NBSIR 88-3813

Initial Graphics Exchange Specification (IGES) Version 4.0

Bradford Smith*
Gaylen R. Rinaudot*
Kent A. Reed**
Thomas Wright*

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

June 1988

Final Report

75 Years Stimulating America's Progress
1913-1988

U.S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

NBSIR 88-3813

INITIAL GRAPHICS EXCHANGE SPECIFICATION (IGES) VERSION 4.0

Bradford Smith*
Gaylen R. Rinaudot*
Kent A. Reed**
Thomas Wright*

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

June 1988

Final Report

U.S. DEPARTMENT OF COMMERCE, C. William Verity, *Secretary*

NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

First printing: June, 1988.

Dedication

This Version 4.0 of the Initial Graphics Exchange Specification is dedicated to the memory of Robert Colsher, IGES Project Manager and close friend. In the IGES team noted for its hard work and dedication to cause, Bob distinguished himself as a leader and an expert revered by all. His knowledge of the Specification was based on years of implementation experience of translators for CAD systems, and he is credited with the design and implementation of the first IGES file analyzer in the world. His personal commitment to this project went beyond anything expected of a member of a volunteer professional organization. He provided guidance, direction and vision to our work, and his absence at regular meetings causes many a silent pause in conversation. We will miss our good friend and can only hope that each of us in our lifetime can make as significant a contribution to our field of work as Bob has done through IGES.

Bradford M. Smith
Chairman, IGES/PDES Organization
June, 1988

Acknowledgment

This Specification could not have been prepared without the many technical experts and the tremendous cooperative nature of the IGES/PDES Organization. Much of the burden, of course, has fallen on the Officers and the Committee Chairpersons listed here in the preface pages. But, several individuals have made special and significant contributions to the preparation and the encoding of this Specification in LaTeX and deserve separate mention.

First, I would like to recognize the efforts of my secretary, Mary Marello, without whose help and thorough attention to detail this work would lack the quality it represents. Second, I wish to acknowledge the work of Tom Wright and Kent Reed of the National Bureau of Standards to encode the text of this Specification and a number of its illustrations in LaTeX. Third, my special thanks go to William R. Oakes and Mary A. Lattin of Los Alamos National Laboratory for their help in the LaTeX encoding of all the mathematical formulae. Finally, the excellent quality of the CAD-produced illustrations within this Specification is the result of work by Clarence Johnson, on my staff, to prepare CAD models of each and by Dennette Harrod, Jr., of Computervision Corporation for his help with the many changes to the illustrations made during the edit review process and for the final plotting of each.

I would like to commend our IGES/PDES Coordinator, Gaylen Rinaudot, for her exceptional ability and tremendous effort put into the scheduling and execution of IGES meetings, and for the coordination of the voting process which led to the preparation of this document.

Bradford M. Smith
Chairman, IGES/PDES Organization

Officers of the IGES/PDES Organization

Special recognition is accorded to the following officers of the Initial Graphics Exchange Specification/Product Data Exchange Specification (IGES/PDES) Organization.

Chairman	Bradford Smith
Coordinator	Gaylen R. Rinaudot
IGES Project Manager	Robert Colsher, Constance H. Panzica
PDES Project Manager	Kalman Brauner, Thurber Moffett, Anthony Day
ANSI Project Manager	Earl Weaver
ISO Project Manager	Jerry Weiss
Change Control	Curt Parks
Acceptance Testing	James Fleming
Application Validation	Mark Palmer
Architecture, Engineering, and Construction	Kent Reed, Pat Rourke, Barbara Warthen
Curves and Surfaces	Edward Clapp
Drafting	Robert Colsher, Bill Turcotte
Edit	Philip Kennicott
Electrical Applications	Larry O'Connell
Finite Element Modeling	Harvey Gray
Implementations	Bill Loye
Logical Layer	Doug Schenck
Manufacturing	William Burkett
Mechanical Product Definition	Spencer DePauw, Tom Voegeli
Methodology Testing	Dave Remington
Physical File Format	Jeff Altemueller
Recommended Practices	Dennette Harrod Jr.
Software Support	Raymond Barker
Solids Geometry	Noel Christensen
Technical Publications	Jeff Altemueller, Kelly Chi, Marc Durnin
Test Case Development	Julia Terry
User Information	Tom Wright
Verification Testing	Stan Briggs

Members of the IGES/PDES Organization

Joseph	Aebischer	Martin Marietta Data Systems
Ajay	Agrawal	Alco Intercon - X
Mark F.	Allison	General Dynamics - DSD Fort Worth
Jeff	Altemueller	McDonnell Douglas (MDAIS)
Charles G.	Altmann	IBM Corp.
Kevin	Andersen	Duke Power Company
Bill D.	Anderson	General Dynamics Corp.
Dennis E.	Anderson	Puget Sound Naval Shipyard
Robert E.	Anderson	NAVAIR Engineering Support Office
Ignatius G.	Ang	Proctor and Gamble Company
Larry	Arnold	Hughes Aircraft
Kofi	Baidoo	Boeing
Bruce C.	Bailey	Automation Technology Products
George	Baker	International TechneGroup Inc.
John L.	Baker	Aluminum Co of America
Janice M.	Banocy	McDonnell Douglas MIS Co
Raymond E.	Barker	Caterpillar, Inc.
William G.	Beazley	W. G. Beazley & Assoc.
Boyd	Belnap	Hughes Aircraft
Peter	Benjamin	Lockheed Missiles And Space Co.
Tibor A.	Berenyi	Deere and Company
Joe	Bernstein	Boeing Computer Services
Anuradha	Bhagat	Electronic Data System
Dan	Billingsley	Department of the Navy
James R.	Blaha	IIT Research Institute
David J.	Blue	Prime Computer
Thomas A.	Boldt	Optigraphics Corp.
David	Boudreau	D. Appleton and Company
James R.	Bradford	Allied Bendix Aerospace
Jack	Brainin	Naval Ship R & D Center
Ray	Brandt	BDM Corporation
Kalman	Brauner	Boeing Commercial Airplane Co.
Jeff	Braunstein	Tandem Computer
M L	Brei	BITE Inc.
Raymond A.	Brengs	Naval Ship R & D Center
David	Briggs	Boeing Commercial Airplane Co.
Stan	Briggs	Naval Aviation Depot
Clare	Bronder	Adra Systems, Inc.
Richard	Brooks	McDonnell Douglas Corp.
Frederick	Bsharah	Rockwell International NAAO
Andrew	Burke	Tektronix Inc.
William C.	Burkett	McDonnell Aircraft Co.
William D.	Cain	Martin Marietta Energy Systems

Members of the IGES/PDES Organization

Bradley K.	Call	PDA Engineering
Rudy	Camarillo	General Dynamics Corp.
Vickie	Carpenter	Electronic Data Systems
William D.	Carr	Digital Equipment Corp.
Robert	Carringer	Int'l Techne Group Inc.
Mark	Chamberlain	Hughes Aircraft
Barry	Chapman	VSE Corporation
Kelly	Chi	McDonnell Douglas (MDAIS)
Ken	Christensen	Ford Motor Company
Noel	Christensen	Allied Bendix Aerospace
Linda	Cieska	Eaton Corporation
Edward	Clapp	IBM Corp.
Margaret	Cline	Lockheed Georgia
Robert	Colsher	IGES Data Analysis Corp
Tandy	Cook	Boeing Computer Services
Earleen	Corn	Bath Iron Works
Margery J.	Cox	Newport News Shipbuilding
Jeffrey K.	Crowell	General CAD/CAM Inc.
Neal	Davis	US Army Corps of Engineers
Laurel C.	Dawson	McDonnell Douglas
Anthony James	Day	Sikorsky Aircraft UTC
James	Delk	US Army Materiel Command
David Leigh	Dellinger	Boeing Military Airplane Company
Allen R.	DelRosario	McDonnell Douglas
Mark	Demeranville	Unisys Defense Products Group
Spencer	DePauw	Caterpillar, Inc.
Joyce	DeTolla	Navy
Richard N.	DiFronzo	Metagraphics Inc.
Rose	DiGeronimo	Department of the Navy
Ralph	Disa	Pratt & Whitney
Jim	Doar	Boeing Computer Services
Gary	Douglas	US Army Corps of Engineers
Ron	Downer	Hughes - MSG
Alan E.	Dragoo	McDonnell Douglas MIS Co
Mark	Dunn	United Technologies
Marc W.	Durnin	Lockheed-Georgia Company
Andrew	Dvorak	Newport News Shipbuilding
Colin R.	Earl	Automation Technology Products
Lee W.	Eason	Department of Transportation
Andrew	Eliachevsky	Electronic Data Systems
Ned	Eliasen	Honeywell
Richard	Ellermeier	Mentor Graphics
Richard	Endo	Versacad Corporation
Dr. Max	Engeli	FIDES Treuhandgesellschaft Electrical
Thomas C.	Estervog	Boeing Computer Services
John C.	Faulkner	S D R C
Ayron L.	Fears	IBM Corp.
Nick	Fkiaras	Tektronix Inc.
John	Flaherty	Boeing Computer Services
Jim	Fleming	Cummins Engine Co. Inc.
Peter	Fosselman	McDonnell Douglas Corp.

Leland H.	Frayseth	Bechtel Petroleum Inc.
Donald E.	Fusco	Boeing Computer Services
Roger	Gale	D. Appleton Co.
William	Garner	Martin Marietta
Ulrich	Gengenbach	Kernforschungszentrum Karlsruhe
William	Gerrein	General Electric Co.
Elena	Gersht	Prime Computer Inc.
Albert J.	Gibbons	Westinghouse Electric Corp.
W. F.	Gielingh	IBBC - IND
Marshall	Giguere	Computervision Corporation
Chip	Gilbert	Martin Marietta
Mitchell	Gilbert	Grumman Aerospace Corp.
Alfred B.	Gillard	Chrysler Corp.
Burton	Gischner	General Dynamics - Electric Boat
Rainer	Glatz	VDMA Dept of Informatic
George	Goldsmith	McDonnell Douglas Astronautics
Mike	Golish	US Army - CERL-FS
Scott A.	Gordon	NASA Goddard Space Flight Center
R.J.	Goult	Cranfield Institute of Technology
William H.	Gray	Martin Marietta Energy Systems
Ronald D.	Green	Boeing Military Airplanes
William B.	Gruttke	McDonnell Douglas (MDAIS)
Eric J.	Gunther	U S Air Force
Eugene F.	Gurga	Computervision
C. Hayden	Hamilton III	PDA Engineering
William	Hammon	Tandem Computer
Yosef	Haridim	Boeing Electronics Co.
C. Scott	Harris	Hughes Aircraft Co
Don E.	Harrison	LTV Aircraft-Products Group
Jessie A.	Harrison	Rohr Industries, Inc.
Dennette A.	Harrod Jr	Computervision Corporation
Brett	Hartman	Boeing Computer Services
Marie T.	Harvey	Naval Electronic Systems Center
Barry	Hass	Raytheon Company
Don	Hemmelgarn	International TechneGroup Inc
Mark A.	Hollingshead	McDonnell Douglas AEC
Paul	Hollingshead	McDonnell Aircraft
R. John	Hooper	Eastman Kodak Company
Kazuo	Hori	McDonnell Douglas Astronautics Co
William A.	Hussong	Honeywell
Diane	Iacovelli	Northrop ASD
Greg	Ippolito	General Dynamics
Madeleine R.	Isenberg	Northrop Corporation
Robert	Ivey	Westinghouse Electric Corp.
Dwight L.	Jaeger	Los Alamos National Laboratory
Jon L.	Judd	General Dynamics Corp.
Kevin	Jurrens	Allied-Bendix Aerospace
Diane	Kaminski	Northrop Corp.
Esfandiar	Kamvar	A T & T Bell Laboratories
Toru	Kato	Toyota Motors
Dewayne	Kelley	Menasco

Members of the IGES/PDES Organization

Michael	Kelley	Auto-Trol Technology
J.C.	Kelly	Sandia National Laboratories
Debbie	Kengott	AutoTrol Technology Corp.
Philip	Kennicott	General Electric Co.
Paul	Kieffer	Pratt and Whitney
Fumihiko	Kimura	University of Tokyo
Elbert	King	General Motors-Truck & Bus Group
Ed	Klages	Martin Marietta
Frank	Knight	The Aerospace Corp.
Michael	Kniley	Applicon
Ronald	Kockler	Northrop Aircraft
Rick	Kohler	Dana Corp.
Toshio	Kojima	Mechanical Engineering Laboratory
Karen L.	Kontry	Electronic Data Systems
Gregory	Kovacicn	Holguin
Diane	Koziol	Pratt & Whitney Aircraft
Ravi	Krishnaswami	Electronic Data Systems
Vincent	Krocinski	IBM
Sudhir	Kshirsagar	The Proctor and Gamble Company
M. K.	LaColla	Northrop Advanced Systems
Harry	Ladd	DuPont
Fred	Langhorst	Mentor Graphics
Deborah	LaPay	Westinghouse Electric Corp.
Kaiman	Lee	Naval Facilities Engineering Command
Tony	Lee	Aries Technology Inc.
Richard	Leung	A T & T
C. Richard	Lewis	Electronic Data Systems
Dr. Larry	Lichten	California State Univ. Northridge
Olga	Lichten	IBM Corp.
Maureen	Little	Naval Civil Engineering Laboratory
Patricia	Lock	Northrop Coporation
Julie	Long	Electronic Data Systems
Mark	Losinski	Control Data Corporation
William	Loye	Loye and Associates
Scott	Mackelprang	Rocketdyne
Robert	MacLatchie	PlanPrint Company
Larry	Mankins	Boeing Military Airplane Company
William D.	Marks	Hughes Aircraft Co
Douglas J.	Martin	NASSCO
Robert	Martin	Ontologic
Linda	Martino	IBM Corp.
Howard G.	Mason	British Aerospace
Raphael	McBain	General Dynamics Corp.
Dennis	McBurney	Boeing Military Airplane Co
Patrick	McFadden	Boeing Computer Services
C. Kevin	McIntyre	Artecon
Stanley	McMillen	Gerber Systems Technology
Jim	McMullen	Fluor Engineers Inc.
Monte	Meals	CALMA Co.
Robert	Meersman	KHT-TILBURG University - INFOLAB
Carl A.	Mellenthin	McDonnell Aircraft Co.

Thomas G.	Melson	McDonnell Aircraft Company
Telis K.	Menas	
Chris	Merrill	Corp of Engineers
Walter	Messcher	US Department of Transportation
H. W. Guy	Meyer	Wisconsin Dept of Transportation
Lyle	Meyer	Eastman Kodak
Jay A.	Miller	CADNETIX Inc
Joan	Milloy	Control Data Corporation
Carolyn F.	Mindel	S D R C
Thurber J.	Moffett	Northrop Corporation
Richard	Moore	Eaton Corporation
Greg	Morea	General Dynamics Corporation
Donald E.	Morgan	GE AEBG 24043
Charles B.	Morrill	IBM Corp.
Sorour	Moshirvaziri	IBM Corp.
Joseph A.	Mudd	Bechtel Petroleum Inc.
James G.	Nell	Westinghouse Electric Corp.
Bruce R.	Nelson	Control Data Corporation
Erick	Nelson	McDonnell Douglas Astronautics Co.
Paul A.	Nelson	Hughes Aircraft Co.
William W.	Niemi	General Dynamics Corp.
Mike	Nolan	JIA Associates
David	Norling	Boeing Computer Services
Brigitte	Nugteren	Ford Aerospace
Larry	O'Connell	Sandia National Laboratories
W. Rob	Oakes	Los Alamos National Laboratory
Wesley W.	Obermeier	LTV Aerospace and Defense Company
Dennis	Ordiway	Northrop Corporation
Jeffrey	Otten	General Electric Co
Mike	Overbeek	Auto-Trol Technology Corp
Jon	Owen	CAD/CAM Data Exchange Cntr
Mark	Palmer	National Bureau of Standards
Connie	Panzica	General Motors BOC Hqts
Lawrence O.	Parker	Hughes Aircraft Co.
Curtis H.	Parks	General Dynamics Corp.
Jeff L.	Parks	Boeing Computer Services
Robert E.	Parks	Sandia National Laboratories
James S.	Pate	Boeing Computer Services
Lawrence J.	Peck	Cimlinc Inc.
Alan	Peltzman	Peltzman & Associates
Kim	Perlotto	Pratt & Whitney
Tom	Peters	Computervision Corporation
Freda	Phelps	Lawrence Livermore National Lab
Brock	Pilkey	Accugraph Corp.
Jon	Pittman	HOK Computer Services Corp.
Patrick T.	Plunkett	Naval Air System Command
David A.	Poland	Northrop Aircraft
Gary W.	Pollak	Society of Automotive Engineers
Richard A.	Price	Boeing Military Airplane Company
Tim	Prohaska	OPTI-COPY
James C.	Purdon	Applicon Inc.

William	Putnam	Rockwell International NAAO
Guus	Ranke	Nederlandse Philips Bedryven
Paul	Ranyak	General Dynamics Corp.
Kent	Reed	National Bureau of Standards
E. A.	Reid	Caterpillar Inc.
David O.	Remington	Naval Surface Weapons Center
Jim	Reyer	Cadnetix
Gaylen R.	Rinaudot	National Bureau of Standards
Phil	Rios	CALMA Co.
Mayford B.	Roark	Boeing Computer Services
Jon	Robertson	Graphic Information Exchange Service Inc
Rob	Robinson	McDonnell Douglas
Ellen	Rosen	USC
Phillip	Rosol	Martin Marietta Data Systems
Patrick W.	Rourke	Newport News Shipbuilding
Mojgan	Royer	FLUOR Inc.
Herbert F.	Ryan	McDonnell Aircraft Company
Walter J.	Rygiel	Ford Motor Company
Charles	Savage	D. Appleton Co.
Steven R.	Schachtner	Johns Hopkins University
Robert W.	Schaffran	Department of the Navy
Joseph	Schartman	General Electric
Bob	Scheller	McDonnell Aircraft Company
Douglas	Schenck	McDonnell Douglas
E. G.	Schlechtendah	Kernforschungszentrum Karlsruhe
John	Schmarr	General Electric
Randy	Schmid	Hughes Aircraft Co EDSG
Dirk	Schroeter	Martin Marietta
Gary M.	Seibert	US Army Corps of Engineers
Henry A.	Seymour	Martin Marietta
Arnold	Shak	Grumman Data Systems
Nigel K.	Shaw	CAD/CAM Data Exchange Center
Joseph R.	Shematek	Northrop Corporation
Jim	Sheridan	General Dynamics
Chia Hui	Shih	S D R C
John C.	Simon	McDonnell Douglas MIS Co
Dave	Simpson	CADAM Inc.
J.	Singh	Computervision Corporation
Bradford	Smith	National Bureau of Standards
Myron	Smith	Eastman Kodak
Robert F.	Smith	CALMA Co.
Neil	Snodgrass	D. Appleton and Company
Michael V.	Spinosa	Computervision Corp.
Fred	Stahl	IBM Corp.
Oren	Stephans	Naval Ship R & D Center
Charles L.	Stoddard	Pratt & Whitney
Jon	Stonecash	Control Data Corporation
Michael H.	Swanger	Georgia Tech
Emery	Szmrecsanyi	Chrysler Motors
Farhad	Tahmasebi	Intercad Corporation
Kishinami	Takeshi	Hokkaido Univ. - JAPAN

James "Steve"	Tate	Boeing Military
Dennis	Taylor	Marshall Space Flight Center
Boris	Tenenbaum	WANG Laboratories
Richard	Tepe	International TechneGroup Inc
Julia	Terry	Martin Marietta Energy Systems
David	Theilen	Allied Bendix Aerospace
Debbie	Thomas	Rutherford Appleton Laboratory
Ken D.	Thomas	Duke Power Company
J. Garth	Thompson	Kansas State University
Paul	Thompson	Control Data Corporation
Vince E.	Thompson	Burns & McDonnell Engineering Co
C.B. "Bill"	Thompson, Jr.	Southern Company Services Inc
Henry G.	Timmer	Northrop
Fremont	Tittle	Control Data Corporation
Dietmar	Trippner	BMW
Bill	Turcotte	IGES Data Analysis
James A.	Turner	University of Michigan
Monica	Turner	CADAM Inc.
Joan	Tyler	National Bureau of Standards
Jan	Van Maanen	Rutherford Appleton Laboratory
James R.	Vander Schaaf	Bath Iron Works
Robert J.	Ventura	Martin Marietta Aerospace
Gail	Vermilyea	Vertek Associates
Tom	Voegeli	Caterpiller Tractor Co.
Reza	Wajih	SoftCAD, Inc.
Barbara	Warthen	CALMA Co.
Earl P.	Weaver	Ballistics Research Laboratory
Thomas K.	Weebe	Northrop Advanced Systems
Jerry A.	Weiss	McDonnell Aircraft Co.
Uwe	Weissflog	IBM Corp.
Ronald O.	Wester	General Electric Co.
Anna	Whelan	McDonnell Douglas
Tracy	Whelan	Control Data Corporation
Gary	Whitaker	Ford Motor Company
George	Whitehurst	NASA Langley Research Center
Gary	Whitt	Intergraph Corp.
Robert	Willis	National Computer Graphics Assoc.
Peter	Wilson	General Electric Co
Robert M.	Wilson	Martin Marietta Energy Systems
Carolyn	Wimple	Lawrence Livermore National Lab
Richard C.	Winfrey	Digital Equipment Corp.
Edward R.	Wittenberg	Ford Motor Company
L. Stephen	Wolfe	Computer Aided Design Report
Vivian	Wong	Douglas Aircraft
Dan	Wooley	Newport News Shipbuilding
Debbie	Wright	Sikorsky Aircraft
Tom	Wright	National Bureau of Standards
Akio	Yajima	Hitachi
Sheree	Yang	Ford Aerospace & Communications Co
Yuhwei	Yang	Rockwell International NAAO
Michael A.	Yinger	Northrop Advanced Systems

Herbert	Zapomueel	Siemens AG ZTZFAAUT4
Philip M.	Zaretzki	Electronic Data Systems
Fred	Zeiler	Infodetics Corporation
John	Zimmerman	Allied Bendix Aerospace
Glen	Ziolko	LTV Aerospace & Defense

Foreword

This version of the Initial Graphics Exchange Specification improves on the earlier Version 3.0 (NBSIR86-3359), published in April, 1986, by extending and refining the technical content of the Specification. In addition, many changes have been made to improve the syntax, clarity, and consistency of the text and figures contained in the document.

All extensions to the IGES document represent the best thinking of the IGES/PDES Organization. In past versions, all of this material has been added to the main body of the document and submitted to ANSI for standardization approval. Although we strongly encourage implementors to utilize all technical additions to the document where there is a need, we have provided for a distinction between that material that has stood the test of practical implementation experience and that which has not. Some of the extensions to Version 3.0 have not been proven stable by sufficient implementation experience and are documented in Appendix J, Untested Entities, instead of the main body of the Specification document. This material is subject to change as a result of implementation experience. Until sufficient implementation experience has been gained and reported to the IGES/PDES Organization, this material will not be submitted for ANSI standardization approval. When the implementation and reporting requirements have been satisfied, the material will be moved to the main body of the document and submitted to ANSI for standardization approval.

Constructive Solid Geometry (CSG) is introduced to the Specification in this version. Specifically, ways of representing the regularized operations for union, intersection, and difference have been defined. Primitive representations have been established for block, wedge, cylinder, cone, sphere, torus, solid of linear extrusion, solid of revolution, and ellipsoid. It should be noted that the other popular approach to solids, called Boundary Representation (B-Rep), is not included in this version. B-Rep will be in the next version of this Specification, Version 5.0.

Electrical/electronic applications have been extended to include the ability to attach predefined electrical attributes and properties. Nominal values, maximum ratings, propagation delays, and other pertinent data about bipolar transistors and other devices may now be defined using the parameters contained in the Attribute Table Entity (Type 322, Electrical Attribute List, ALT = 2). Any public-domain behavioral parameter data that might be added to the table will be welcomed by the Electrical Application Committee.

Two major contributions have been made for Architecture/Engineering/Construction (AEC) applications. First, the Attribute Table Entities (Type 322 and Type 422) can now be used to define attribute data and associated graphic representations that are often necessary for the various AEC applications (e.g., a pattern fill used to denote where concrete is used for construction purposes). Second, Appendix D contains a three-dimensional piping model that shows how existing entities can be used to define piping information.

The Finite Element Modeling (FEM) capability of the Specification has been expanded to describe FEM results data. Specifically, the Nodel Results Entity (Type 146) is to be used for defining nodal temperature and displacement results and the Element Results Entity (Type 148) is to be used for defining elemental stress and strain results.

Basic geometry entities have had no format changes from Version 3.0. However, minor changes to their descriptions have been made to clarify existing entities. For example, the weights of B-spline

curves and surfaces must now be greater than zero. A new form has been added to the Plane Entity (Type 108, Form 0).

Classic drafting entities, including annotation and some of the geometry entities have also remained the same as in Version 3.0. Some clarifications and error corrections have been made to the annotation section. New forms have been added to the Ordinate Dimension Entity (Type 218, Form 1), Radius Dimension Entity (Type 222, Form 1), and General Symbol Entity (Type 228, Forms 1, 2, and 3). Note that the new General Symbol forms are to be used to add feature control information to the entity.

Contents

List of Figures

List of Tables

1. General

1.1 Purpose

This Specification establishes information structures to be used for the digital representation and communication of product definition data. Use of this Specification permits the compatible exchange of product definition data used by various Computer-Aided Design and Computer-Aided Manufacturing (CAD/CAM) 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 through 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.

Chapter 1 is general in nature and defines the overall purpose and objectives of this Specification. Chapter 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 Chapter 3 deals with two- and three-dimensional edge-vertex models, with simple surface representations and Constructive Solid Geometry (CSG) representations. Chapter 4 specifies non-geometric representations, including common drafting practices, data organization methods, and data definition methods.

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

1.3 Untested Entities

It is the policy of the organization to ensure that entities are tested before being introduced into the Specification. In cases where this testing is not yet complete, the entity is included in Appendix J. A prospective implementor is warned that, despite the fact that Appendix J entities represent the best judgment of the organization, there is a chance that changes will be required before these entities are introduced into the body of the Specification. If these entities are judged useful and implementation

is attempted, the results of the attempt will be useful to the IGES/PDES Organization. Contact the IGES Office at the National Bureau of Standards to report problems and successes.

1.4 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, 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 solid modeling. 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, allowing exchange of product definition data, 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. This Specification specifies communication formats (information structures) for subsets of the product definition.

1.5 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 is 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 data in the file is the entity. Entities are categorized as geometry and non-geometry. Geometry entities represent the definition of the physical shape and include points, curves, surfaces, solids, and relations which are collections of similarly structured entities. Non-geometry 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-geometry 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 file. Typical non-geometry entities for drawing definition, annotation, and dimensioning are the view, drawing, general note, witness line, and leader. Typical non-geometry entities for attributes and groupings are the property and the associativity entities.

- ADMINISTRATIVE
  - Product Identification
  - Product Structure
- DESIGN/ANALYSIS
  - Idealized Models
- BASIC SHAPE
  - Geometric
  - Topological
- AUGMENTING PHYSICAL CHARACTERISTICS
  - Dimensions and Tolerances
  - Intrinsic Properties
- PROCESSING INFORMATION
- PRESENTATIONAL INFORMATION

Figure 1. Categories of Product Definition

A file consists of five or six sections: Flag (in the case of the binary or compressed ASCII form), 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. 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 in Chapter 2 has a specific assigned entity type number. While not all are assigned at this time, entity type 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 1.6.7. 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 Entity provides a mechanism to establish relationships among entities and to define the meaning of the relationship. The Property Entity allows specific characteristics, such as line widening, 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 predefined associativities and properties to be included in the Specification and unique definitions which will be defined by the user.

1.6 Concepts of Information Structures for Geometric Models

The geometric model refers to the entity set defined by Chapters 3 and 4, and comprises an entity-based product definition file. The entity types, as described above are categorized as geometry and non-geometry. In general, the geometry 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-geometry entities include structures in which an entity may be defined by a collection of other entities and structures which are independent.

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.

1.6.1 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. In addition, a value which is contained in a property may be displayed as text when an additional pointer (See Section 2.2.4.4.2) of the property points to a Text Display Template Entity (Type 432).

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.

1.6.2 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 specifically defined as part of this Specification in Section 4.3.3.3.

1.6.3 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 associativities control such representations. These provide information for orientation, clipping, line removal, and other characteristics associated with individual views rather than with the model itself.

1.6.4 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.

1.6.5 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.

1.6.6 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 Section 4.3.6.

1.6.7 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.

1.7 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 a three-dimensional piping model example while Appendix E gives explanation of spline representation and approaches for conversion techniques. Appendix F discusses the numerical stability of conic arcs. Appendix G provides mappings between color spaces. Appendix H provides a set of FORTRAN utilities to convert physical file structures in the ASCII Form from the regular ASCII Format to the Compressed ASCII Format and back. Appendix I itemizes entities from previous versions which have been made obsolete by this version. Appendix J includes new entities which have not received sufficient implementation testing for inclusion in the main body of the Specification. In addition, a List of References, a Glossary, an Index of Topics, and a Numerical Index of Entities are included.

2. Data Form

2.1 General

Two different forms are defined in this Specification to represent IGES data. These forms 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).

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 (not always present)	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 six types of constants: integer (or fixed point), real (or floating point), string, pointer, language statement, and logical. Regardless of whether the constants appear in a fixed or free format, certain rules apply to their formation, interpretation, and display as text:

- 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.
- Numeric constants shall not contain embedded commas.
- 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).

