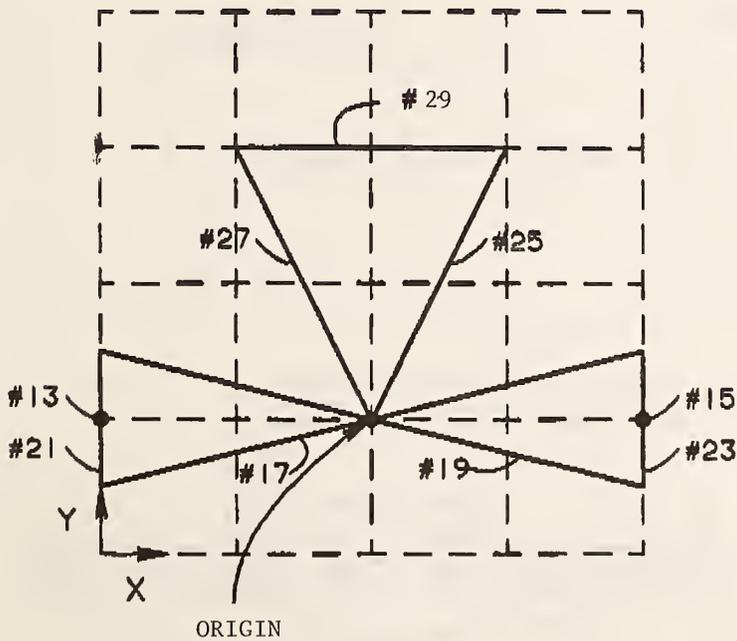
FIGURE C-3b ENTITY #3 VALVE
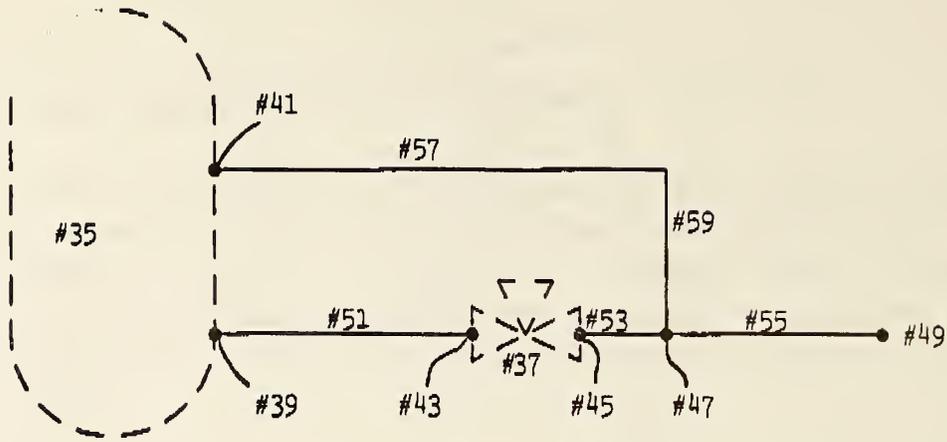
FIGURE C-4a GEOMETRY ENTITIES
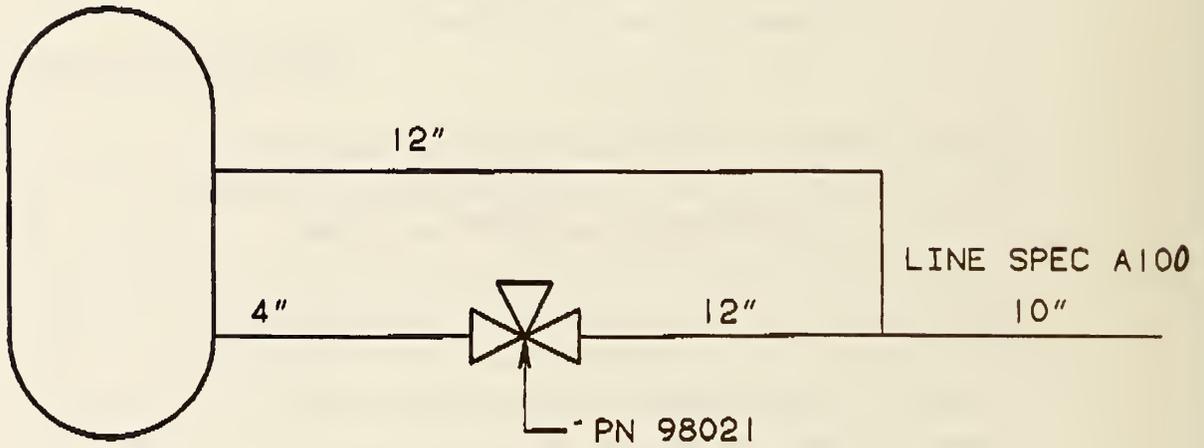
FIGURE C-4b ATTRIBUTES
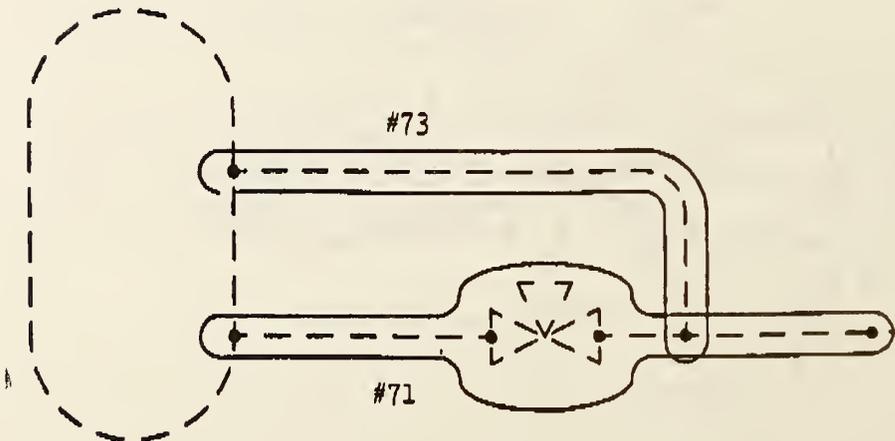
FIGURE C-4c FLOW PATHS

460

Attributes

Three specific properties (attributes) are provided for flowsheets. These are part number, nominal size, and (flow) line specification. Figure C-4b shows the properties used for the test case. These are implemented by three forms of the property entity. The entities in the test case are #61 through 67 and #91.

Flow Paths

Figure C-4c shows an assignment of flow paths. Two flow path entities are required, #71 and 73. Each Flow Path Associativity entity contains an ordered list of the connect points along the path and a separate list of the geometry entities representing the path. Where a branch occurs one path may point to one or more other paths. In the example flow path #71 lists path #73 as a continuation. These points are bi-directional, so flow path #73 contains a list of the entities (#71) which point to it. The flow path is also the entity to which the Line Specification property is attached. Note that a rather arbitrary flow path assignment was made by the process engineer. Other assignments are possible and could be encoded.

Text Templates

Text templates are used to control display of the text which is actually contained in the Subfigure Instance and connect point entities. There are two uses of Text Template entities in this example. The first type of use is to control display of the names (reference designators) of the tank and valve instances. Absolute templates #75 and #77 are used for this. The tank nozzles (connect points #39 and #41) both have their nozzle names (connect point function identifier) displayed with the same text size and location offset relative to the connect point location. Both connect point entities point to the same Incremental Text Template entity #79.

External References

Flow path #71 continues offpage, i.e., the path is logically continued on another flowsheet. Three pieces of information must be matched to make this continuation:

1. Line designation or number.
2. Drawing on which the path is continued, and
3. "Incoming" or "outgoing" connectors which match to the start or end of the path on the page.

This is handled by two mechanisms:

1. The External Reference entity which points (using a unique symbolic name) to the next connect point of the path upon which continuation is made, and to the file in which the drawing is contained. The connect point function identifier field is used to carry the drawing name and a text template is used to display it. File names are recommended to be constructed from the drawing name by the pre-processor.

2. The position of connect point within a flow path to the external flow path on the external flowsheet will determine whether it is logically "incoming" or "outcoming.

We will now implement these pointer structures for the example (see figure C-5.)

Flow path #71 ends at external reference entity #69. It points by a unique name 'I834-6"-C121' to a connect point in file 'XYZOIL.PROCESS.5678'. These external reference names must be unique. Typical CAD systems that do not maintain a globally unique name for a connect point may construct one using some set of conventions. The unique name 'I834-6"-C121' is typical of what can be constructed by a pre-processor using 'I' for incoming and '834-6"-C121', the line number, size, and spec. The file name 'XYZOIL.PROCESS.5678' is constructed from the connect point function identifier string '5678' using the root string file name entry in the External Reference Entity. The External Reference File List entity #93 carries this file name also, in the referencing file.

FILE XYZOIL P&ID 5678

FILE XYZOIL P&ID 1234
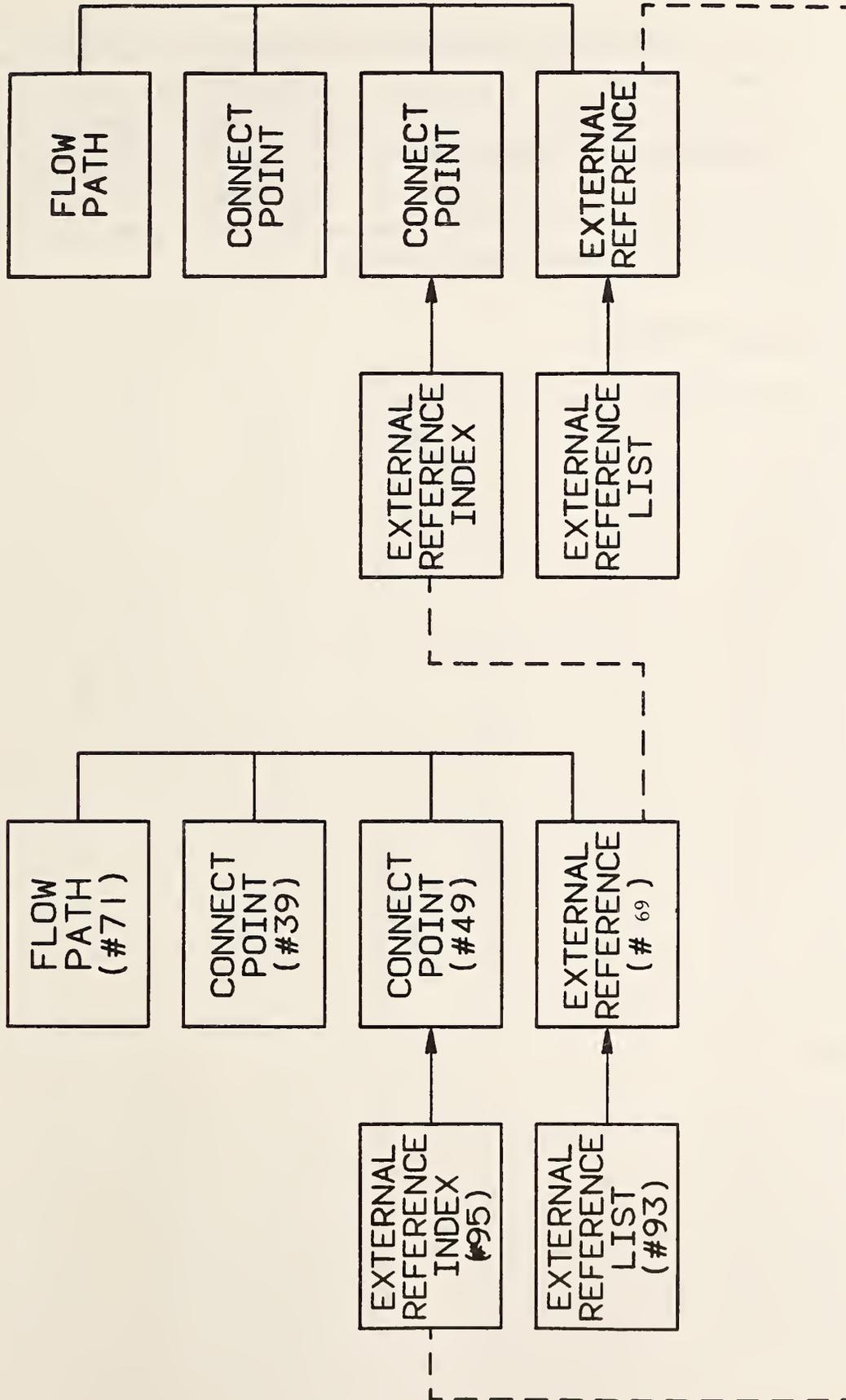
FIGURE C-5

In the external flowsheet, an External Reference File Index entity will list the symbolic name 'I834-6-C121' and reciprocal entities also exist to allow the external flowsheet to continue back into the referencing flowsheet. The flow path begins with an external reference entity '0834-6"C121' in file 'XYZOIL.PROCESS.1234'. It will also contain an external reference file list repeating this file name. There will also be an External Reference File Index entity #95 containing the symbolic name '0834-6-C121' a pointer to connect point #49.