  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).
- 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).
- 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
```

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.

The following rules and examples apply to real constants as parameter data or as processed for text display.

- 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        -1.3E-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).

2.2.2.6 Logical Constants. A logical constant has only two values. These values are specified in the current FORTRAN standard [ANSI78]. In IGES files, where logical constants are expected, the unsigned integer 0 will denote the logical value .FALSE. and the unsigned integer 1 will denote the value .TRUE. .

The form of a logical constant is an unsigned integer consisting of a single digit. The digits 2 through 9 may not be used as logical constants. The digit is interpreted as a logical value.

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):

- The parameter delimiter (Global Parameter 1—defaulting to a comma) is used to separate parameters.
- The record delimiter (Global Parameter 2—defaulting to a semicolon) is used to terminate the record (*i.e.*, to terminate a list of parameters).
- When two parameter delimiters, or a parameter and record delimiter, appear adjacent to each other, or are separated by only blanks, the delimited 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.
- 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).
- 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.
- The parameter delimiter and record delimiter characters do not maintain their special significance when included within a string constant.
- A numeric constant, including its trailing delimiter, cannot extend across a line boundary.

2.2.3.1 Parameter and Record Delimiter Combinations. The following ASCII characters are prohibited from being used as either Global Parameter 1 (Parameter Delimiter) or Global Parameter 2 (Record Delimiter) because they will cause parsing difficulties in the post-processor.

Name	Octal Range
The Control Symbols	0...37, 177
The Space Character	40
The Digits 0 through 9	60...71
The Characters + - .	53, 55, 56
The Letters D E H	104, 105, 110

Only four forms of legitimate syntax are allowed for the first characters of the Global Section defining the two delimiters. They are (where α and β represent ASCII characters):

Form	Interpretation	
	Parameter Delimiter Character	Record Delimiter Character
1. ,,	,	;
2. 1Hα1H$\beta\alpha$	α	β
3. 1H$\alpha\alpha\alpha$	α	;
4. ,1Hβ,	,	β

2.2.4 File Structure. The file contains six subsections which must appear in order as follows:

a. Flag Section (Not always present)
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 (see Section 2.4.2.1) or in the Compressed ASCII Format (see Section 2.3.1).

2.2.4.1 Start Section. The Start Section of the file is designed to provide a human-readable prologue to the file. There must be at least one start record. All records in the section shall have the letter S in Column 73 and a sequence number in Column 74 through 80 (see Section 2.2.1). The information in Columns 1 through 72 need not be formatted in any special way except that the ASCII character set shall be used. An example of a Start Section is shown in Figure 2.

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 Section 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 Section 2.2.3. As implied in Section 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 1 and the paragraphs that follow. Unless explicitly stated, no defaults are provided.

1	72	73 80
This section is a human readable prologue to the file. It can contain an arbitrary number of lines ⋮ using ASCII characters in columns 1-72.		S0000001 S0000002 S0000003 ⋮ S000000N

Figure 2. Format of the Start section in the ASCII Form

Table 1. Parameters in the Global Section

Index	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 binary 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. Eleven values of the unit flag have been defined. (see Section 2.2.4.2.14).
16	Integer	Maximum number of line weight gradations (1–32768). Refer to the Directory Entry Parameter 12 (see Section 2.2.4.3) for use of this parameter.
17	Real	Size of maximum line width in units. Refer to the Directory Entry Parameter 12 (see Section 2.2.4.3) for use of this parameter.
18	String	Date & time of file generation `13HYYMMDD.HHNNSS` where: `YY` is last 2 digits of year `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 $\|X\|, \|Y\|, \|Z\| \leq 1000$.)
21	String	Name of author
22	String	Author's organization
23	Integer	Integer value corresponding to the version of IGES used to create this file. (See Section 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 Section 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 Section 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 Section 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 software which generated this file. It includes the complete vendor's name, the name by which the system is marketed, and the product ID/version number and/or release date of software.

2.2.4.2.6 Preprocessor Version. This parameter identifies the version and/or release date of the preprocessor which created this file.

2.2.4.2.7 Number of Binary 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 floating point representation (see [IEEE85]) 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:

Value	Measuring System
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 Units. A string constant naming the model units in the system:

Constant	Model Units
2HIN or 4HINCH	Inches
2HMM	Millimeters
2HFT	Feet
2HMI	Miles
1HM	Meters
2HKM	Kilometers
3HMIL	Mils
2HUM	Microns
2HCM	Centimeters
3HUIN	Microinches

When the unit flag is given a value of 3, the string constant naming the desired unit should conform to [MIL12] or [IEEE260].

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

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

2.2.4.2.18 Date and Time of File Generation. This is a time stamp of when the file was generated.

2.2.4.2.19 Minimum User-Intended Resolution. 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 Approximate Maximum Coordinate Value. 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 creation of the data contained in this file.

2.2.4.2.22 Author's 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 values in the table below are valid for this Specification version, and will be added to for each successive version or ANSI Specification. If no valid integer number is entered in this field, the default value of 3 (corresponding to Version 2.0 [NBS83]) should be assumed. See the List of References for a full citation for each version.

Value	Version	Reference
1	1.0	[NBS80]
2	ANSI Y14.26M - 1981	[ANSI81]
3	2.0	[NBS83]
4	3.0	[NBS86]
5	ANSI Y14.26M - 1987	[ANSI88]
6	4.0	This document

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

Code	Drafting Standard	
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 character 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 or a pointer to an entity containing 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.

Figure 3 gives an abbreviated listing of the fields making up the directory entry for each entity. Table 2 and the following paragraphs describe each directory entry field. Note that Table 2 contains references to entities and form numbers defined in Appendix J.

1 8	9 16	17 24	25 32	33 40	41 48	49 56	57 64	65 72	73 80
(1) Entity Type Number #	(2) Parameter Data ⇒	(3) Structure #,⇒	(4) Line Font Pattern #,⇒	(5) Level #,⇒	(6) View 0,⇒	(7) Transformation Matrix 0,⇒	(8) Label Display Assoc. 0,⇒	(9) Status Number #	(10) Sequence Number D #
(11) Entity Type Number #	(12) Line Weight Number #	(13) Color Number #,⇒	(14) Parameter Line Count #	(15) Form Number #	(16) Reserved	(17) Reserved	(18) Entity Label	(19) Entity Subscript Number #	(20) Sequence Number D # + 1

Nomenclature:

(n) - Field number n
\# - Integer
⇒ - Pointer
#,⇒ - Integer or pointer (pointer has negative sign)
0,⇒ - Zero or pointer

Figure 3. Format of the Directory Entry (DE) Section in the ASCII Form

2.2 ASCII FORM

Table 2. Directory Entry (DE) Section

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	Negated 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 Section 2.2.4.3.3)
4	Line Font Pattern	Line font pattern or negated pointer to the directory entry of a Line Font Definition Entity (Type 304).
5	Level	Number of the level upon which the entity resides, or a negated pointer to the directory entry of a Property Entity (Type 406, Form 1) which contains a list of levels upon which the entity resides.
6	View	Pointer to the directory entry of a View Entity (Type 410), or pointer to a Views Visible Associativity Instance (Type 402, Form 3 or 4), or integer zero (default).
7	Transformation Matrix	Pointer to the directory entry of a Transformation Matrix Entity (Type 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 (Type 402, Form 5). The value of zero indicates no label display associativity.
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 negated pointer to the directory entry of a Color Definition Entity (Type 314).
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	Reserved for future use	
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, centerline, and dotted 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.

Value	Pattern
1	Solid
2	Dashed
3	Phantom
4	Centerline
5	Dotted

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. (Type 402, 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. A zero value indicates no Label Display Associativity is specified.

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 both 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. The additional pointers as defined in Section 2.2.4.4.2 are ignored for the purposes of this definition. 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.

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:

- A Leader Line Entity pointed to by a Linear Dimension Entity.
- A Circular Arc Entity pointed to by a Plane Entity.
- A Circular Arc Entity pointed to by a Composite Curve Entity.

- A Composite Curve Entity pointed to by a Subfigure Definition Entity (note that the subfigure definition would NOT point to the constituent entities of the composite curve).

Consider the following example:

- Entity A is physically subordinate to entity B.
- Entity A points to a Transformation Matrix M1.
- Transformation Matrix M1 points to a Transformation Matrix M2.
- Entity B is subordinate to a Subfigure Definition Entity C.
- Entity B points to a Transformation Matrix M3.
- Entity C is instanced in a Subfigure Instance D.
- The parameter data of entity D specifies its scale factor as Sd and position as (Xd,Yd,Zd).
- Entity D points to a Transformation Matrix M4.
- Entity D points to a View Entity E.
- The view scale factor defined in the parameter data of entity E is Se.
- Entity E occurs within a drawing F at drawing coordinates (Fx,Fy).
- 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 preceding step are transformed by M2.
3. The coordinates resulting from the preceding step are transformed by M3.
4. The coordinates resulting from the preceding step are scaled by Sd.
5. The coordinates resulting from the preceding step are transformed by M4.
6. The coordinates resulting from the preceding 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 preceding step are transformed by M5.
8. The coordinates resulting from the preceding step are scaled by the scale factor Se.
9. The coordinates resulting from the preceding 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 occurring in a subfigure definition and pointed to by a Group Associativity Entity.

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 (Type 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 Node, Connect Point, and Point when their primary use is as a reference.

05: 2-D 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 3.14.2, $T_3 = R_{13} = R_{31} = R_{32} = R_{23} = 0.0$ 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 (Type 406, Form 10) (see Section 4.3.7.3.9) 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 either is a color number 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. See Section 2.2.4.3.10.

2.2.4.3.21 Table 3 summarizes DE Field Requirements. 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. Zero 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 is not a valid setting for this field; therefore no default is allowed.

Table 3. Entity DE Field Requirements

Entity types: 100, 104, 106 (Forms 1–3, 11–13, 63), 108, 110, 112, 114, 116, 118, 120, 122, 126, 128, 130, 140, 142, 144

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font			X	Ignored for 106 (Forms 1,2,3), 116
5. Level		0		
6. View pointer		0		0 = display in all views
7. Matrix pointer		0		
8. Label pointer		0		
9a. Blank Status		00		
9b. Subord. Sw.		00		
9c. Entity use		00		00,01,02, and 05 are only valid values; 01 implies entity used for annotation
9d. Hierarchy	X			See Note 1
10. Sequence No.			X	
11. Type			X	
12. Line weight		0		0 = use the receiving system's default line weight
13. Color		0		0 = use any color
14. PD count			X	
15. Form		0		Non-zero reqd. for 106
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Notes:

1. Hierarchy is valid for entity types 118, 120, 122, 130, 140, 144.

Table 3. Entity DE Field Requirements (continued)

Entity Type: 102

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font			X	See Note 1
5. Level		0		See Note 1
6. View pointer		0		See Note 1
7. Matrix pointer		0		
8. Label pointer		0		
9a. Blank Status		00		See Note 1
9b. Subord. Sw.		00		
9c. Entity use		00		
9d. Hierarchy		00		
10. Sequence No.			X	
11. Type			X	
12. Line weight		0		See Note 1
13. Color		0		See Note 1
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Notes:

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

Table 3. Entity DE Field Requirements (continued)

Entity Type: 124 (Forms 0, 1)

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer		0		
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.	X			
9c. Entity use	X			
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Type: 124 (Forms 10–12)

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer		0		
8. Label pointer	X			
9a. Blank Status		00		
9b. Subord. Sw.		00		
9c. Entity use	X			
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color		0		
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Types: 200 Series except 212 and 214

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font			X	
5. Level		0		
6. View pointer		0		
7. Matrix pointer		0		
8. Label pointer		0		
9a. Blank Status		00		
9b. Subord. Sw.		00		
9c. Entity use			X	Must be 01
9d. Hierarchy		00		
10. Sequence No.			X	
11. Type			X	
12. Line weight		0		
13. Color		0		
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Types: 212, 214, 106 (Forms 20, 21, 31–38, 40), and 312

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font			X	Ignored for 106, 212
5. Level		0		
6. View pointer		0		
7. Matrix pointer		0		
8. Label pointer		0		
9a. Blank Status		00		
9b. Subord. Sw.		00		For 312, cannot be 00
9c. Entity use			X	Must be 01
9d. Hierarchy		00		
10. Sequence No.			X	
11. Type			X	
12. Line weight		0		
13. Color		0		
14. PD count			X	
15. Form		0		Non-zero reqd for 106
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Types: 302, 306, 310

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer	X			
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Type: 304

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font			X	
5. Level	X			
6. View pointer	X			
7. Matrix pointer		0		
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form			X	
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Types: 308, 320

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font			X	See Note 1
5. Level	X			See Note 1
6. View pointer	X			
7. Matrix pointer		0		
8. Label pointer		0		
9a. Blank Status	X			
9b. Subord. Sw.		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy		00		
10. Sequence No.			X	
11. Type			X	
12. Line weight		0		See Note 1
13. Color		0		See Note 1
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Notes:

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

Table 3. Entity DE Field Requirements (continued)

Entity Type: 314

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer	X			
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		Must be 00
9c. Entity use			X	Must be 02
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color		0		
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Type: 402

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure			X	Ignored for form numbers to 5000
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer	X			
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		00 required for Forms 3 and 4
9c. Entity use				03 required for Form 18
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form			X	
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Type: 404

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer	X			
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		Must be independent
9c. Entity use			X	
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Type: 406

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level		0		
6. View pointer	X			
7. Matrix pointer	X			
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		See Note 1
9c. Entity use	X			
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form			X	
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Notes:

1. In most cases, the setting for this field should be 02. The Specification currently states a setting of 00 implies the Level Function Property should be applied to all entities occurring at the same level (see Section 4.3.7.3.3).

Table 3. Entity DE Field Requirements (continued)

Entity Types: 408, 412, 414, 420

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level		0		
6. View pointer		0		
7. Matrix pointer		0		
8. Label pointer		0		
9a. Blank Status		00		
9b. Subord. Sw.		00		
9c. Entity use		00		
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			X	

Table 3. Entity DE Field Requirements (continued)

Entity Type: 410

ATTRIBUTE	IGNORE	VALID: May be defaulted to	VALID: Non-zero required	COMMENT
1. Type			X	
2. PD pointer			X	
3. Structure	X			
4. Line font	X			
5. Level	X			
6. View pointer	X			
7. Matrix pointer		0		
8. Label pointer	X			
9a. Blank Status	X			
9b. Subord. Sw.		00		
9c. Entity use			X	
9d. Hierarchy	X			
10. Sequence No.			X	
11. Type			X	
12. Line weight	X			
13. Color	X			
14. PD count			X	
15. Form		0		
16. Reserved	X			
17. Reserved	X			
18. Label		Blank		
19. Subscript		0		
20. Sequence No.			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 Section 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 Section 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 Instance, General Note, and/or Text Template Entities. 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 (Type 212) 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 (Type 312) 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.

The second group of parameters may contain pointers to one or more properties or attribute tables.

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	NP	Integer	Number of pointers to properties or attribute tables
NV+NA+2	DE	Pointer	Property or attribute table 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 including them in the count of parameter lines for the entity (DE Field 14). Figure 4 shows a Parameter Data Section.

1 — 64	66 — 72	73 — 80
Entity type number followed by parameter delimiter followed by parameters separated by parameter delimiters	DE Pointer	P0000001
Parameters separated by parameter delimiters followed by record delimiter	DE Pointer	P0000002
⋮	⋮	⋮

Note: The DE pointer is the sequence number of the first directory entry line for this entity

Figure 4. Format of the Parameter Data (PD) section in the ASCII Form

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. Column 73 of the line contains a "T", 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 5.

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

1 — 8	9 — 16	17 — 24	25 — 32	33 — 40	41 — 48	49 — 56	57 — 64	65 — 72	73 — 80
S0000020	G0000003	D0000500	P0000261	Not Used					T0000001

Figure 5. Format of the Terminate section in the ASCII Form

2.3 Compressed ASCII Format

The format described here is intended to serve as an alternative to the fixed line length ASCII Format when the size of a file is a problem. The Compressed ASCII Format is intended to be simply converted to and from the fixed line length ASCII Format. An example of software to perform such conversions is presented in Appendix H.

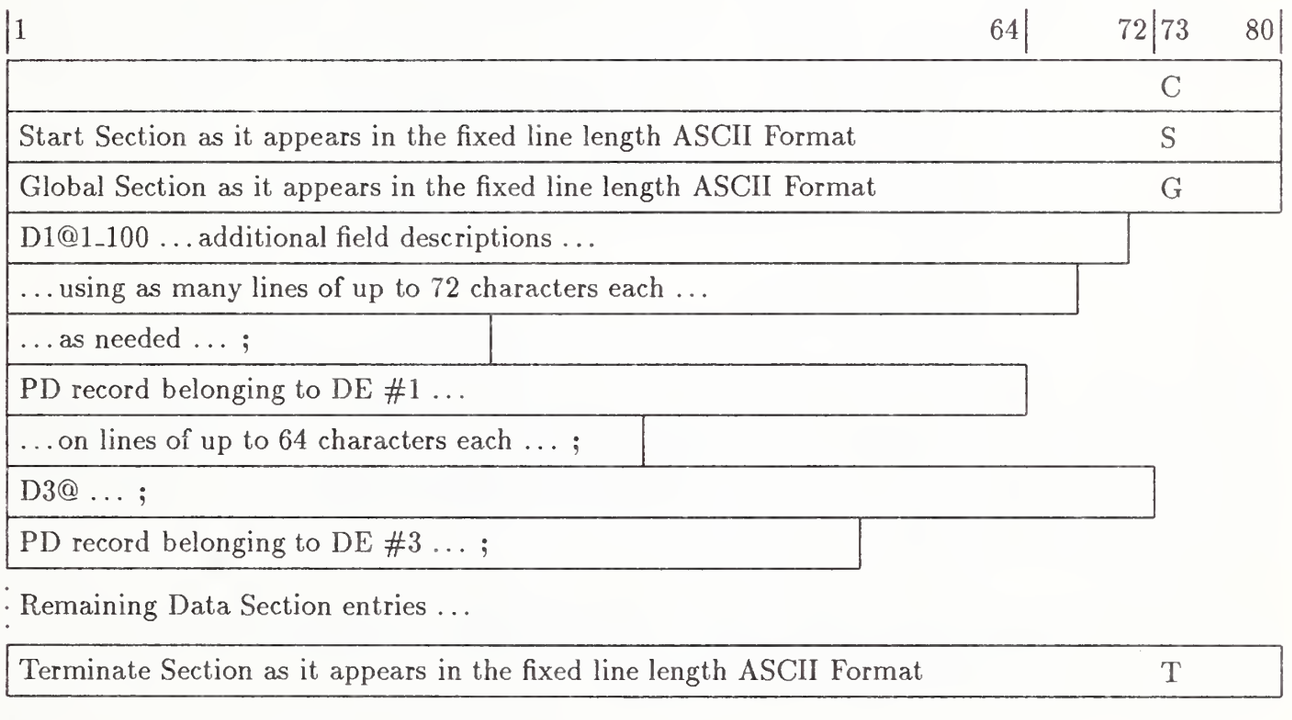
2.3.1 File Structure. A single line Flag Section shall precede the Start Section with the character "C" in character position 73 to identify the file as the Compressed ASCII Format. The Start, Global and Terminate Sections remain the same as those for the ASCII Format, while the DE and PD Sections are combined into a single Data Section. The PD portion of the Data Section is written in free format in a manner similar to that of 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 6. 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, because they are either redundant or meaningless in the Compressed ASCII Format. 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 "@".

DE field values will be specified only when they change. Thus, a DE field retains its value from entity to entity unless it is changed. 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 ";").

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



Note: Default record delimiter assumed

Figure 6. General file structure in the Compressed ASCII Format

2.4 Binary Form

The formats defined in Section 2.2 and Section 2.3, referred to collectively as the ASCII Form, character oriented record lines. This section describes a bit stream binary representation of data which may be used as an alternative format to the ASCII Form. The binary representation of data, including ASCII characters, is organized in multiples of 8-bit bytes.

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:

- Integer numbers
- Real numbers
- String constants
- Pointers
- 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.

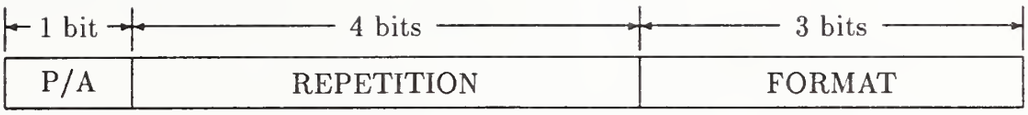
The format of the control byte is shown in Figure 7.

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 8.

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).

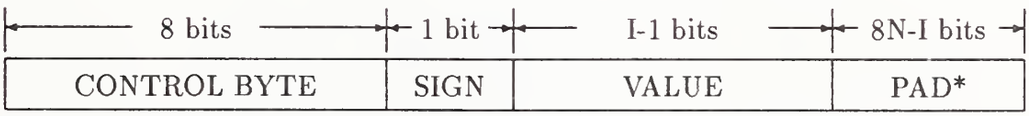


P/A = 0 If only the first of a set of repeated values is physically present
= 1 If all expected values are physically present

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 7. Format of the Control Byte Used in the Binary Form



*The PAD of zeroes from 1 to 7 bits is included only if the length I of the integer number is not a multiple of 8 bits

Figure 8. Format of an Integer Number in the Binary Form

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 9.

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 structure of the string constant is shown in Figure 10.

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 to indicate a default value.

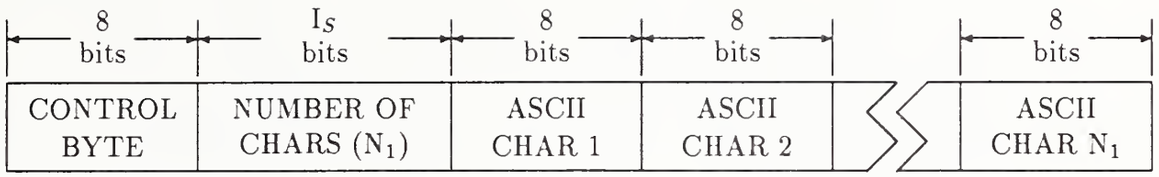
2.4.1.5 Language Constants. Language constants are the string constants of the Macro Definition Entity which, in the ASCII Form, are not preceded by nH and are terminated with a record delimiter. In the Binary Form, the format of language constants will be identical to string constants. Each language constant (Macro Statement) will be an individual string constant.

8 bits	1 bit	NX bits	NF bits	8N-(NX+NF+1) bits
CONTROL BYTE	SIGN	EXPONENT	BINARY FRACTION	PAD*

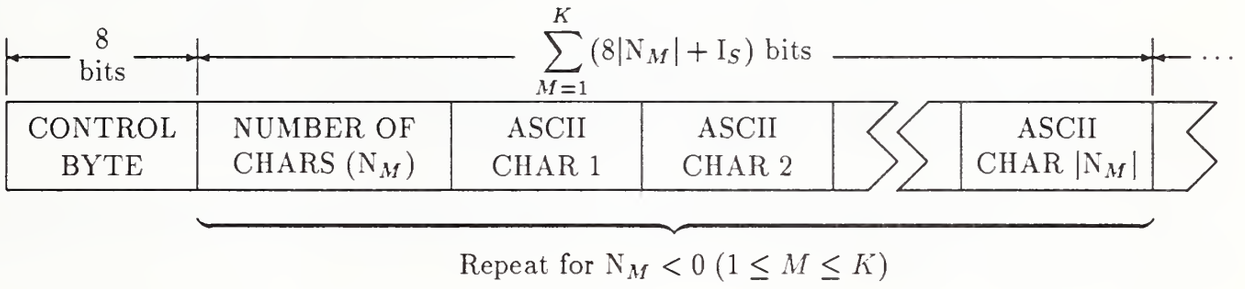
*The PAD of zeroes from 1 to 7 bits is included only if the length NX+NF+1 of the floating point number is not a multiple of 8 bits

Figure 9. Format of a Real Number in the Binary Form

For $N_1 > 0$



For $N_1 < 0, \ldots, N_K < 0$, $N_R > 0$



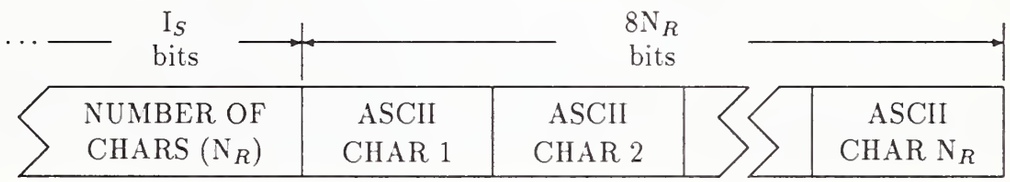


Figure 10. Structure of a String Constant in the Binary Form

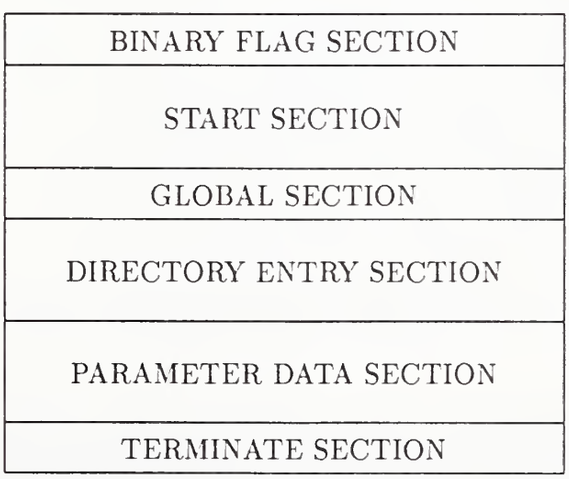


Figure 11. General File Structure in the Binary Form

2.4.2 File Structure. The general file structure is shown in Figure 11 and comprises the following six sections:

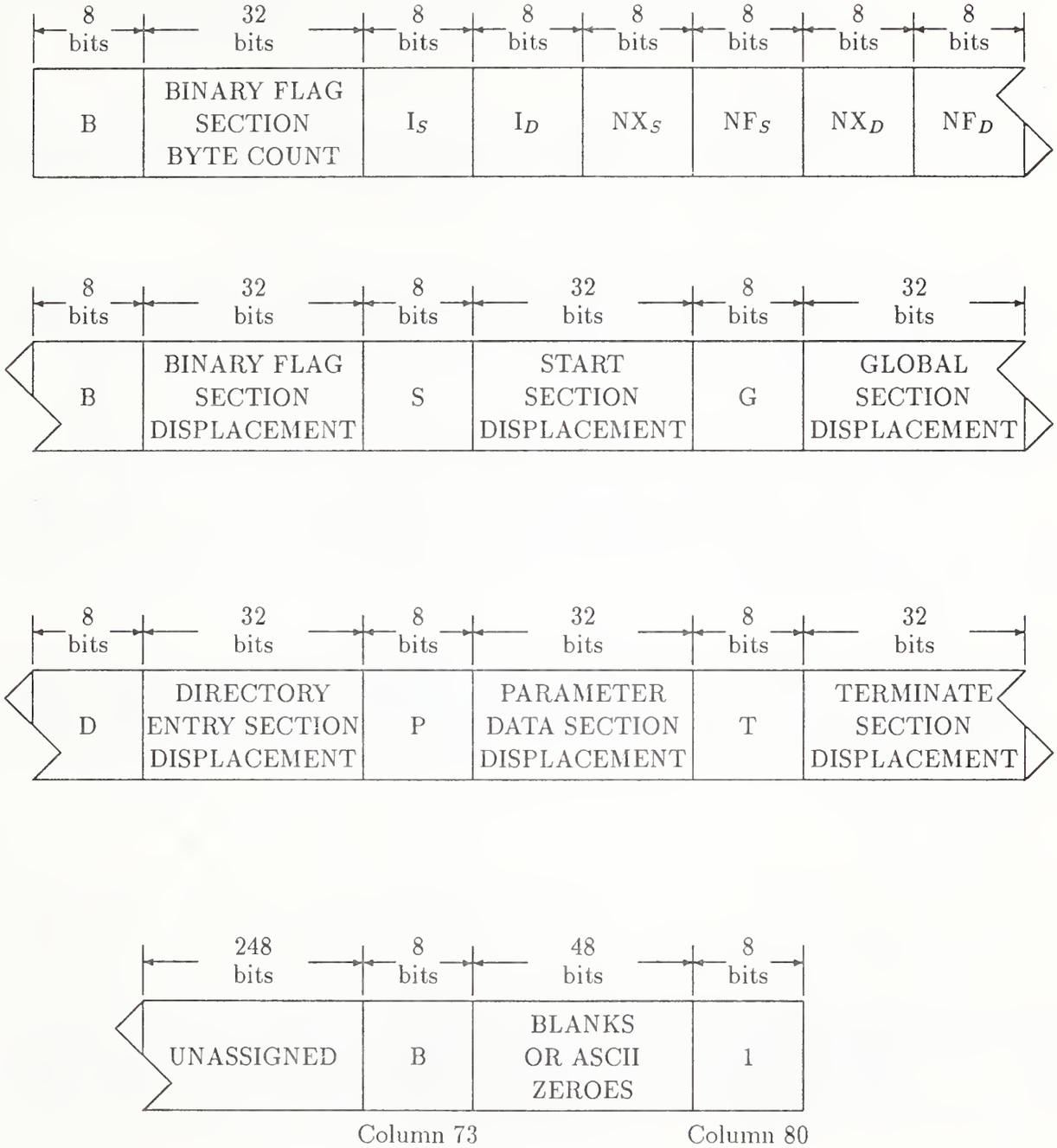
- Binary Flag Section
- Start Section
- Global Section
- Directory Entry Section
- Parameter Data Section
- Terminate Section

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 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 Flag Section. The format of the Binary Flag Section is shown in Figure 12. The Binary Flag Section contains a letter code indicating that the file is in Binary Form and also contains information required by a postprocessor to decode the file. (In previous versions of this Specification, this section was called the Binary Information Section.) The Binary Flag Section comprises the following data items, all of which are integers unless otherwise specified:

- Binary Flag Section identifier consisting of the ASCII letter B.
- Binary Flag Section byte count. This byte count (a 32 bit unsigned integer) 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.
- Length I_s of single length integer primitives.
- Length I_d of double length integer primitives.
- Length NX_s of exponent of single precision real primitives.
- Length NF_s of binary fraction of single precision real primitives.
- Length NX_d of exponent of double precision real primitives.
- Length NF_d of binary fraction of double precision real primitives.
- ASCII letter B.
- Binary Flag Section displacement. This is the byte count of the total length of the Binary Flag Section including all null padding characters. This length is the actual length from the initial B of the Binary Flag Section up to but not including the S of the Start Section.
- ASCII letter S.



NOTE: No fields in the Binary Flag Section have control bytes

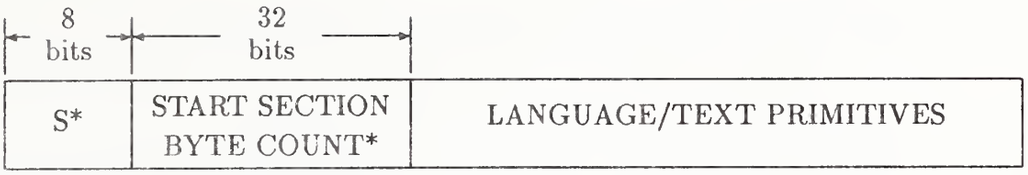
Figure 12. Format of the Binary Flag Section in the Binary Form

- 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 Global Section.
- ASCII letter G.
- 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.
- ASCII letter D.
- Directory Entry Section displacement. This is the byte count of the total length of the Directory Entry 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.
- ASCII letter P.
- 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.
- ASCII letter T.
- 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.
- 31 unassigned bytes.
- ASCII letter B.
- 6 ASCII blanks or zeroes.
- 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 Flag 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 compressed ASCII if a C). If it contains a B, it is binary.

2.4.2.2 Start Section. The format of the start section is shown in Figure 13. It comprises the following data items:

- A Start Section identifier consisting of the ASCII letter S.
- 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.



*These fields do not have control bytes

Figure 13. Format of the Start Section in the Binary Form

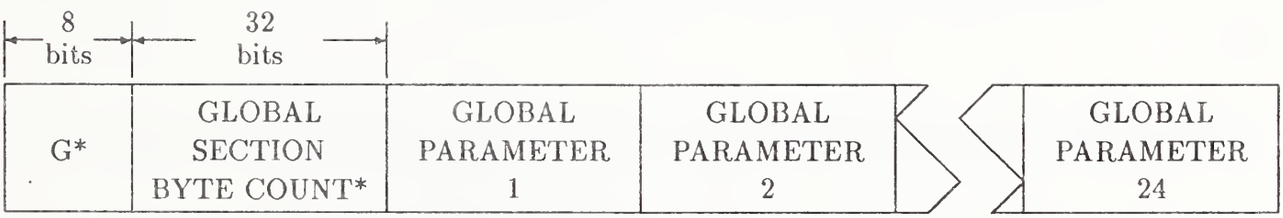
- One or more language or text primitives which are logically equivalent to Columns 1 through 72 of the ASCII Form. There is no required physical correspondence between the ASCII Form and language/text primitives. One language/text primitive may contain the equivalent of several complete or partial ASCII 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 shown in Figure 14. The Global Section comprises the following data items:

- Global Section identifier consisting of the ASCII letter G.
- 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.
- 24 global parameters.

Control bytes apply to only 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 Flag Section shall supersede these global parameters.



*These fields do not have control bytes

Figure 14. Format of the Global Section in the Binary Form

2.4.2.4 Directory Entry Section. The format of the Directory Entry Section is shown in Figure 15. The Directory Entry Section comprises the following data items:

- Directory Entry Section identifier consisting of the ASCII letter D.
- 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.
- 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 number
  - parameter data
  - structure
  - line font pattern
  - level
  - view
  - transformation matrix
  - label display associativity
  - status number
  - line weight number
  - color number
  - form number
  - reserved field 1
  - reserved field 2
  - entity label
  - entity subscript number

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 the 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 Data Section. The format of the Parameter Data Section is shown in Figure 16. The Parameter Data Section comprises the following data items:

- Parameter Data Section identifier consisting of the ASCII letter P.
- 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.

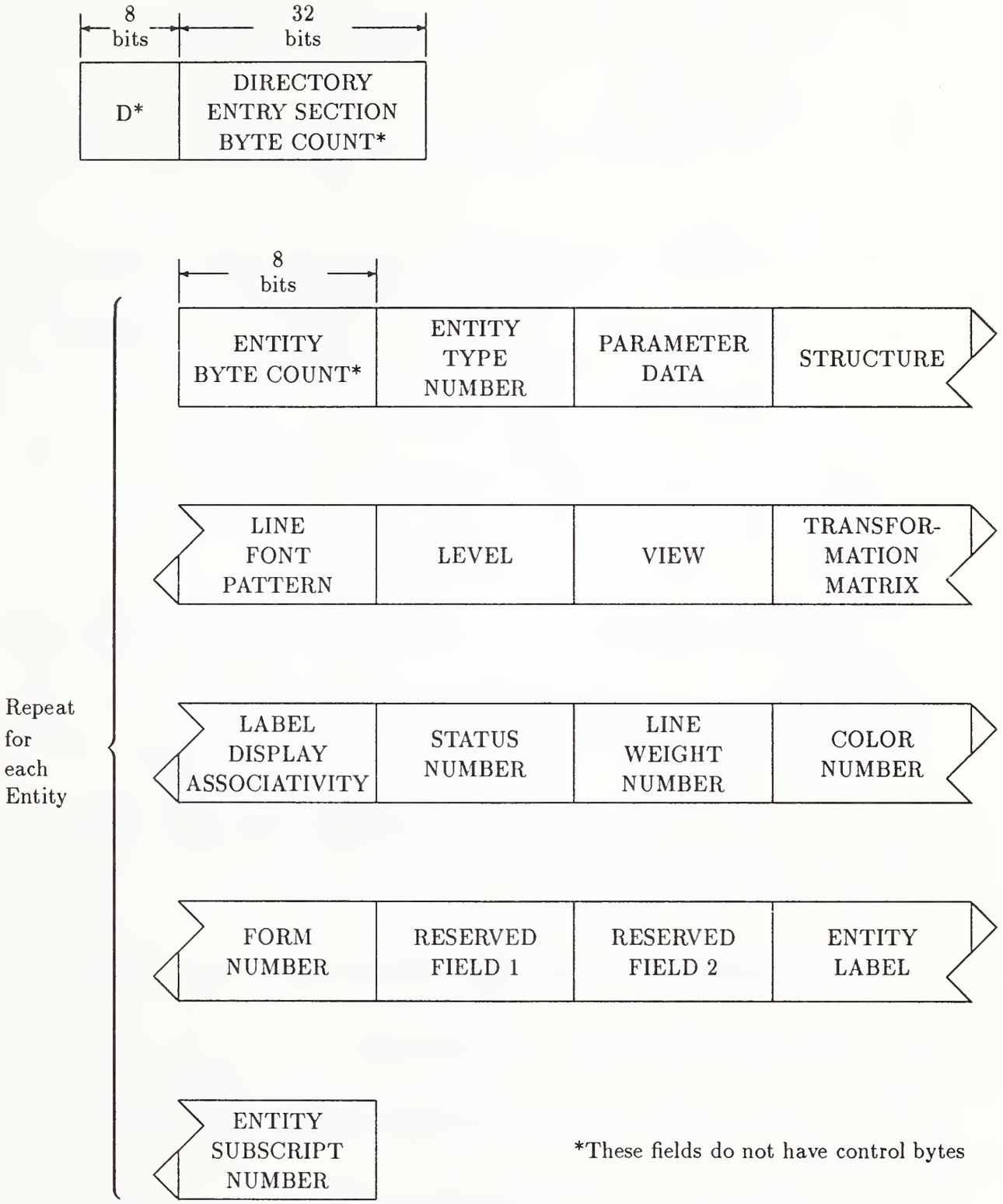


Figure 15. Format of the Directory Entry (DE) Section in the Binary Form

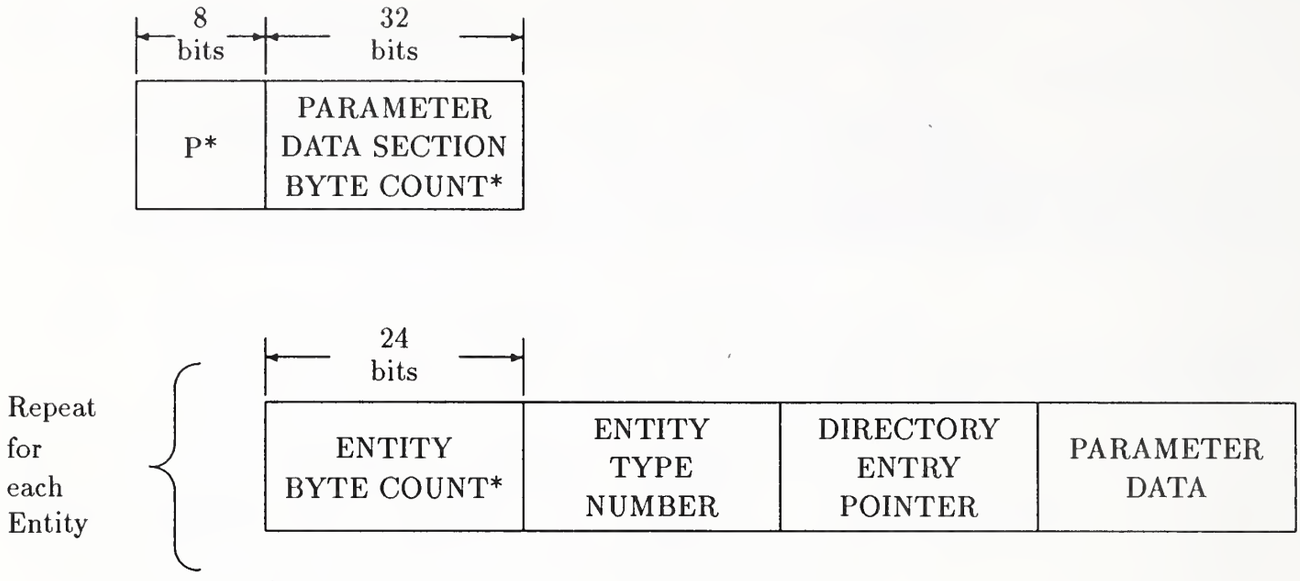


Figure 16. Format of the Parameter Data (PD) Section in the Binary Form

- For each Parameter Data entry, 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.

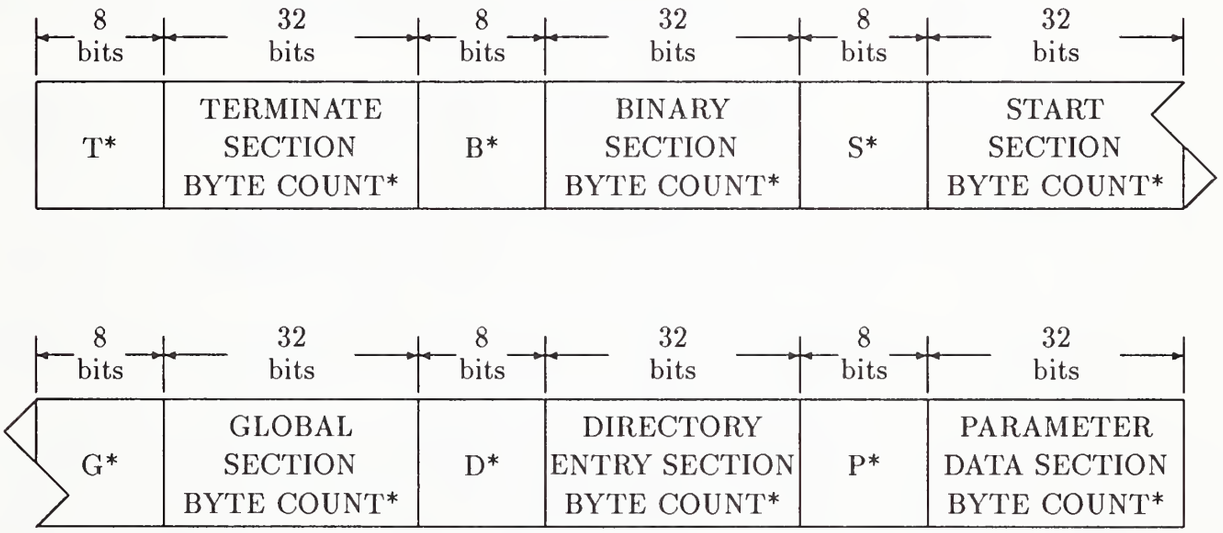
Control bytes apply only to the entity type, Directory Entry pointer and Parameter Data fields.

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

2.4.2.6 Terminate Section. The format of the Terminate Section is shown in Figure 17. The Terminate Section comprises the following data items:

- Terminate Section identifier consisting of the ASCII letter T
- 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.
- ASCII letter B.
- Binary Flag Section byte count, including the section identifier, and section byte count, but excluding any null padding characters.
- ASCII letter S.
- Start Section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.

- ASCII letter G.
- Global Section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.
- ASCII letter D.
- Directory Entry Section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.
- ASCII letter P.
- Parameter Data Section byte count, including the section identifier, section byte count, and all control bytes but excluding any null padding characters.



*These fields do not have control bytes

Figure 17. Format of the Terminate Section in the Binary Form

2.5 Specific File Structures

Individual entities of the file structure are described in Chapters 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. 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 (Type 308) and each placement of the collection is specified by a Singular Subfigure Instance Entity (Type 408). The Network Subfigure Definition (Type 320) and Instance (Type 420) Entity pair is similar in concept but has some special features to accommodate the notion of connect point in a network. (Section 2.5.2 provides additional information about network subfigures.) In other cases, a Rectangular Array (Type 412) or a Circular Array (Type 414) 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 18 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.

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).

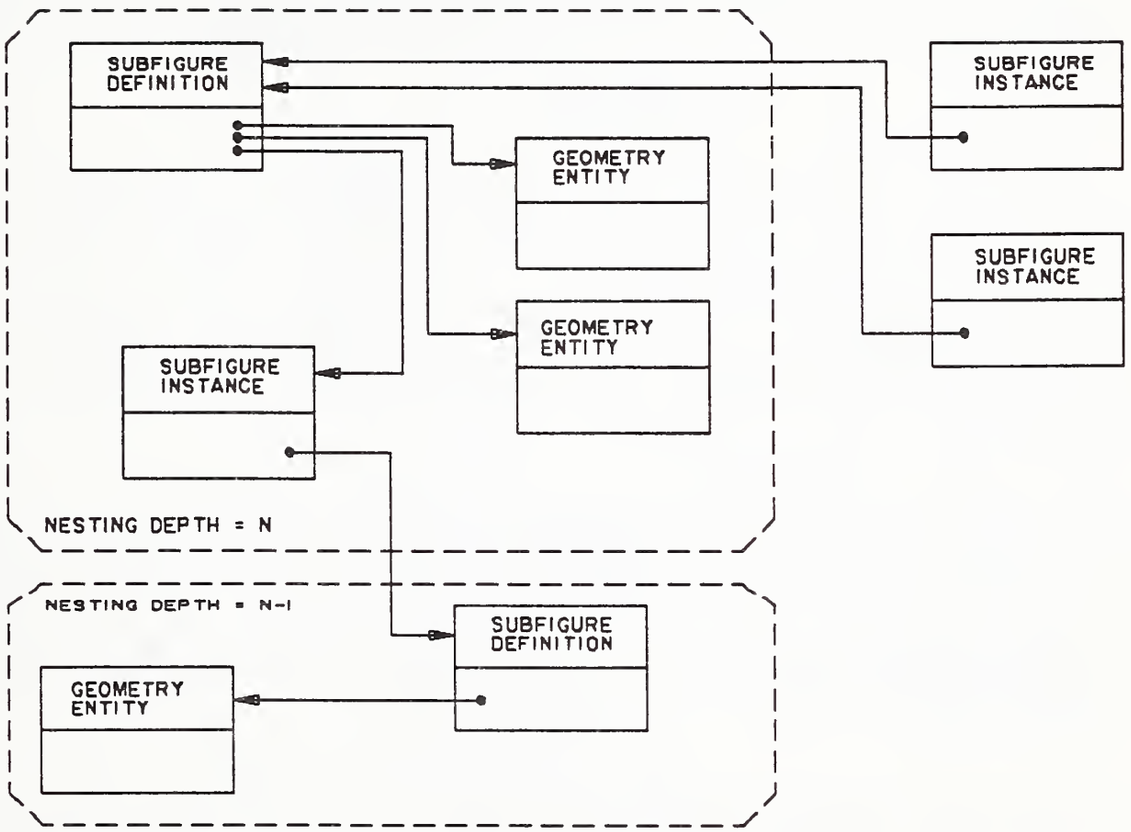


Figure 18. Subfigure Structures

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 (Type 132) 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 19 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 (Type 312) 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 (Type 402, Form 18) 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.

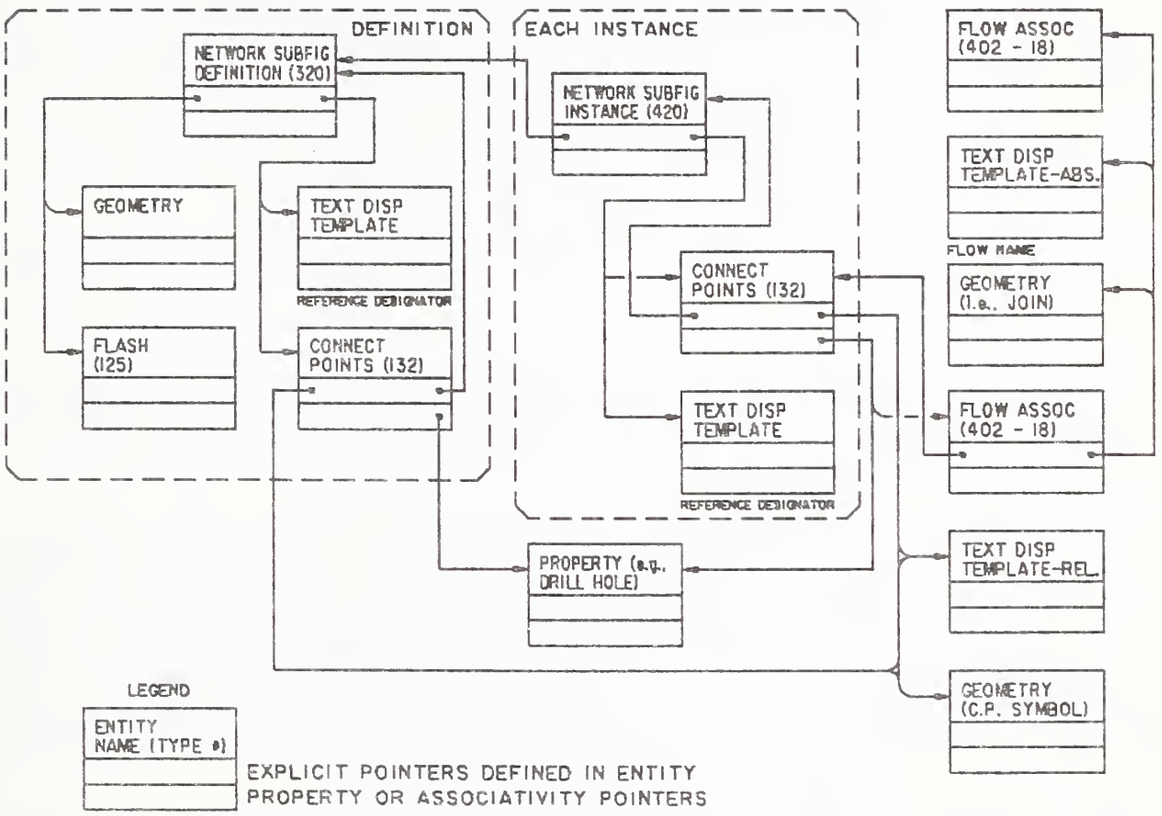


Figure 19. General Connectivity Pointer Diagram

2.5.3 Macro. 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 (Type 306) which are pointed to by Macro Instance Entities (Types 600–699 or 10000–99999) as shown in Figure 20. 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.

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 21). 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 21). 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). That External Reference Entity 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 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.

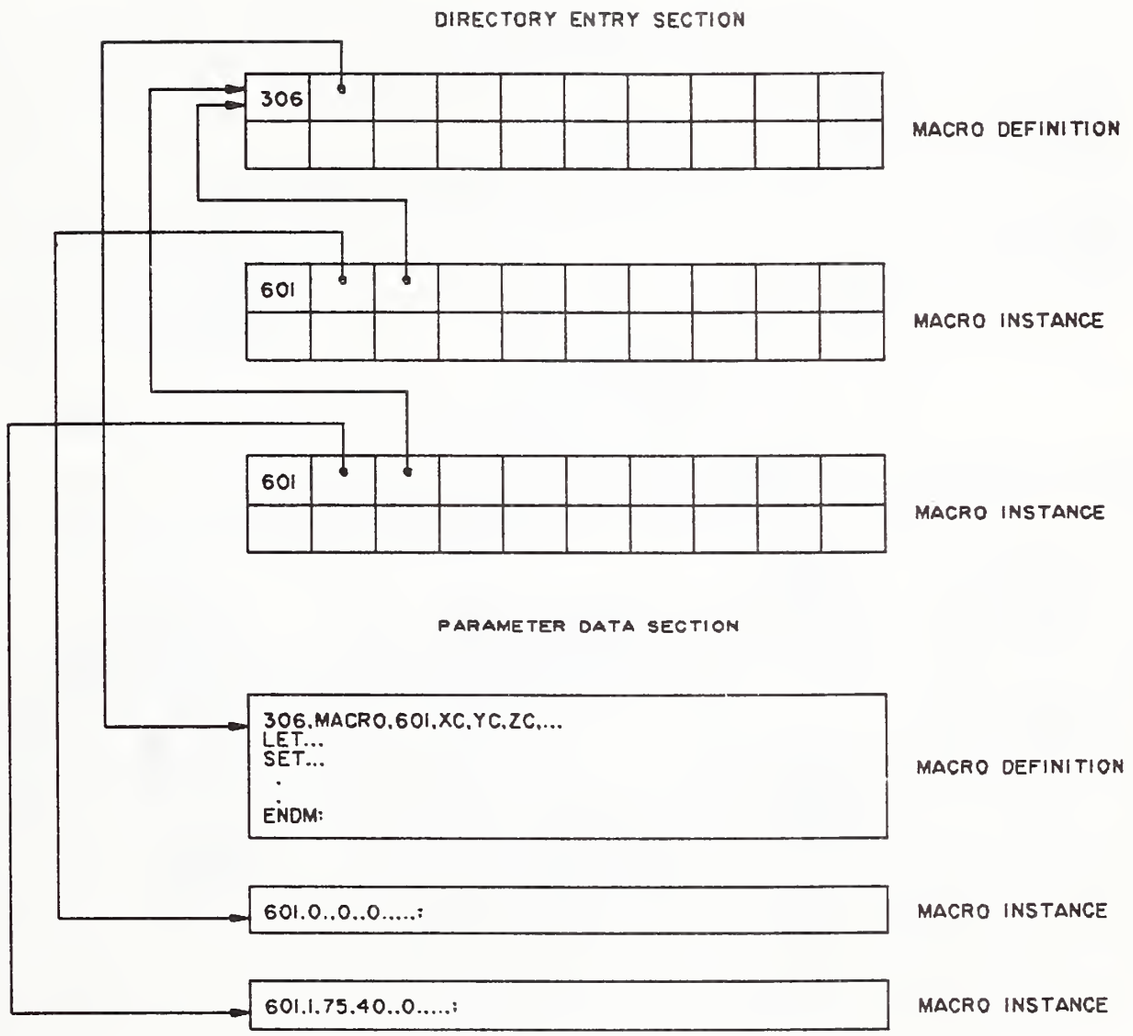


Figure 20. Macro Definition and Instance Structure

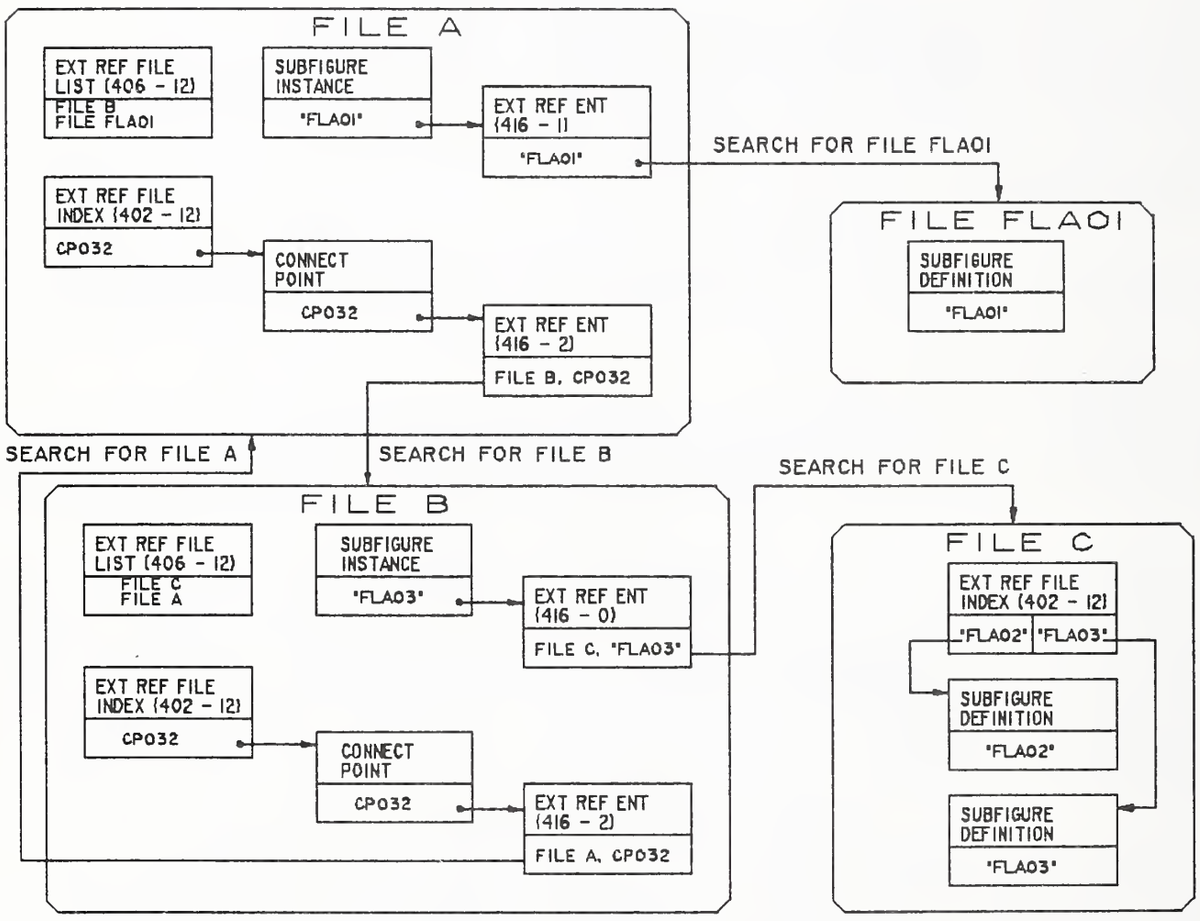


Figure 21. External Linkages

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 21).

A preprocessor implementor should use the external 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 3-D model space to the two-dimensional 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 22 and 23. The left side of Figure 22 illustrates the relationships between the entities that define the model's parametric attributes. The right side illustrates the addition of the analysis results. Figure 23 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 Entity (Type 136), Node Entity (Type 134), Load/Constraint Entity (Type 418), Tabular Data Property Entity (Type 406, Form 11), Nodal Results Entity (Type 146) and Element Results Entity (Type 148). The Nodal Results Entity is intended to supercede the Nodal Displacement and Rotation Entity (Type 138).

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 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.

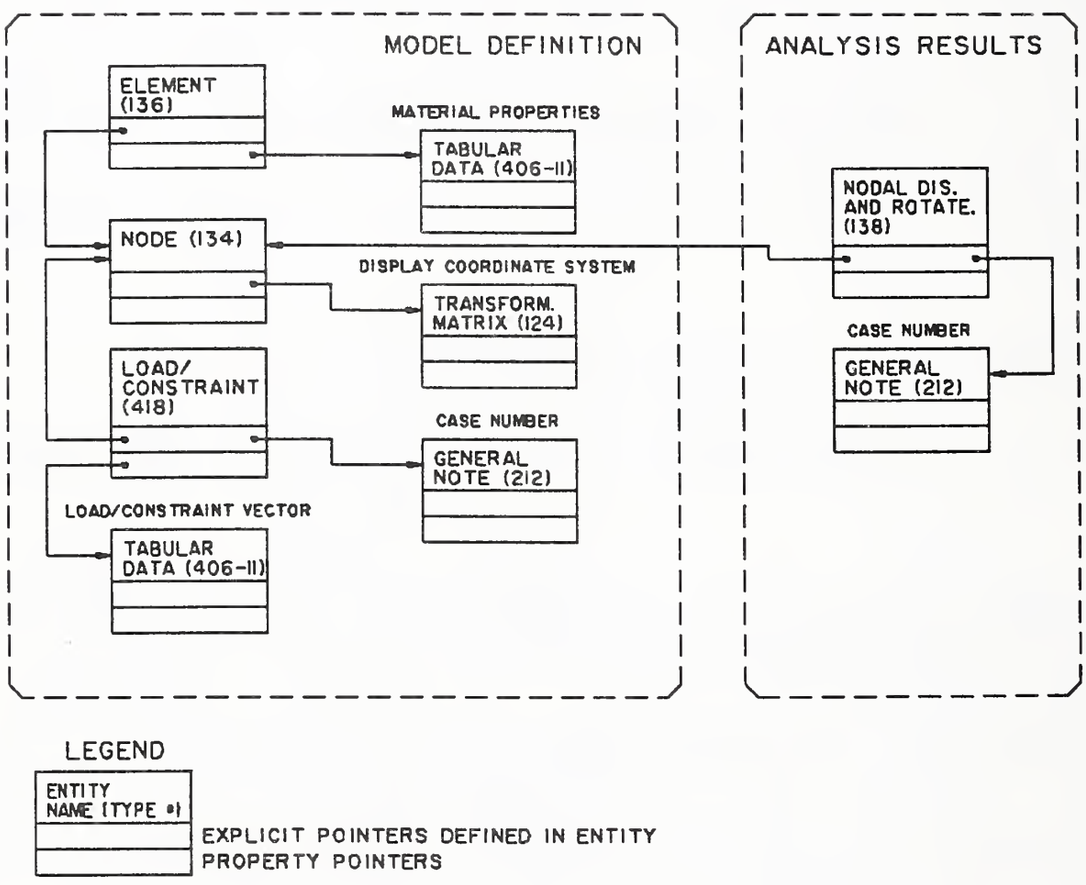


Figure 22. Finite Element Modeling File Structure

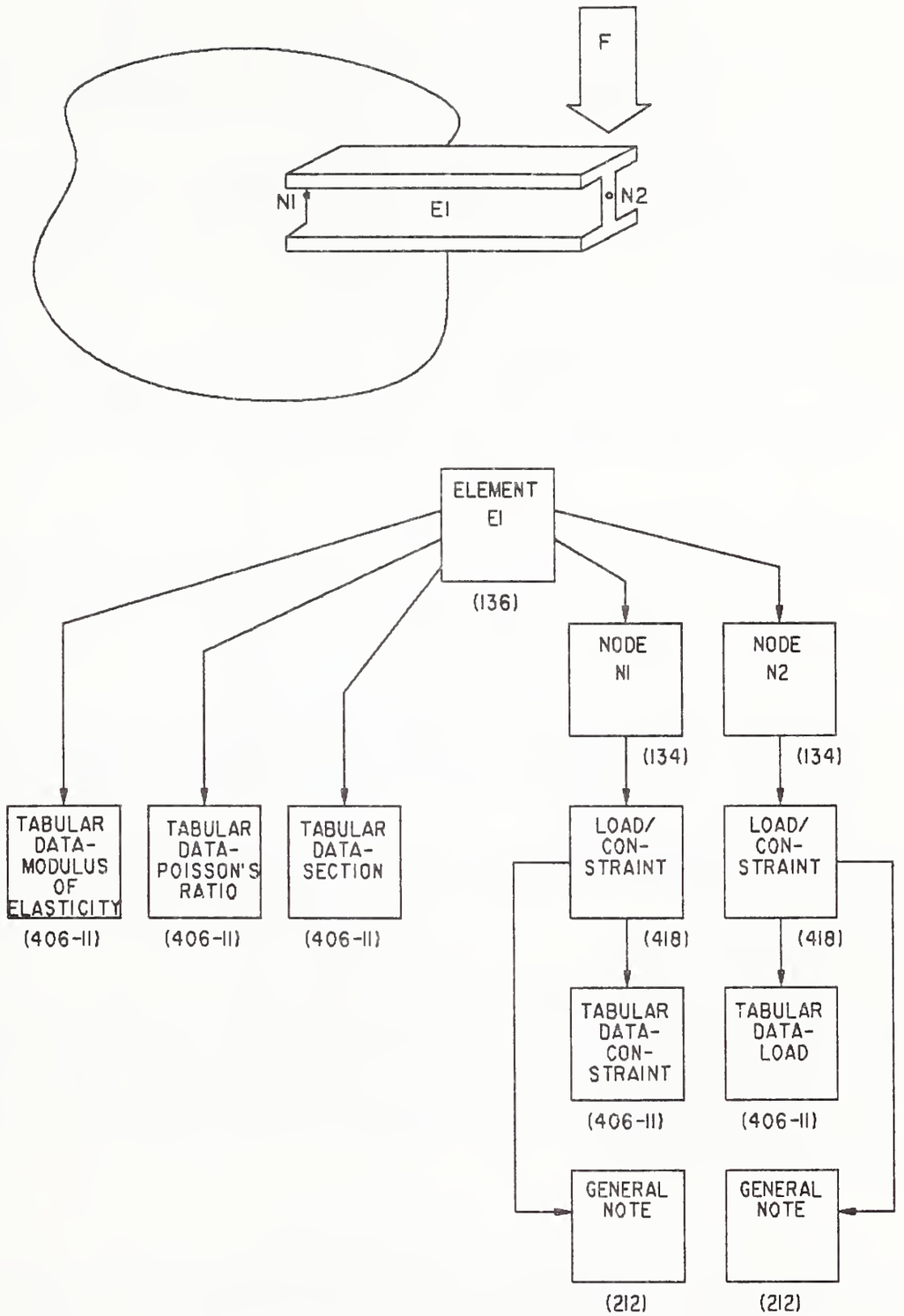


Figure 23. Finite Element Modeling Logical Structure

Tabular Data Property Entity (Type 406, Form 11) contains the material property data of the elements and the load/constraint data as required.

The Nodal Results Entity is used to communicate nodal finite element analysis results data. It contains analysis results at FEM nodes that are independent of the FEM elements that are attached to them. (The Element Results Entity should be used if the analysis results data are dependent on FEM elements.) It is intended to supercede the old Nodal Displacement and Rotation Entity, as it permits far greater flexibility in the transfer of nodal results.

The Element Results Entity is used to communicate FEM element results that vary within a FEM element. The data communicated may be results at various layers within the FEM element: at the FEM element/nodes, at the FEM centroid, at the FEM element gauss points, or any combination of these locations.

For example, consider the extrapolated stress values at the nodes of several quadratic, plane-stress FEM elements. There is no guarantee that the nodal values of stress will be identical for adjacent FEM elements at common nodes. There are at least as many possible FEM element result values as there are finite elements that contain common nodes in their topologies. These data are different from the results data expressed at the same node in the Nodal Results Entity.

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 24(a).

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 24(b).

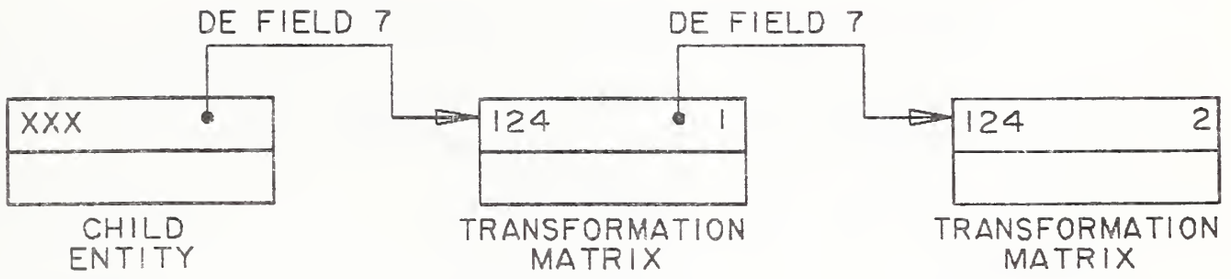
In the case illustrated by Figure 24(b) the points represented by entity XXX are operated upon by matrix 2 and from that point on are transformed like the points in entity YYY, using matrix 1.

A parent/child relationship between entities may also be created with a Single Parent Associativity Instance Entity (Type 402, Form 9).

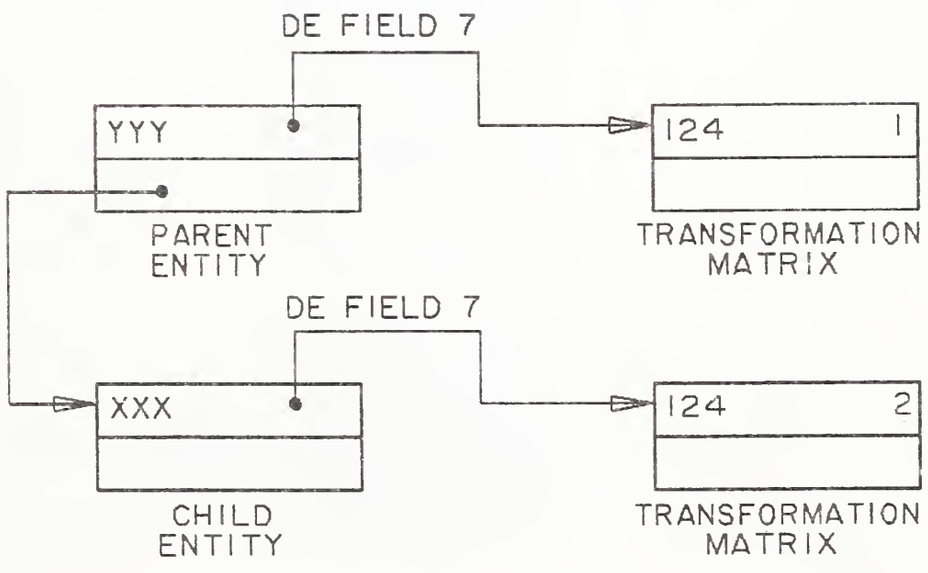
When the specific parent/child relationships shown in Table 4 occur the implicit relation rule shall apply. Each of the relationships in Table 4 ordinarily results in the subordinate entity switch of the child entity being set to 01 (physically dependent). The exception is the case in which a preprocessor 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).

2.5.8 Attribute Tables. An attribute table (see Appendix J) is a collection of attribute definitions and values in the form of a single row or table. The structure consists of an Attribute Table Definition Entity (Type 322), where each attribute is defined by a name, a data-type, and a count. The attribute values are either supplied immediately after the attribute definition, or "instanced" via the Attribute Table Instance Entity (Type 422). One or more Attribute Table Instance Entities may point to the Attribute Table Definition Entity using the third field of their Directory Entry.

Three types of Attribute Table Definition Entities and two types of Attribute Table Instance Entities are defined. The Attribute Table Definition Entity can have: (1) attribute definitions only, (2)



a) EXPLICIT CASE



b) PARENT-CHILD IMPLICIT CASE

Figure 24. Multiple Transformation Cases

attribute definitions followed immediately by the attribute values, or (3) attribute definitions followed by attribute values with each value followed by a Text Display Template. The Attribute Table Instance Entity can store: (1) a single row of attribute values, or (2) a table of rows of attribute values, stored in row-major order.

Table 4. Examples of 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

3. Geometry

3.1 General

This Chapter 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 Section 1.6.5, 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 Section 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 Geometry 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 Simple Closed Area Linear Path Witness Line Section 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
146	Nodal Results (see Appendix J)
148	Element Results (see Appendix J)
150	Block
152	Right Angular Wedge
154	Right Circular Cylinder
156	Right Circular Cone Frustum
158	Sphere
160	Torus
162	Solid of Revolution
164	Solid of Linear Extrusion
168	Ellipsoid
180	Boolean Tree
184	Solid Assembly
430	Solid Instance

3.2 Circular Arc Entity (Type 100)

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.

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.

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.

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

$$C(t) = (X_1 + R * \cos t,\ Y_1 + R * \sin t,\ ZT)$$
$$\text{for } t_2 \leq t \leq t_3$$

where, for $i = 2$ and 3,

$(i)\ R = \sqrt{(X_i - X_1)^2 + (Y_i - Y_1)^2}$

$(ii)\ t_i$ is such that $(R * \cos t_i, R * \sin t_i) = (X_i - X_1,\ Y_i - Y_1)$

and

$$0 \leq t_2 < 2 * \pi$$
$$0 \leq t_3 - t_2 \leq 2 * \pi$$

Examples of the Circular Arc Entity are shown in Figure 25. In Example 2 of Figure 25, 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 start point listed in the parameter data entry would be B, and the terminate point would be A.

3.2.1 Directory Data

Entity Type Number: 100

3.2.2 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 Section 2.2.4.4.2).

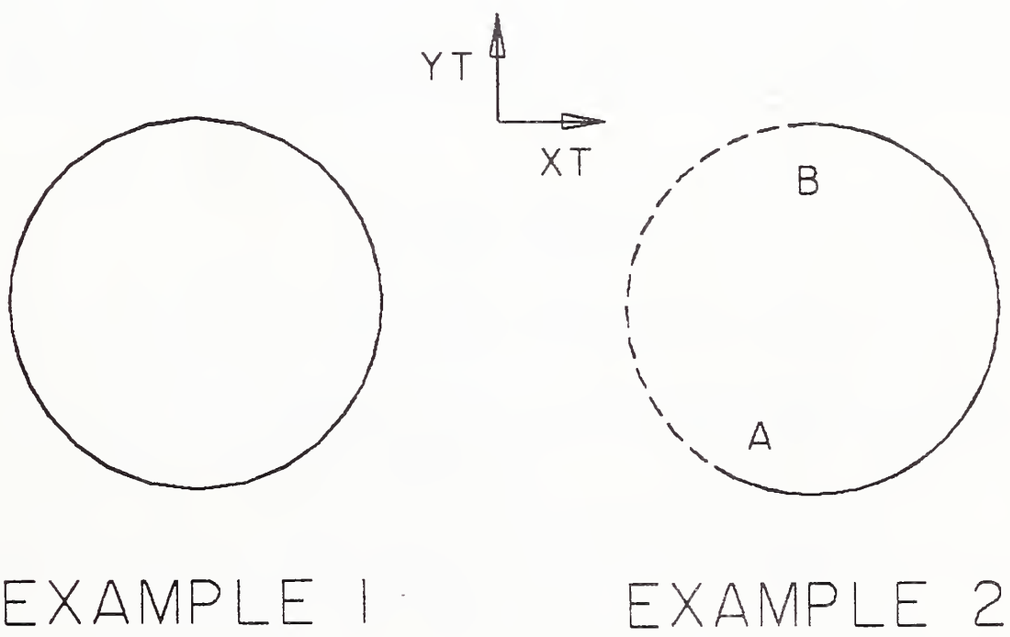


Figure 25. Examples Defined Using the Circular Arc Entity

3.3 Composite Curve Entity (Type 102)

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

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.

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.

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.

The Point and Connect Point Entites 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 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:

1. Two Point or Connect Point Entities cannot appear consecutively in the defining list unless they are the only entities in the composite curve.
2. 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.
3. A composite curve cannot consist of a Point Entity alone or a single connect point.

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

Let

C	be the composite curve;
N	be the number of constituent curves ($N \geq 1$);
$CC(i)$	be the i-th constituent curve, for each i such that $1 \leq i \leq N$;
$PS(i)$	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 $\sum_{j=1}^{i}(PE(j) - PS(j))$, for each i such that $1 \leq i \leq N$

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 $T(i-1) \leq u \leq T(i)$.

A composite curve consisting solely of Point and/or Connect Point Entities will not be given a parameterization.

As an example of a parameterization of a composite curve entity, let $N = 3$, and for each i such that $1 \leq i \leq 3$, 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 is illustrated in Figure 26.

The curve combining $CC(1)$, $CC(2)$, and $CC(3)$ represents the composite curve C.

An example of a composite curve and its parameterization is shown in Figure 27.

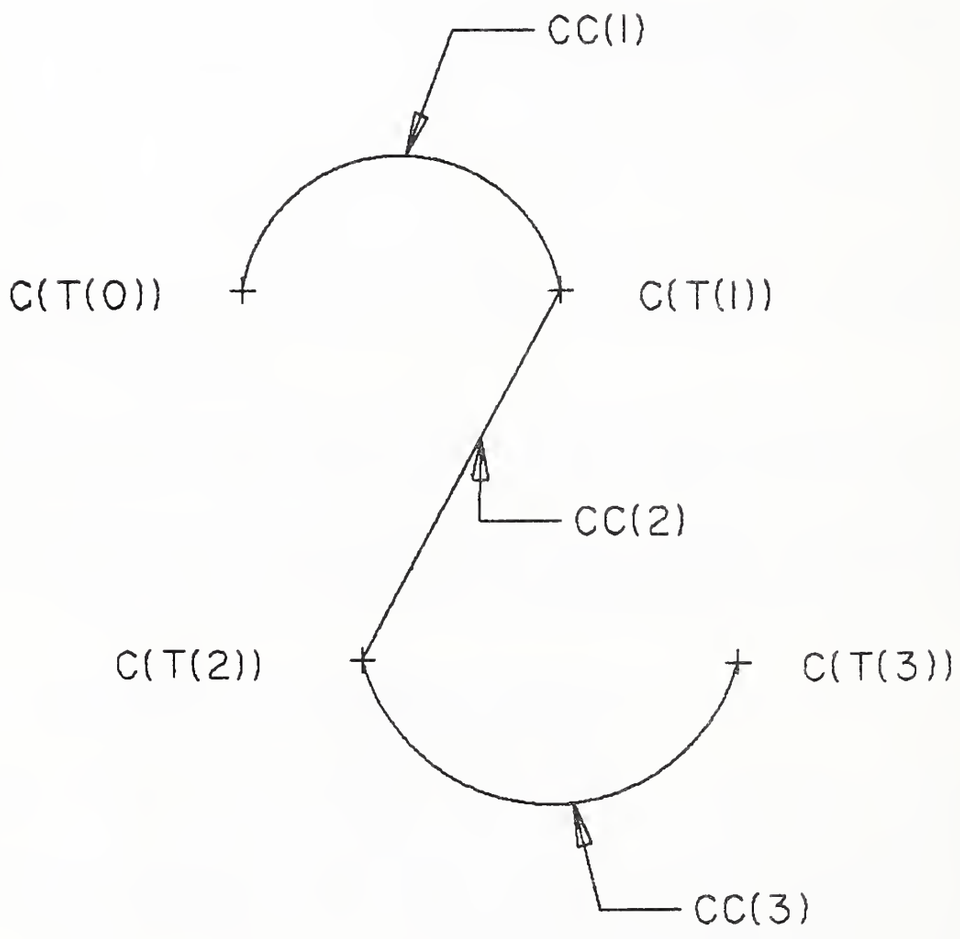


Figure 26. Parameterization of the Composite Curve

3.3.1 Directory Data

Entity Type Number: 102

3.3.2 Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entities
2	DE1	Pointer	Pointers to directory entries for the constituent entities
.	.	.	.
.	.	.	.
.	.	.	.
N+1	DEN	Pointer	

Additional pointers as required (see Section 2.2.4.4.2).

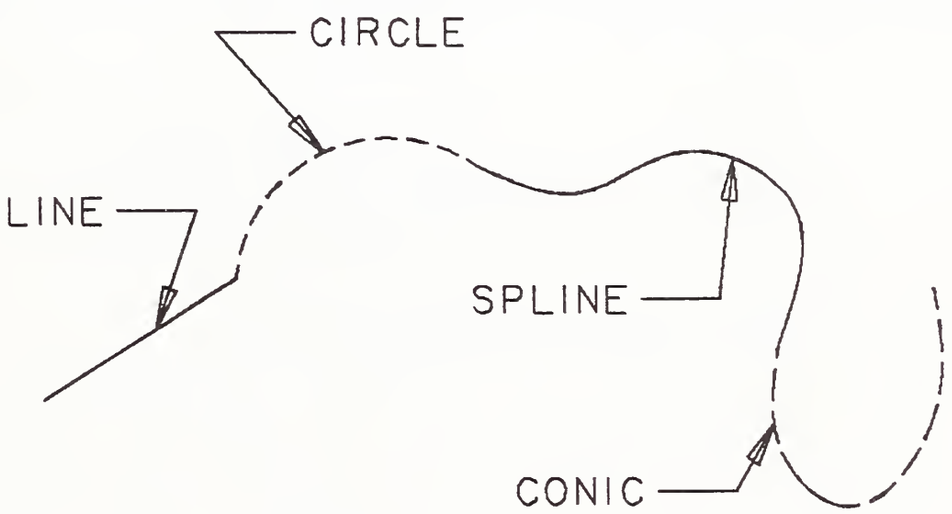


Figure 27. Example Defined Using the Composite Curve Entity

3.4 Conic Arc Entity (Type 104)

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.

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

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

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 distinguished 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.")

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.

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

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

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

$$Q3 = A + C$$

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 28.

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 F.

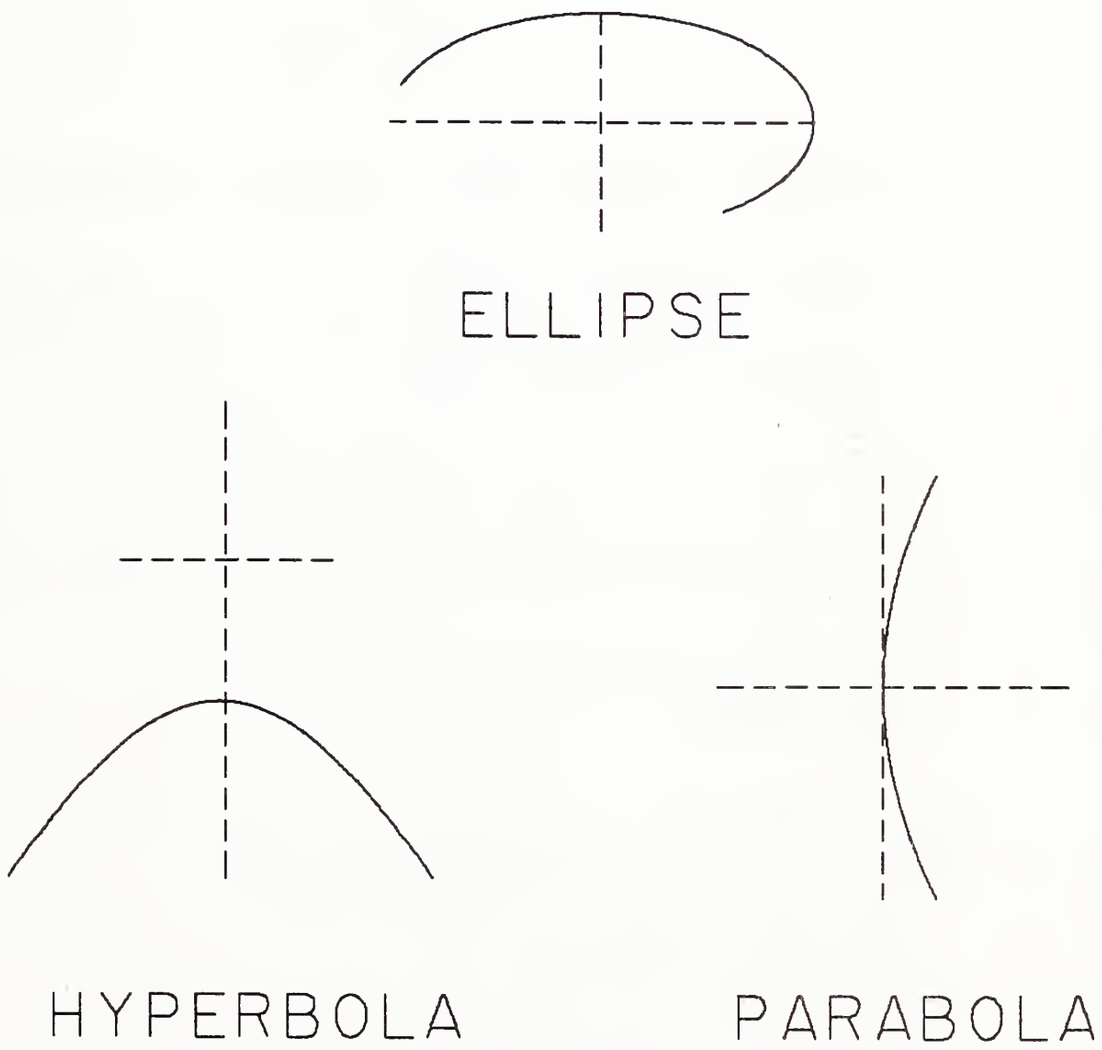


Figure 28. Examples Defined Using the Conic Arc Entity

3.4 CONIC ARC ENTITY (TYPE 104)

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

Parabola

case A and E $\neq 0.0$

if $X_1 < X_2$

$$C(t) = (t, -(A/E) * t^2, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

where, for $i = 1$ and 2, $t_i = X_i$.

if $X_2 < X_1$

$$C(t) = (-t, -(A/E) * t^2, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

where, for $i = 1$ and 2, $t_i = -X_i$.

case C and D $\neq 0.0$

if $Y_1 < Y_2$

$$C(t) = (-(C/D) * t^2, t, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

where, for $i = 1$ and 2, $t_i = Y_i$.

if $Y_2 < Y_1$

$$C(t) = (-(C/D) * t^2, -t, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

where, for $i = 1$ and 2, $t_i = -Y_i$.

Ellipse

$$C(t) = (a * \cos t, b * \sin t, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

where

$$a = \sqrt{-F/A}$$
$$b = \sqrt{-F/C}$$

and, for $i = 1$ and 2, t_i is such that

(i) $(a * \cos t_i, b * \sin t_i, ZT) = (X_i, Y_i, ZT)$
(ii) $0 \leq t_1 \leq 2 * \pi$
(iii) $0 \leq t_2 - t_1 \leq 2 * \pi$

Hyperbola

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

let

$a = \sqrt{-F/A}$
$b = \sqrt{F/C}$
and, for $i = 1, 2$,
t_i is such that
(i) $(a * \sec t_i, b * \tan t_i, ZT) = (X_i, Y_i, ZT)$
(ii) $-\pi/2 < t_1, t_2 < \pi/2$

if $t_1 < t_2$

$$C(t) = (a * \sec t, b * \tan t, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

if $t_2 < t_1$

$$C(t) = (a * \sec(-t), b * \tan(-t), ZT) \qquad \text{for } -t_1 \leq t \leq -t_2$$

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

let

$a = \sqrt{F/A}$
$b = \sqrt{-F/C}$
and, for $i = 1, 2$
t_i is such that
(i) $(a * \tan t_i, b * \sec t_i, ZT) = (X_i, Y_i, ZT)$
(ii) $-\pi/2 < t_1, t_2 < \pi/2$

if $t_1 < t_2$

$$C(t) = (a * \tan t, b * \sec t, ZT) \qquad \text{for } t_1 \leq t \leq t_2$$

if $t_2 < t_1$

$$C(t) = (a * \tan(-t), b * \sec(-t), ZT) \qquad \text{for } -t_1 \leq t \leq -t_2$$

3.4 CONIC ARC ENTITY (TYPE 104)

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 Figure 28)
2	Parent conic curve is a hyperbola (See Figure 28)
3	Parent conic curve is a parabola (See Figure 28)

3.4.1 Directory Data
Entity Type Number: 104

3.4.2 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 Section 2.2.4.4.2).

3.5 Copious Data Entity (Type 106)

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.

3.5 COPIOUS DATA ENTITY (TYPE 106)

The linear path is implemented as two forms of the Copious Data Entity (Type 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 compatibility with previous versions of this 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 Restriction.