Encoded Flowsheet

The encoded flowsheet is shown in Table C-2.

TABLE C-2   PLANT FLOWSHEET EXAMPLE

```
PIPING AND INSTRUMENTATION DIAGRAM IGES EXAMPLE                                      S      1
1H,,1H;,12HPI&D EXAMPLE,19HXYZOIL.PROCESS.1234,10HHYBRID 1.0,1H2,32,38,  G      1
6,308,15,12HPI&D EXAMPLE,0.100000E+01,1,4HINCH,3,.06,13H851115.233700,   G      2
0.0001,10000.,10HP.W.ROURKE,3HORG,4,0;                                   G      3
          320       1       1       0       1       0       0       00010200D      1
          320       0       1       1       4                                 D      2
          320       2       1       0       1       0       0       00010200D      3
          320       0       1       1       4                                 D      4
          110       3       1       0       1       0       0       00010000D      5
          110       0       1       1       0                                 D      6
          100       4       1       1       2       5       0       00010000D      7
          100       2       1       1       0                                 D      8
          110       5       1       1       1       5       0       00010000D      9
          110       2       1       1       0                                 D     10
          100       6       1       1       1       5       0       00010000D     11
          100       2       1       1       0                                 D     12
          132       7       1       1       2       5       0       00020400D     13
          132       2       1       1       0                                 D     14
          132       8       1       1       2       5       0       00020400D     15
          132       2       1       1       0                                 D     16
          110       9       1       1       2       5       0       00010000D     17
          110       2       1       1       0                                 D     18
          110      10       1       1       1       5       0       00010000D     19
          110       2       1       1       0                                 D     20
          110      11       1       1       1       5       0       00010000D     21
          110       2       1       1       0                                 D     22
          110      12       1       1       2       5       0       00010000D     23
          110       2       1       1       0                                 D     24
          110      13       1       1       2       5       0       00010000D     25
          110       2       1       1       0                                 D     26
          110      14       1       1       1       5       0       00010000D     27
          110       2       1       1       0                                 D     28
          110      15       1       1       2       5       0       00010000D     29
          110       2       1       1       0                                 D     30
          212      16       1       1       1       5       0       00000100D     31
          212       2       1       1       0                                 D     32
          312      18       1       1       1       5       0       00010201D     33
          312       2       1       1       0                                 D     34
          420      19       1       1       1       5       0       00000300D     35
          420       2       1       1       0                                 D     36
          420      20       1       1       2       5       0       00000300D     37
          420       2       1       1       0                                 D     38
          132      21       1       1       2       5       0       00020400D     39
          132       2       1       1       0                                 D     40
          132      22       1       1       1       5       0       00020400D     41
          132       2       1       1       0                                 D     42
          132      23       1       1       2       5       0       00020400D     43
          132       2       1       1       0                                 D     44
          132      24       1       1       1       5       0       00020400D     45
          132       2       1       1       0                                 D     46
          132      25       1       1       1       5       0       00020400D     47
          132       2       1       1       0                                 D     48
          132      26       1       1       1       5       0       00020400D     49
          132       2       1       1       0                                 D     50
```

110	27	1	1	2	5	0	00000000D	51
110	2	1	1	0			D	52
110	28	1	1	2	5	0	00000000D	53
110	2	1	1	0			D	54
110	29	1	1	1	5	0	00000000D	55
110	2	1	1	0			D	56
110	30	1	1	2	5	0	00000000D	57
110	2	1	1	0			D	58
110	31	1	1	1	5	0	00000000D	59
110	2	1	1	0			D	60
406	32	1	1	1	5	0	01000000D	61
406	2	1	1	13			D	62
406	33	1	1	2	5	0	01000000D	63
406	2	1	1	13			D	64
406	34	1	1	1	5	0	01000000D	65
406	2	1	1	13			D	66
406	35	1	1	1	5	0	01000000D	67
406	2	1	1	9			D	68
416	36	1	1	1	5	0	01000000D	69
416	2	1	1	2			D	70
402	37	1	1	2	5	0	01000300D	71
402	2	1	1	18			D	72
402	39	1	1	2	5	0	01000300D	73
402	2	1	1	18			D	74
312	40	1	1	1	5	0	00010201D	75
312	2	1	1	0			D	76
312	41	1	1	2	5	0	00010201D	77
312	2	1	1	0			D	78
312	42	1	1	1	5	0	00010201D	79
312	2	1	1	0			D	80
110	43	1	1	2	5	0	00000000D	81
110	2	1	1	0			D	82
110	44	1	1	2	5	0	00000000D	83
110	2	1	1	0			D	84
110	45	1	1	1	5	0	00000000D	85
110	2	1	1	0			D	86
110	46	1	1	2	5	0	00000000D	87
110	2	1	1	0			D	88
110	47	1	1	1	5	0	00000000D	89
110	2	1	1	0			D	90
406	48	1	1	1	5	0	01000000D	91
406	2	1	1	14			D	92
406	49	1	1	2	5	0	01000000D	93
406	2	1	1	12			D	94
402	50	1	1	1	5	0	01000000D	95
402	2	1	1	12			D	96

```
320,0,4HTANK,4,5,7,9,11,1;                                    1P      1
320,0,5HVALVE,7,17,19,21,23,25,27,28,1,,,2,13,15;             3P      2
110, -0.5,2.5,0.0, -0.5,0.0,0.0;                              5P      3
100, 0.0, 0.0,0.0, -0.5,0.0, 0.5,0.0;                         7P      4
110, 0.5,0.0,0.0,  0.5,2.5,0.0;                               9P      5
100, 0.0, 0.0,2.5,  0.5,2.5, -0.5,2.5;                       11P      6
132, -2.0,0.0,0.0,,1,2,1HA,,,,,,,3;                          13P      7
132,  2.0,0.0,0.0,,1,2,1HB,,,,,,,3;                          15P      8
110, 2.0,0.5,0.0,  -2.0,-0.5,0.0;                            17P      9
```

```
 110, -2.0,0.5,0.0,  -2.0,-0.5,0.0;                                    21P    11
 110, 2.0,0.5,0.0,    2.0,-0.5,0.0;                                    23P    12
 110, 0.0,0.0,0.0,    1.0,2.0,0.0;                                     25P    13
 110, 0.0,0.0,0.0,   -1.0,2.0,0.0;                                     27P    14
 110, 1.0,2.0,0.0,   -1.0,2.0,0.0;                                     29P    15
 212,1,12,0.200000E+01,0.164063E+00,1,1.570796,0.0,0,0,                31P    16
0.400000E+01,0.800000E+01,0.0,12HXYZ OIL PI&D;                         31P    17
 312,1.2,0.1,,,0.,0,0,4.0,8.0,0.0;                                     33P    18
 420,1,2.0,2.0,0.0,2.0,,,1,,,2,39,41;                                  35P    19
 420,3,5.5,3.0,0.0,,,,1,,,2,43,45,1,71,2,67,61;                        37P    20
 132,5.0,0.0,0.0,,1,2,6HOUTLET,79,,,,2,,35;                            39P    21
 132,3.0,3.0,0.0,,1,2,5HINLET,79,,,,1,,35;                             41P    22
 132,5.0,3.0,0.0,,1,2,,,,,,,,37;                                       43P    23
 132,6.0,3.0,0.0,,1,2,,,,,,,,37;                                       45P    24
 132,8.0,3.0,0.0,,1,2;                                                 47P    25
 132,10.0,3.0,0.0,1,2;                                                 49P    26
 110, 3.0,3.0,0.0, 5.0,3.0,0.0, 1,71, 1,61;                            51P    27
 110, 6.0,3.0,0.0, 8.0,3.0,0.0, 1,71, 1,63;                            53P    28
 110, 8.0,3.0,0.0, 10.0,3.0,0.0, 1,71, 1,63;                           55P    29
 110, 3.0,6.0,0.0, 8.0,6.0,0.0, 1,73, 1,65;                            57P    30
 110, 8.0,3.0,0.0, 8.0,6.0,0.0, 1,73, 1,65;                            59P    31
 406,2,12.,3HIPS;                                                      61P    32
 406,2,10.,3HIPS;                                                      63P    33
 406,2,4.,3HIPS;                                                       65P    34
 406,4,5H98021,,,;                                                     67P    35
 416,1,19HXYZOIL.PROCESS.5678,11HO834-6"C121;                          69P    36
 402,2,0,6,4,1,1,1,1,2,39,43,45,47,49,69,51,37,53,55,                  71P    37
 11H834-6"-C121,33,73;                                                 71P    38
 402,2,0,2,2,1,0,0,1,2,47,41,57,59,1,71,1,77;                          73P    39
 312,0.9,0.1,,,0.,0,0,1.5,4.5,0.;                                      75P    40
 312,0.4,0.1,,,0.,0,0,5.2,4.1,0.;                                      77P    41
 312,1.0,0.1,,,0.,0,0,0.1,-0.1,0.;                                     79P    42
 110, 10.0,2.75,0.0, 10.75,2.75,0.0;                                   81P    43
 110, 10.75,2.75,0.0, 11.0,3.0,0.0;                                    83P    44
 110, 11.0,3.0,0.0,  10.75,3.25,0.0;                                   85P    45
 110, 10.75,3.25,0.0, 10.0,3.25,0.0;                                   87P    46
 110, 10.0,3.25,0.0, 10.0,2.75,0.0;                                    89P    47
 406,1,4HA100;                                                         91P    48
 406,1,19HXYZOIL.PROCESS.5678;                                         93P    49
 402,1,11HI834-6"C121,49;                                              95P    50
S       1G       3D       96P       50                                  T      1
```

THIS PAGE LEFT BLANK

APPENDIX D
SPLINE REPRESENTATIONS

D1   INTRODUCTION

Section 3 of the Specification includes four different types of spline representations:

    a. A Parametric Piecewise Cubic Polynomial (for curves)

    b. A Rational B-Spline Curve

    c. A Grid of General Bicubic Patches (for surfaces).

    d. A Rational B-Spline Surface

Most of the spline types used in CAD/CAM systems can be mapped into these representations without change in shape. Spline types supported in Section 3 of this Specification include parametric cubics, piecewise linear, Wilson-Fowler, modified Wilson-Fowler, rational and non-rational B-splines, rational and non-rational cartesian product B-spline surfaces, and Coons patches. Spline types not supported include splines under tension and extended Coons patches.