The simple closed area is implemented as Form 63 of the Copious Data Entity (Type 106).

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), *e.g.*, Forms 1, 11, 20–63

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), *e.g.*, Forms 2, 12

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), *e.g.*, Forms 3, 13

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 Section 2.2.4.4.2).

3.6 Plane Entity (Type 108)

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).

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 the unbounded plane example of Figure 29, the curves and the crosshair together constitute the display symbol.) Setting the size parameter to zero indicates that a display symbol is not intended.

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.

The case of a bounded portion of a fixed plane minus some portion(s) of that plane, such as those shown in Figure 30, 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 file and has a backpointer to the associativity structure. The child plane entity will have a subordinate entity switch class of 01 (Physically Dependent).

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. PTR must not be zero.
0	Plane is unbounded. PTR must be zero.
-1	Bounded planar portion is considered negative (hole). PTR must not be zero.

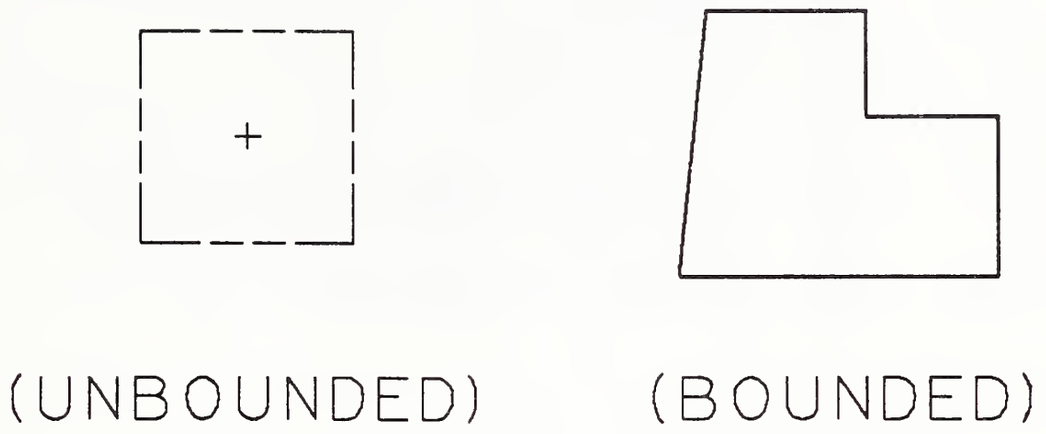


Figure 29. Examples Defined Using the Plane Entity

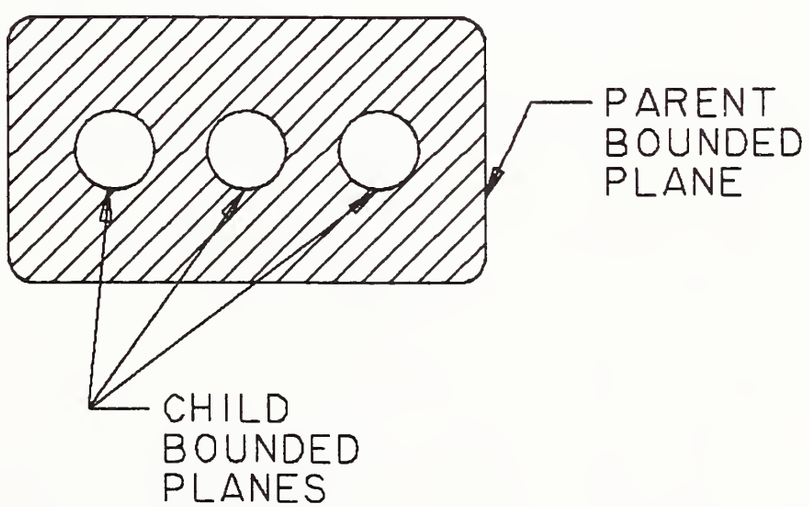


Figure 30. Single Parent Associativity Used with a Collection of Bounded Planes

3.6 PLANE ENTITY (TYPE 108)

3.6.1 Directory Data
Entity Type Number: 108

3.6.2 Parameter Data

Index	Name	Type	Description
1	A	Real	Coefficients of Plane
2	B	Real	Coefficients of Plane
3	C	Real	Coefficients of Plane
4	D	Real	Coefficients of Plane
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 Section 2.2.4.4.2).

3.7 Line Entity (Type 110)

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 triple 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 31.

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

$$C(t) = P_1 + t * (P_2 - P_1) \qquad \text{for} \qquad 0 \leq t \leq 1$$

3.7.1 Directory Data

Entity Type Number: 110

3.7.2 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 Section 2.2.4.4.2).

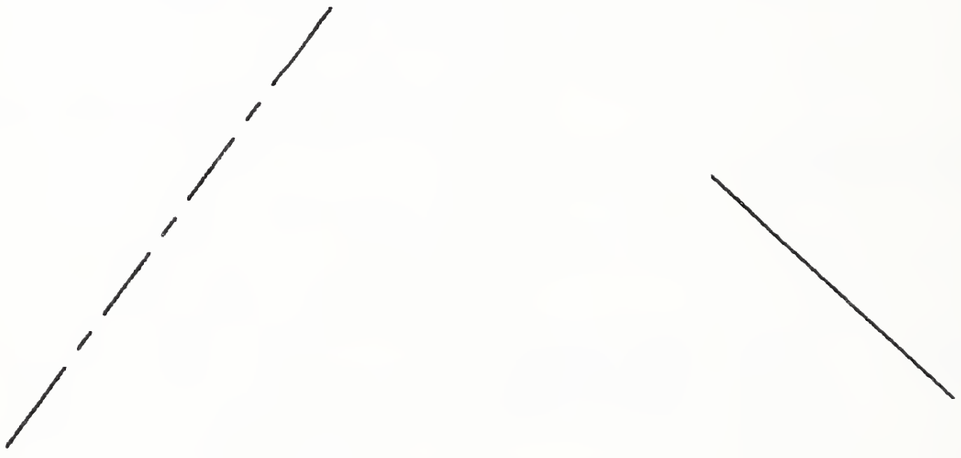


Figure 31. Examples Defined Using the Line Entity

3.8 Parametric Spline Curve Entity (Type 112)

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 (preprocessing) system before conversion to this entity.

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):

$$\begin{aligned} X(u) &= AX(i) + BX(i) * s + CX(i) * s^2 + DX(i) * s^3 \\ Y(u) &= AY(i) + BY(i) * s + CY(i) * s^2 + DY(i) * s^3 \\ Z(u) &= AZ(i) + BZ(i) * s + CZ(i) * s^2 + DZ(i) * s^3 \end{aligned}$$

where

$$\begin{aligned} & T(i) \leq u \leq T(i+1),\ i = 1, ..., N \\ & s = u - T(i) \end{aligned}$$

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

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

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]).

To enable determination of the terminate point and derivatives without computing the polynomials, the N-th 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 N-th 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 N-th polynomial segment.

Examples of a parametric spline are shown in Figure 32 and Figure 33; see Appendix E for additional mathematical details.

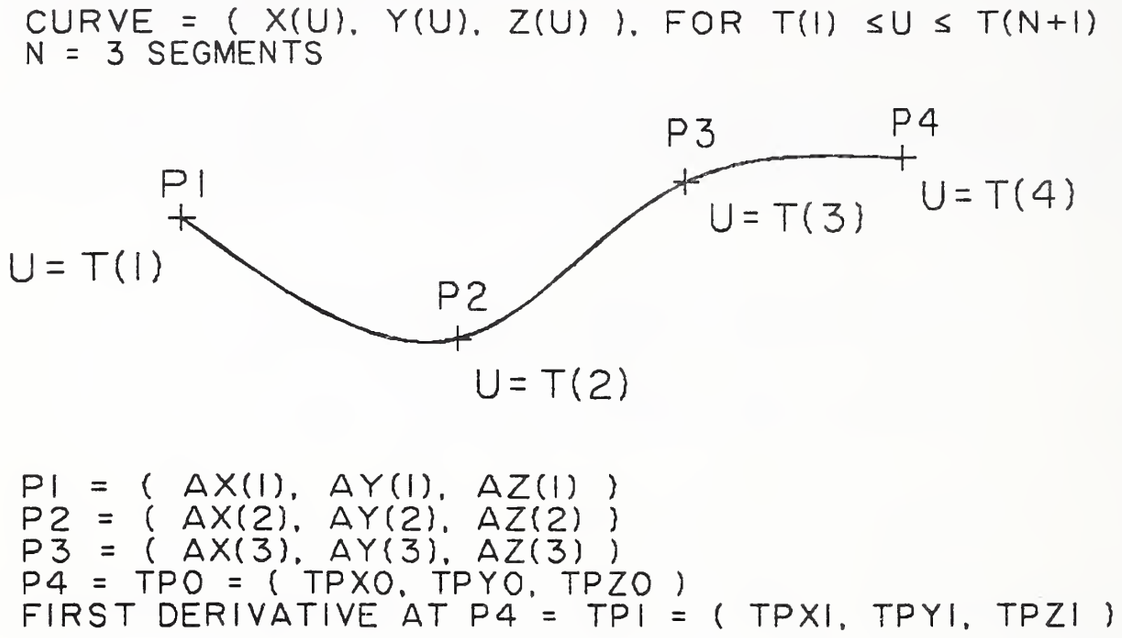


Figure 32. Parameters of the Parametric Spline Curve Entity

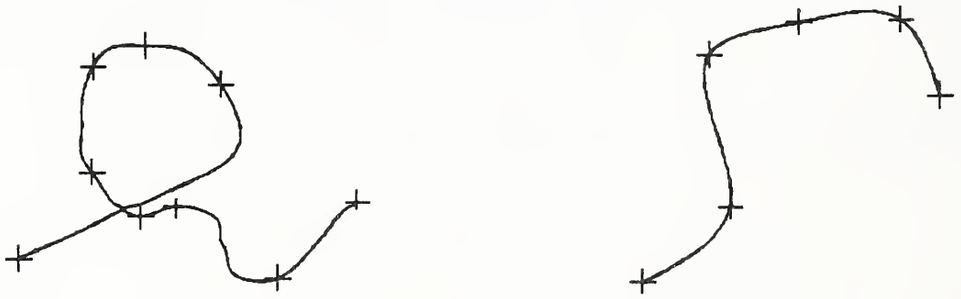


Figure 33. Examples Defined Using the Parametric Spline Curve Entity

3.8.1 Directory Data

Entity Type Number: 112

3.8.2 Parameter Data

Index	Name	Type	Description
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	Number of dimensions 2=planar 3=non-planar
4	N	Integer	Number of segments
5	T(1)	Real	Break points of piecewise polynomial
.	.	.	.
.	.	.	.
.	.	.	.
5+N	T(N+1)	.	.
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 polynomial
11+N	BY(1)	.	.
12+N	CY(1)	.	.
13+N	DY(1)	.	.
14+N	AZ(1)	.	Z coordinate polynomial
15+N	BZ(1)	.	.
16+N	CZ(1)	.	.
17+N	DZ(1)	.	.
.	.	.	Subsequent X, Y, Z polynomials concluding with the twelve coefficients of the Nth polynomial segment.
.	.	.	.
.	.	.	.
.	.	.	.

3.8 PARAMETRIC SPLINE CURVE ENTITY (TYPE 112)

The parameters that follow comprise the evaluations of the polynomials of the N-th 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.

Index	Name	Type	Description
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 Section 2.2.4.4.2).

3.9 Parametric Spline Surface Entity (Type 114)

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

The $M \times N$ 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).

$$
\begin{aligned}
X(u,v) &= AX(i,j) + BX(i,j)*s + CX(i,j)*s^2 + DX(i,j)*s^3 \\
&+ EX(i,j)*t + FX(i,j)*t*s + GX(i,j)*t*s^2 + HX(i,j)*t*s^3 \\
&+ KX(i,j)*t^2 + LX(i,j)*t^2*s + MX(i,j)*t^2*s^2 + NX(i,j)*t^2*s^3 \\
&+ PX(i,j)*t^3 + QX(i,j)*t^3*s + RX(i,j)*t^3*s^2 + SX(i,j)*t^3*s^3 \\
Y(u,v) &= \ldots \\
Z(u,v) &= \ldots
\end{aligned}
$$

where

$$TU(i) \le u \le TU(i+1), \quad i = 1, \ldots, M$$
$$s = u - TU(i)$$

and

$$TV(j) \le v \le TV(j+1), \quad j = 1, \ldots, N$$
$$t = v - TV(j)$$

Postprocessors shall ignore parameters with the indices `7+M+N+48*(k*N+(k-1))` through `6+M+N+48*(k*(N+1))`, where `k=1,2,3,...,M` (*i.e.*, the (N+1)-th row of patches) as well as `7+M+N+48*(M*(N+1))` through `6+M+N+ 48*(M+1)*(N+1)` (*i.e.*, the (M+1)-th column of patches).

To maintain upward compatibility with previous versions of this Specification, the preprocessors must either enter a real number for each of these parameters or a series of parameter delimiters (see Section 2.2.3). These values act as placeholders in the parameter list. These parameters were intended to handle first, second, and third partial derivatives of the N-th 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.

An example of the bicubic surface is shown in Figure 34; consult Appendix E for additional details.

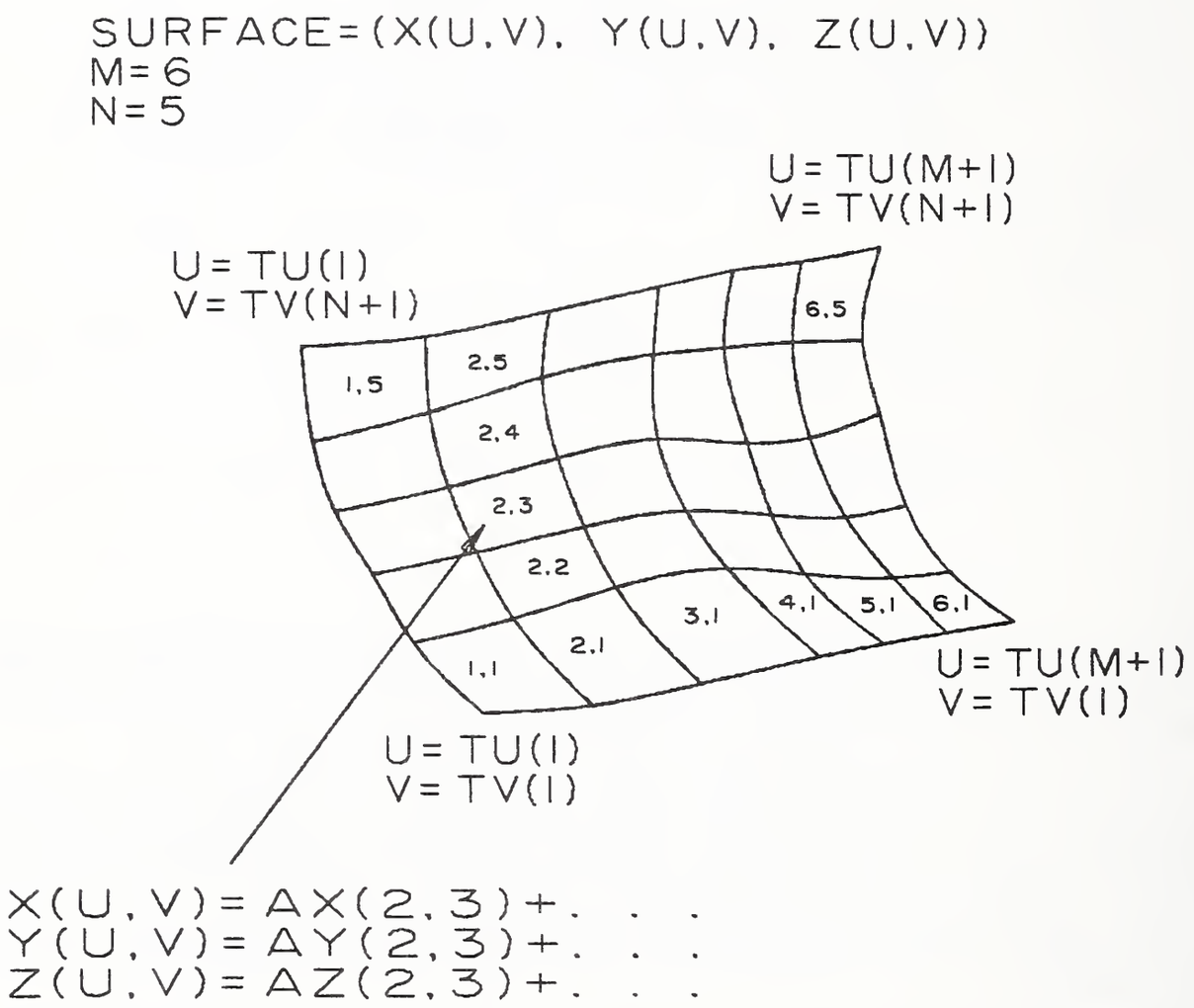


Figure 34. Parameters of the Parametric Spline Surface Entity

3.9 PARAMETRIC SPLINE SURFACE ENTITY (TYPE 114)

3.9.1 Directory Data

Entity Type Number: 114

3.9.2 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)	.	.
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)	Real	Y Coefficients of (1,1) Patch
.	.	.	.
38+M+N	SY(1,1)	.	.
39+M+N	AZ(1,1)	Real	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
.	.	.	.

3.9 PARAMETRIC SPLINE SURFACE ENTITY (TYPE 114)

Index	Name	Type	Description
6+M+N+48*(N+2)	SZ(2,1)	Real	.
.	.	.	.
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 Section 2.2.4.4.2).

3.10 Point Entity (Type 116)

A point is defined by its coordinates in definition space. Examples of the point entity are shown in Figure 35.

3.10.1 Directory Data
Entity Type Number: 116

3.10.2 Parameter Data

Index	Name	Type	Description
1	X	Real	Coordinates of point
2	Y	Real	
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 Section 2.2.4.4.2).

Figure 35. Examples Defined Using the Point Entity

3.11 Ruled Surface Entity (Type 118)

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 nonplanar).

Form 0: In this case, DE1 and DE2 specify the defining rail curves, but their given parameterizations are not the ones used to generate the ruled surface. Instead, their arc length reparameterizations, 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 parameterizations are the ones used to generate the ruled surface.

Both Forms: In either case, if two curves are expressed parameterically by the functions $(C1_x(t), C1_y(t), C1_z(t))$ and $(C2_x(s), C2_y(s), C2_z(s))$, where $a \leq t \leq b$ and $c \leq s \leq d$, then the coordinates of the points on the ruled surface can be written as:

$$
\begin{aligned}
X(u,v) &= (1-v) * C1_x(t) + v * C2_x(s) \\
Y(u,v) &= (1-v) * C1_y(t) + v * C2_y(s) \\
Z(u,v) &= (1-v) * C1_z(t) + v * C2_z(s)
\end{aligned}
$$

where

$$
\begin{aligned}
&0 \leq u \leq 1, \\
&0 \leq v \leq 1, \\
&t = a + u * (b-a) \\
&s = c + u * (d-c), \text{ if } DIRFLG = 0 \\
&s = d + u * (c-d), \text{ if } DIRFLG = 1
\end{aligned}
$$

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.

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.

If DEVFLG=1, then the surface is a developable surface (see [DOCA76]); if DEVFLG=0, the surface may or may not be a developable surface.

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. Examples of the Ruled Surface Entity are shown in Figures 36 and 37.

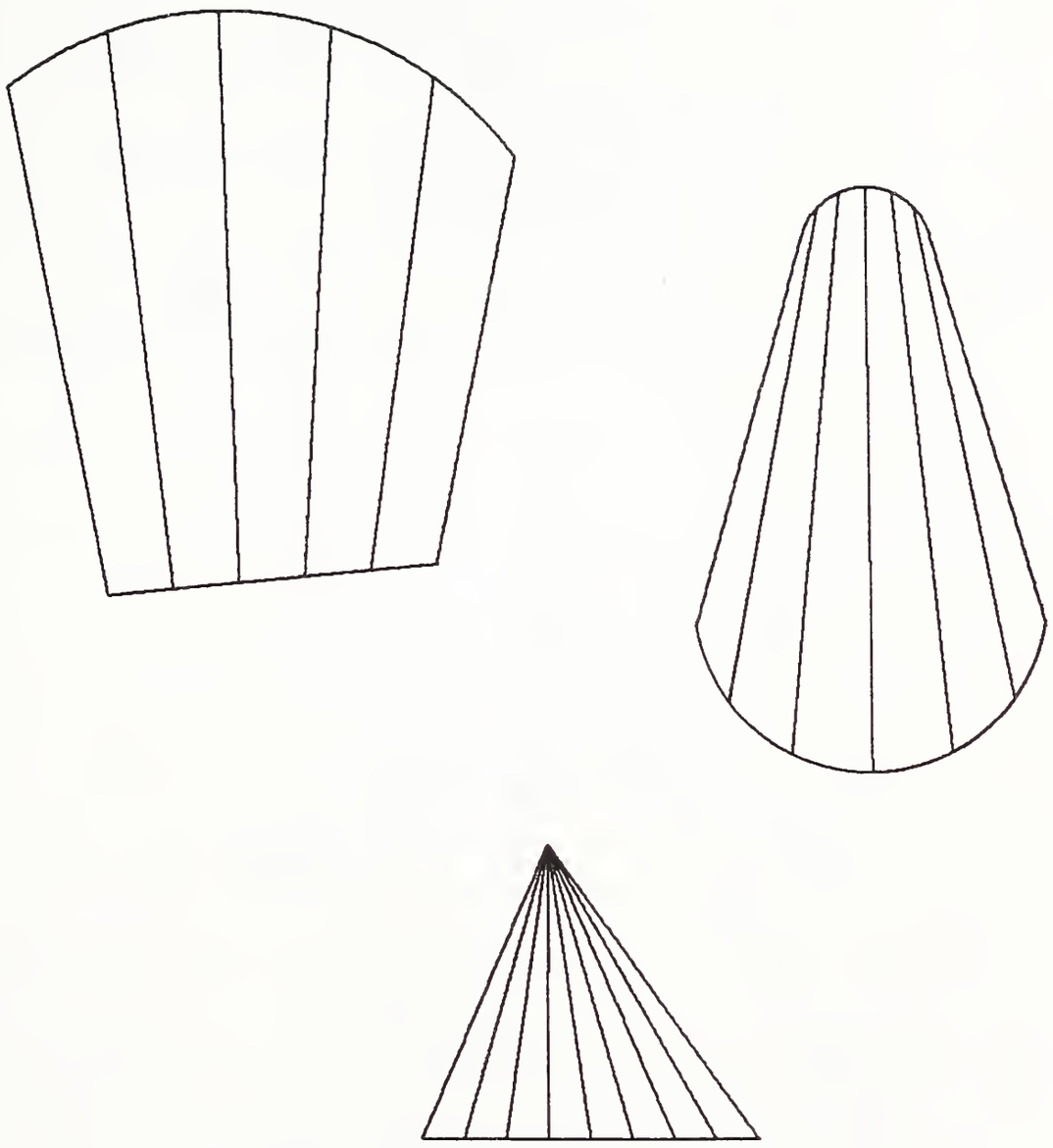


Figure 36. Examples Defined Using the Ruled Surface Entity

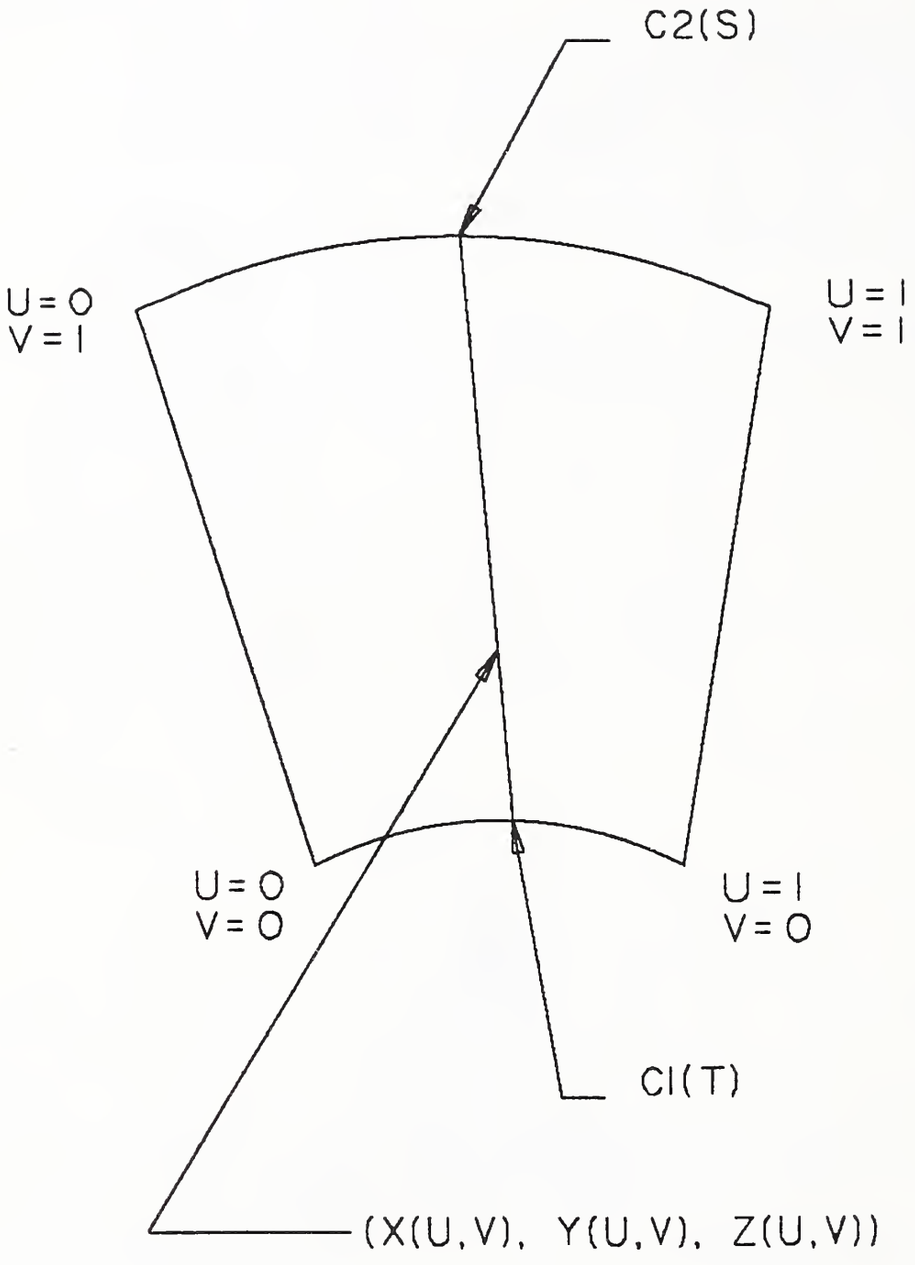


Figure 37. Parameters of the Ruled Surface Entity

3.11.1 Directory Data

Entity Type Number: 118

3.11.2 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 Section 2.2.4.4.2).

3.12 Surface of Revolution Entity (Type 120)

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 (Type 110), 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 measured counterclockwise while looking in the direction of the start point of the Line Entity defining the axis of revolution from the terminate point of this line. The generatrix curve may be any curve entity to which a parameterization has been assigned. Examples of surfaces of revolution are given in Figure 38.

The various parameters defining the Surface of Revolution Entity are illustrated in Figure 39. The Line Entity L defines a unique straight line. This straight line defines the axis of revolution. The axis is given the same direction as the direction assigned to the Line Entity L . Let R_θ be the unique rigid motion leaving each point of the axis of revolution fixed and rotating each point in three dimensional Euclidean space, θ radians counterclockwise about the axis of revolution. R_θ assigns to each element of three dimensional Euclidean space another element of three dimensional Euclidean space.

The curve C is the generatrix of the surface of revolution. For each real number in the interval [a,b] that defines its domain, C assigns an element of three dimensional Euclidean space.

SA and TA denote the start angle and terminate angle, measured in radians, of the surface of revolution to be defined. SA and TA are constrained so that $0 < TA - SA \leq 2\pi$.

The surface of revolution S defined by this entity is the surface that is swept by rotating the generatrix curve C from the angle SA to the angle TA, counterclockwise about the directed axis of revolution.

The default parameterization for the surface of revolution S shall be given by

$$S(x, \theta) = R_\theta \ (C(x))$$

for each pair of real numbers (x, θ) such that $a \leq x \leq b$ and $SA \leq \theta \leq TA$.

3.12.1 Directory Data

Entity Type Number: 120

3.12.2 Parameter Data

Index	Name	Type	Description
1	L	Pointer	Pointer to a Line Entity (axis of revolution)
2	C	Pointer	Pointer to generatrix
3	SA	Real	Start angle in radians
4	TA	Real	Terminate angle in radians

Additional pointers as required (see Section 2.2.4.4.2).

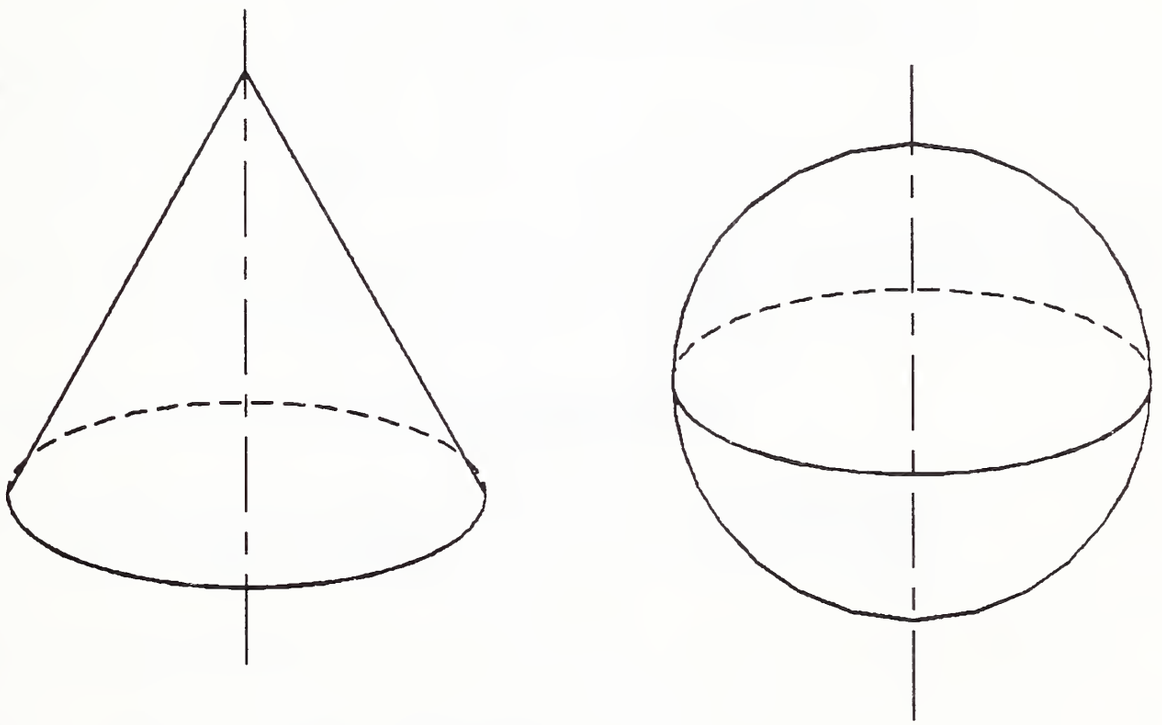


Figure 38. Examples Defined Using the Surface of Revolution Entity

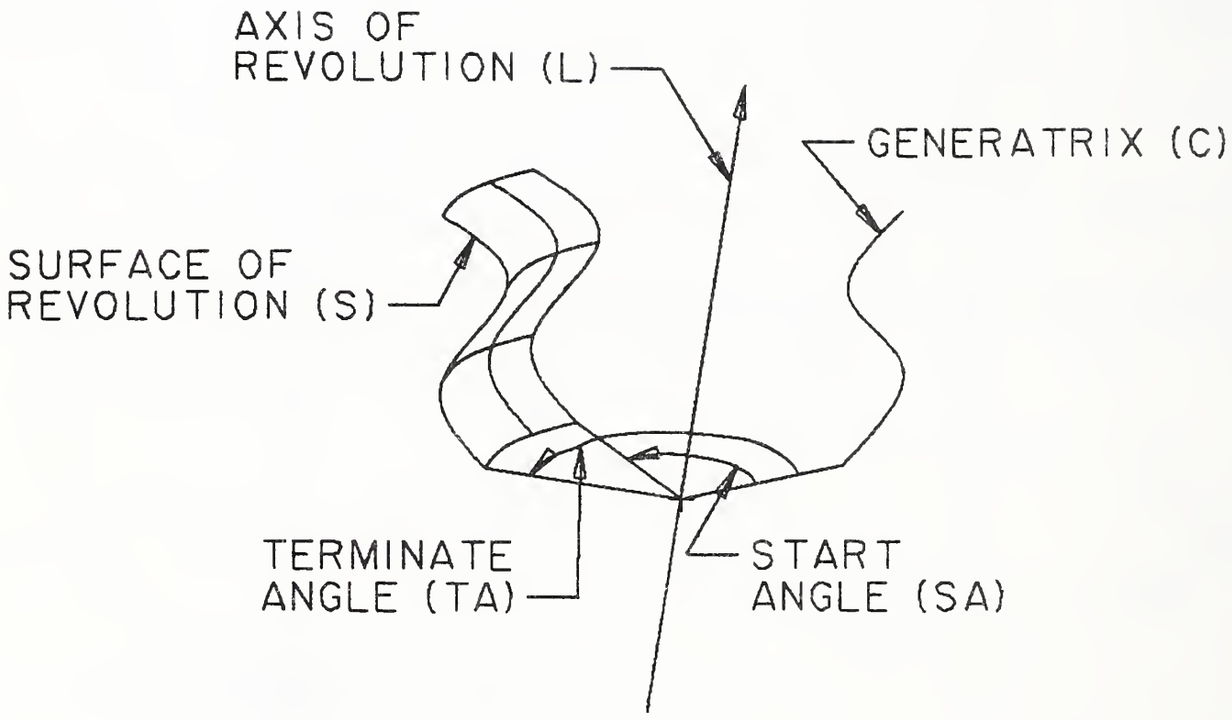


Figure 39. Parameters of the Surface of Revolution Entity

3.13 Tabulated Cylinder Entity (Type 122)

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.

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:

$$\begin{aligned} X(u,v) &= CX(u) + v * (LX - CX(0)) \\ Y(u,v) &= CY(u) + v * (LY - CY(0)) \\ Z(u,v) &= CZ(u) + v * (LZ - CZ(0)) \end{aligned}$$

where $0 \leq u \leq 1, \quad 0 \leq v \leq 1$

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.

An example of the tabulated cylinder is shown in Figure 40.

3.13.1 Directory Data
Entity Type Number: 122

3.13.2 Parameter Data

Index	Name	Type	Description
1	DE	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 Section 2.2.4.4.2).

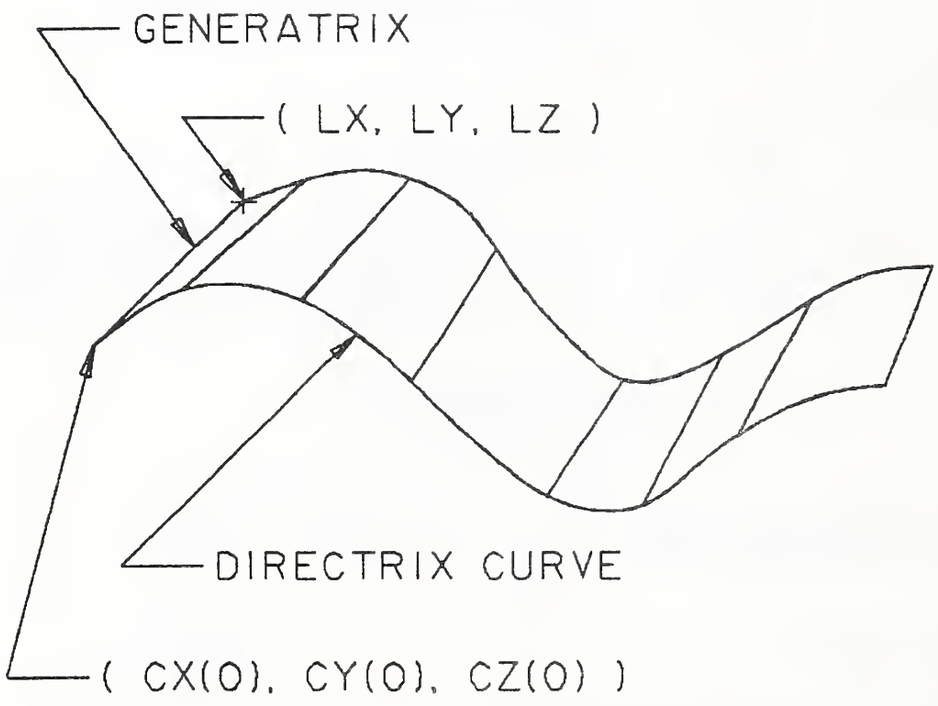


Figure 40. Parameters of the Tabulated Cylinder Entity

3.14 Transformation Matrix Entity (Type 124)

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, col [XINPUT, YINPUT, ZINPUT] (*i.e.*, the column vector) is the vector being transformed, and col [XOUTPUT, YOUTPUT, ZOUTPUT] is the column vector resulting from this transformation. $R = [R_{ij}]$ is a 3 row by 3 column matrix of real numbers, and T = col [T1,T2,T3] 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.

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 considering special input vectors such as col [1,0,0], col [0,1,0], and col [0,0,1] and computing the corresponding output results, a geometric appreciation of the spatial relationship between the two coordinate systems can be gained.

For example, for

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

the spatial relationship of the input and output coordinate systems is given in Figure 41.

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

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 systems. 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 Section 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 Section 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 Section 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.

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

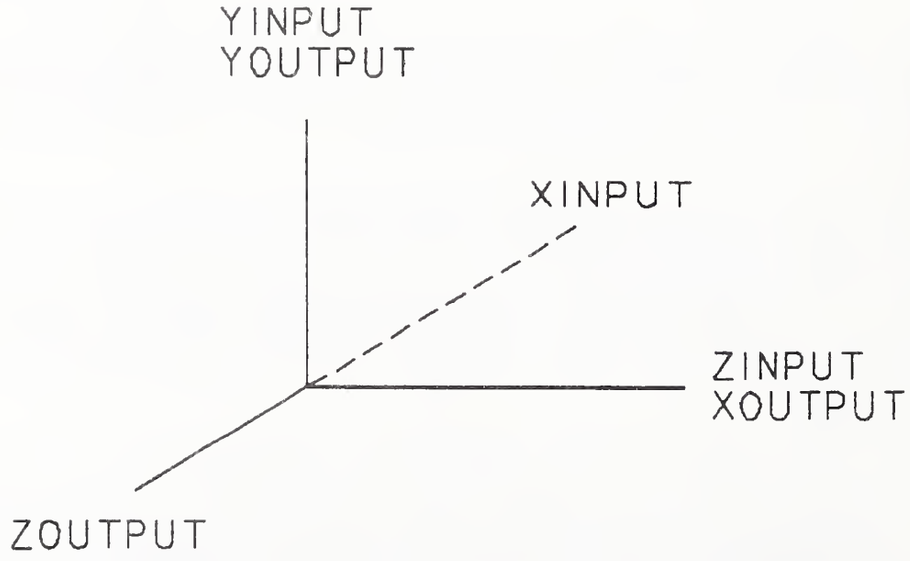


Figure 41. Example of the Transformation Matrix Coordinate Systems

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.

A 3x3 matrix R is called orthogonal provided its transpose, R^t, 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.

Allowable Form Numbers The defining matrix of an entity must use either Form 0 or Form 1. A defining matrix associated with a View Entity (Type 410) must use Form 0. Special matrices representing Node Entity (Type 134) local coordinate systems must use Forms 10, 11, or 12.

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.

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

Refer to Figure 42(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,

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

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 134) in Finite Element applications.

Refer to Figure 42(b) for notation. The matrix R and the vector T are used to transform coordinate data from the u1, u2, u3 (node point) coordinate system to the x,y,z (local system) coordinate system.

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

$$\begin{array}{lcll} XOFFSET & = & r_0 \cos\theta_0 & r_0 > 0 \\ YOFFSET & = & r_0 \sin\theta_0 & 0 \leq \theta_0 \leq 360^0 \\ ZOFFSET & = & z_0 & -\infty < z_0 < \infty \end{array}$$

$$\text{for } r_0 = 0, \text{ take } \theta = 0^0$$

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

$$\begin{aligned} x &= r\cos\theta \\ y &= r\sin\theta \\ z &= z, \end{aligned}$$

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 134) in Finite Element applications.

Refer to Figure 42(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

$$\begin{array}{lcll} XOFFSET & = & r_0 \sin\theta_0 \sin\phi_0 & r_0 \geq 0 \\ YOFFSET & = & r_0 \sin\theta_0 \cos\phi_0 & 0 \leq \theta_0 \leq 180^0 \\ ZOFFSET & = & r_0 \cos\theta_0 & 0 \leq \phi_0 < 360^0 \end{array}$$

$$\text{for } r_0 = 0, \text{ take } \theta_0 = \phi_0 = 0^0$$
$$\text{for } \theta_0 = 0^0 \text{ or } 180^0, \text{ take } \phi_0 = 0^0$$

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 spherical coordinates (r,θ,ϕ) with

$$\begin{array}{lcl} x & = & r \sin\theta \cos\phi \\ y & = & r \sin\theta \sin\phi \\ z & = & r \cos\theta \end{array}$$

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:

$$\begin{bmatrix} \sin\theta_0 * \cos\phi_0 & \cos\theta_0 * \cos\phi_0 & -\sin\phi_0 \\ \sin\theta_0 * \sin\phi_0 & \cos\theta_0 * \sin\phi_0 & \cos\phi_0 \\ \cos\theta_0 & -\sin\theta_0 & 0 \end{bmatrix} \begin{bmatrix} u1 \\ u2 \\ u3 \end{bmatrix} + \begin{bmatrix} XOFFSET \\ YOFFSET \\ ZOFFSET \end{bmatrix} = \begin{bmatrix} XLOCAL \\ YLOCAL \\ ZLOCAL \end{bmatrix}$$

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

3.14 TRANSFORMATION MATRIX ENTITY (TYPE 124)

3.14.1 Directory Data

Entity Type Number: 124

3.14.2 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 Section 2.2.4.4.2).

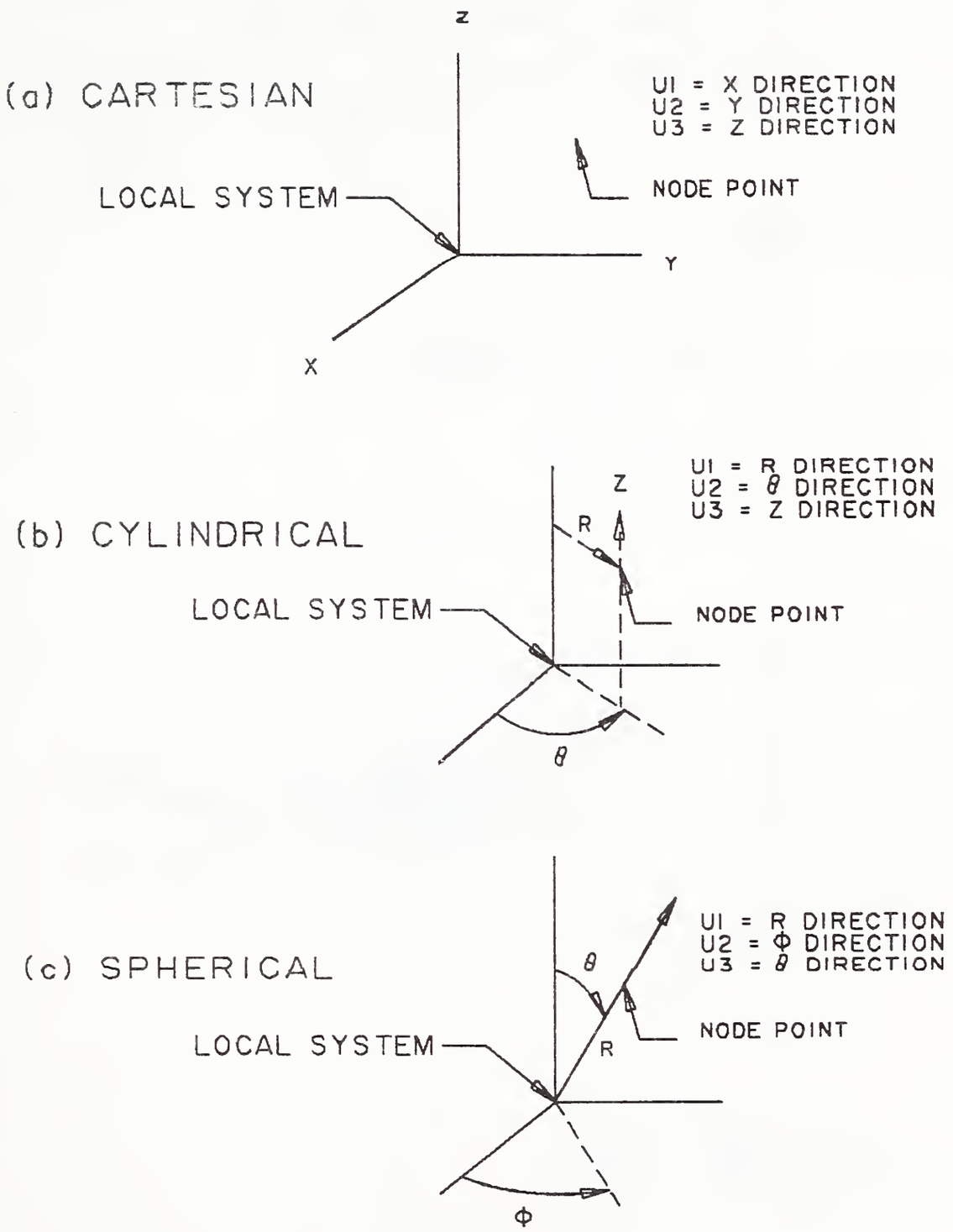


Figure 42. Notation for FEM-specific Forms of the Transformation Matrix Entity

3.15 Flash Entity (Type 125)

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 43 indicates the usage of those parameters for the specific flash forms. Parameters 3 through 5 are ignored for Form 0.

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

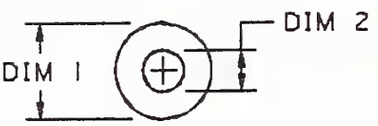
Index	Name	Type	Description
1	X	Real	X reference of flash
2	Y	Real	Y reference of flash
3	DIM1	Real	First flash sizing parameter
4	DIM2	Real	Second flash sizing parameter
5	ROT	Real	Rotation of flash about reference point in radians
6	DE	Pointer	DE of referenced entity or zero

Additional pointers as required (see Section 2.2.4.4.2).

FORM 1 - CIRCULAR

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

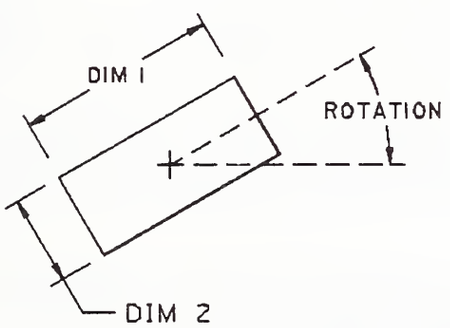
REFERENCE POINT IS CIRCLE CENTER



FORM 3 - DONUT

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

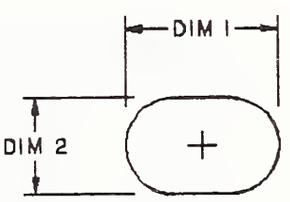
REFERENCE POINT IS CIRCLE CENTER



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 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 43. Definition of Shapes for the Flash Entity

3.16 Rational B-Spline Curve Entity (Type 126)

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 rational B-spline curves, see Appendix E.

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 nonplanar.

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 1 (see Appendix E). The weights must be positive real numbers.

If the curve is periodic with respect to its parametric variable, set PROP4 to 1; otherwise set PROP4 to 0. The periodic flag is to be interpreted as purely informational. The curves which are flagged to be periodic are to be evaluated exactly the same as in the nonperiodic case.

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 E
2	M	Integer	Degree of basis functions
3	PROP1	Integer	= 0 - nonplanar = 1 - planar
4	PROP2	Integer	= 0 - open curve

			= 1 - closed curve
5	PROP3	Integer	= 0 - rational
			= 1 - polynomial
6	PROP4	Integer	= 0 - nonperiodic
			= 1 - periodic

Let N=K-M+1 and A=N+2*M

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	X0	Real	Control Points
10+A+K	Y0	.	.
11+A+K	Z0	.	.
.	.	.	.
.	.	.	.
.	.	.	.
9+A+4*K	XK	.	.
10+A+4*K	YK	.	.
11+A+4*K	ZK	.	.
12+A+4*K	V(0)	Real	Starting parameter value
13+A+4*K	V(1)	Real	Ending parameter value
14+A+4*K	XNORM	Real	Unit Normal (if curve is planar)
15+A+4*K	YNORM	.	.
16+A+4*K	ZNORM	.	.

Additional pointers as required (see Section 2.2.4.4.2).

3.17 Rational B-Spline Surface Entity (Type 128)

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 Appendix E.

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 E). The weights must be positive real numbers.

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. The periodic flags are to be interpreted as purely informational. The surfaces which are flagged to be periodic are to be evaluated exactly the same as in the non-periodic case.

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 E
2	K2	Integer	Upper index of second sum. See Appendix E
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 + 2*M1,
B = N2 + 2*M2,
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+4*C	X(K1,K2)	.	.
10+A+B+4*C	Y(K1,K2)	.	.
11+A+B+4*C	Z(K1,K2)	.	.
12+A+B+4*C	U(0)	Real	Starting value for first parametric direction
13+A+B+4*C	U(1)	Real	Ending value for first parametric direction
14+A+B+4*C	V(0)	Real	Starting value for second parametric direction
15+A+B+4*C	V(1)	Real	Ending value for second parametric direction

Additional pointers as required (see Section 2.2.4.4.2).

3.18 Offset Curve Entity (Type 130)

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:

$$O(t) = r(t) + f(s) * (V \times T(t)); \qquad TT1 \leq t \leq TT2$$

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 coordinate 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)
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 Section 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.

3.19 Connect Point Entity (Type 132)

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 boards). The Connect Point Entity is referenced from either the Network Subfigure Definition (Type 320), Network Subfigure Instance (Type 420), or the Flow Associativity Instance (Type 402, Form 18); or it may stand alone in a file. The connect point may be displayed by the receiving system using default display parameters and/or symbols. Also see Section 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
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 Section 2.2.4.4.2).

3.20 Node Entity (Type 134)

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.

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. Figure 44 illustrates the definition of a node in the three coordinate systems.

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 44(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 44(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 44(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 a cylindrical system the first axis is the $\theta = 0$ axis and the third axis is the z axis. For a spherical system the first axis is the $\phi = 0$ 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.

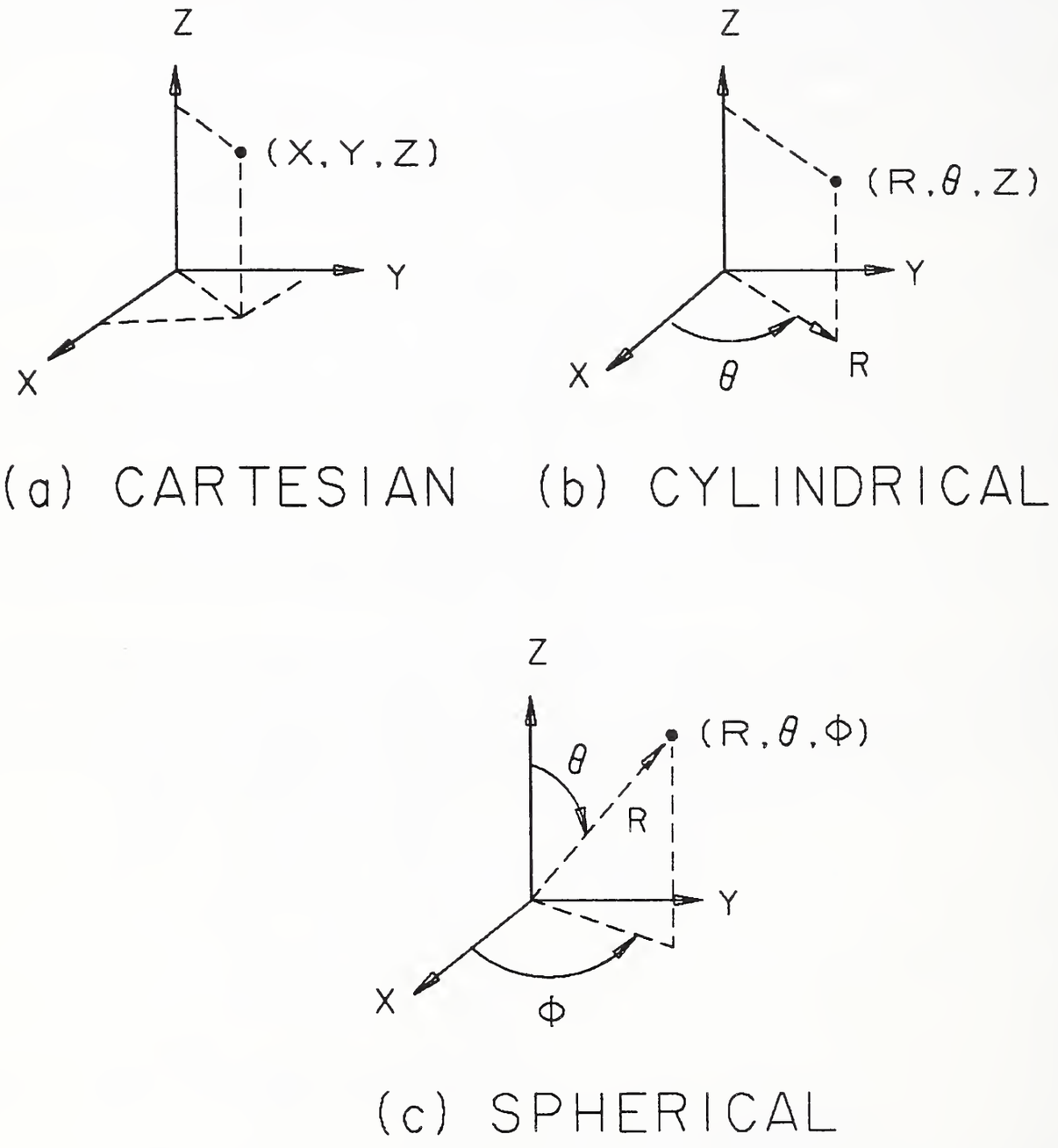


Figure 44. Nodal Displacement Coordinate Systems

3.20.1 Directory Data
Entity Type Number: 134

Entity Label:	Node Label (Optional)
Entity Subscript:	Node Number (Required)

3.20.2 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/ϕ	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 Section 2.2.4.4.2).

3.21 Finite Element Entity (Type 136)

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

Table 5 lists the data to define the element topology. Figure 45 illustrates the node connectivity for each element topology.

In Table 5 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:

Value	Order of Edge
0	Not applicable
1	Linear
2	Parabolic
3	Cubic

The number of nodes from Table 5 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.

3.21.1 Directory Data
Entity Type Number: 136

```
Entity Label:        Element Label (Optional)
Entity Subscript:    Element Number (Required)
```

3.21.2 Parameter Data

Index	Name	Type	Description
1	ITOP	Integer	Topology type
2	N	Integer	Number of nodes defining element (See Section 3.20).
3	DE1	Pointer	Pointer to first node defining element (See Section 3.20).
.	.	.	.
.	.	.	.
.	.	.	.
N+2	DEN	Pointer	Pointer to last node defining element
N+3	ETYP	String	Element type name

Additional pointers as required (see Section 2.2.4.4.2).

Table 5. Finite Element Topology Set

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

3.21 FINITE ELEMENT ENTITY (TYPE 136)

```
1. BEAM
   E1=1,2
```

```
2. LTRIA - Linear Triangle
   E1=1,2          F1=1,2,3
   E2=2,3
   E3=3,1
```

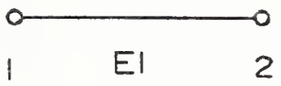
```
3. PTRIA - Parabolic Triangle
   E1=1,2,3        F1=1,2,3,4,5,6
   E2=3,4,5
   E3=5,6,1
```

```
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
```

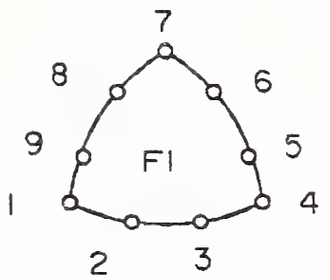
```
5. LQUAD - Linear Quadrilateral
   E1=1,2          F1=1,2,3,4
   E2=2,3
   E3=3,4
   E4=4,1
```

```
6. PQUAD - Parabolic Quadrilateral
   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
```

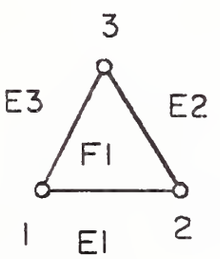
Figure 45. Finite Element Topology Set



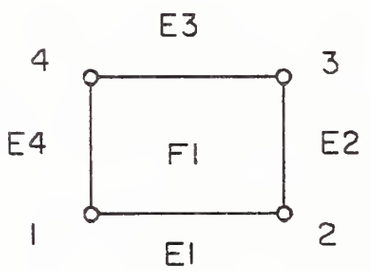
1. BEAM



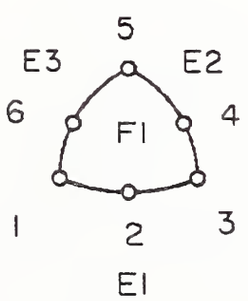
4. CUBIC TRIANGLE



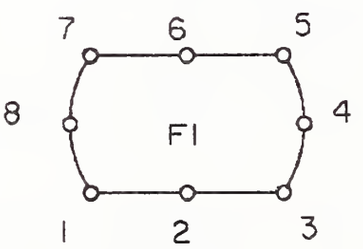
2. LINEAR TRIANGLE



5. LINEAR QUADRILATERAL



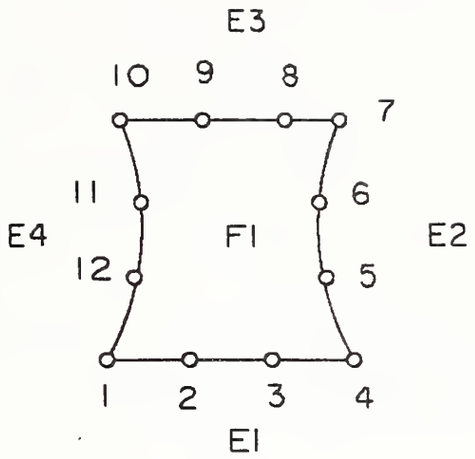
3. PARABOLIC TRIANGLE



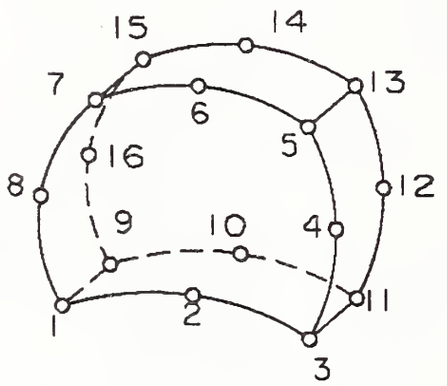
6. PARABOLIC QUADRILATERAL

Figure 45. Finite Element Topology Set (continued)

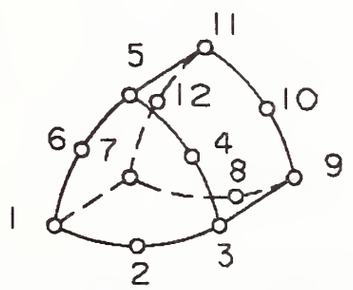
```
7. CQUAD - Cubic Quadrilateral
   E1=1,2,3,4
   E2=4,5,6,7
   E3=7,8,9,10
   E4=10,11,12,1

   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
```



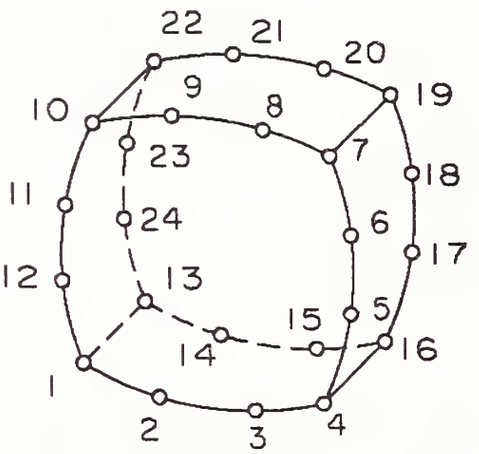
7. CUBIC QUADRILATERAL



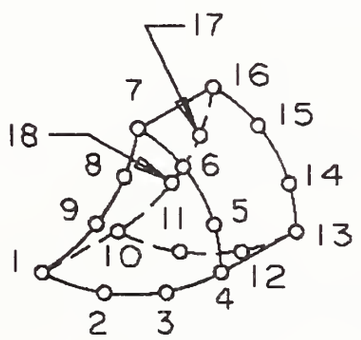
10. PARABOLIC THICK SHELL



8. PARABOLIC THICK SHELL WEDGE



11. CUBIC THICK SHELL



9. CUBIC THICK SHELL WEDGE

Figure 45. Finite Element Topology Set (continued)

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

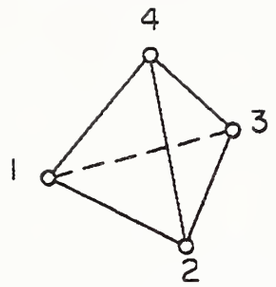
```
E1=1,2        E4=4,5     E7=1,4
E2=2,3        E5=5,6     E8=2,5
E3=3,1        E6=6,4     E9=3,6
F1=1,2,3
F2=4,5,6
F3=1,2,5,4
F4=2,3,6,5
F5=3,1,4,6
```

15. PSOW - Parabolic Solid Wedge

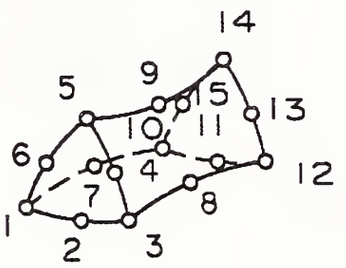
```
E1=1,2,3    E4=10,11,12    E7=1,7,10
E2=3,4,5    E5=12,13,14    E8=3,8,12
E3=5,6,1    E6=14,15,10    E9=5,9,14
F1=1,2,3,4,5,6
F2=10,11,12,13,14,15
F3=1,2,3,8,12,11,10,7
F4=3,4,5,9,14,13,12,8
F5=5,6,1,7,10,15,14,9
```

16. CSOW - Cubic Solid Wedge

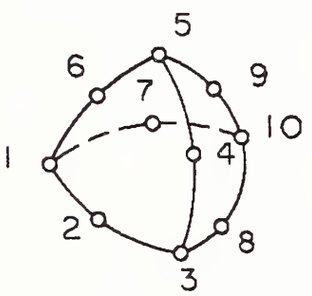
```
E1=1,2,3,4   E4=16,17,18,19   E7=1,10,13,16
E2=4,5,6,7   E5=19,20,21,22   E8=4,11,14,19
E3=7,8,9,1   E6=22,23,24,16   E9=7,12,15,22
F1=1,2,3,4,5,6,7,8,9
F2=16,17,18,19,20,21,22,23,24
F3=1,2,3,4,11,14,19,18,17,16,13,10
F4=4,5,6,7,12,15,22,21,20,19,14,11
F5=7,8,9,1,10,13,16,24,23,22,15,12
```



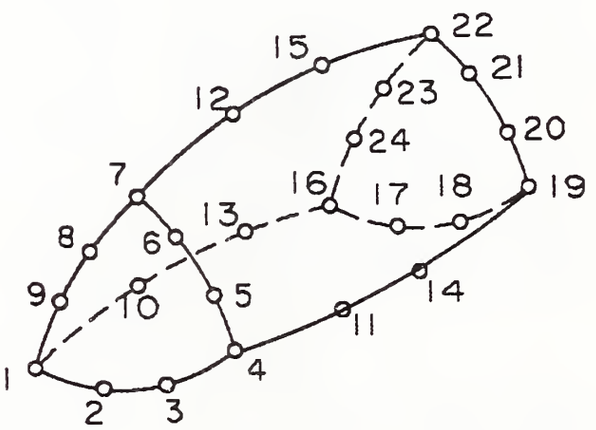
12. LINEAR SOLID TETRAHEDRON



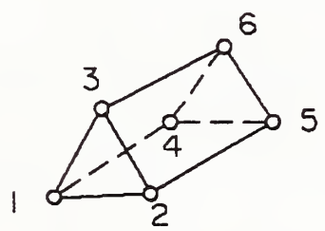
15. PARABOLIC SOLID WEDGE



13. PARABOLIC SOLID TETRAHEDRON



16. CUBIC SOLID WEDGE



14. LINEAR SOLID WEDGE

Figure 45. Finite Element Topology Set (continued)

3.21 FINITE ELEMENT ENTITY (TYPE 136)

17. LSO - Linear Solid

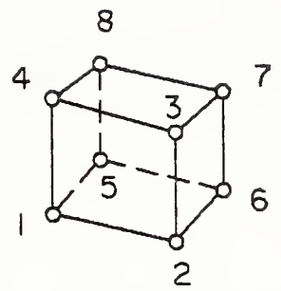
```
    E1=1,2    E5=5,6   E9=1,5
    E2=2,3    E6=6,7   E10=2,6
    E3=3,4    E7=7,8   E11=3,7
    E4=4,1    E8=8,5   E12=4,8
    F1=1,2,3,4
    F2=5,6,7,8
    F3=1,2,6,5
    F4=2,3,7,6
    F5=3,4,8,7
    F6=4,1,5,8
```

18. PSO - Parabolic Solid

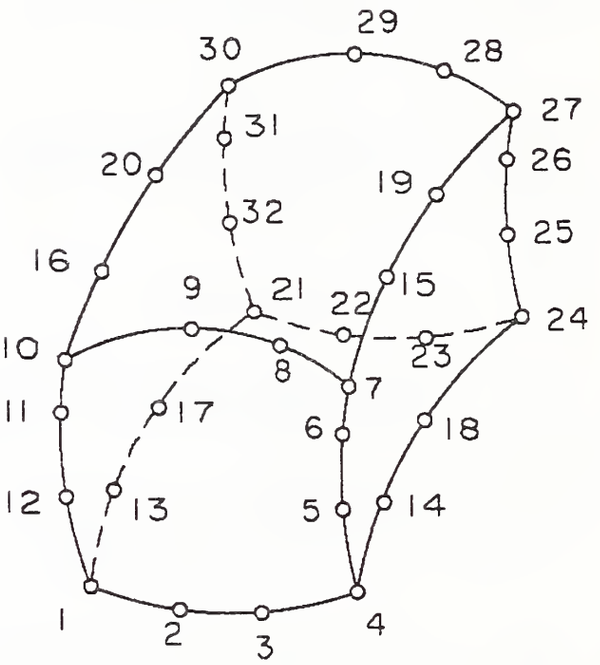
```
    E1=1,2,3       E7=17,18,19
    E2=3,4,5       E8=19,20,13
    E3=5,6,7       E9=1,9,13
    E4=7,8,1       E10=3,10,15
    E5=13,14,15    E11=5,11,17
    E6=15,16,17    E12=7,12,19
    F1=1,2,3,4,5,6,7,8
    F2=13,14,15,16,17,18,19,20
    F3=1,2,3,10,15,14,13,9
    F4=3,4,5,11,17,16,15,10
    F5=5,6,7,12,19,18,17,11
    F6=7,8,1,9,13,20,19,12
```

19. CSO - Cubic Solid

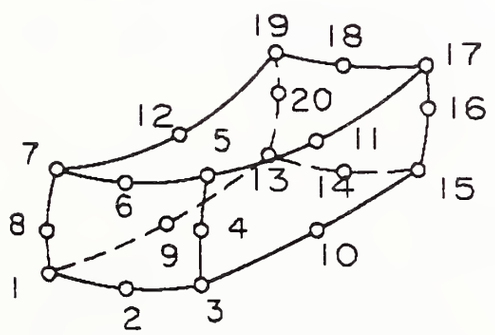
```
    E1=1,2,3,4
    E2=4,5,6,7
    E3=7,8,9,10
    E4=10,11,12,1
    E5=21,22,23,24
    E6=24,25,26,27
    E7=27,28,29,30
    E8=30,31,32,21
    E9=1,13,17,21
    E10=4,14,18,24
    E11=7,15, 19, 27
    E12=10,16,20,30
    F1=1,2,3,4,5,6,7,8,9,10,11,12
    F2=21,22,23,24,25,26,27,28,29,30,31,32
    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
```



17. LINEAR SOLID



19. CUBIC SOLID

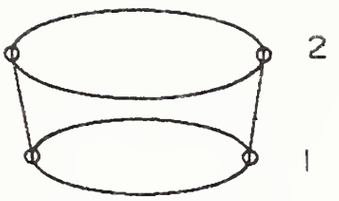


18. PARABOLIC SOLID

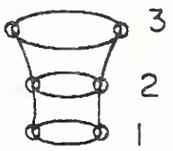
Figure 45. Finite Element Topology Set (continued)

3.21 FINITE ELEMENT ENTITY (TYPE 136)

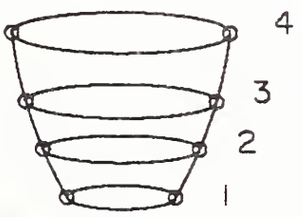
```
20. ALLIN - Axisymmetric Linear Line
    E1=1,2      No Faces