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

D2   SPLINE FUNCTIONS

In Section 3.8 of this Specification, spline curves are represented by a number of cubic spline functions, one for each of the X,Y,Z coordinates. Each cubic spline function S(u) is defined by

a.   N: The number of segments,

b. $T(1),...,T(N+1)$: The endpoints and the breakpoints separating the cubic polynomial segments,

c. $A(i),B(i),C(i),D(i)$, $i=1,...,N$: The coefficients of the polynomials representing the spline in each of the N segments,

d. CTYPE: The spline type (1=linear, 2=quadratic, 3=cubic, 4=Wilson-Fowler, 5=Modified Wilson-Fowler, 6=B-spline) of the originating system.
See section 3.8

e. H: Degree of continuity. See section 3.8.3.

To evaluate the spline at a point "u", first determine the segment containing "u", i.e., the segment "i" such that $T(i) \leqslant u \leqslant T(i+1)$, then evaluate the cubic polynomial in that segment, i.e., compute

S(u) = A(i) + B(i)*(uu) + C(i)*(uu)**2 + D(i)*(uu)**3

where uu = u–T(i).

The polynomial is written in terms of the relative displacement uu (rather than u) so that the values of the spline at the breakpoints can be read directly out of the representation (i.e., S(T(i)) = A(i), i=1, ..., N, and S(T̄(N+1)) = TP0). Computations using the relative displacement also have less floating-point roundoff error.

This particular "piecewise polynomial" form is only one of many used to represent the spline segments in CAD/CAM systems. Other representations employed include:

a. End points E1,E2 and end slopes S1,S2: The spline can be evaluated using the "Hermite" basis (DEBO78, p. 59).

b. Values at four points: The spline value can be computed from the Lagrange or Newton interpolation formulae (DEBO78).

c. End points and "control" points: There are a number of schemes for computing splines from control points which will not be described here.

DeBoor (DEBO78) gives techniques for conversion between these representations.

Splines can also be represented as a linear combination of the B-spline basis functions. In CAD/CAM systems, B-splines have been used directly in curve fitting (e.g., the B-spline Bezier polygon (GORD74)) and indirectly in various spline calculations (e.g., computing a cubic spline interpolate). For every set of breakpoints $T(1),...,T(N+1)$ and degree of continuity H, a set of B-spline functions $B(1,u),B(2,u),...,B(n',u)$ can be constructed (DEBO78). Then, for any piecewise polynomial $S(u)$ with these breakpoints and continuity there is a set of B-spline coefficients $a(1),..., a(n')$ such that $S(u)$ can be represented as a linear combination of these B-splines

$$S(u) = a(1)*B(1,u) + a(2)*B(2,u) + ... + a(n')*B(n',u)$$

where $n' = (N-1)*(3-H)+4$.

B-splines can be computed from piecewise polynomials and vice versa (DEBO78, p. 116 and Subroutine BSPLPP).

Several other types of spline representations (e.g., cardinal bases) have been employed, but they are much less common and do not appear to present a problem for this Specification.

D3   **SPLINE CURVES**

The comments in this section pertain primarily to section 3.8.

Since curves in CAD/CAM problems are frequently many-valued, spline functions cannot represent such curves adequately. The most common approach to curve fitting is to parameterize the curves, i.e., to represent each curve as either two or three spline functions (one for each coordinate)

$$X(u) = Sx(u),$$
$$Y(u) = Sy(u), \text{ and}$$
$$Z(u) = Sz(u)$$

which sketch out the curve as the parameter u varies from $T(1)$ to $T(N+1)$.

All of the spline function representations of the previous section can be generalized to parametric curves and the algorithms for converting spline curves from one representation to the other follow easily from multiple applications of the corresponding function conversion algorithms.

Wilson-Fowler Curves: In the early sixties, the Wilson-Fowler spline (a special case of parametric cubics) was developed for curve fitting (IITR68). It is still used in many turnkey drafting systems.   In the Wilson-Fowler representation, each spline segment is defined in a separate coordinate system whose X-axis begins at one endpoint of the segment and passes through the other.   Each spline segment is then defined by a cubic spline function Swf(x) and the coordinates of the two endpoints.   These Wilson-Fowler splines can be converted to splines defined in Section 3.8 by rotating the parametric spline (u,Swf(u)) back into the current coordinate system; however, most types of splines defined in Section 3.8 cannot be converted to Wilson-Fowler splines.

D4   RATIONAL B-SPLINE CURVES

The comments in this section pertain primarily to section 3.16.

A rational B-spline curve is expressed parametrically in the form

$$G(t) = \frac{\sum_{i=0}^{K} W(i)P(i)b_i(t)}{\sum_{i=0}^{K} W(i)b_i(t)}$$

where the notation is interpreted as follows.

The $W(i)$ are the _weights_ (non-zero real numbers).

The $P(i)$ are the _control points_ (points in R^3).

The b_i are the <u>B-spline basis functions.</u> These are defined as soon as their <u>degree</u>, M, and underlying <u>knot sequence</u>, T, are specified.

This is done as follows:

Let N = K - M + 1. Then, the knot sequence consists of the non-decreasing set of real numbers; T(-M), ..., T(0), ..., T(N), ..., T(N+M)

The curve itself is parametrized for $V(0) \leqslant t \leqslant V(1)$ where
$T(0) \leqslant V(0) < V(1) \leqslant T(N)$.

The B-spline basis functions b_i are each non-negative piecewise polynomials of degree M. The function b_i is supported by the interval $[T(i-M), T(i+1)]$. Between any two adjacent knot values T(j), T(j+1) the function can be expressed as a single polynomial of degree M.

For any parameter value t between T(0) and T(N) the basis functions satisfy the identity

$$\sum_{i=0}^{K} b_i(t) = 1.$$

If the weights are all positive, the curve G(t) is contained within the convex hull of its control points.

There are a number of ways to precisely define the B-spline basis functions. A recursive approach proceeds as follows.

Let $N(t \mid t_{i-m}, ..., t_{i+1})$ denote the B-spline basis function of degree m supported by the interval $\left[t_{i-m}, t_{i+1} \right]_1$.

With this notation, the degree 0 functions are simply characteristic functions of a half-open interval.

$$N(t \mid a,b) = \begin{cases} 1 & \text{if } a \leqslant t < b \\ 0 & \text{otherwise} \end{cases}$$

The degree k functions are defined in terms of those of degree k-1.

$$N(t \mid s_0, \ldots, s_k) = \frac{(t-s_0) N(t \mid s_0, \ldots, s_{k-1})}{s_{k-1} - s_0} + \frac{(s_k-t) N(t \mid s_1, \ldots, s_k)}{s_k - s_1}$$

Since some of the denominators will be 0 in the case of multiple knots, the convention 0/0 = 0 is adopted in the above definition.

Rational Bezier curves (and surfaces) can be expressed exactly as rational B-spline curves (and surfaces). (BLOM82).

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

D5   SPLINE SURFACES

The spline surface defined in Section 3.9 is the analog of the spline curve defined in section 3.8, i.e., it is also pieced together out of other primitive functions. The surface is a grid of parametric bicubic patches defined by:

a.   M: The number of grid lines in u,

b.   TU(1),...,TU(M+1): The breakpoints in u (u values of grid lines),

c.   N: The number of grid lines in v,

d.   TV(1),...,TV(N+1): The breakpoints in v (v values of grid lines),

e.   Ax(i,j),Bx(i,j),...,Ay(i,j),...,Az(i,j),..., for i=1,...,M; j=1,...,N:

The M*N sets of 3*16 coefficients defining the bicubic polynomial for each of the three coordinates of the patch,

f.   CTYPE: The spline type. (1=linear, 2=quadratic, 3=cubic, 4=Wilson-Fowler, 5=Modified Wilson-Fowler, 6 = B-spline), and

g.   PTYPE: The patch type. (1=Cartesian product, 0=unspecified).

To evaluate the spline at a point "u,v", first determine the patch containing the point "u,v" in the parameter grid, i.e., the patch "i,j" such that $TU(i) \leqslant u \leqslant TU(i+1)$ and $TV(j) \leqslant v \leqslant TV(j+1)$, then evaluate the bicubic polynomial in that patch, i.e., compute

$$
\begin{aligned}
X(u,v) = {} & Ax(i,j) *vv**0 *uu**0 + Bx(i,j) *vv**0 *uu**1 \\
& + Cx(i,j) *vv**0 *uu**2 + Dx(i,j) *vv**0 *uu**3 \\[6pt]
& + Ex(i,j) *vv**1 *uu**0 + Fx(i,j) *vv**1 *uu**1 \\
& + Gx(i,j) *vv**1 *uu**2 + Hx(i,j) *vv**1 *uu**3 \\[6pt]
& + Kx(i,j) *vv**2 *uu**0 + Lx(i,j) *vv**2 *uu**1 \\
& + Mx(i,j) *vv**2 *uu**2 + Nx(i,j) *vv**2 *uu**3 \\[6pt]
& + Px(i,j) *vv**3 *uu**0 + Qx(i,j) *vv**3 *uu**1 \\
& + Rx(i,j) *vv**3 *uu**2 + Sx(i,j) *vv**3 *uu**3
\end{aligned}
$$

$Y(u,v) = Ay(i,j) \ldots$

$Z(u,v) = Az(i,j) \ldots$
where $uu = u - TU(i)$ and $vv = v - TV(j)$

The patches in the spline surface are equivalent to the bicubic surface patch (or the Coons patch, see (ROGE76, p. 170) for the conversion details). The parameters of the Coons patch are given as the corner points, corner slopes, and twist vectors (similar in spirit to the point/slope representation for curves).

However, because the Specification spline is more general than splines found in many CAD/CAM systems (e.g., the APT Wilson-Fowler spline), shape-preserving transformations out of the Specification spline format may not be possible.   Difficulties encountered include restrictions such as uniform

breakpoint spacing and smooth second derivatives. In these cases, the conversion must be accomplished by an interpolation or smoothing process.

D6   RATIONAL B-SPLINE SURFACES

The comments in this section pertain primarily to section 3.17.

A rational B-spline surface is expressed parametrically in the form

$$G(s,t) = \frac{\displaystyle\sum_{i=0}^{K1} \sum_{j=0}^{K2} W(i,j)P(i,j)b_i(s)b_j(t)}{\displaystyle\sum_{i=0}^{K1} \sum_{j=0}^{K2} W(i,j)b_i(s)b_j(t)}$$

where the notation is analogous to that used for rational B-spline curves.

The $W(i,j)$ are the weights (non-zero real numbers).

The $P(i,j)$ are the control points (points in R^3).

The b_i are the B-spline basis functions of degree M1 determined by the knot sequence $S(-M1),...,S(N1+M1)$. The b_j are the B-spline basis functions of degree M2 determined by the knot sequence $T(-M2),...,T(N2+M2)$. Here, $N1=K1-M1+1$ and $N2=K2-M2+1$.

The surface itself is parameterized for $U(0) \leqslant s \leqslant U(1)$ and for $V(0) \leqslant t \leqslant V(1)$ where $S(0) \leqslant U(0) < U(1) \leqslant S(N1)$ and $T(0) \leqslant V(0) < V(1) \leqslant T(N2)$. Rational Bezier surfaces (and curves) can be expressed exactly as rational B-spline surfaces (and curves). (BLOM82).

If the surface is periodic with respect to the first parametric variable, set PROP4 to 1; otherwise set PROP4 to 0. If the surface is periodic with respect to the second parametric variable, set PROP5 to 1; otherwise set PROP5 to 0.

Software to convert between parametric spline curves or surfaces and the corresponding rational B-spline curves or surfaces is available from the IGES office at the National Bureau of Standards. Materials provided include a magnetic tape of Pascal source code, a listing of the code, and accompanying documentation.

THIS PAGE LEFT BLANK

APPENDIX E
CONIC ARCS

Conic arcs as specified by the IGES standard are extremely sensitive to the data in two distinct ways:

(a)   Accuracy

It is numerically sensitive; small changes in the coefficients can cause large changes in the locations of the points satisfying the conic equation.

(b)   Stability

The determination of the conic type depends upon whether certain invariants are positive, zero or negative. Working in floating point arithmetic, a machine value of 0.0 is unlikely to be encountered. Furthermore, small changes in coefficient values can easily result in positive values when negative ones are intended and conversely.

It is assumed that data is put into a conic arc entity with the intent of preserving the geometric properties of the data (major and minor semi-axes, asymptotes, directrices, etc.) in addition to describing the points on the curve.

If the geometric properties are desired, the 104 entity should be used as described below.

This method primarily addresses the stability problem, though the accuracy of the conic should improve because the range of coefficient values will decrease. While the geometric properties are not explicitly defined in this representation, they can be obtained from it in a direct and arithmetically stable manner.

If both the sending and intended receiving system are known to use the A-F form of the 104 entity (Conic Arc) in their own databases the preprocessor may put the data into in the unchanged form. This minimizes the loss of information caused by truncation and roundoff errors as no changes are made to the data. The stability problem is presumably not of concern in this case.

Here is one suggested set of values:

(1) Ellipse

$A := AXISY^2$ $B := 0$

$C := AXISX^2$ $D := 0$

$E := 0$ $F := -A*C$

where AXISY and AXISX are the lengths of the major and minor semi-axes (not necessarily in order).

(2) Hyperbola

$A := - AXISY^2$ (or $+ AXISY^2$) $B := 0$

$C := + AXISX^2$ (or $- AXISX^2$, if $A < 0$) $D := 0$

$E := 0$ $F := -A*C.$

where AXISY and AXISX are the lengths of the major and minor semi-axes (not necessarily in order).

(3) Parabola

$A := 0$ (or 1) $B := 0$

$C := 1$ (or 0, if $A = 1$) $D := 4*DIST$
                                                      (or 0, if $A = 1$)

$E := 0$ (or $4*DIST$, if $A = 1$) $F := 0$

where DIST is the distance of the vertex from the focus.

Preprocessor Conic Handling

The conic arc must be put into standard form, parallel to the X and/or Y axis(axes) and centered about the origin. An 124 transformation matrix must be used to move the conic arc into its desired position in space. In this form the coefficients in the format that should be 0.0 will be exactly so. In particular, for the ellipse and hyperbola B, D, and E must be 0.0, and for the parabola B and F and either A and E or C and D must be 0.0.

Determination of the conic type from the equations becomes straight forward for the postprocessor.

For further mathematical details, see (THOM60).

APPENDIX F
COLOR SPACE VARIATIONS

It is often more convenient to operate in some color space other than RGB. The relationship between RGB and CMY is given by:

	where:	
R = 100.0 - C	R = red	C = cyan
G = 100.0 - M	G = green	M = magenta
B = 100.0 - Y	B = blue	Y = yellow

The HSL color space can be defined in terms of RGB in several ways with subtle variations. A typical approach is given by the following transformation:

$$H = \frac{1}{2\pi} \arctan\left[\frac{2R-G-B}{\sqrt{3}(G-B)}\right]$$

$$S = (R^2 + G^2 + B^2 - RG - RB - BG)^{\frac{1}{2}}$$

$$L = \frac{1}{3}(R + G + B)$$

where:

H = Hue

S = Saturation

L = Lightness

Variations on this transformation are given in (JOBL78) and SMIT78).

THIS PAGE LEFT BLANK

FILE CONVERSION UTILITIES

APPENDIX G

This Appendix gives details of utility programs to convert
a file from the ASCII Form to the Compressed ASCII Form
and back again. These utilities are written in Fortran 77
code, have been tested at several sites and are available
from the IGES office at The National Bureau of Standards.