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=3,1


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
```



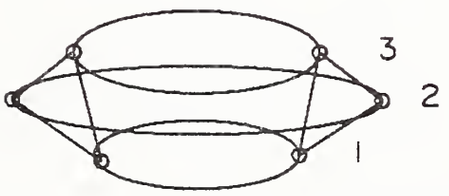
20. AXISYMMETRIC LINEAR LINE



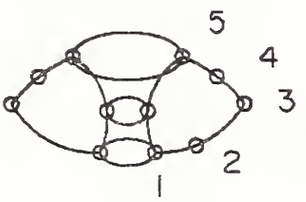
21. AXISYMMETRIC PARABOLIC LINE



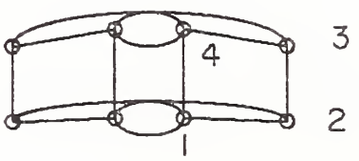
22. AXISYMMETRIC CUBIC LINE



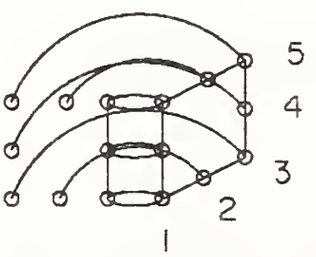
23. AXISYMMETRIC LINEAR TRIANGLE



24. AXISYMMETRIC PARABOLIC TRIANGLE



25. AXISYMMETRIC LINEAR QUADRILATERAL



26. AXISYMMETRIC PARABOLIC QUADRILATERAL

Figure 45. Finite Element Topology Set (continued)

27. SPR - Spring
    No edges or faces

28. GSPR - Grounded Spring

29. DAMP - Damper

30. GDAMP - Grounded damper

31. MASS - Mass

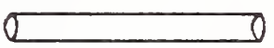
32. RBDY - Rigid Body

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

27. SPRING

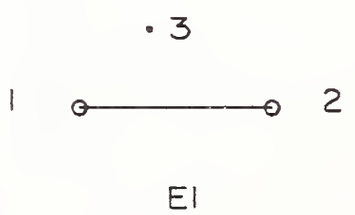
31. MASS

28. GROUNDED SPRING

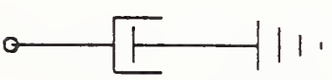


32. RIGID BODY

29. DAMPER



33. THREE NODED BODY



30. GROUNDED DAMPER

Figure 45. Finite Element Topology Set (continued)

3.22 Nodal Displacement and Rotation Entity (Type 138)

The Nodal Displacement and Rotation Entity is used to communicate finite element postprocessing 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 (Type 212) 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 (Type 134).

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
5 + NC	X11	Real	X-Incr. translation, first analysis case
6 + NC	Y11	Real	Y-Incr. translation
7 + NC	Z11	Real	Z-Incr. translation
8 + NC	RX11	Real	RX-Incr. rotation
9 + NC	RY11	Real	RY-Incr. rotation
10 + NC	RZ11	Real	RZ-Incr. rotation
.	.	.	.
.	.	.	.
7*NC - 1	X1NC	Real	X-Incr. translation, last analysis case
.	Y1NC	Real	Y-Incr. translation
.	Z1NC	Real	Z-Incr. translation
.	RX1NC	Real	RX-Incr. rotation
.	.	.	.
.	.	.	.
3+NC+(NN-1)*(6*NC+2)	NONN	Integer	Node number identifier for NNth node
.	NPNN	Pointer	Pointer to Node Directory Entry
.	XNN1	Real	X-Incr. translation, first analysis case
.	YNN1	Real	Y- " "
.	ZNN1	Real	Z- " "
.	RXNN1	Real	RX-Incr. rotation, first analysis case
.	RYNN1	Real	RY- " "
10+NC+(NN-1)*(6*NC+2)	RZNN1	Real	RZ- " "

3.22 NODAL DISPLACEMENT AND ROTATION ENTITY (TYPE 138)

Index	Name	Type	Description
.	.	.	.
.	.	.	.
.	XNNNC	Real	X-Incremental translation, last analysis case
.	YNNNC	Real	Y- " "
.	ZNNNC	Real	Z- " "
.	RXNNNC	Real	RX-Incremental rotation, last analysis case
.	RYNNNC	Real	RY- " "
2+NC+NN*(6*NC+2)	RZNNNC	Real	RZ- " "

Additional pointers as required (see Section 2.2.4.4.2).

3.23 Offset Surface Entity (Type 140)

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 parameterized 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 parameterized surface $S(u, v)$ given by:

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

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

$$N(u, v) = \frac{\partial S/\partial u \times \partial S/\partial v}{||\partial S/\partial u \times \partial S/\partial v||}.$$

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 46, consists of the vector (Nx, Ny, Nz) defined by:

$$(Nx, Ny, Nz) = N(Um, Vm)/||N(Um, Vm)||.$$

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

where, if the surface is bounded,

$$Um = (u1 + u2)/2 \quad \text{and} \quad Vm = (v1 + v2)/2$$

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 the formula for O that defines the offset surface.

3.23.1 Directory Data

Entity Type Number: 140

3.23.2 Parameter Data

Index	Name	Type	Description
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 Section 2.2.4.4.2).

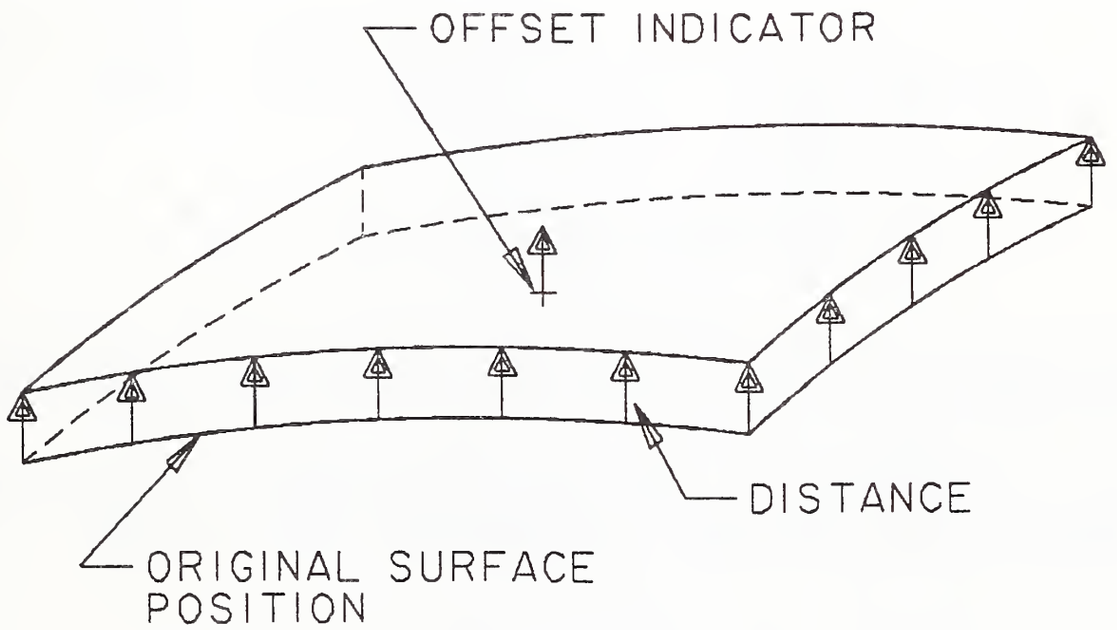


Figure 46. Offset Surface in 3-D Euclidean Space

3.24 Curve On A Parametric Surface Entity (Type 142)

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

$$\text{Let } S = S(u,v) = (x(u,v), y(u,v), z(u,v))$$

be a regular parameterized surface whose domain is a rectangle defined by

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

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

$$B(t) = (u(t), v(t)) \text{ for } a \leq t \leq b$$

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:

$$\begin{aligned} C(t) &\triangleq S \circ B(t) \\ &\triangleq S(B(t)) \\ &\triangleq S(u(t), v(t)) \\ &\triangleq (x(u(t),v(t)),\ y(u(t),v(t)),\ z(u(t),v(t))) \quad a \leq t \leq b. \end{aligned}$$

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 $S(u,v)$,

(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	BPTR	Pointer	Pointer to the entity that contains the definition of the curve B 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 = $S \circ B$ is preferred 2 = C is preferred 3 = C and $S \circ B$ are equally preferred

Additional pointers as required (see Section 2.2.4.4.2).

3.25 Trimmed (Parametric) Surface Entity (Type 144)

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 parameterized surface, whose untrimmed domain is a rectangle D consisting of those points (u, v) such that $a \leq u \leq b$ and $c \leq v \leq d$ 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:

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

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.

1. Outer boundary: there is exactly one. It lies in D, and in particular, it can be the boundary curve of D.
2. Inner boundary: there can be any number of them including zero. The set of inner boundaries satisfies two criteria:
   (a) The curves as well as their interiors are mutually disjoint.
   (b) 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, the trimmed surface being defined is untrimmed.

3.25 TRIMMED (PARAMETRIC) SURFACE ENTITY (TYPE 144)

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.
4	PTO	Pointer	Pointer to the simple closed curve (Curve on a Parametric Surface Entity), that constitutes the outer boundary of the trimmed surface or zero.
5	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.
.	.	.	.
.	.	.	.
.	.	.	.
4+N2	PTIN2	Pointer	Pointer to the last simple closed inner boundary curve (Curve on a Parametric Surface Entity).

Additional pointers as required (see Section 2.2.4.4.2).

3.26 Nodal Results Entity (Type 146)

The definition of this entity can be found in Appendix J (see Section J.3).

3.27 Element Results Entity (Type 148)

The definition of this entity can be found in Appendix J (see Section J.4).

3.28 Constructive Solid Geometry Models

The Constructive Solid Geometry (CSG) section contains standard formats for one of the two mostly widely used solid model representations—CSG. The other widely used representation, boundary representation (B-REP), is the subject of current definition effort.

The entities in this section can be thought of as being one of two types—geometric or structural. The geometric entities are volumetric primitives. These primitives include a block, wedge, cylinder, cone, sphere, torus, ellipsoid, solid of revolution and solid of linear extrusion. The model information for a primitive contains dimensions that define the shape of the primitive, point and vector coordinates that define the local coordinate system of the primitive, and an optional Directory Entry pointer to a transformation matrix which may be used to further position the primitive. If the point and vector coordinates defining the local coordinate system are not given values, the local coordinate system defaults to the global coordinate system. For the Solid of Revolution and Solid of Linear Extrusion Entities, the shape is partly defined indirectly, via a pointer to a planar boundary curve.

The structural entities are the Boolean Tree, Solid Instance, and Solid Assembly Entities. The Boolean Tree Entity contains pointers to the elements of the tree, and operations such as union, difference, and intersection to be performed on these elements. Elements may be primitives, other boolean trees or solid instances. There may also be a Directory Entry pointer to a transformation matrix to relocate the entire boolean resultant.

The Solid Instance Entity contains a pointer to an entity representing a solid and a Directory Entry pointer to a transformation matrix by which the entity is to be transformed. It is a copy of the solid entity relocated in global space. The solid entity may be a primitive, boolean tree, another solid instance, or an assembly.

A solid assembly is a collection of items that share a fixed geometric relationship. The relationship is a logical one and is not to be confused with a boolean union. If the faces of different items in an assembly touch, they are not removed, as they would be in a boolean union. The items of an assembly may include primitives, boolean trees, other assemblies, and solid instances. Corresponding to each item pointed to by the assembly is an optional pointer to a transformation matrix to be applied to that item. Thus, each item of the assembly can be moved independently. There is also an optional directory entry pointer to a global transformation matrix to be applied to the entire assembly of items. This global transformation matrix is applied after each of the individual transformation matrices are applied.

The description of a solid model is an acyclic directed graph. The nodes in the graph are the various geometric and structural entities. This type of graph is like a tree structure, except that the branches of this graph may reconvene as a move is made down the graph, where down is the general direction from root to terminal node. There may be any number of root nodes, which represent the actual solid models. A root may even be within the branches of another root's graph.

The terminal nodes are the primitives—the geometric entities. All the other nodes are structural entities. The structural entities are all able to point to each of the other structural entities as well as to primitives, with one exception. The boolean tree cannot point to an assembly.

A CSG solid model is thus represented by appropriately combining geometric entities with structural entities to create a graph structure.

The CSG Primitive Entities are a defined set of solid modeling primitive constructs to be used in all solid modelers—either directly in CSG modelers or in other types of modelers after conversion. CSG primitive entities include the following:

Entity Type Number	Entity Type
150	Block
152	Right Angular Wedge
154	Right Circular Cylinder
156	Right Circular Cone Frustum
158	Sphere
160	Torus
162	Solid of Revolution
164	Solid of Linear Extrusion
168	Ellipsoid

These primitive entities can be combined into more complex CSG solids using the following entities:

Entity Type Number	Entity Type
180	Boolean Tree
430	Solid Instance
184	Solid Assembly

3.28.1 Block Entity (Type 150). The block is a rectangular parallelepiped, defined with one vertex at (X1,Y1,Z1) and three edges lying along the local +X, +Y, and +Z axes. The local X-axis is defined by the unit vector (I1,J1,K1) and the local Z-axis by (I2,J2,K2). The local Y-axis is derived by taking the cross product of Z into X. The resulting local system must be orthogonal, with (I1,J1,K1) values having the highest accuracy precedence. The block is specified by the positive lengths (LX,LY,LZ) along these axes as shown in Figure 47.

3.28.1.1 Directory Data

Entity Type Number: 150

3.28.1.2 Parameter Data

Index	Name	Type	Description
1	LX	Real	Length in the local X-direction
2	LY	Real	Length in the local Y-direction
3	LZ	Real	Length in the local Z-direction
4	X1	Real	Corner point coordinates (default (0,0,0))
5	Y1	Real	
6	Z1	Real	
7	I1	Real	Unit vector defining local X-axis (default (1,0,0))
8	J1	Real	
9	K1	Real	
10	I2	Real	Unit vector defining local Z-axis (default (0,0,1))
11	J2	Real	
12	K2	Real	Must be orthogonal (see above) to (I1,J1,K1)

Additional pointers as required (see Section 2.2.4.4.2).

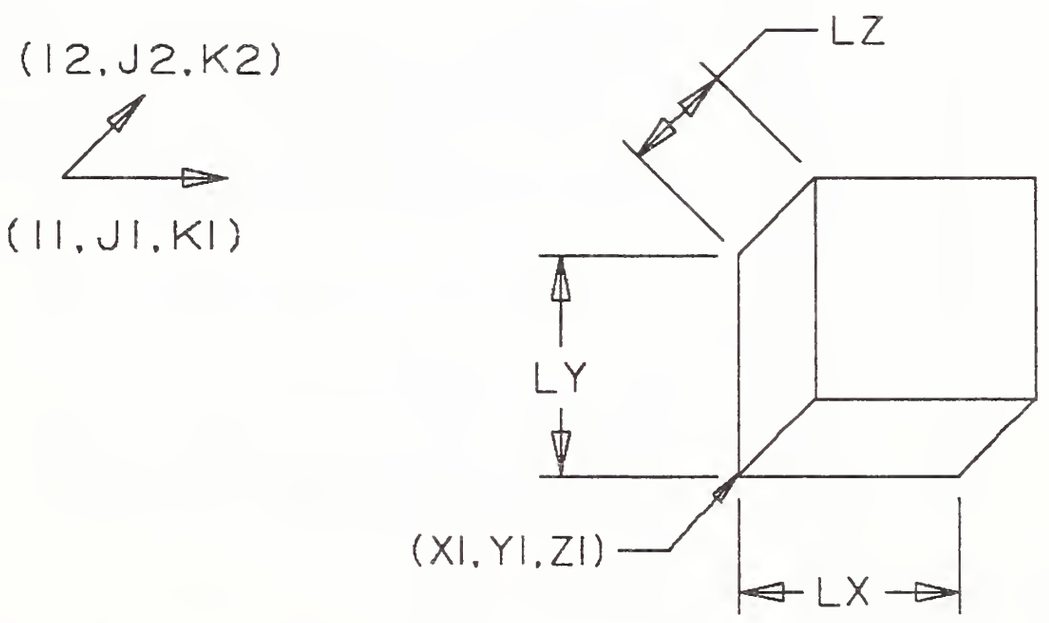


Figure 47. Parameters of the CSG Block Entity

3.28.2 Right Angular Wedge Entity (Type 152). The right angular wedge is defined with one vertex at (X1,Y1,Z1) and three orthogonal edges lying along the local +X, +Y, and +Z axes. Figure 48 shows one example. A triangular/trapezoidal face lies in the local XY-plane. The local X-axis is defined by the unit vector (I1,J1,K1) and the local Z-axis by (I2,J2,K2). The local Y-axis is derived by taking the cross product of Z into X. The resulting local system must be orthogonal, with (I1,J1,K1) values having the highest accuracy precedence. The wedge is specified by the positive lengths LX, LY, LZ along these axes and the length LTX (where LTX<LX) in the local positive X-direction at a distance LY (in the local Y-direction) from the local X-axis. If LTX=0, the wedge has five faces. Otherwise, it has six faces.

3.28.2.1 Directory Data

Entity Type Number: 152

3.28.2.2 Parameter Data

Index	Name	Type	Description
1	LX	Real	Length in the local X-direction at Y=0
2	LY	Real	Length in the local Y-direction
3	LZ	Real	Length in the local Z-direction
4	LTX	Real	Length in the local X-direction at distance LY from local X-axis
5	X1	Real	Corner point coordinates (default (0,0,0))
6	Y1	Real	
7	Z1	Real	
8	I1	Real	Unit vector defining local X-axis (default (1,0,0))
9	J1	Real	
10	K1	Real	
11	I2	Real	Unit vector defining local Z-axis (default (0,0,1))
12	J2	Real	
13	K2	Real	Must be orthogonal (see above) to (I1,J1,K1)

Additional pointers as required (see Section 2.2.4.4.2).

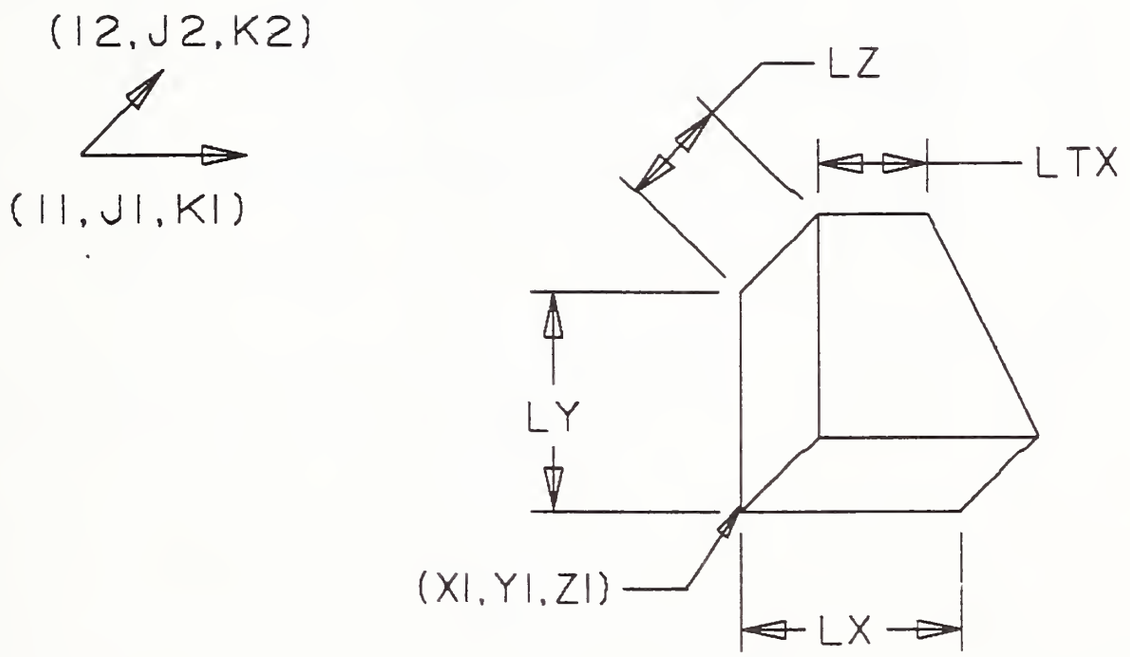


Figure 48. Parameters of the CSG Right Angular Wedge Entity

3.28.3 Right Circular Cylinder Entity (Type 154). The right circular cylinder is defined by the center of one circular cylinder face, a unit vector, a height, and a radius as shown in Figure 49. The faces are perpendicular to the unit vector in the axis direction (I1,J1,K1) and are circular discs with the specified radius R (where R>0). The height H (where H> 0) is the distance from the first circular face center in the positive direction of the unit vector to the second circular face center.

3.28.3.1 Directory Data

Entity Type Number: 154

3.28.3.2 Parameter Data

Index	Name	Type	Description
1	H	Real	Cylinder height
2	R	Real	Cylinder radius
3	X1	Real	First face center coordinates (default (0,0,0))
4	Y1	Real	
5	Z1	Real	
6	I1	Real	Unit vector in axis direction (default (0,0,1))
7	J1	Real	
8	K1	Real	

Additional pointers as required (see Section 2.2.4.4.2).

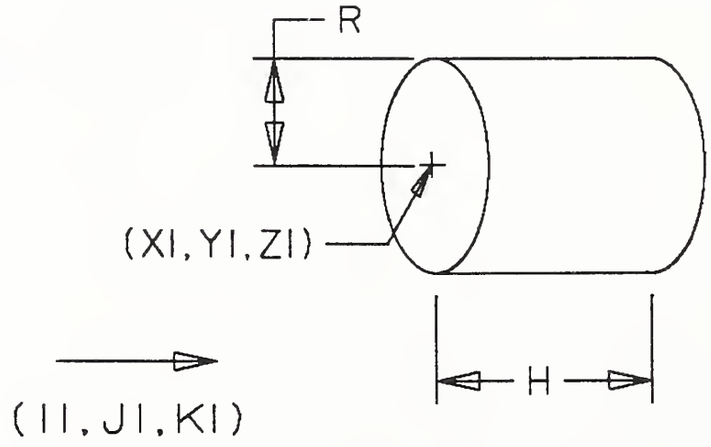


Figure 49. Parameters of the CSG Right Circular Cylinder Entity

3.28.4 Right Circular Cone Frustum Entity (Type 156). The right circular cone frustum is defined by the center of the larger circular face of the frustum (X1,Y1,Z1), its radius R1, a unit vector in the axis direction (I1,J1,K1), a height H in this direction, and a second circular face with radius R2, where R1 > R2 $\geq$ 0 and H > 0. As shown by Figure 50, the circular faces are perpendicular to the unit vector (I1,J1,K1).

3.28.4.1 Directory Data

Entity Type Number: 156

3.28.4.2 Parameter Data

Index	Name	Type	Description
1	H	Real	Height
2	R1	Real	Larger face radius
3	R2	Real	Smaller face radius (zero for cone apex - default)
4	X1	Real	Larger face center coordinates (default (0,0,0))
5	Y1	Real	
6	Z1	Real	
7	I1	Real	Unit vector in axis direction (default (0,0,1))
8	J1	Real	
9	K1	Real	

Additional pointers as required (see Section 2.2.4.4.2).

3.28.4 RIGHT CIRCULAR CONE FRUSTUM ENTITY (TYPE 156)

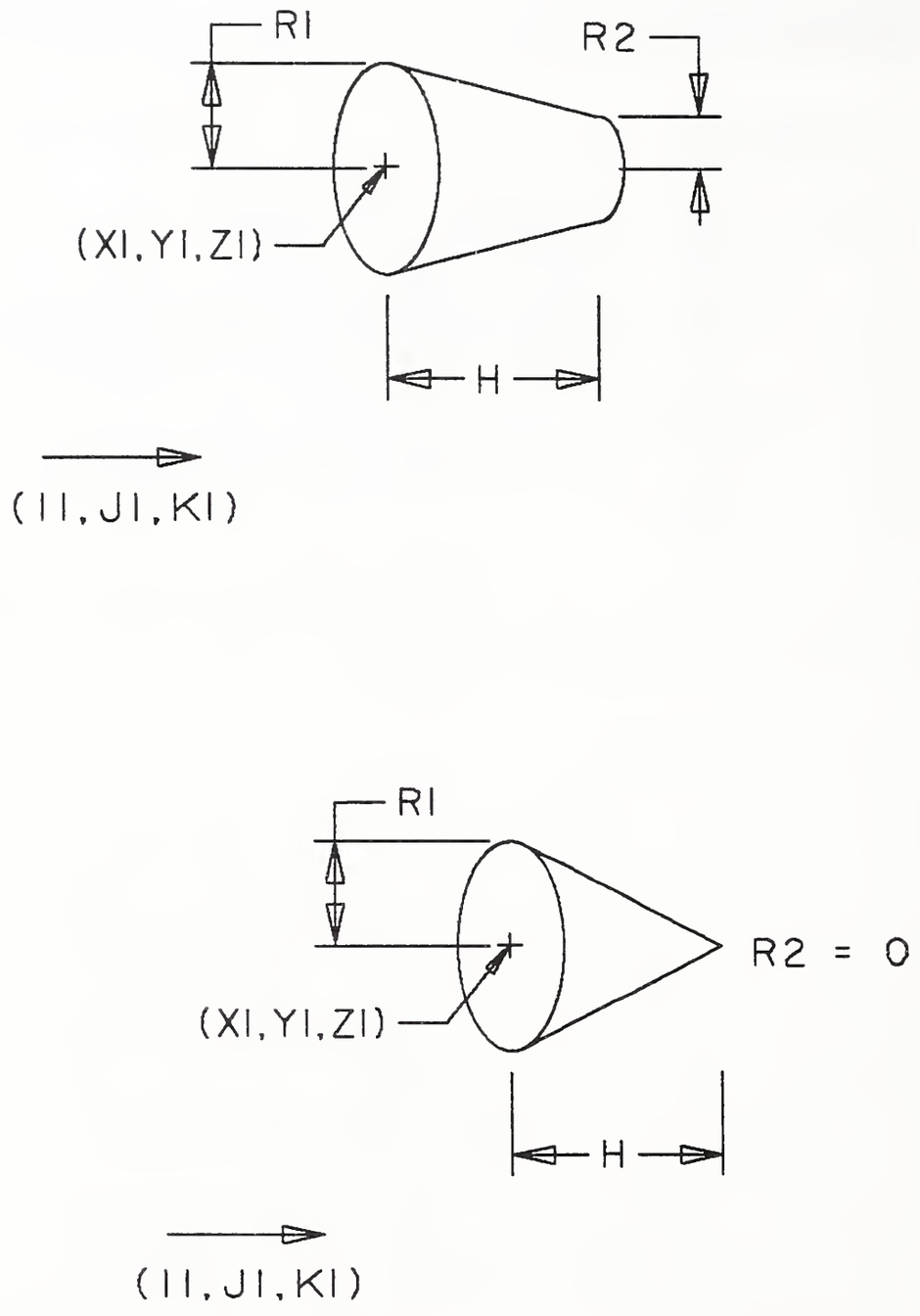


Figure 50. Parameters of the CSG Right Circular Cone Frustum Entity

3.28.5 Sphere Entity (Type 158). The sphere is defined with its center coordinates at (X1,Y1,Z1) and a radius R, where R > 0. Figure 51 shows one example.

3.28.5.1 Directory Data

Entity Type Number: 158

3.28.5.2 Parameter Data

Index	Name	Type	Description
1	R	Real	Radius
2	X1	Real	Center coordinates (default (0,0,0))
3	Y1	Real	
4	Z1	Real	

Additional pointers as required (see Section 2.2.4.4.2).

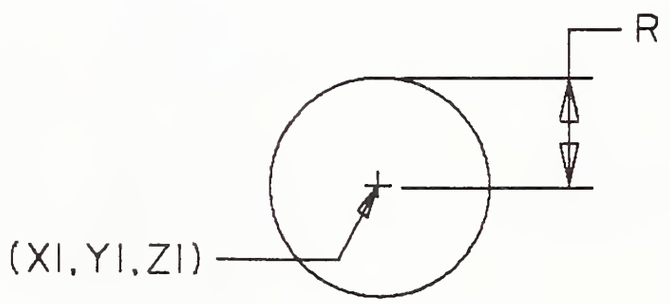


Figure 51. Parameters of the CSG Sphere Entity

3.28.6 Torus Entity (Type 160). The torus is the solid formed by revolving a circular disc about a specified coplanar axis. R1 is the distance from the axis to the center of the defining disc and R2 is the radius of the defining disc, where R1 > R2 > 0. The torus is located with its center at (X1, Y1,Z1), and its axis is oriented in the (I1,J1,K1) direction, as shown in Figure 52.

3.28.6.1 Directory Data

Entity Type Number: 160

3.28.6.2 Parameter Data

Index	Name	Type	Description
1	R1	Real	Distance from center of torus to center of circular disc to be revolved (perpendicular to axis)
2	R2	Real	Radius of circular disc
3	X1	Real	Torus center coordinates (default (0,0,0))
4	Y1	Real	
5	Z1	Real	
6	I1	Real	Unit vector in axis direction (default (0,0,1))
7	J1	Real	
8	K1	Real	

Additional pointers as required (see Section 2.2.4.4.2).

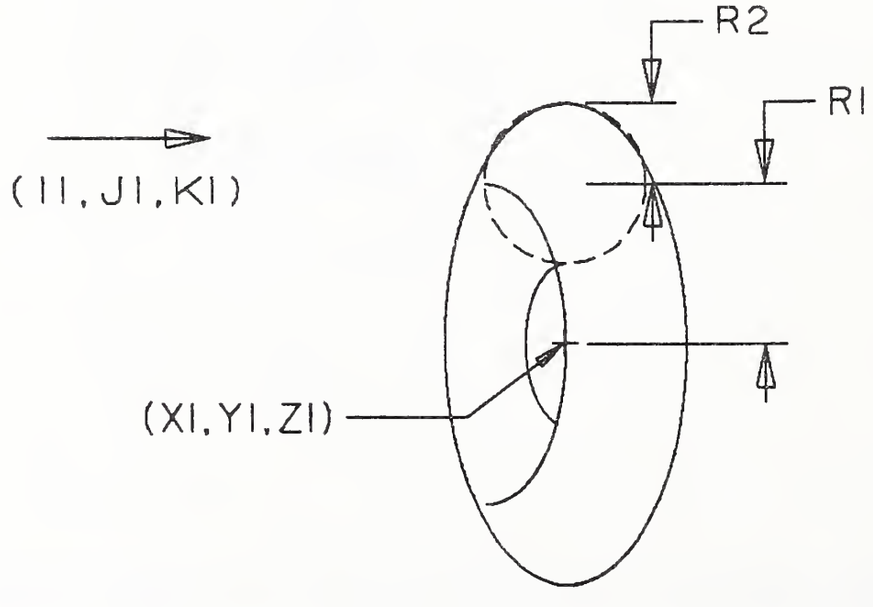


Figure 52. Parameters of the CSG Torus Entity

3.28.7 Solid of Revolution Entity (Type 162). The solid of revolution is defined by revolving the area determined by a planar curve about a specified axis (which must be in the same plane) through a given fraction of full rotation F, $(0 < F \leq 1)$ using the right hand rule (counterclockwise when viewed from the positive direction). The curve must not intersect itself. It must not cross the axis but may touch it. Figure 53 shows one example.

Two form numbers are used to indicate how the area is determined from the curve. If the curve is closed, the form number shall be set to 1, and the area enclosed by the curve is used. If the curve is not closed and the form number = 0, projections are made from the ends of the curve to the rotation axis and the area enclosed by the curve, the projections, and the axis is used. In this case, the curve must be such that it does not intersect the projections, except at the end points. If the curve is not closed and the form number = 1, the curve is closed by adding a line connecting its end points and the area enclosed by the curve and the added line is used. In this case, the curve must not intersect the added line, except at the end points.

3.28.7.1 Directory Data

Entity Type Number: 162

Form	Meaning
0	Curve closed to axis
1	Curve closed to itself

3.28.7.2 Parameter Data

Index	Name	Type	Description
1	PTR	Pointer	DE Sequence Number of curve entity to be revolved. Must be coplanar with rotation axis.
2	F	Real	Fraction of full rotation through which the curve entity will be revolved $(0 < F \leq 1)$ - counterclockwise when viewed from the positive direction; default 1
3	X1	Real	Coordinates of point on axis (default (0,0,0))
4	Y1	Real	
5	Z1	Real	
6	I1	Real	Unit vector in axis direction (default (0,0,1))
7	J1	Real	
8	K1	Real	

Additional pointers as required (see Section 2.2.4.4.2).

3.28.7 SOLID OF REVOLUTION ENTITY (TYPE 162)

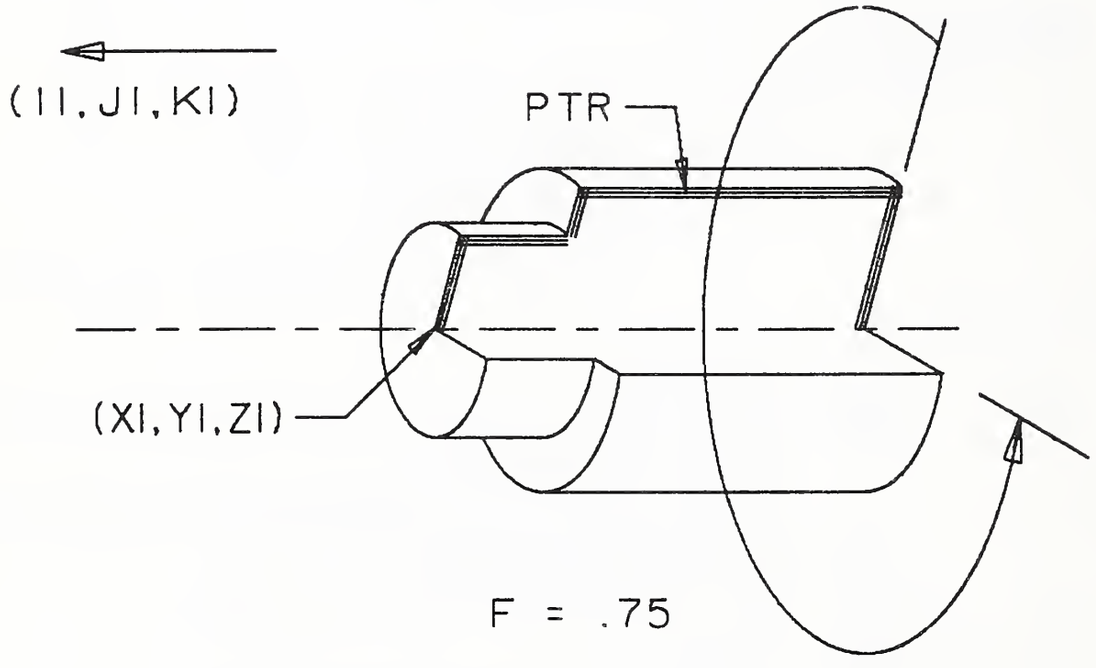


Figure 53. Parameters of the CSG Solid of Revolution Entity

3.28.8 Solid of Linear Extrusion Entity (Type 164). The solid of linear extrusion is defined by translating an area determined by a planar curve. The curve as indicated by PTR in Figure 54 must be closed and non-intersecting. The direction of the translation is defined by a unit vector (I1,J1,K1) and the length of the translation is defined by L, where L > 0. The vector (I1,J1,K1) must not be coplanar with the closed curve.

3.28.8.1 Directory Data
Entity Type Number: 164

3.28.8.2 Parameter Data

Index	Name	Type	Description
1	PTR	Pointer	Pointer to closed curve entity
2	L	Real	Length of extrusion along the vector positive direction
3	I1	Real	Unit vector specifying direction of extrusion (default (0,0,1))
4	J1	Real	
5	K1	Real	

Additional pointers as required (see Section 2.2.4.4.2).

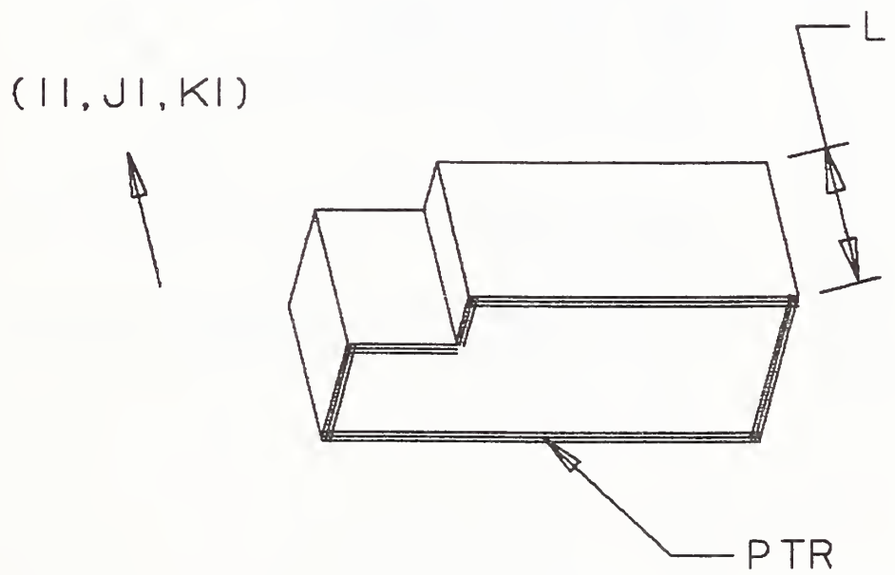


Figure 54. Parameters of the CSG Solid of Linear Extrusion Entity

3.28.9 Ellipsoid Entity (Type 168). The ellipsoid is a solid bounded by the surface defined by:

$$\frac{X^2}{LX^2} + \frac{Y^2}{LY^2} + \frac{Z^2}{LZ^2} = 1$$

when centered at the origin and aligned with its major axis (LX) in the X direction and with the minor axis (LZ) in the Z direction. A major axis of an ellipsoid can be found by choosing a point on the surface farthest from the center and constructing the line from that point through the center. The plane through the center perpendicular to this major axis intersects the surface of the ellipsoid in an ellipse. The other two axes of the ellipsoid are the axes of this ellipse.