APPENDIX G   FILE CONVERSION

```
C********************************************************************
C      THIS PROGRAM CONVERTS BETWEEN THE IGES REGULAR ASCII FORM AND
C         THE IGES COMPRESSED ASCII FORM
C      THE PROGRAM IS WRITTEN IN FORTRAN 77 SOURCE.
C
       PROGRAM IGES
C
C      PROGRAM ORIGANALLY WRITTEN BY J. M. SPAETH 7-24-84
C                                  GENERAL ELECTRIC CORP. RE. & DEV.
C      RE-WRITTEN BY LEE KLEIN 9-20-84
C                                  GENERAL DYNAMICS CAD/CAM POMONA DIV.
C      PURPOSE:
C           TO CONVERT NEW FORM OF IGES OUTPUT TO OLD FORM AND
C           OLD FORM TO NEW.
C
C      NOTE:   THIS CODE ASSUMES THAT THE ENTITY ORDER IS IDENTICAL
C              IN THE DE AND PD SECTIONS OF THE OLD FORM TO BE CONVERTED
C      INPUT:
C              YOU MUST GIVE THE NAME INCLUDING DIRECTORY IF DIFFERENT OF
C              THE FILE CONTAINING THE NEW FORM OF OUTPUT.  YOU MUST ALSO
C              GIVE THE NAME OF THE FILE TO CONTAIN THE CONVERTED OUTPUT.
C********************************************************************
C
       CHARACTER * 80 LINE1
       CHARACTER * 60 INFILE,OUTFILE
C
C      FORMATS.....
C
10     FORMAT($,' Enter input file name: ')
20     FORMAT($,' Enter output file name: ')
30     FORMAT(A60)
40     FORMAT(A80)
50     FORMAT(A)
60     FORMAT(/1X,'*** IGES FILE CONVERSION PROGRAM ***'/)
70     FORMAT(/1X,'Error in filename. Try again.')
C
C      PROMPT AND GET FILE NAMES...
C
       GOTO 110
100    WRITE(*,70)
110    WRITE(*,60)
       WRITE(*,10)
       READ(*,30)INFILE
       WRITE(*,20)
       READ(*,30)OUTFILE
C
C      READ THE FIRST LINE...
C
       OPEN(UNIT=10,FILE=INFILE,STATUS='OLD',ERR=100)
       READ(10,40)LINE1
       CLOSE(10)
C
C      CHECK TO SEE WHICH FORM THE INPUT IS IN...
C
       IF (LINE1(73:73).EQ.'C') THEN
```

```
        CALL OLDFORM(INFILE,OUTFILE)
       ELSE IF ((LINE1(73:73).EQ.'S').OR.
      +         (LINE1(73:73).EQ.'G').OR.
      +         (LINE1(73:73).EQ.'T').OR.
      +         (LINE1(73:73).EQ.'@')) THEN
        CALL NEWFORM(INFILE,OUTFILE)
       ELSE
        WRITE(*,50)' File contains ILLEGAL record format'
        CLOSE(10)
        STOP
       ENDIF
      WRITE(*,50)' '
      WRITE(*,50)' IGES conversion complete'
      WRITE(*,50)' '
      END
C
C
C      OLD FORM CONVERSION
C
       SUBROUTINE OLDFORM(INFILE,OUTFILE)
C
C      PROGRAM ORIGANALLY WRITTEN BY J. M. SPAETH 7-24-84
C                                  GENERAL ELECTRIC CORP. RE. & DEV.
C      RE-WRITTEN BY LEE KLEIN 9-20-84
C                                  GENERAL DYNAMICS CAD/CAM POMONA DIV.
C      PURPOSE:
C              TO CONVERT NEW FORM OF IGES OUTPUT TO OLD FORM...
C
C      VARIABLE DECLARATIONS...
C
       CHARACTER * 60  INFILE,OUTFILE
       CHARACTER * 8   BLNK,INARR(20),NEWARR(20)
       CHARACTER
```

```
       CALL OLDFORM(INFILE,OUTFILE)
      ELSE IF ((LINE1(73:73).EQ.'S').OR.
     +         (LINE1(73:73).EQ.'G').OR.
     +         (LINE1(73:73).EQ.'T').OR.
     +         (LINE1(73:73).EQ.'@')) THEN
       CALL NEWFORM(INFILE,OUTFILE)
      ELSE
       WRITE(*,50)' File contains ILLEGAL record format'
       CLOSE(10)
       STOP
      ENDIF
      WRITE(*,50)' '
      WRITE(*,50)' IGES conversion complete'
      WRITE(*,50)' '
      END

C
C      OLD FORM COMVERSION
C
       SUBROUTINE OLDFORM(INFILE,OUTFILE)
C
C      PROGRAM ORIGANALLY WRITTEN BY J. M. SPAETH 7-24-84
C                                   GENERAL ELECTRIC CORP. RE. & DEV.
C      RE-WRITTEN BY LEE KLEIN 9-20-84
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C      PURPOSE:
C               TO CONVERT NEW FORM OF IGES OUTPUT TO OLD FORM...
C
C      VARIABLE DECLARATIONS...
C
       CHARACTER * 60 INFILE,OUTFILE
       CHARACTER * 8   BLNK,INARR(20),NEWARR(20)
       CHARACTER * 80  INLINE
       CHARACTER * 160 OUTARR
       INTEGER         ICOUNT1,ICOUNT2,ICOUNT3,IA,IB,IC
       INTEGER         IJ,IK,IL,IT
       DATA IPEE/1HP/
C
C      FORMATS...
C
1      FORMAT(A80)
2      FORMAT(A64,1I8,1A1,1I7)
3      FORMAT(10A8)
5      FORMAT(9A8,A,I7)
6      FORMAT(A8,I8,A64)
C
C      INITIALIZE OUTARR TO BLANKS...
C
       OUTARR(1:160)=' '
C
C      OPEN INPUT AND TEMP FILES...
C
       OPEN(UNIT=1,FILE='FILE1.TMP',STATUS='NEW',CARRIAGECONTROL='LIST')
       OPEN(UNIT=2,FILE='FILE2.TMP',STATUS='NEW',CARRIAGECONTROL='LIST')
       OPEN(UNIT=3,FILE='FILE3.TMP',STATUS='NEW',CARRIAGECONTROL='LIST')
```

```
      OPEN(UNIT=4,FILE='FILE4.TMP',STATUS='NEW',CARRIAGECONTROL='LIST')
      OPEN(UNIT=9,FILE=OUTFILE,STATUS='NEW',CARRIAGECONTROL='LIST')
      OPEN(UNIT=10,FILE=INFILE,STATUS='OLD')
C
C     INITIALIZE COUNTERS...
C
      ICOUNT1 = -1
      ICOUNT2 = 1
      ICOUNT3 = 0
C
C     READ THE FILE AND SEPARATE INTO PARTS...
C
100   READ(10,1,END=600)INLINE
      IF ((INLINE(73:73).EQ.'S').OR.(INLINE(73:73).EQ.'G')) GOTO 101
      IF (INLINE(73:73).EQ.'T') GOTO 102
      IF ((INLINE(1:1).EQ.'@').OR.(INLINE(1:1).EQ.'D')) GOTO 400
C
C     IF IT IS AN C THAN DELETE IT OFF BE READING THE NEXT LINE
C
      IF (INLINE(73:73).EQ.'C') GOTO 100
C
C     PUT THE PARAMETER DATA INTO FILE4.TMP...
C
      WRITE(4,2) (INLINE(1:64),ICOUNT1,IPEE,ICOUNT2)
      ICOUNT2 = ICOUNT2 + 1
      GOTO 100
C
C     WRITE HEADER LINES INTO FILE1.TMP...
C
101   WRITE(1,1)INLINE
      GOTO 100
C
C     WRITE TERMINATION LINES INTO FILE2.TMP...
C
102   WRITE(2,1)INLINE
      GOTO 100
C
C     WRITE DIRECTORY ENTRY LINES INTO FILE3.TMP...
C
103   WRITE(3,6) (OUTARR(1:8),ICOUNT2,OUTARR(17:80))
      WRITE(3,1) OUTARR(81:160)
      ICOUNT1 = ICOUNT1 + 2
      GOTO 100
C
C     REWRITE TO DE LINES IN THE NEW FORM...
C
C     GO THRU INLINE ONE CHAR. AT A TIME LOOKING FOR THE
C     DELIMETER (@, ,_)...
C
400   DO IL=1,80
          IF (INLINE(IL:IL).EQ.';') GOTO 103
          IF (INLINE(IL:IL).EQ.' ') GOTO 100
          IF (INLINE(IL:IL).NE.'@') GOTO 501
C
C     DETERMINE IF THE FIELD IS ONE OR TWO CHARACTERS...
```

```
C
            IF (INLINE(IL+2:IL+2).EQ.' ') THEN
             IC=ICHAR(INLINE(IL+1:IL+1))-48
             IL=IL+2
            ELSE
             IA=ICHAR(INLINE(IL+1:IL+1))-48
             IB=ICHAR(INLINE(IL+2:IL+2))-48
             IC=10*IA+IB
             IL=IL+3
            ENDIF
C
C     AT THIS POINT IC IS THE NUMBER OF THE RECORD FIELD BEING
C     PROCESSED, AND INLINE(IL)=USCORE
C
      IT=0
      IK=0
      IJ=(IC-1)*8+1
C
C     RESET THE FIELD TO BE CHANGED TO ALL BLANKS IN ORDER TO CREATE
C     A COMPLETELY NEW FILED...
C
      OUTARR(IJ:IJ+7)='
C
C     WE WILL NOW CONTINUE THRU THE LINE PICKING OFF THE CHAR.
C     OF THE RECORD FIELD ONE AT A TIME UNTIL A DELIMETER IS HIT...
C
450   IK=IK+1
       IF (INLINE(IL+IK:IL+IK).EQ.'@') GOTO 500
       IF (INLINE(IL+IK:IL+IK).EQ.' ') THEN
        IF (INLINE(IL+IK+1:IL+IK+1).EQ.' ') GOTO 500
       ENDIF
       IF (INLINE(IL+IK:IL+IK).EQ.';') GOTO 500
       IT=IT+1
      IJ=(IC-1)*8+IT
      OUTARR(IJ:IJ)=INLINE(IL+IK:IL+IK)
      IF (IC.EQ.1) THEN
       OUTARR(IJ+80:IJ+80)=INLINE(IL+IK:IL+IK)
      ENDIF
      GOTO 450
500   IL=IL+IT
501   END DO
      GOTO 100
C
C     REWIND ALL FILES BEFORE WE WRITE THEM TO OUTPUT...
C
600   REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
C
C     WRITE START AND GLOBAL RECORDS TO OUTPUT FILE...
C
601   READ(1,1,END=602) INLINE
      WRITE(9,1) INLINE
```

```
        GOTO 601
C
C       WRITE THE DE RECORDS TO OUTPUT FILE. THEY NOW BECOME RE-FORMATTED
C
602     READ(3,3,END=603) (INARR(I),I=1,10)
        READ(3,3) (INARR(I),I=11,20)
        DO IP=1,20
         IF ((IP.NE.9).AND.(IP.NE.18)) GOTO 200
         NEWARR(IP)=INARR(IP)
         GOTO 201
C
C       CHANGE THOSE FIELDS THAT MUST BE RIGHT JUSTIFIED
C
200      NEWARR(IP)=BLNK(INARR(IP))
201     END DO
        ICOUNT3=ICOUNT3+1
        WRITE(9,5) ((NEWARR(J),J=1,9),'D',ICOUNT3)
        ICOUNT3=ICOUNT3+1
        WRITE(9,5) ((NEWARR(J),J=11,19),'D',ICOUNT3)
        GOTO 602
C
C       WRITE THE PD LINES TO THE OUTPUT FILE...
C
603     READ(4,1,END=604) INLINE
        WRITE(9,1) INLINE
        GOTO 603
C
C       WRITE THE TERMINATE LINES TO THE OUTPUT FILE...
C
604     READ(2,1,END=605) INLINE
        WRITE(9,1) INLINE
        GOTO 604
C
C       NOW CLOSE THE FILES AND DELETE THE TEMP ONES...
C
605     CLOSE(UNIT=1,STATUS='DELETE')
        CLOSE(UNIT=2,STATUS='DELETE')
        CLOSE(UNIT=3,STATUS='DELETE')
        CLOSE(UNIT=4,STATUS='DELETE')
        CLOSE(9)
        CLOSE(10)
        RETURN
        END

C       START FUNCTION BLNK HERE
C
C       WRITTEN BY P. R. KENNICOTT 9-29-83.
C       RE-WRITTEN BY LEE KLEIN 9-2-84.
C
C       PURPOSE:
C           TO REMOVE BLANKS FROM END OF A CHARACTER STRING (RIGHT JUSTIFY)
C
C       INPUT:
C           BUF STRING WITH TRAILING BLANKS
C
```

```fortran
C      OUTPUT:
C         BLNK STRING WITH TRAILING BLANKS REMOVED
C
C      METHOD:
C         FIND FIRST BLANK, THEN TRANSLATE OUTPUT STRING
C
C      RESTRICTIONS:
C                  1. BUF <= 512 CHARACTERS.
C                  2. LENGTHS OF BUF & BLNK MUST BE =.
C                  3. FIRST CHARACTER MUST NOT BE BLANK OR NO CONVERSION.
C
C
C      VARIABLE DECLARATIONS...
C
       CHARACTER*(*) FUNCTION BLNK(BUF)
       CHARACTER*(*) BUF
       INTEGER I
       CHARACTER*512 IBUF
C
C      SET UP COUNTERS...
C
       N=INDEX(BUF(1:),' ')-1
       M=LEN(BUF)
C
C      CHECK FOR SIZE TOO BIG...
C
       IF (M.GT.512) STOP 'Buffer too big at function BLNK'
C
C      CHECK FOR FIRST CHAR A BLANK...
C
       IF (BUF(1:1).EQ.' '.OR.BUF(M:M).NE.' ') THEN
        BLNK=BUF
        RETURN
       ENDIF
C
C      OK PROCESS STRING...
C
       DO 10 I=1,N
10      IBUF(M-I+1:M-I+1)=BUF(N-I+1:N-I+1)
       IBUF(1:M-N)=' '
       BLNK=IBUF
       RETURN
       END

C      START NEW SUBROUTINE HERE
C
C      PROGRAM ORIGANALLY WRITTEN BY J. M. SPAETH 7-24-84
C                                   GENERAL ELECTRIC CORP. RE. & DEV.
C      RE-WRITTEN BY LEE KLEIN 9-20-84
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C      PURPOSE:
C               TO CONVERT OLD FORM OF IGES OUTPUT TO NEW FORM...
C
       SUBROUTINE NEWFORM(INFILE,OUTFILE)
C
```

```
C       VARIABLE DECLARATIONS...
C
        DIMENSION INLINE(80)
        CHARACTER * 60 INFILE,OUTFILE
        DATA IPEE/1HP/
        DATA IDEE/1HD/
        DATA ITEE/1HT/
C
C       OPEN THE INPUT AND TEMP FILES...
C
        OPEN(UNIT=10,FILE=INFILE,STATUS='OLD')
        OPEN(UNIT=1,FILE='TEST.TMP',STATUS='NEW',CARRIAGECONTROL='LIST')
        OPEN(UNIT=2,FILE='FILE2.TMP',STATUS='NEW')
        OPEN(UNIT=3,FILE='FILE3.TMP',STATUS='NEW')
        OPEN(UNIT=7,FILE='FILE5.TMP',STATUS='NEW')
C
C       WRITE THE HEADER WITH A "C" TO SHOW COMPRESSED ASCII FORM...
C
        WRITE(1,99)'C'
99      FORMAT(72X,A)
C
C       SEPERATE THE PD AND DE RECORDS, WHILE WRITING G,S, &T LINES
C       TO THE OUTPUT FILE...
C
102     FORMAT(80A1)
200     READ(10,102,END=204) (INLINE(I),I=1,80)
        IF (INLINE(73).EQ.IDEE) GOTO 201
        IF (INLINE(73).EQ.IPEE) GOTO 202
        IF (INLINE(73).EQ.ITEE) GOTO 203
C
C       WRITE HEADER LINES INTO A TEST.TMP...
C
        WRITE (1,102) (INLINE(I),I=1,80)
        GOTO 200
C
C       WRITE DIRECTORY LINES INTO FILE3.TMP...
C
201     WRITE (3,102) (INLINE(I),I=1,80)
        GOTO 200
C
C       WRITE PARAMETER DATA INOT FILE2.TMP...
C
202     WRITE (2,102) (INLINE(J),J=1,80)
        GOTO 200

203     WRITE (7,102) (INLINE(J),J=1,80)
        GOTO 200

204     CALL XPD
        REWIND 7

205     READ(7,102,END=206) (INLINE(I),I=1,80)
        WRITE (1,102) (INLINE(J),J=1,80)
        GOTO 205
```

```
206     CALL COMPRESS(OUTFILE)
C
C       CLOSE FILES AND DELETE TEMP ONES...
C
        CLOSE(UNIT=1,STATUS='DELETE')
        CLOSE(UNIT=2,STATUS='DELETE')
        CLOSE(UNIT=3,STATUS='DELETE')
        CLOSE(UNIT=4)
        CLOSE(UNIT=7,STATUS='DELETE')
        CLOSE(UNIT=10)
        RETURN
        END

        SUBROUTINE XPD
C
C       PURPOSE:
C               TO TRANSFER ALL PD & DE RECORDS FORM TEMPORY FILES TO
C               OUTPUT FILE IN MERGED FORM...
C
C       VARIABLE DECLARATIONS...
C
        CHARACTER * 1    SCOLN/';'/
        CHARACTER * 8    BLNK,LSTDAT(20),NEWDAT(20)
        CHARACTER * 80   PDLINE,DELIN1,DELIN2
        CHARACTER * 160  NEWDE
        INTEGER          LFLD(20),FLDNUM,FLDBEG,FLDEND
        INTEGER          CHRPTR,NEWPTR,LSTPTR
1       FORMAT(A80)
C
C       REWIND THE FILES...
C
        REWIND 3
        REWIND 2
C
C       INITIALIZE LAST DATA SO AS NOT OT EQUAL NEXT DATA...
C
        LSTPTR = -1
        DO FLDNUM=1,20
         LSTDAT(FLDNUM) = 'XXXXXXXX'
        END DO
C
C       START LOOP.. GET NEXT PD RECORD.
C
10      READ (2,1,END=120)PDLINE
C
C       CLEAR OUT THE DE RECORD BUFFER...
C
        NEWDE(1:160) = ' '
C
C       IF BACK-POINTER TO DE RECORD IS DIFFERENT FORM LAST PD
C       GET NEW DE RECORD
C
        READ (PDLINE(65:73),20)NEWPTR
20      FORMAT(I8)
        IF (NEWPTR.EQ.LSTPTR) GOTO 115
```

```
      READ(3,1,END=120) DELIN1
      READ(3,1) DELIN2
      DELIN1(73:73)=' '
      DELIN2(73:73)=' '
C
C     GET THE DATA FROM EACH FIELD OF THE DE RECORD SET...
C
      FLDBEG = -7
      FLDEND = 1
      DO FLDNUM=1,10
       FLDBEG=FLDBEG + 8
       FLDEND=FLDEND + 8
       READ(DELIN1(FLDBEG:FLDEND),30) NEWDAT(FLDNUM)
       READ(DELIN2(FLDBEG:FLDEND),30) NEWDAT(FLDNUM+10)
30     FORMAT(A8)
      END DO
C
C     FIELD 18 MUST BE RIGHT JUSTIFIED...
C
      NEWDAT(18)=BLNK(NEWDAT(18))
C
C     DETERMINE THE LENGTH OF THE DATA WITHIN EACH FIELD
C
      DO FLDNUM=1,20
       DO I=1,8
        IF (NEWDAT(FLDNUM)(I:I).NE.' ') THEN
         LFLD(FLDNUM)= 9 - I
         GOTO 40
        ENDIF
       END DO
40    END DO
C
C     WRITE THE DE SEQUENCE NUMBER AT THE BEGINNING OF THE OUTPUT DE RECORD
C
      ENCODE(LFLD(10)+1,60,NEWDE(1:LFLD(10)+1)) NEWDAT(10)(9-LFLD(10):8)
60    FORMAT ('D',A)
      CHRPTR = LFLD(10) + 2
C
C     SEARCH NEW DE RECORD SET FOR CHANGED DATA; WHEN FOUNT WRITE CHANGED
C     DATA TO OUTPUT DE RECORD...
C
      DO FLDNUM=1,20
C
C     SKIP FIELDS THAT NO LONGER NEED PROCESSING...
C
       IF ((FLDNUM.EQ. 2).OR.
     +     (FLDNUM.EQ.10).OR.
     +     (FLDNUM.EQ.11).OR.
     +     (FLDNUM.EQ.20)) GOTO 100

       IF (NEWDAT(FLDNUM).NE.LSTDAT(FLDNUM)) THEN
        IF (FLDNUM.GT.9) THEN
         ENCODE(4,70,NEWDE(CHRPTR:CHRPTR+3)) FLDNUM
70       FORMAT('@',I2,'_')
         CHRPTR=CHRPTR+4
```

```
          ELSE
            ENCODE(3,80,NEWDE(CHRPTR:CHRPTR+2)) FLDNUM
80          FORMAT('@',I1,'_')
            CHRPTR=CHRPTR+3
          ENDIF
          IF (LFLD(FLDNUM).NE.0) THEN
           IF (FLDNUM.NE.9) THEN
            READ(NEWDAT(FLDNUM)(9-LFLD(FLDNUM):8),90)
     +           NEWDE(CHRPTR:CHRPTR-1+LFLD(FLDNUM))
90          FORMAT(A)
            CHRPTR=CHRPTR+LFLD(FLDNUM)
           ELSE
C
C      FIELD 9 IS A SPECIAL CASE...
C
            READ(NEWDAT(9)(1:8),90) NEWDE(CHRPTR:CHRPTR+7)
            CHRPTR=CHRPTR+8
           ENDIF
          ENDIF
         ENDIF
C
C      STORE DATA FORM CURRENT DE RECORD SET TO COMPARE WITH NEXT SET
C
          LSTDAT(FLDNUM)=NEWDAT(FLDNUM)
100      END DO
         NEWDE(CHRPTR:CHRPTR) = SCOLN
C
C      IF OUTPUT DE RECORD > 80 CHAR'S, WRITE 2 LINES...
C
         IF (CHRPTR.GT.80) THEN
          DO I=1,11
           IF (NEWDE(82-I:82-I).EQ.'@') GOTO 110
          END DO
110       WRITE(1,111)NEWDE(1:81-I)
111       FORMAT(A<81-I>)
          NEWDE(1:80)=NEWDE(82-I:161-I)
         ENDIF
         WRITE(1,1) NEWDE(1:80)
C
C      ERASE UNNECESSARY DATA FORM PD RECORD AND WRITE TO OUTPUT FILE;
C      THEN PURGE
C
115      PDLINE(65:80)= ' '
         WRITE(1,1) PDLINE
         PDLINE(1:80) = ' '
C
C      STORE CURRENT BACK-POINTER TO DE RECORD TO COMPARE WITH NEXT...
C
         LSTPTR = NEWPTR
C
C      END OF LOOP GET NEXT PD RECORD...
C
         GOTO 10
120      CONTINUE
999      RETURN
         END
```

```fortran
C        START OF SUBROUTINE COMPRESS HERE
         SUBROUTINE COMPRESS(OUTFILE)
C
C        PURPOSE:
C                TO CLEAR AWAY ALL TRAILING BLANKS FROM THE OUTUT FILE
C
C        VARIABLE DECLARATIONS...
C
         CHARACTER * 80 TEXT
         CHARACTER * 60 OUTFILE
         INTEGER        LENGTH
C
C        REWIND THE INPUT FILE AND OPEN THE OUTPUT FILE
C
         REWIND 1
         OPEN (UNIT=4,NAME=OUTFILE,STATUS='NEW',CARRIAGECONTROL='LIST',
     +         ERR=1000)
         GOTO 100
C
C        GETS HERE IF THERE IS AN ERROR IN THE OUTPUT FILE NAME...
C
1000     WRITE(*,1001)'Error in OUTPUT file name. Output written to file',
     +         ' IGES.OUT'
1001     FORMAT(1X,A,A)
         OPEN (UNIT=4,NAME='IGES.OUT',STATUS='NEW',CARRIAGECONTROL='LIST')
C
C        READ RECORD LINES INTO BUFER ONE AT A TIME...
C
100      READ(1,30,END=400) TEXT
30       FORMAT(A80)
         LENGTH = 80
C
C        GO THRU EACH LINE DELETEING TRAILING BLANKS
C
200      IF (TEXT(LENGTH:LENGTH).NE.' ') GOTO 300
         LENGTH=LENGTH-1
         IF (LENGTH.GT.1) GOTO 200
C
C        WRITE PROCESSED LINES TO THE OUTPUT FILE
C
300      WRITE(4,40) TEXT(1:LENGTH)
40       FORMAT(A<LENGTH>)
C
         GOTO 100
C
400      CONTINUE
         RETURN
         END
```

THIS PAGE LEFT BLANK

LIST OF REFERENCES

ANSI68   American National Standards Institute, Code for Information Interchange (X3.4-1968), ANSI, 1968.

ANSI74   American National Standards Institute, Code Extension Techniques for Use with the 7-Bit Coded Character Set of American National Standard Code for Information Interchange (ASCII), X3.4-1968 (X3.41-1974), ANSI, 1974.

ANSI77   American National Standards Institute, Code for Information Interchange (X3.4-1977), ANSI, 1977.

ANSI78   American National Standards Institute, Line Conventions and Lettering (Y14.2-1979), ANSI, 1978.

ANSI79   American National Standards Institute, FORTRAN (X3.9-1978), ANSI, 1978.

BLOM82   Blomgren, R. and R. Fuhr, Algorithms to Convert Between Rational B-spline and Rational Bezier Representations of Curves and Surfaces, National Bureau of Standards, 1982.

COON67   Coons, S. A., Surfaces for Computer Aided Design of Space Forms, MIT Project MAC TR-41, June 1967.

DEBO78   deBoor, C., A Practical Guide to Splines, Springer-Verlag, 1978.

DOD12D   Department of Defense, Abbreviations for Use on Drawing, Specifications, Standards, and in Technical Documents (MIL-STD-12D).

DOCA76   DoCarmo, M. P., Differential Geometry of Curves and Surfaces, Prentice Hall, 1976.

FAUX79   Faux, I., and M. J. Pratt, Computational Geometry for Design and Manufacture, John Wiley and Sons, 1979.

GORD 74 Gordon, W. J. and R. F. Riesenfeld, "B-Spline Curves and Surfaces", published in Barnhill, R. E. and R. F. Riesenfeld, ed., Computer Aided Geometric Design, Academic Press, 1974.

HILD76   Hildebrand, F., Advanced Calculus for Applications, Prentice Hall, 1976.

IEEE260 Institute of Electrical and Electronics Engineers, IEEE Standard Letter Symbols for Units of Measurement (ANSI/IEEE Std 260), IEEE.

IEEE85   Institute of Electrical and Electronics Engineers, Standard for Binary Floating-Point Arithmetic (ANSI/IEEE Std 754-1985), IEEE, 1985.

IITR68   Illinois Institute of Technology Research Institute, APT Computer System Manual: Volume 2 - Subroutine Library, IITRI, 1968.

JOBL78   Joblove, G. H. and D. Greenberg, Color Spaces for Computer Graphics, SIGGRAPH Proceedings, 1978.

LIST OF REFERENCES

KAPL52   Kaplan, W., Advanced Calculus, Addison-Wesley, 1952.

ROGE76   Rogers, D. F. and J. A. Adams, Mathematical Elements for Computer Graphics, McGraw-Hill, 1976.

SMIT78   Smith, A. R., Color Gamut Transformation Pairs, Computer Graphics, 1978.

THOM60   Thomas, G., Calculus and Analytic Geometry, Addison-Wesley, 1960.

IGES GLOSSARY

The spirit of this Glossary is to provide general, sometimes intuitive information pertaining to certain phrases and concepts either appearing in or alluded to by this document. The spirit is not to provide detailed mathematical definitions such as may be found within the document itself.

ANGULAR DIMENSION ENTITY

An annotation entity designating the measurement of the angle between two geometric lines.

ANNOTATION

Text or symbols, not part of the geometric model, which provide information.

ASSEMBLY (IEEE 200-1975)

A number of basic parts or subassemblies, or any combination thereof, joined together to perform a specific function.

ASSOCIATIVITY

A structure entity, which defines a logical link or relationship between different entities.

ASSOCIATIVITY DEFINITION ENTITY

A structure entity which designates the type (link structure) and generic meaning of a relationship. (See PRE-DEFINED ASSOCIATIVITIES)

ASSOCIATIVITY INSTANCE ENTITY

A structure entity formed by assigning specific values to the data items defining an associativity.