The ellipsoid is defined with its center at (X1,Y1,Z1) and its three axes coincident with the local X, Y, Z axes, as shown in Figure 55. The local X-axis is defined by the unit vector (I1,J1,K1) and the local Z-axis by (I2,J2,K2). The local Y-axis is derived by taking the cross product of Z into X. The resulting local system must be orthogonal, with (I1,J1,K1) values having the highest accuracy precedence. The ellipsoid is specified by positive lengths (LX, LY, and LZ respectively, where LX $\geq$ LY $\geq$ LZ $>$ 0) from the local origin to the surface along the local +X, +Y, +Z axes.

3.28.9.1 Directory Data
Entity Type Number: 168

3.28.9.2 Parameter Data

Index	Name	Type	Description
1	LX	Real	Length in the local X-direction
2	LY	Real	Length in the local Y-direction
3	LZ	Real	Length in the local Z-direction
4	X1	Real	Coordinates of point in center of ellipsoid
5	Y1	Real	(default (0,0,0))
6	Z1	Real	
7	I1	Real	Unit vector defining local X-axis (Ellipsoid major axis)
8	J1	Real	(default (1,0,0))
9	K1	Real	
10	I2	Real	Unit vector defining local Z-axis (Ellipsoid minor axis)
11	J2	Real	(default (0,0,1)) Must be orthogonal (see above) to (I1,J1,K1)
12	K2	Real	

Additional pointers as required (see Section 2.2.4.4.2).

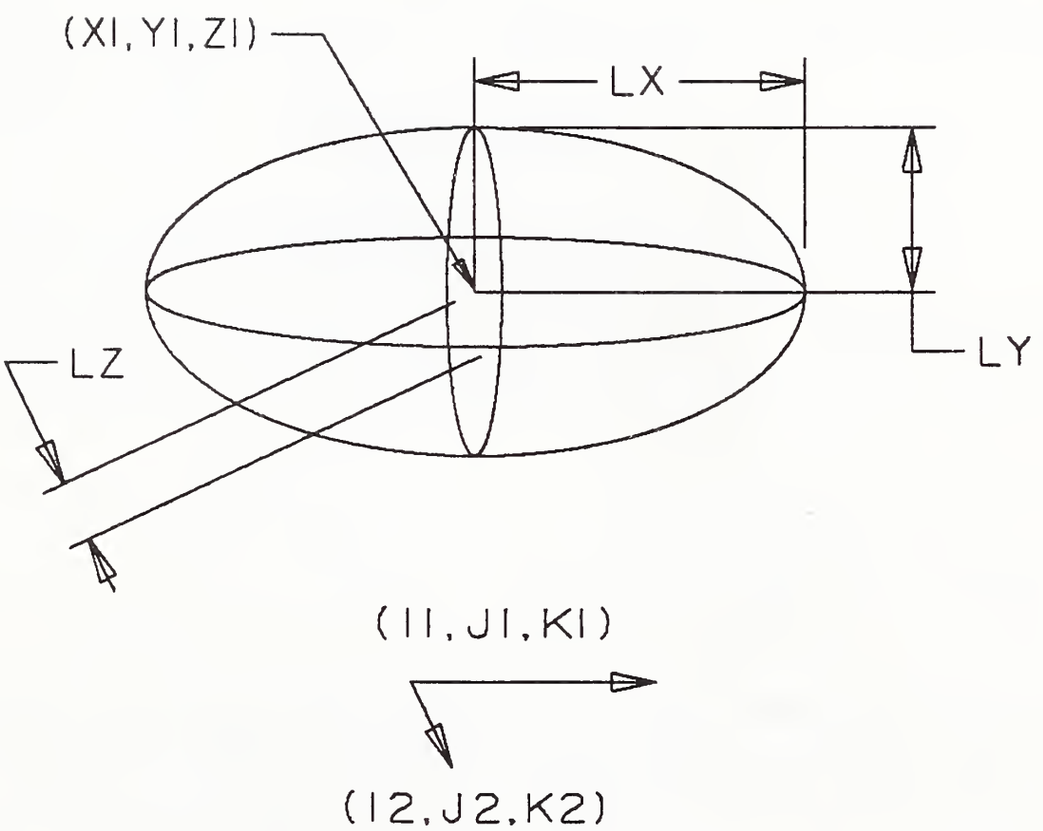


Figure 55. Parameters of the CSG Ellipsoid Entity

3.28.10 Boolean Tree Entity (Type 180). The Boolean tree describes a binary tree structure composed of regularized Boolean operations and operands, in postorder notation. A regularized Boolean operation is defined as the closure of the interior of the result of a Boolean set operation. Specifically, denote the interior of a set X by X_o, the closure of X by $\overline{X}$, and use $\cup^*$, $\cap^*$, and $-^*$ to denote the regularized Boolean operations union, intersection, and difference, respectively. Then:

$$\begin{aligned} X \cup^* Y &= \overline{(X \cup Y)_o} \\ X \cap^* Y &= \overline{(X \cap Y)_o} \\ X -^* Y &= \overline{(X - Y)_o} \end{aligned}$$

Since the topological space under consideration is a 3-dimensional space, all lower dimensional entities resulting from these operations will disappear. A discussion of regularized Boolean operations can be found in [TILO80].

All operations are assigned integers as follows:

Integer	Operation
1	Union
2	Intersection
3	Difference

Allowable operands are:

- Primitive entities
- Boolean Tree Entities
- Solid Instance Entities

The parameter data entries for the Boolean Tree Entity can be operation codes (integers) or pointers to operands. A positive (or unsigned) value in a parameter data entry implies an operation code; a negative value implies the absolute value is to be taken as a pointer to an operand.

A transformation matrix may be pointed to by Field 7 of the DE to position the resulting solid in any desired manner.

3.28.10.1 Directory Data
Entity Type Number: 180

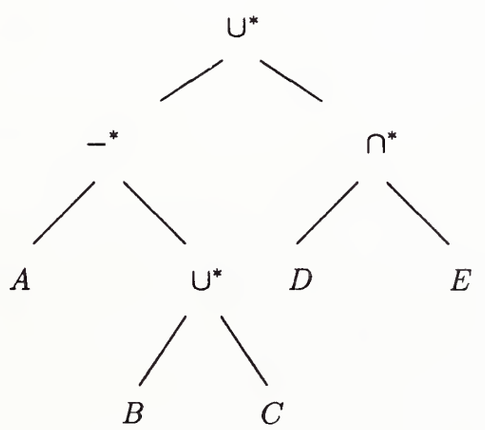
3.28.10.2 Parameter Data

Index	Name	Type	Description
1	N	Integer	Length of post-order notation, including operations and operands (N > 2)
2	PTR(1)	Pointer	Negated pointer to the first operand
3	PTR(2)	Pointer	Negated pointer to the second operand
4	PTR(3)	Pointer	Negated pointer for the third operand
	or	or	or
	IOP(1)	Integer	Integer for the first operation
.	.	.	.
.	.	.	.
.	.	.	.
N	PTR(M)	Pointer	Negated pointer for last operand
	or	or	or
	IOP(L-1)	Integer	Integer for next-to-last operation
N+1	IOP(L)	Integer	Integer for last operation

Notes: Parameters 2 and 3 will always be operands and thus will be negative numbers.
As L is the number of operations, and M is the number of operands, N = L+M.

Additional pointers as required (see Section 2.2.4.4.2).

The following is an example of a Boolean tree composed of five operands and four operations.



Ordinary infix notation:

$(A -^* (B \cup^* C)) \cup^* (D \cap^* E)$

Postorder notation:

$A\ B\ C\ \cup^*\ -^*\ D\ E\ \cap^*\ \cup^*$

Parameters: 9 A B C 1 3 D E 2 1

(A, B, C, D, & E are negative values representing pointers to operands.)

3.28.10 BOOLEAN TREE ENTITY (TYPE 180)

For the preceding example, the values are:

PARAMETER	VALUE
1	9
2	PTRA (negative)
3	PTRB (negative)
4	PTRC (negative)
5	1
6	3
7	PTRD (negative)
8	PTRE (negative)
9	2
10	1

Additional pointers as required (see Section 2.2.4.4.2).

3.28.11 Solid Instance Entity (Type 430). The Solid Instance Entity provides a mechanism for replicating a solid representation. The solid pointed to in this entity is allowed to be:

- Primitive Entity
- Boolean Tree Entity
- Solid Assembly Entity
- Solid Instance Entity

Note that a transformation matrix may be pointed to by Field 7 of the DE to position this instance in any desired manner.

3.28.11.1 Directory Data
Entity Type Number: 430

3.28.11.2 Parameter Data

Index	Name	Type	Description
1	PTR	Pointer	Pointer to solid

Additional pointers as required (see Section 2.2.4.4.2).

3.28.12 Solid Assembly Entity (Type 184). A solid assembly is a collection of items which possess a shared fixed geometric relationship. It differs from a union of the items in that each item retains its own structure, even if the items touch.

The transformation matrices are applied to the items individually before a matrix pointed to in Field 7 of the DE is applied to the collection. A value of zero in the pointer field indicates the identity matrix.

3.28.12.1 Directory Data

Entity Type Number: 184

3.28.12.2 Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of items
2	PTR1	Pointer	Pointer to item 1
.	.	.	.
.	.	.	.
.	.	.	.
N+1	PTRN	Pointer	Pointer to item N
N+2	PTRM1	Pointer	Pointer to Transformation Matrix for item 1
.	.	.	.
.	.	.	.
.	.	.	.
2N+1	PTRMN	Pointer	Pointer to Transformation Matrix for item N

Additional pointers as required (see Section 2.2.4.4.2).

4. Non-geometry

4.1 General

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

- Annotation Entities
- Structure Entities

Entity numbers from 200 through 499 are reserved for this Section. In addition, some non-geometric entities make use of the Copious Data Entity (Type 106).

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) Entities and a pointer to a General Note Entity.

For some annotation entities, a witness line or leader, although allowed, 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 56.

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 comprise several subordinate entities, each subordinate entity by its definition has the ability to stand alone and may require its own ZT displacement; it is likely, though not necessary, that each ZT displacement is identical.

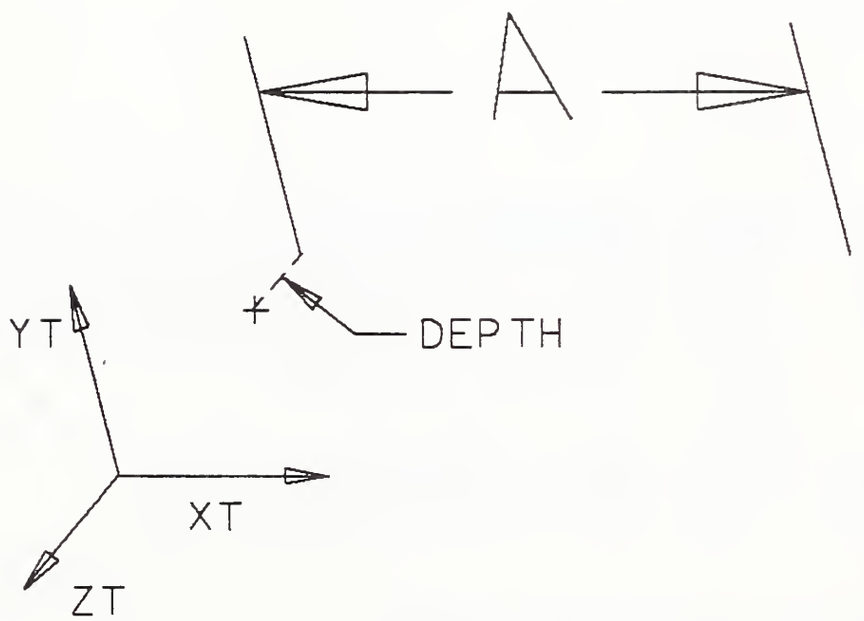


Figure 56. Interpretation of ZT Displacement (Depth) for Annotation Entities

4.2.4 Angular Dimension Entity (Type 202). An Angular Dimension Entity consists of a general note; zero, one, or two witness lines; two leaders; and an angle vertex point. Figure 57 indicates the construction used. Refer to Figure 58 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 Section 3.1.2 for information relating to the use of the term counterclockwise).

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 57. The first example in Figure 58 illustrates a leader with three segments.

4.2.4.1 Directory Data
Entity Type Number: 202

4.2.4.2 Parameter Data

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to General Note Directory Entry
2	DEWIT1	Pointer	Pointer to first Witness Line Directory Entry or 0
3	DEWIT2	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	DEARRW1	Pointer	Pointer to 1st Leader Directory Entry
8	DEARRW2	Pointer	Pointer to 2nd Leader Directory Entry

Additional pointers as required (see Section 2.2.4.4.2).

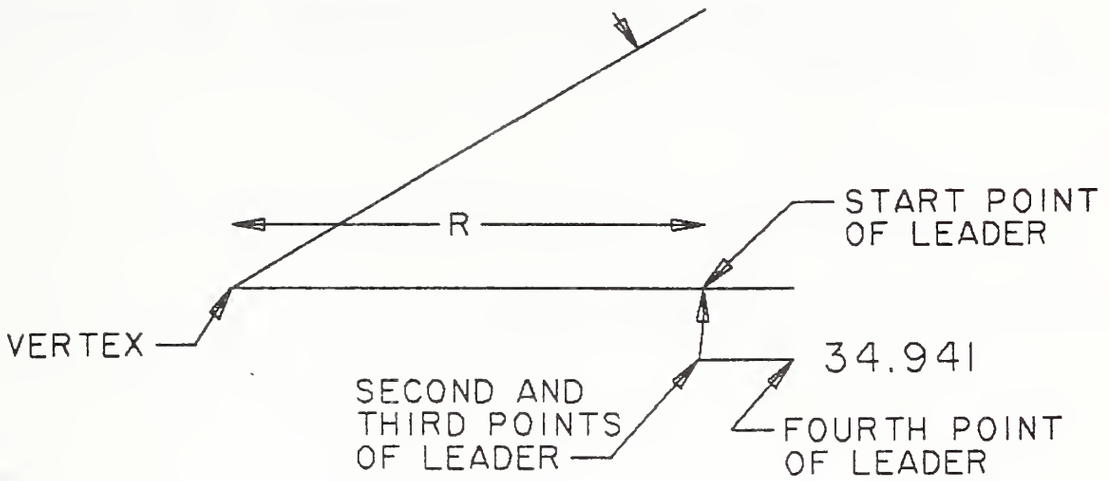


Figure 57. Construction of Leaders for the Angular Dimension Entity. The radius of the arc in the leader must be calculated between the vertex point and the start point of the leader.

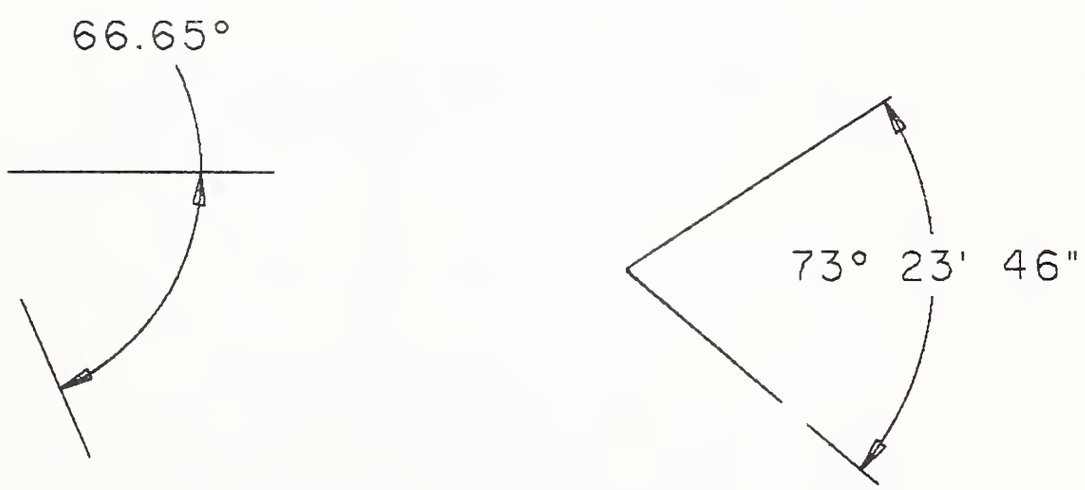


Figure 58. Examples Defined Using the Angular Dimension Entity

4.2.5 Centerline Entity (Type 106, Form 20-21). The Centerline Entity takes one of two forms. The first, as illustrated in Example 1 of Figure 59 appears as crosshairs and is normally used in conjunction with circles. The second type (Example 2) is a construction between 2 positions.

The Centerline entities are defined as Form 20 or 21 of the Copious Data Entity. 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

$$(X_n, Y_n, Z_n) \text{ to } (X_{n+1}, Y_{n+1}, Z_{n+1}) \quad \text{where } n = 1,3,5,...,N-1.$$

See Section 3.5 for parameters of the Centerline Entity.

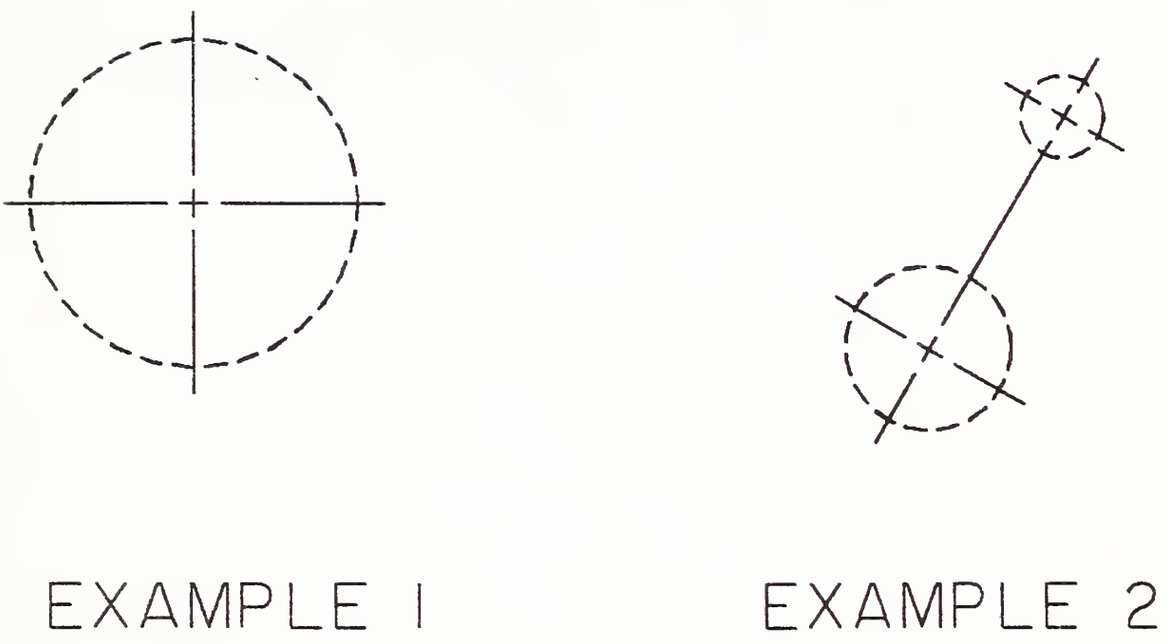


Figure 59. Examples Defined Using the Centerline Entity

4.2.6 Diameter Dimension Entity (Type 206). A Diameter Dimension Entity consists of a general note, one or two leaders, and an arc center point. Refer to Figure 60 for examples of the Diameter Dimension Entity. The arc center coordinates are used as reference in constructing the diameter dimension but have no effect on the dimension components.

4.2.6.1 Directory Data
Entity Type Number: 206

4.2.6.2 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 or zero
4	XT	Real	Arc center coordinates
5	YT	Real	

Additional pointers as required (see Section 2.2.4.4.2).

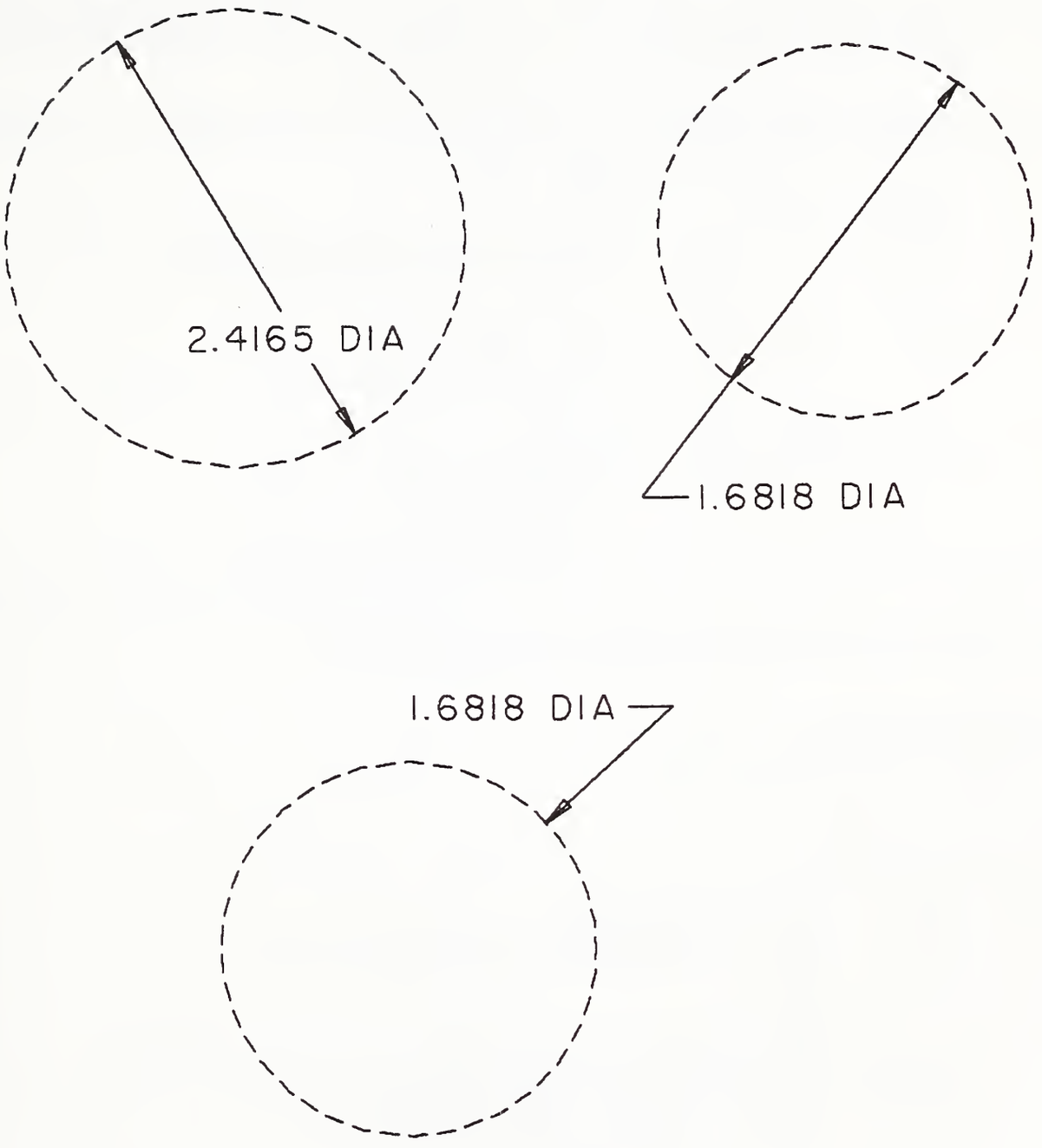


Figure 60. Examples Defined Using the Diameter Dimension Entity

4.2.7 Flag Note Entity (Type 208). A Flag Note Entity is label information formatted as shown in Figure 61. The rotation angle and location of the lower left corner coordinate in the Flag Note Entity override the General Note Entity (Type 212) rotation angle and placement.

The Flag Note Entity may be defined with or without leaders.

The flag note is constructed from information defined in the General Note Entity. This data is the character box height and character box width. For this reason, no geometric definition is explicit within the definition of the Flag Note Entity. The box containing the text (as defined in the General Note Entity) shall be centered in the flag note box of size (H x L).

The general note may consist of multiple text strings; however, they must share a common baseline. The number of characters shall not be greater than 10.

Variables:	H	=	Height
	CH	=	Character Height (from General Note)
	L	=	Length
	TW	=	Text Width (from General Note)
	T	=	Tip Length
	A	=	Rotation Angle (in radians)
Formulas:	H	=	2 * CH
	L	=	TW + 0.4 * CH
	T	=	0.5 * H / tan 35^0
Restrictions:			H shall never be less than 0.3 in.
			L shall never be less than 0.6 in.

Examples defined using the Flag Note Entity are shown in Figure 62.

4.2.7.1Directory Data

Entity Type Number: 208

4.2.7.2 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	DEARRW1	Pointer	Pointers to associated Leaders
.	.	.	.
.	.	.	.
.	.	.	.
6+N	DEARRWN	Pointer	Pointer to last Leader

Additional pointers as required (see Section 2.2.4.4.2).

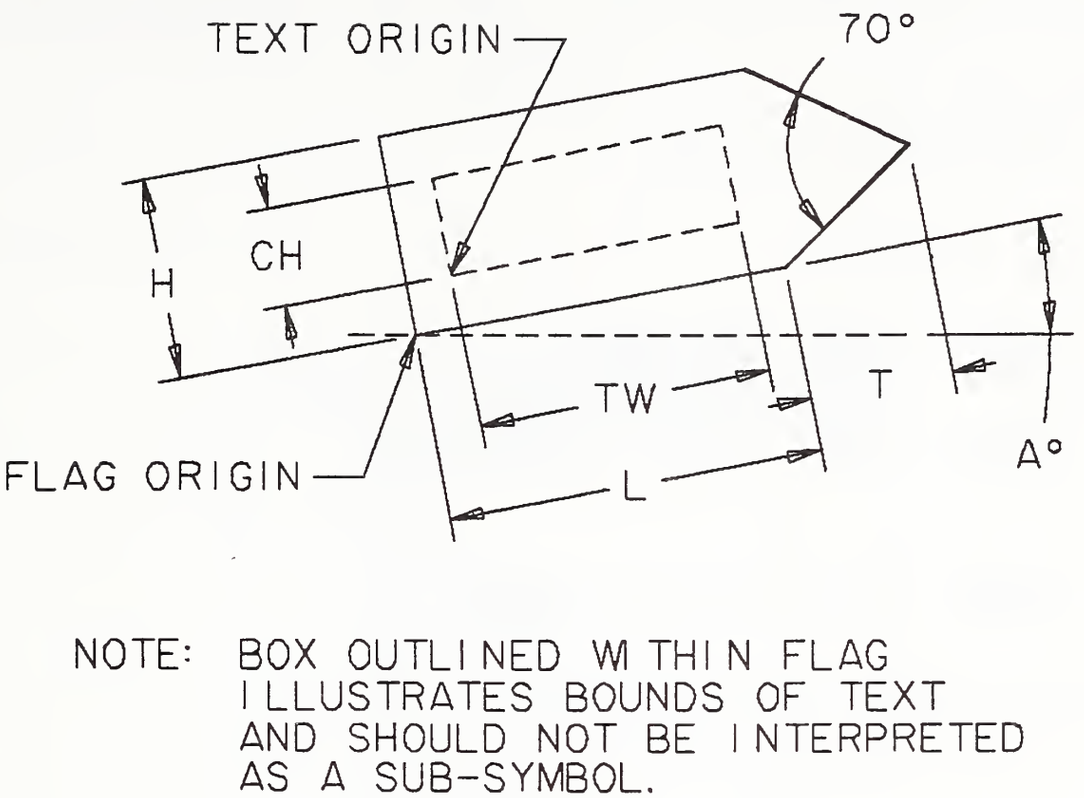


Figure 61. Parameters of the Flag Note Entity. Note that the box outlined within the flag illustrates the bounds of the text and is not a subsymbol.

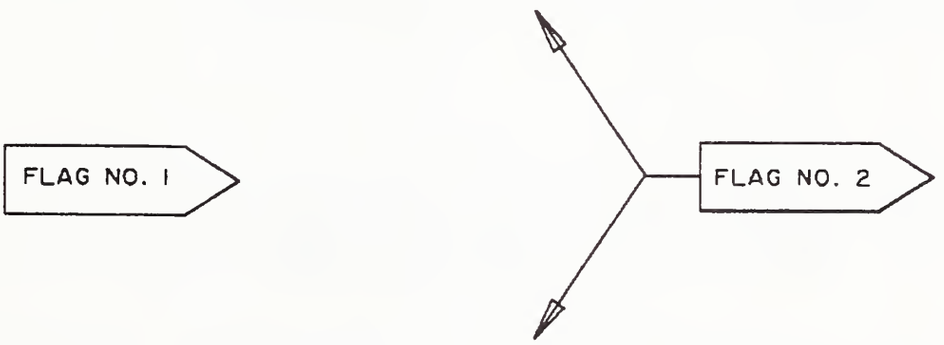


Figure 62. Examples Defined Using the Flag Note Entity

4.2.8 General Label Entity (Type 210). A General Label Entity consists of a general note with one or more associated leaders. Examples of general labels are shown in Figure 63.

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	DEARRW1	Pointer	Pointers to associated Leaders
.	.	.	.
.	.	.	.
.	.	.	.
N+2	DEARRWN	Pointer	Pointer to last Leader

Additional pointers as required (see Section 2.2.4.4.2).

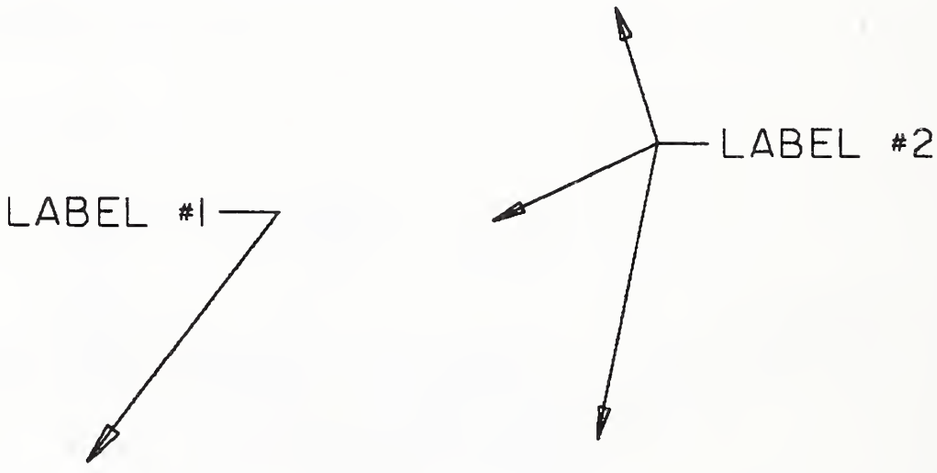


Figure 63. Examples Defined Using the General Label Entity

4.2.9 General Note Entity (Type 212). A General Note Entity consists of one or more text strings. Each text string contains text, a starting point, a text size, and an angle of rotation of the text. Examples of general notes are shown in Figure 64. 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 are defined:

Font	Description
0	Symbol Font (no longer recommended)
1	Standard Block
2	LeRoy
3	Futura
6	Comp 80
12	News Gothic
13	Lightline Gothic
14	Simplex Roman
17	Century Schoolbook
18	Helvetica
19	OCR-B [ISO1073] (see Appendix J)
1001	Symbol Font 1
1002	Symbol Font 2
1003	Drafting Font (see Appendix J)

Font 0 is an old symbol font and should no longer be used. Figure I1 in Appendix I is a mapping symbol definition for Font 0.

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 format characters. The intent of this font is for usage when the actual display of the characters is not critical for the application.

Font 19 is shown in Figure 65 and is defined in Appendix J (see Section J.5).

Fonts in the 1000 series display symbols mapped onto ASCII characters as shown in Figures 66, 67 and 68. They do not specify a character display font. Font 1003 is defined in Appendix J (see Section J.5).

Table 6 provides names for the graphical characters generated for each valid code.

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 69.

Within definition space, the parameters for the text block are applied in the following order (See Figure 70).

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 start of the left-most (first) text character/symbol in the positive XT direction along the text base line, and extends to the end of the

right-most (last) character/symbol, extending N characters/symbols and N-1 intercharacter spaces. The box height is measured in the positive YT direction and is the height of capital letters. It is equivalent to the symbol "h" used in Appendix C of [ANSI82]. Special symbols, such as those appearing in Appendix C of [ANSI82], which exceed "h" in height are centered vertically. Descenders and portions of symbols exceeding "h" extend outside the lower and/or upper borders of the box. The box height and width are measured before the rotation angle (A) is applied. The text start point is defined as the lower left corner of the first character/symbol box.

When a receiving system cannot fit a text string inside its defined text box, it shall give precedence to maintaining the full original character height, as defined by the text box height.

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).

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) text 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).

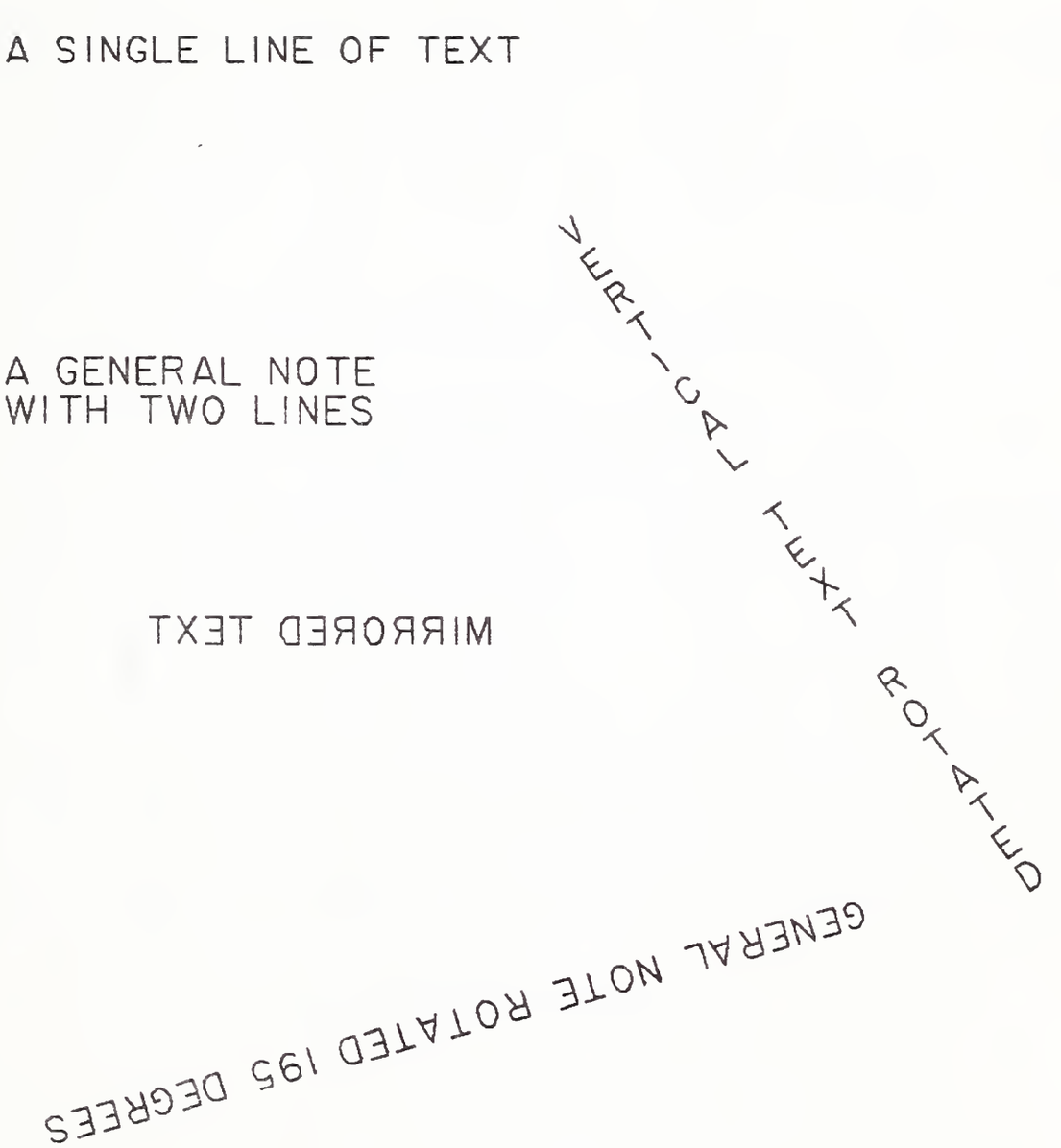


Figure 64. Examples of the General Note Entity

The definition of this font can be found in Appendix J (see Section J.5).

Figure 65. General Note Font 19

BL		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	○		

Figure 66. General Note Font 1001

4.2.9 GENERAL NOTE ENTITY (TYPE 212)

BL		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	ϕ
%	%	5	5	E	E	U	U	e	△	u	θ
&	&	6	6	F	F	V	V	f	√	v	γ
’	´	7	7	G	G	W	W	g	×	w	ψ
(	(	8	8	H	H	X	X	h	≡	x	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	∑		

Figure 67. General Note Font 1002

The definition of this font can be found
in Appendix J (see Section J.5).

Figure 68. General Note Font 1003

Table 6. Character Names for the Symbol and Drafting Fonts

Name	Symbol	Font†			
		1	1001	1002	1003
Space		32	32	32	see
Exclamation mark	!	33	33	33	App. J
Quotation marks	"	34	34	34	for this
Pound sign	#	35	35		Font
Plus/minus	±			35	
Dollar sign	$	36	36		
Degree symbol	°			36	
Percent sign	%	37	37	37	
Ampersand	&	38	38	38	
Apostrophe	’	39	39	39	
Left parenthesis	(	40	40	40	
Right parenthesis	)	41	41	41	
Asterisk	*	42	42	42	
Plus sign	+	43	43	43	
Comma	,	44	44	44	
Minus sign/hyphen	-	45	45	45	
Period	.	46	46	46	
Slash	/	47	47	47	
Numeric 0	0	48	48	48	
Numeric 1	1	49	49	49	
Numeric 2	2	50	50	50	
Numeric 3	3	51	51	51	
Numeric 4	4	52	52	52	
Numeric 5	5	53	53	53	
Numeric 6	6	54	54	54	
Numeric 7	7	55	55	55	
Numeric 8	8	56	56	56	
Numeric 9	9	57	57	57	
Colon	:	58	58	58	
Semi-colon	;	59	59	59	
Less than	<	60	60	60	
Equal sign	=	61	61	61	
Greater than	>	62	62	62	
Question mark	?	63	63	63	
Commercial at	@	64	64	64	
Upper case letter A	A	65	65	65	
Upper case letter B	B	66	66	66	
Upper case letter C	C	67	67	67	
Upper case letter D	D	68	68	68	
Upper case letter E	E	69	69	69	
Upper case letter F	F	70	70	70	
Upper case letter G	G	71	71	71	
Upper case letter H	H	72	72	72	

†Entries for each Font are decimal ASCII equivalent

Table 6. Character Names for the Symbol and Drafting Fonts (continued)

Name	Symbol	Font†			
		1	1001	1002	1003
Upper case letter I	I	73	73	73	see
Upper case letter J	J	74	74	74	App. J
Upper case letter K	K	75	75	75	for this
Upper case letter L	L	76	76	76	Font
Upper case letter M	M	77	77	77	
Upper case letter N	N	78	78	78	
Upper case letter O	O	79	79	79	
Upper case letter P	P	80	80	80	
Upper case letter Q	Q	81	81	81	
Upper case letter R	R	82	82	82	
Upper case letter S	S	83	83	83	
Upper case letter T	T	84	84	84	
Upper case letter U	U	85	85	85	
Upper case letter V	V	86	86	86	
Upper case letter W	W	87	87	87	
Upper case letter X	X	88	88	88	
Upper case letter Y	Y	89	89	89	
Upper case letter Z	Z	90	90	90	
Left bracket	[	91	91	91	
Backward slash	\	92	92	92	
Right bracket	]	93	93	93	
Caret	^	94	94	94	
Underscore	_	95	95	95	
Reverse quote	`	96	96	96	
Lower case letter a	a	97			
Angularity	∠		97		
Marker/symbol	⧖			97	
Lower case letter b	b	98			
Marker/symbol	⌖		98		
Division symbol	÷			98	
Lower case letter c	c	99			
Flatness	⏥		99		
Less than or equal	≤			99	
Lower case letter d	d	100			
Profile of a surface	⌓		100		
Greater than or equal	≥			100	
Lower case letter e	e	101			
Circularity	○		101		
Marker/symbol	△			101	
Lower case letter f	f	102			
Parallelism	//		102		
Radical	√			102	

†Entries for each Font are decimal ASCII equivalent

Table 6. Character Names for the Symbol and Drafting Fonts (continued)

Name	Symbol	Font†			
		1	1001	1002	1003
Lower case letter g	g	103			see
Cylindricity	⌭		103		App. J
Cross product	×			103	for this
Lower case letter h	h	104			Font
Runout	↗		104		
Congruence	≡			104	
Lower case letter i	i	105			
Symmetry	⌯		105		
Not equal	≠			105	
Lower case letter j	j	106			
Position	⌖		106		
Integral	∫			106	
Lower case letter k	k	107			
Profile of a line	⌒		107		
Implication	⊃			107	
Lower case letter l	l	108			
Perpendicularity	⊥		108		
Union	∨			108	
Lower case letter m	m	109			
Maximum material condition	Ⓜ		109		
Intersection	∧			109	
Lower case letter n	n	110			
Diameter	∅		110		
Approximately equal	≈			110	
Lower case letter o	o	111			
All around applicability	○		111		
Greek letter sigma (Sum)	∑			111	
Lower case letter p	p	112			
Projected tolerance zone	Ⓟ		112		
Up arrow	↑			112	
Lower case letter q	q	113			
Centerline	℄		113		
Down arrow	↓			113	
Lower case letter r	r	114			
Concentricity	◎		114		
Right arrow	→			114	
Lower case letter s	s	115			
Regardless of feature side	Ⓢ		115		
Left arrow	←			115	
Lower case letter t	t	116			
Marker/symbol	⍁		116		
Greek letter phi	ϕ			116	

†Entries for Each Font are decimal ASCII equivalent

Table 6. Character Names for the Symbol and Drafting Fonts (continued)

Name	Symbol	Font†			
		1	1001	1002	1003
Lower case letter u	u	117			see
Marker/symbol	⏲		117		App. J
Greek letter theta	θ			117	for this
Lower case letter v	v	118			Font
Marker/symbol	△		118		
Greek letter gamma	γ			118	
Lower case letter w	w	119			
Marker/symbol	◈		119		
Greek letter psi	ψ			119	
Lower case letter x	x	120			
Marker/symbol	⍋		120		
Greek letter omega	ω			120	
Lower case letter y	y	121			
Marker/symbol	⧖		121		
Greek letter lambda	λ			121	
Lower case letter z	z	122			
Marker/symbol	Y		122		
Greek letter alpha	α			122	
Left brace	{	123	123		
Greek letter delta	δ			123	
Vertical bar	|	124	124		
Greek letter mu	μ			124	
Right brace	}	125	125		
Greek letter pi	π			125	
Tilde	~	126	126		
Overscore	—			126	

†Entries for each Font are decimal ASCII equivalent

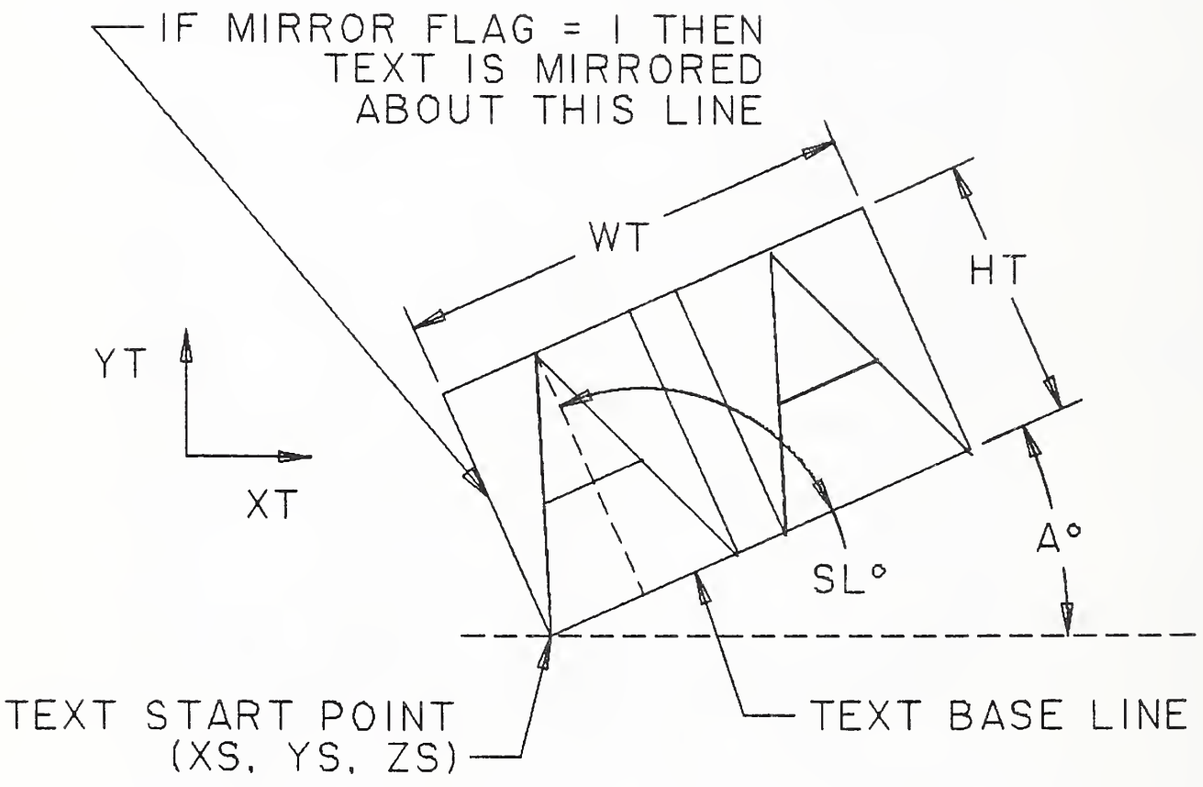


Figure 69. General Note Text Construction

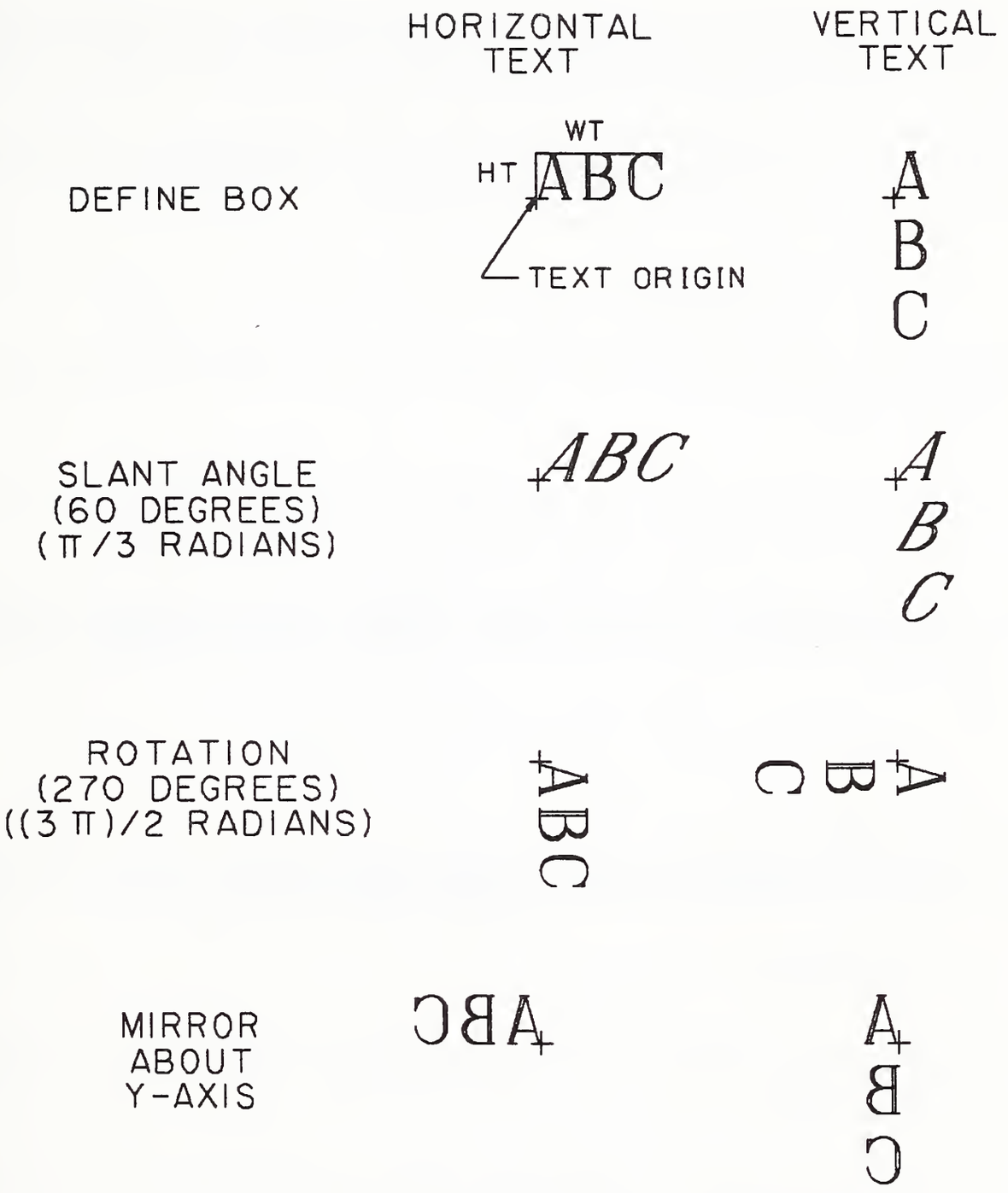


Figure 70. General Note Example of Text Operations

4.2.9 GENERAL NOTE ENTITY (TYPE 212)

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.

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.

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_
```