ATTRIBUTE

Information, provided in specific fields within the directory entry of an entity, which serves to qualify the entity definition.

AXONOMETRIC PROJECTION

A projection in which only one plane is used, the object being turned so that three faces show. The main axonometric positions are isometric, dimetric, and trimetric.

BACK POINTER

A pointer in the parameter data section of an entity pointing to an associativity instance of which it is a member.

BASIC PART (IEEE 200-1975)

One piece, or two or more pieces joined together, which are not normally subject to disassembly without destruction of designed use.

BLANK STATUS FLAG

A portion of the status number field of the directory entry of an entity designating whether a data item is to be displayed on the output device.

BOUNDED PLANE

A finite region defined in a plane.

BREAKPOINT

A member of an increasing sequence of real numbers which is a subsequence of the knot sequence used to specify parametric spline curves.

B-SPLINE BASIS

A set of functions which form a basis for the set of splines of specified degree on a specified knot sequence. B-spline basis functions are characterized by being splines of minimal support. See appendix A4 for more details.

CENTERLINE ENTITY

An annotation entity for representing the axis of symmetry for all symmetric views or portions of views, such as the axis of a cylinder or a cone.

CIRCULAR ARC ENTITY

A geometric entity which is a connected portion of a circle or the entire circle.

CLASS

A group of data items pertinent to a common logical relationship in an associativity definition.

CLIP

To abbreviate or terminate the intended display of an entity along an intersecting curve or surface.

CLIPPING BOX

A bounding set of surfaces which abbreviate the intended display of data to that portion which lies within the box.

CLIPPING PLANE

A bounding plane surface which abbreviates the intended display of data to that portion which lies on one or the other side of the plane.

CLOSED CURVE

A curve with coincident start and terminate points.

COMPLEMENTARY ARC

Either of the two connected components of a closed connected curve which has been divided by two distinct points lying on the curve.

COMPONENT

Typically a synonym for part (e.g., resistor, capacitor, microcircuit, etc.), but also may refer to a subassembly being treated as a part. The IGES representation of a component may be a collection of entities, associativities, and properties.

COMPOSITE CURVE

A connected curve which is formed by concatenating two or more curve segments.

CONIC ARC ENTITY

A geometric entity which is a finite connected portion of an ellipse, a parabola, or a hyperbola.

CONNECTED CURVE

A curve such that for any two points P1 and P2, one can travel from P1 to P2 without leaving the curve.

CONNECTION POINT ENTITY

A geometric entity giving the XYZ location and other information (e.g., text labels) of a point of connection. May be independent or subordinate to a Network subfigure definition and/or instance. Used for netlist information.

CONSTITUENT

A member of a set.

CONTROL POINT

A point in definition space which appears in the numerator of the expression for a rational B-spline curve or surface. If the weights are all positive, the resulting curve or surface lies within the convex hull of the control points. Its shape resembles that of the polygon or polyhedron whose vertices are the control points. A control point is sometimes referred to as a B-spline coefficient. See appendices A4 and A6 for more details.

COONS PATCH

A three dimensional surface.

COPIOUS DATA ENTITY

A geometric entity sometimes used as an annotation entity, containing arrays of types of real numbers to which a specific meaning has been assigned. One form number corresponds to one special meaning.

DEFINITION LEVEL (or DISPLAY LEVEL)

The graphics display level (or layer) on which one or more entities have been defined.

DEFINITION MATRIX

The matrix which transforms the coordinates represented in the definition space into the coordinates represented in the model space.

DEFINITION SPACE

A local Cartesian coordinate system chosen to represent a geometric entity for the purpose of mathematical simplicity.

DEFINITION SPACE SCALE

A scale factor applied within an entity definition space.

DEVELOPABLE SURFACE

A surface which can be unrolled onto a plane.

DIAMETER DIMENSION ENTITY

An annotation entity designating the measurement of a diameter of a circular arc.

DIRECTED CURVE

A curve with an associated direction derived from the start and terminate points.

DIRECTORY ENTRY SECTION

The section of an IGES file, consisting of fixed field data items for an index and attribute list of all entities in the file.

DIRECTRIX

The curve entity used in the definition of a tabulated cylinder entity.

DISPLAY SYMBOL

A method for graphically representing certain entities (plane, point, section) for identification purposes.

DRAWING ENTITY

A structure entity which specifies the projection(s) of a model onto a plane, with any required annotation and/or dimension.

DRILLED HOLE PROPERTY

A predefined property that assigns the physical attribute of a hole that can be made

by a drill. May be used in electrical applications to 1) define a via from one printed circuit board, PCB, layer to another, 2) define a plated via hole, and 3) give the first physical drill diameter and/or the finished hole diameters. It is usually attached to a point, circle, subfigure definition or instance.

ENGE VERTEX

A method of geometric modeling in which a two- or three-dimensional object is represented by curve segments (edges of the object) connected to points or vertices of the object. A higher level of topological information can be contained in such a model than is implied by a 'wire-frame' terminology, but in the context of this specification the terms are used interchangeably.

ENTITY

The basic unit of information in a file. The term applies to single items which may be individual elements of geometry, collections of annotation to form dimensions, or collections of entities to form structured entities.

ENTITY LABEL

A one to eight character identifier for an entity. This term may implicitly include the entity subscript, providing for additional characters.

ENTITY SUBSCRIPT

A one to eight digit unsigned integer associated with the entity label. The label and subscript specify a unique instance of an entity within an array of entities.

ENTITY TYPE NUMBER

An integer used to specify the kind of the entity. For example, the circular arc entity has an entity type number of 100.

ENTITY USE FLAG

A portion of the status number field of the directory entry of an entity to designate whether the entity is used as geometry, annotation, structure, logical, or other. For example, a circle used as part of a point dimension would have an entity use flag which designates annotation.

EXTERNAL REFERENCE ENTITY

A mechanism for referencing definitions which do not reside in the same IGES file as the instances of those definitions.

FINITE ELEMENT

A small part of a structure defined by the connection of nodes, material, and physical properties.

FLAG NOTE ENTITY

An annotation entity which takes label information and formats it such that the text is circumscribed by a flag symbol.

FLASH

The Flash entity may be used to represent a repetitive artwork feature which is usually produced by photo-optical means. Examples include PC pads, targets, clearances and IC features.

FLASH ENTITY

A geometric entity used for photo-plotting apertures and other filled areas. May be used for representing metallic conductive material on a printed circuit board such as pads and traces. Also, may be used in integrated circuit, IC, chip masks.

FLEXIBLE PRINTED CIRCUIT

An arrangement of printed circuit and components utilizing flexible base materials with or without flexible cover layers.

FLOW ASSOCIATIVITY ENTITY

A predefined associativity that represents a flow path. In electrical applications such as schematics and physical descriptions for Printed Wiring Boards, PWB, Printed Circuit Boards, PCB, PCB assemblies, ICs, etc., it presents a common electrical signal (e.g., voltage). In piping applications it represents a flow path between only one source and sink, but branching is allowed to other Flow Associativities. It provides netlist information for a single flow.

FONT CHARACTERISTIC

An integer which is used to identify a text font. Font characteristic numbers may be positive which indicate an IGES-defined text font or may be negative which is interpreted as a text font definition entity.

FORM NUMBER

An integer which is used when needed to further define a specific entity. This becomes necessary when there are several interpretations of an entity type. For example, the form number of the conic arc entity indicates whether the curve is an ellipse, hyperbola, parabola, or unspecified. The form number is also used when necessary to supply sufficient information in the directory entry of an entity to allow the structure of the parameters in the parameter data entry to be decoded.

GENERAL LABEL ENTITY

An annotation entity consisting of a general note with one or more associated leaders.

GENERAL NOTE ENTITY

An annotation which consists of text which is to be displayed in some specific size and at some specific location and orientation.

GENERATRIX

The defining curve which is to be swept to generate a tabulated cylinder, or revolved to generate a surface of revolution.

GEOMETRIC

> Having to do with the shape information (points, curves, surfaces, and volumes), necessary to represent some object.

GLOBAL SECTION

> The section of an IGES file consisting of general information describing the file, the file generator (pre-processor), and information needed by the file reader (post-processor).

GRID

> The set of (ui, vj) where ui and vj are the breakpoints on the u and v coordinates respectively used to specify a parametric spline or rational B-spline surface. The term grid is also applied to the projected image on the spline surface.

GROUND PLANE

> A conductor layer, or portion of a conductor layer (usually a continuous sheet of metal with suitable clearances), used as a common reference point for circuit returns, shieldings, or heat sinking.

GROUP ASSOCIATIVITY

> A predefined associativity for forming any collection of entities.

HIERARCHY

> A tree structure consisting of a root and one or more dependents. In general, the root may have any number of dependents, each of which may have any number of lower-level dependents, and so on, to any number of levels.

INSTANCE

> A particular occurrance of some item or relationship. Several instances may reference the same item.

KNOT SEQUENCE

> A nondecreasing sequence of real numbers used to specify parametric spline curves.

LABEL DISPLAY ASSOCIATIVITY

> A pre-defined associativity that is used by those entities that have one or more possible displays for their entity label. Entities requiring this associativity will have pointers in their directory entry to a label display associativity instance entity.

LEADER ENTITY

> An annotation entity, also referred to as arrow, which consists of an arrowhead and one or more line segments. In the case of an angular dimension entity, the line segment is replaced by a circular arc segment. In general, these entities are used in connection with other annotation entities to link text with some location.

LEVEL

An entity attribute which defines a graphic display level to be associated with the entity.

LEVEL FUNCTION PROPERTY

A predefined property that assigns an "application data base defined functionality" to a level. This property may stand alone (e.g., DE status is independent), that is no other entity points to it. Also, see the level field in directory entry.

LINE ENTITY

A geometric entity consisting of a straight segment connecting two points in space.

LINE FONT

A pattern for the appearance of a curve. The pattern is a repeating sequence of blanked and unblanked line segments, or of subfigure instances.

LINE FONT DEFINITION ENTITY

A structure entity which defines a line font.

LINE WEIGHT

An entity attribute which is used to determine the line display thickness for that entity.

LINE WIDENING PROPERTY

A predefined property that overrides the line weight given in the directory entry of an entity by providing a physical value for the actual width. May be used in electrical applications to describe metallization on a printed circuit board such as traces and off board connections. Also, see the FLASH and SECTION entities and the Region Fill property.

LINEAR DIMENSION ENTITY

An annotation entity used to represent a distance between two locations.

LINEAR PATH ENTITY

A geometric entity that defines a collection of linear segments that form a path. Also, see Copious Data Entity Forms 11 and 12.

MACRO BODY

The portion of a macro definition containing statements which define the action of the macro.

MACRO DEFINITION ENTITY

The structure entity, containing the macro body within its parameter data section, used to define a specific macro.

MACRO INSTANCE ENTITY

A structure entity which will invoke a macro which has been defined using a macro definition entity.

MIRROR

To reflect through an axis.

MODEL

A particular collection of data in an IGES file which describes a product.

MODEL SPACE

A right-handed three-dimensional Cartesian coordinate space in which the product is represented.

NEGATIVE BOUNDED PLANAR PORTION

A hole.

NETWORK SUBFIGURE DEFINITION ENTITY

A structure entity used to define a schematic symbol, component or pipe in electrical and piping applications. Shall be used whenever associated Connect Point Entities need to be instanced with the Network Subfigure Instance Entity. For physical components it may have subordinate entities (copious data, simple closed area, subfigure definition or instance, etc.) that have attached a Region Restriction Property giving design rules for auto routing a Printed Circuit Board, PCB. Also, 2-D component outlines and 3-D phsical descriptions may be defined.

NETWORK SUBFIGURE INSTANCE ENTITY

A structure entity used to specify an occurrence of a schematic symbol, component or pipe in electrical and piping applications. It has associated "instanced" Connect Point Entities that give the XYZ model space point of connections. Used in netlist information and part lists.

NODAL DISPLACEMENT and ROTATIONAL ENTITY

This entity is used to communicate finite element post processing data. It contains the node identifier, original node coordinates, and incremental displacements and rotations for each node for each load case.

NODAL LOAD/CONSTRAINT ENTITY

An entity used in a finite element model to apply a force, moment, or other loading or constraints at a specific node.

NODE

A point in space used to define a finite element topology.

ORDINATE DIMENSION ENTITY

An annotation entity used to indicate dimensions from a common reference line in the direction of the XT or YT axis.

ORTHONORMAL

A term describing two vectors which are orthogonal and of unit length.

PARAMETER DATA SECTION

A section of an IGES file consisting of specific geometric or annotative information about the entities or pointers to related entities.

PARAMETRIC SPLINE CURVE ENTITY

A geometric entity consisting of polynomial segments subject to certain continuity conditions.

PARAMETRIC SPLINE SURFACE ENTITY

A geometric entity which is a smooth surface made from a grid of patches. The patches are regions between the component parametric curves.

PARENT CURVE

The full curve on which a segment curve lies.

PART NUMBER PROPERTY

A predefined property that provides one or more text strings giving one to four distinct part numbers (Generic, MIL-STD, Vendor, and/or Internal) to an entity representing a physical part. May be used in electrical, piping or other applications. Usually, it is attached to a subfigure definition and/or instance that represents the part. May be used for part lists.

PATCH

A surface represented by parametric functions of two parameters which blend four given boundary curves.

PIN NUMBER PROPERTY

A predefined property that provides a text string giving a component pin number value to an entity representing an electrical component. Also, see the CONNECT POINT Entity.

PLANE ENTITY

A geometric entity which is a surface with the property that the straight line passing through any two distinct points on the surface lies entirely on the surface.

PLATED-THROUGH HOLE (ANSI/IPC-T-50B)

A hole in which electrical connection is made between internal or external conductive patterns, or both, by the deposition of metal on the wall of the hole.

POINT ENTITY

A geometric entity which has no size but possesses a location in space.

POINT DIMENSION ENTITY

An annotation entity consisting of a leader, text, and an optional circle or hexagon enclosing the text.

POINTER

A number that indicates the location of an entity within an IGES file.

POSITIVE BOUNDED PLANAR PORTION

The top of a peg.

POST-PROCESSOR

A program which translates a file of product definition data from the form of this standard into the data base form of a specific CAD/CAM system.

PRE-DEFINED ASSOCIATIVITIES

Associativities which are defined within this standard.

PRE-PROCESSOR

A program which translates a file of product defintiion data from the data base form of a specific CAD/CAM system into the form of this standard.

PRINTED BOARD (ANSI/IPC-T-50B)

The general term for completely processed printed circuit or printed wiring configurations. It includes rigid or flexible, single, double, or multilayer boards.

PRINTED CIRCUIT (ANSI/IPC-T-50B)

A conductive pattern comprised of printed components, printed wiring, or a combination thereof, all formed in a predetermined design and intended to be attached to a common base. (In addition, this is a generic term used to describe a printed board produced by any of a number of techniques.)

PRINTED CIRCUIT BOARD (ANSI/IPC-T-50B)

A part manufactured from rigid base material upon which a completely processed printed circuit has been formed.

PRINTED WIRING (ANSI/IPC-T-50B)

The conductive pattern intended to be formed on a common base, to provide point-to-point connection of discrete components, but not to contain printed components.

PRODUCT DEFINITION

Data required to describe and communicate the characteristics of physical objects as manufactured products.

PROPERTY ENTITY

A structure entity which allows numeric or text information to be related to other entities.

RADIUS DIMENSION ENTITY

An annotation entity which is a measurement of the radius of a circular arc.

RATIONAL B-SPLINE CURVE

A parametric curve which is expressed as the ratio of two linear combinations of B-spline basis functions. Each basis function in the numerator is multiplied by a scalar weight and a vector B-spline coefficient. Each corresponding basis function in the denominator is just multiplied by the corresponding weight.

RATIONAL B-SPLINE SURFACE

A parametric surface which is expressed as the ratio of two linear combinations of products of pairs of B-spline basis functions. Each product of basis functions in the numerator is multiplied by a scalar weight and a vector B-spline coefficient. Each corresponding product of basis functions in the denominator is multiplied by the corresponding weight.

REFERENCE DESIGNATOR PROPERTY

A predefined property that provides a text string giving a component reference designator value to an entity representing an electrical component. Also, see the Network Subfigure entity.

REGION

The bounded area enclosed by a closed curve or a combination of curves.

REGION FILL PROPERTY

A predefined property that is used to solid fill or unfill (nested) a closed area. May be used for cross-section material representations (i.e., concrete, steel, etc.) and artwork. Also, see the SECTION entity. Also, see the FLASH and SECTION entities and the Line Widening property.

REGION RESTRICTION PROPERTY

A predefined property that provides design rules in electrical applications. Especially, region restrictions regarding Printed Circuit Board, PCB routing rules for prohibiting or permitting vias and traces under component outlines and placement of components on the printed circuit board.

RELATION

An aspect or quality that connects two or more things or parts as being or belonging or working together or as being of the same kind.

REPEATING PATTERN

An ordered sequence of items (elements) which, after a certain point, repeats itself.

RIGHT-HANDED CARTESIAN COORDINATE SYSTEM

A coordinate system in which the axes are mutually perpendicular and are positioned in such a way that, when viewed along the positive Z axis toward the origin, the positive X axis can be made to coincide with the positive Y axis by rotating the X axis 90 degrees in the counterclockwise direction.

RULED SURFACE ENTITY

A surface generated by connecting corresponding points on two space curves by a set of lines.

SECTION ENTITY

A pattern used to distinguish a closed region in a diagram. It is represented as a form of the copious data entity.

SECTION DISPLAY SYMBOL

An arrangement of fonted straight lines in a repetitive planar pattern at a specified spacing and angle.

SET (IEEE 200-1975)

A unit or units and necessary assemblies, subassemblies, and basic parts connected or associated together to perform an operational function.

SIMPLE CLOSED AREA ENTITY

A geometric entity used to give a 2-D mathematically simple closed area (no holes or boundary intersections). See Copious Data entity Form 63. May be used for electrical design rules in conjunction with the Region Restriction property.

SINGLE PARENT ASSOCIATIVITY ENTITY

A predefined associativity that provides logical grouping of a single parent entity to its many children entities.

SPLINE

A piecewise continuous polynomial.

START SECTION

The section of an IGES file containing a man-readable file prolog.

SUBASSEMBLY (IEEE 200-1975)

Two or more basic parts which form a portion of an assembly or a unit, replaceable as a whole, but having a part or parts which are individually replaceable.

SUBFIGURE DEFINITION ENTITY

A structure entity which permits a single definition of a detail to be utilized in multiple instances.

SUBFIGURE INSTANCE ENTITY

A structure entity which specifies an occurrence of the subfigure definition.

SUBORDINATE ENTITY SWITCH

A portion of the status number field of the directory entry of an entity. An entity is subordinate if it is an element of a geometric or annotative entity structure or is a member of a logical relationship structure. The terms subordinate and dependent are equivalent within this document.

SURFACE OF REVOLUTION ENTITY

A geometric entity which is a surface generated by rotating a curve, called the generatrix, about an axis, called the axis of rotation.

SYSTEM (IEEE 200-1975)

A combination of two or more sets, generally physically separated when in operation, and other such units, assemblies, and basic parts necessary to perform an operational function or functions.

TABULAR DATA PROPERTY

The tabular data property provides a structure to accommodate point form data. The basic structure is a two-dimensional array containing data list for depenent and independent variable.

TABULATED CYLINDER ENTITY

A geometric entity which is a surface generated by moving a line parallel to itself along a space curve called the generatrix.

TERMINATE SECTION

The final section of an IGES file, indicating the sizes of each of the preceding file sections.

TEXT DISPLAY TEMPLATE ENTITY

An annotation entity used to define the display location of a text string. In electrical applications it gives a "relative" mode of location dependent on a connect point for pin number and/or pin function of components (e.g., Integrated Circuit, IC, chip pins). For electrical schematic symbols and physical components it may give the display location of the reference designator text and/or part number.

TEXT FONT

The specification of the appearance of the characters.

TEXT FONT DEFINITION ENTITY

The entity used to define the appearance of characters in a text font. A character is defined by pairing its character code with a sequence of display strokes and positional information.

TRANSFORMATION MATRIX ENTITY

An entity which allows translation and rotation to be applied to other entities. This is used to define alternate coordinate systems for definition and viewing

TRANSLATION VECTOR

A three element vector which specifies the offsets (along the coordinate axes) required to move an entity linearly in space.

UNIT (IEEE 200-1975)

A major building block for a set or system, consisting of a combination of basic parts, subassemblies, and assemblies packaged together as a physically independent entity.

VERSION NUMBER

A means for uniquely designating one specification definition or translator implementation from a preceding or subsequent one.

VIA HOLE (ANSI/IPC-T-50B)

A plated-through hole used as a through connection, but in which there is no intention to insert a component lead or other reinforcing material.

VIEW ENTITY

A structure entity used to provide the definition of a human-readable representation of a two-dimensional projection of a selected subset of the model and/or non-geometry information.

VIEWING BOX

The clipping box used to define a view.

WEIGHT

A non-zero real number which appears in the numerator and denominator of the expression for a rational B-spline curve or surface. Increasing the weight associated with a particular control point will tend to draw the resulting curve or surface toward that control point. See appendices A4 and A6 for details.

WIRE-FRAME

A method of geometric modeling in which a two- or three-dimensional object is represented by curve segments which are edges of the object. In the context of this specification, 'wire-frame' and 'edge-vertex' models are considered as the same technique and the terms are used interchangeably.

WITNESS LINE

An annotation entity consisting of line segments and used in engineering drawings to indicate the beginning or the end of a measurement.

INDEX OF TOPICS

4. TITLE AND SUBTITLE

Initial Graphics Exchange Specification (IGES) Version 3.0

5. AUTHOR(S)

Smith, B., Briggs, D., Colsher, R., Kennicott, P., Parks, C., Wellington, J.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

This document contains Version 3.0 of the Initial Graphics Exchange Specification, a defined format for the creation of a file which enables data found in today's commercially available CAD/CAM systems to be exchanged or archived.

IGES, Version 1.0, published as NBSIR 80-1978 (R) in January 1980, consisted of entity definitions for geometry, drafting and structural information. Definition entities were provided as a means of expanding the utility of IGES.

Version 3.0 further refines the concept and offers increased capability in both geometry and non-geometry data exchange. The applications of printed wiring boards and finite element models are well supported in addition to enhancements for mechanical products.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

Computer Aided Design (CAD), Computer Aided Manufacturing (CAM), design drawing, electrical information, exchange format, finite element modeling, geometrics, graphics.