The following form numbers for the general note are used to maintain the graphical representation of the originating system's note:

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.

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

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*xx*xxxx

Form 3: Superscript - A general note of two or more strings where the second string is a superscript of the first string.

xxx^{yyy}

Form 4: Subscript - A general note of two or more strings where the second string is a subscript of the first string.

xxx_{yyy}

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.

$$xxx_{zzz}^{yyy}$$

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
yyyyyyy
zzzzzzzzzz
```

Form 7: Multiple Stack/Center Justified - A general note where all strings are center justified to a common axis.

```
 xxxxxxxx
   yyyy
 zzzzzzzz
```

Form 8: Multiple Stack/Right Justified - A general note where all strings are right justified to a common margin.

```
 xxxxxxxxxx
      yyyyy
    zzzzzzzz
```

Form 100: Simple Fraction - A general note of four or more strings where the first four strings define a mixed numeral as defined previously.

```
    yy
xx --
    zz
```

Form 101: Dual Stack Fraction - A general note of eight or more strings which represent two mixed numerals as defined previously. These mixed numerals are related such that the fifth through the eighth strings are displayed below the first through the fourth strings respectively.

4.2.9 GENERAL NOTE ENTITY (TYPE 212)

```
   yy
xx --
   zz

   jj
ii --
   kk
```

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 previously. The fifth string is a character (or characters) that was set apart to accommodate the font change.

```
   yy       jj
xx --  -  ii --
   zz       kk
```

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 previously. The second and third mixed numerals are the superscript and subscript respectively of the first mixed numeral.

```
                ⎛    jj ⎞
                ⎜ ii --⎟
                ⎝    kk ⎠
⎛    yy ⎞
⎜ xx --⎟
⎝    zz ⎠

                ⎛    ss ⎞
                ⎜ rr --⎟
                ⎝    tt ⎠
```

Note: The large parentheses are added to help convey the intent of Form 105. They are not part of the General Note.

4.2.9.1 Directory Data
Entity Type Number: 212

4.2.9.2 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	FC1	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
.	.	.	.
.	.	.	.
.	.	.	.
NS*12-10	NCNS	Integer	Number of characters in last text string
.	.	.	.
.	.	.	.
.	.	.	.
1+NS*12	TEXTNS	String	Last text string

Additional pointers as required (see Section 2.2.4.4.2).

4.2.10 Leader (Arrow) Entity (Type 214). A Leader (Arrow) Entity consists of one or more line segments except when the leader is part of an angular dimension (see Section 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 71.

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 72.

Some leaders (for example, the leader involved with the radius dimension in Figure 72) 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.

There are eleven arrowhead types defined (see Figure 73) and selection is made by entering the form number in Directory Entry Field 15.

4.2.10.1 Directory Data
Entity Type Number: 214

4.2.10.2 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	X1	Real	Segment tail coordinate pairs
.	Y1	Real	.
.	.	.	.
.	.	.	.
6+2N	YN	Real	Last segment coordinate

Additional pointers as required (see Section 2.2.4.4.2).

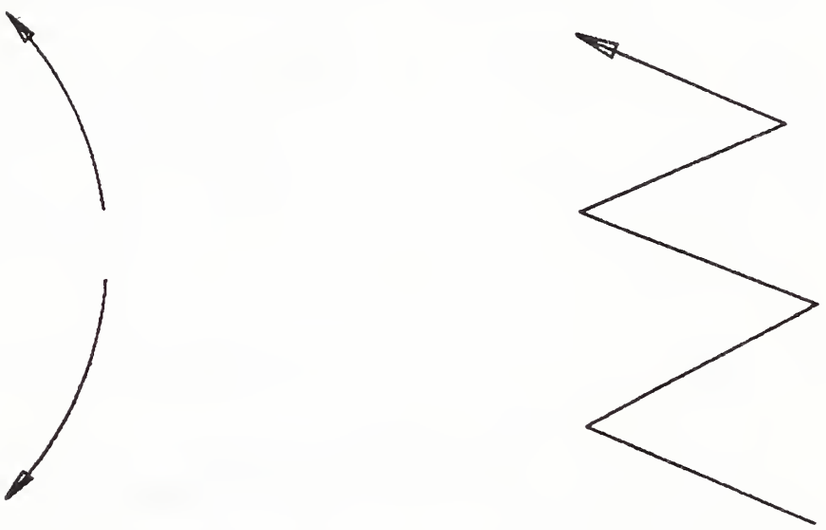
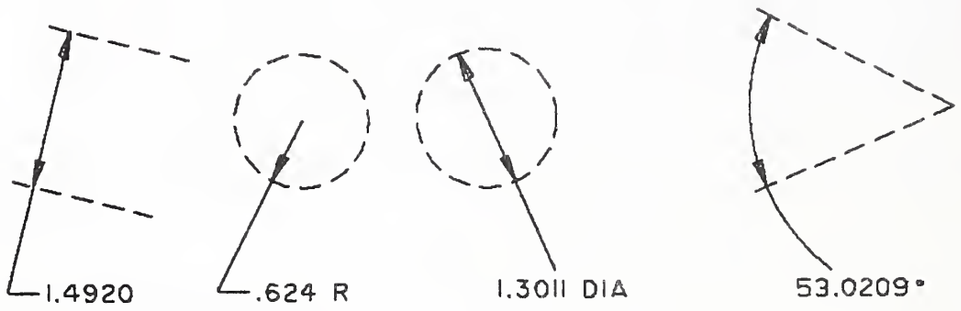


Figure 71. Examples Defined Using the Leader Entity

4.2.10 LEADER (ARROW) ENTITY (TYPE 214)



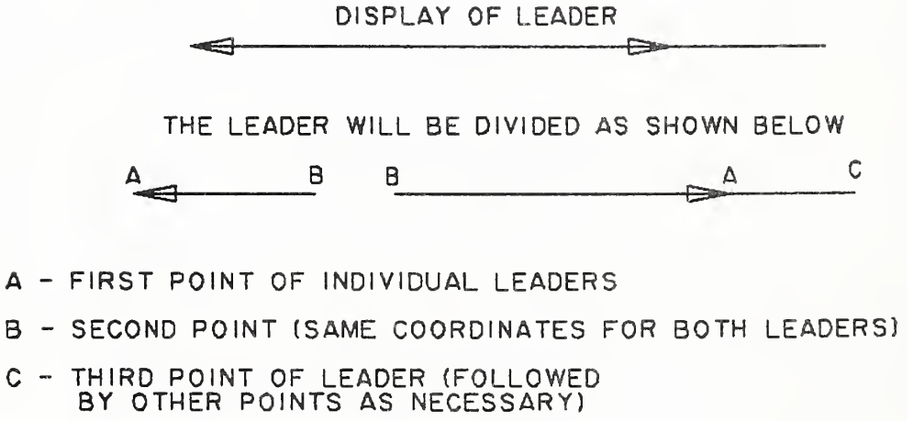


Figure 72. Structure of Leaders Internal to a Dimension

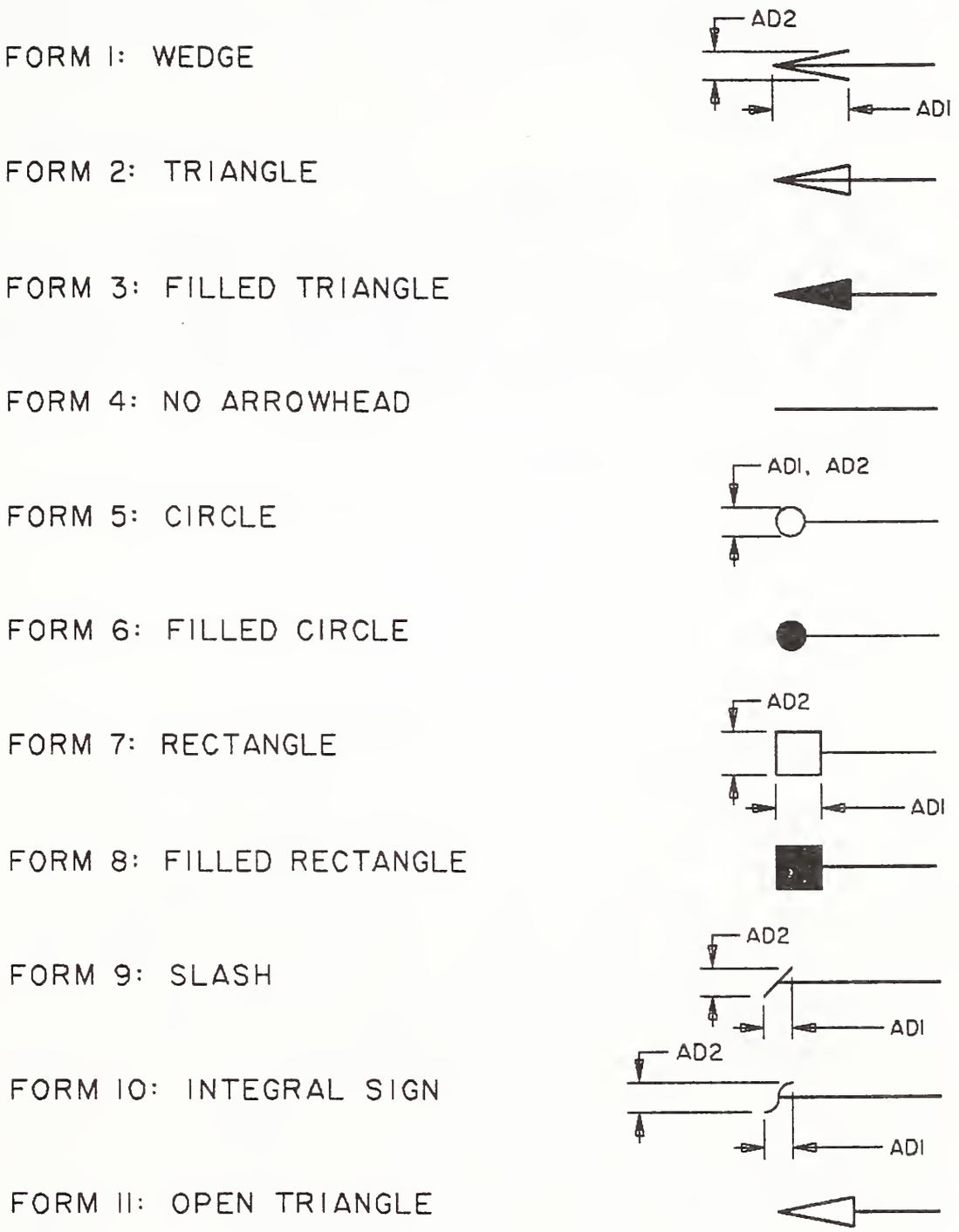


Figure 73. Definition of Arrowhead Types for the Leader (Arrow) Entity

4.2.11 Linear Dimension Entity (Type 216). A Linear Dimension Entity consists of a general note; two leaders; and zero, one or two witness lines. Refer to Figure 74 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; 0 if not defined

Additional pointers as required (see Section 2.2.4.4.2).

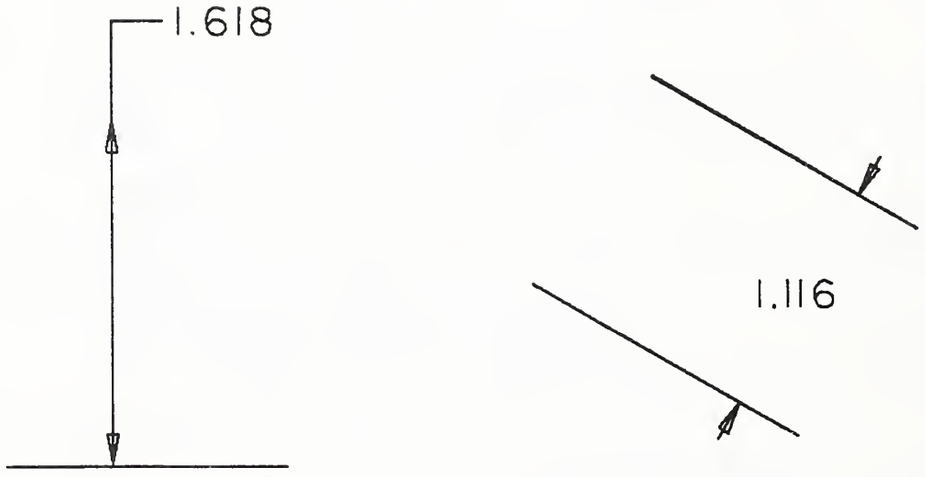


Figure 74. Examples Defined Using the Linear Dimension Entity

4.2.12 Ordinate Dimension Entity (Type 218). The Ordinate Dimension Entity is used to indicate dimensions from a common base line. Dimensioning is only permitted along the XT or YT axis.

An Ordinate Dimension Entity consists of a general note and a witness line or leader. The values stored are pointers to the Directory Entry for the associated General Note and Witness Line or Leader Entities. Examples of ordinate dimensions are shown in Figure 75.

4.2.12.1 Directory Data

Entity Type Number: 218

4.2.12.2 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 Section 2.2.4.4.2).

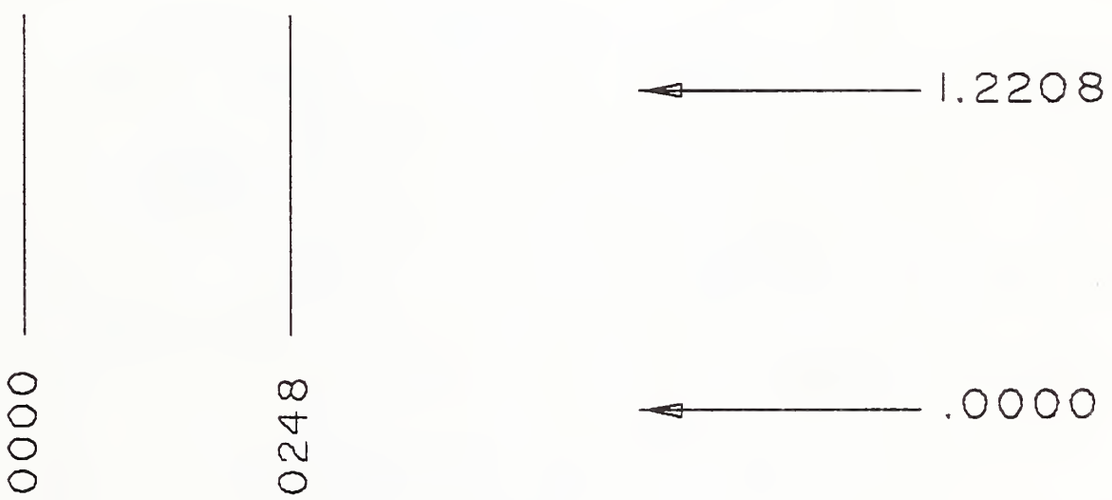


Figure 75. Examples Defined Using the Ordinate Dimension Entity

4.2.13 Point Dimension Entity (Type 220). A Point Dimension Entity consists of a leader, text, and an optional circle or hexagon enclosing the text.

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 (Type 102). 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.

Examples are shown in Figure 76.

4.2.13.1 Directory Data
Entity Type Number: 220

4.2.13.2 Parameter Data

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to General Note Directory Entry
2	DEARRW	Pointer	Pointer to Leader Directory Entry
3	DEGEOM	Pointer	Pointer to Circular Arc, Composite Curve, or 0.

Additional pointers as required (see Section 2.2.4.4.2).

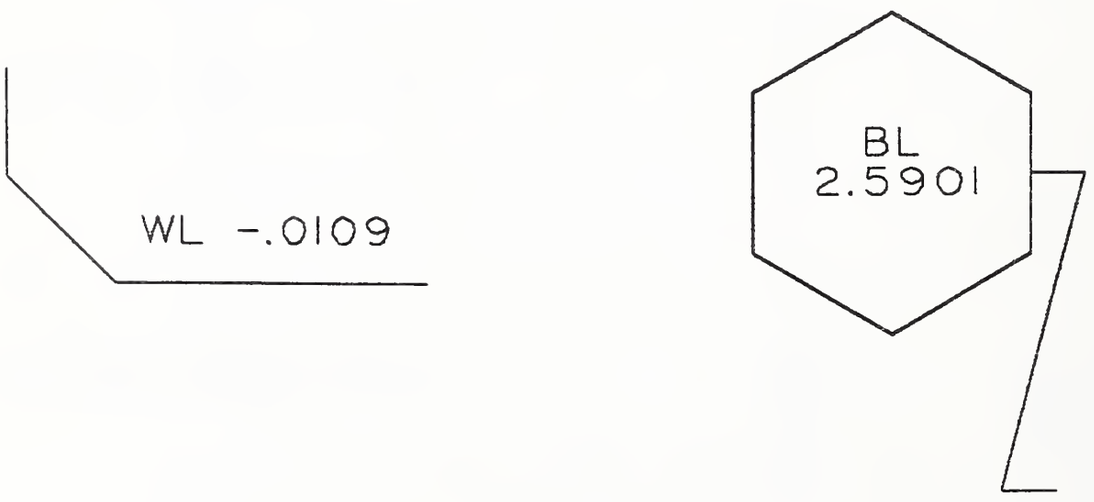


Figure 76. Examples Defined Using the Point Dimension Entity

4.2.14 Radius Dimension Entity (Type 222). A Radius Dimension Entity consists of a general note, a leader, and an arc center point, (XT, YT). Refer to Figure 77 for examples of radius dimensions. A second form of this entity accounts for the occasional need to have two Leader (Arrow) Entities referenced. The definition of this second form can be found in Appendix J (see Section J.6).

The arc center coordinates are used as reference in constructing the radius dimension but have no effect on the dimension components.

4.2.14.1 Directory Data
Entity Type Number: 222
Form Number: 0 Single Leader Format

4.2.14.2 Parameter Data

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to General Note Directory Entry
2	DEARRW	Pointer	Pointer to Leader Directory Entry
3	XT	Real	Arc center coordinates
4	YT	Real	.

Additional pointers as required (see Section 2.2.4.4.2).

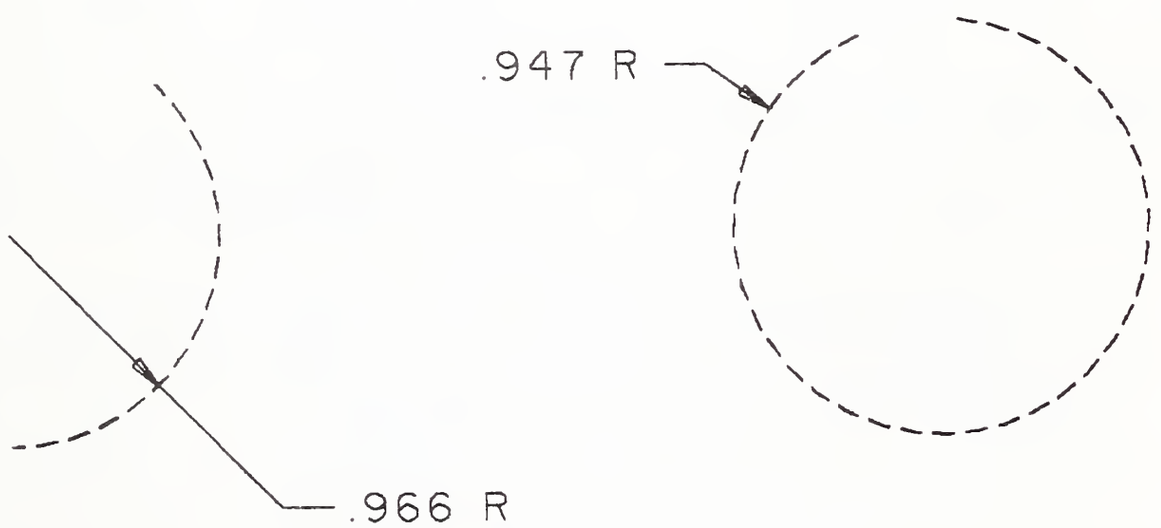


Figure 77. Examples Defined Using the Radius Dimension Entity

4.2.15 General Symbol Entity (Type 228). 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 symbols are shown in Figure 78.

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.

Field 15 of the Directory Entry Section accommodates a form number. For this entity, it is used to maintain the nature of the symbol on the originating system. The definition of the form numbers can be found in Appendix J (see Section J.7).

4.2.15.1 Directory Data
Entity Type Number: 228

4.2.15.2 Parameter Data

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to associated General Note
2	N	Integer	Number of pointers to geometry
3	DEGEOM1	Pointer	Pointer to defining geometry
.	.	.	.
.	.	.	.
.	.	.	.
N+2	DEGEOMN	Pointer	.
N+3	L	Integer	Number of Leaders or zero
N+4	DEARRW1	Pointer	Pointers to associated Leaders
.	.	.	.
.	.	.	.
.	.	.	.
N+3+L	DEARRWL	Pointer	.

Additional pointers as required (see Section 2.2.4.4.2).

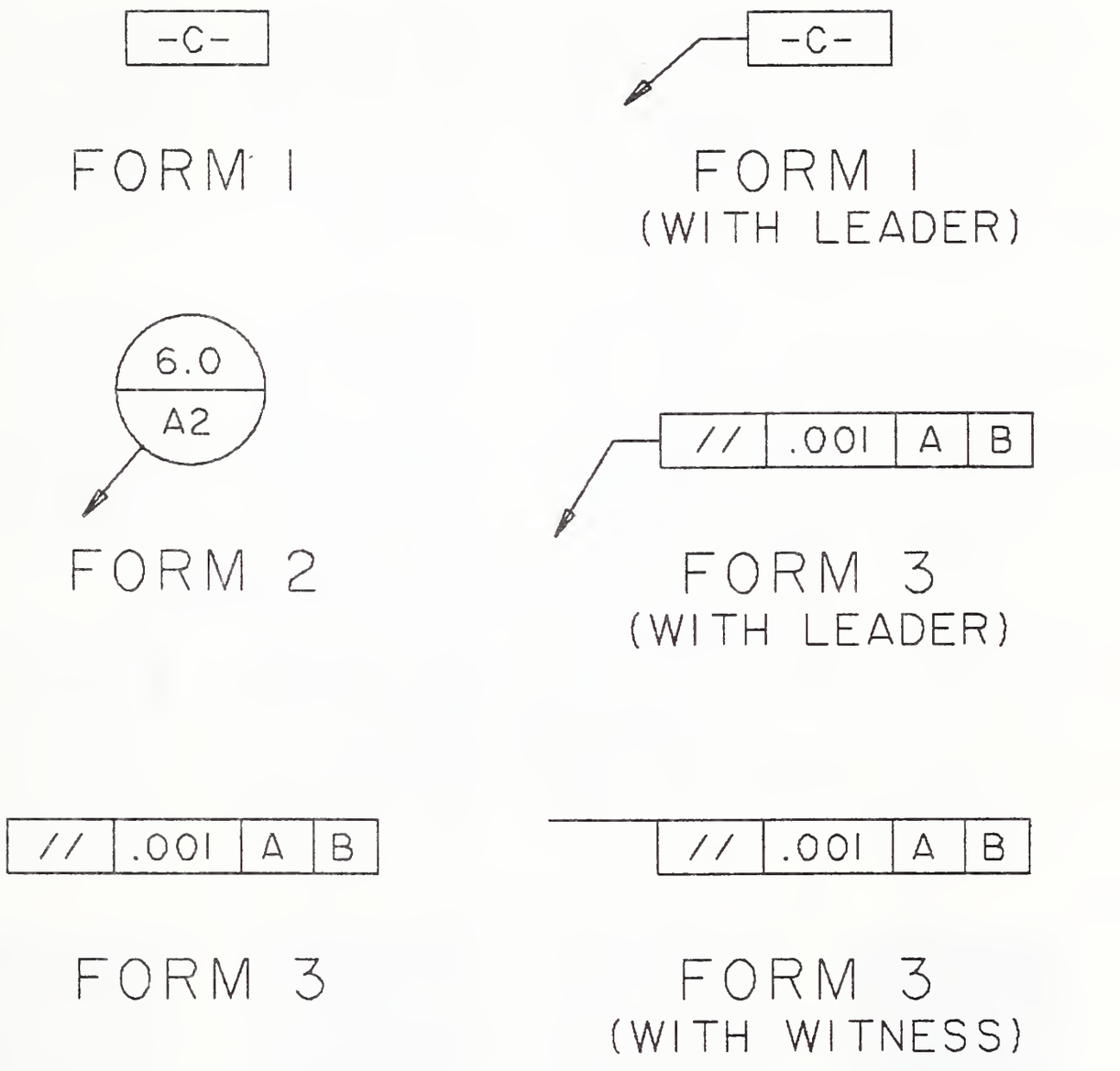


Figure 78. Examples Defined Using the General Symbol Entity (Form Numbers Defined in Appendix J)

4.2.16 Sectioned Area Entity (Type 230). A sectioned area is a portion of a design which is to be filled with a pattern of lines. The Sectioned Area Entity 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).

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.

The angle of the lines has a default value of $\pi/4$, measured in radians.

The fill pattern is specified according to predefined definitions illustrated in Figure 79.

For the fill pattern 0 and 19, the values for indexes 3 through 7 are defaulted to 0.0 because they do not apply to the specified fill patterns.

4.2.16.1 Directory Data

Entity Type Number: 230

4.2.16.2 Parameter Data

Index	Name	Type	Description
1	BNDP	Pointer	Pointer to a boundary curve – a single closed curve or a composite curve
2	PATRN	Integer	Fill 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. Default = $\pi/4$
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 Section 2.2.4.4.2).

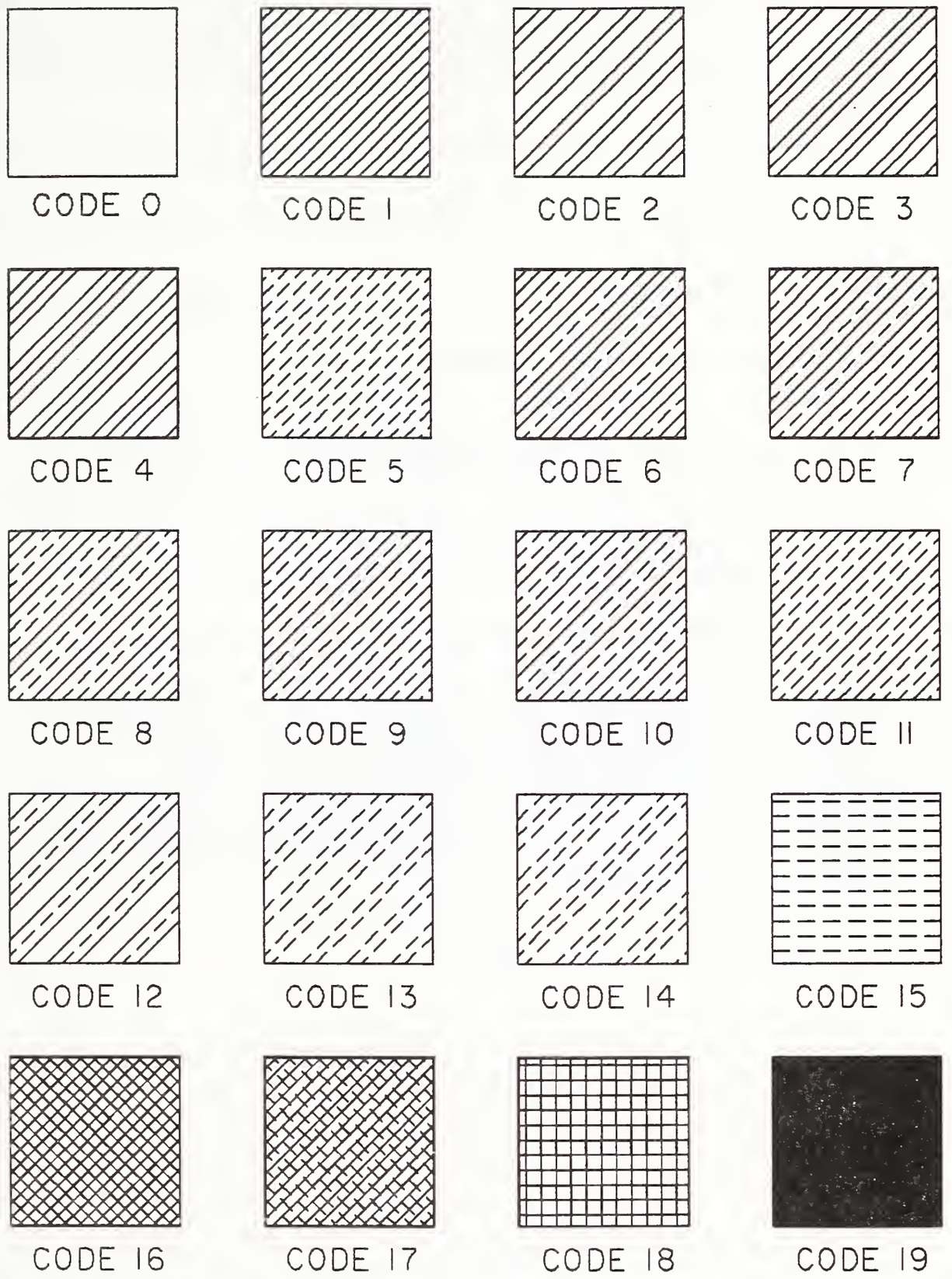


Figure 79. Predefined Fill Patterns for the Sectioned Area Entity

4.2.17 Section Entity (Type 106, Forms 31-38). 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 compatibility with previous versions of the Specification. The Sectioned Area Entity (Type 230) provides a more compact method for transferring this information.

The point data contains a list of points (Xn, Yn), 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 (Xn,Yn,Z) and (Xn+1,Yn+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.

The defined line patterns are described below and illustrated in Figure 80.

Form	Description (see [ANSI79])
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)
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)

See Section 3.5 for parameters of the Section Entity.

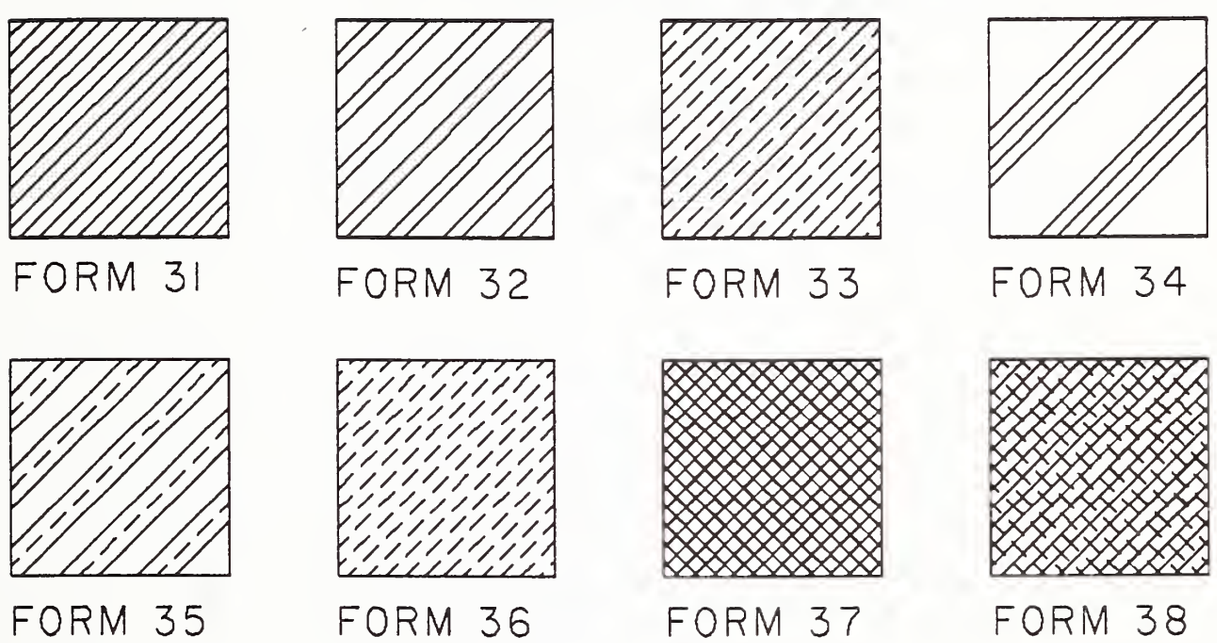


Figure 80. Definition of Patterns for the Section Entity

4.2.18 Witness Line Entity (Type 106, Form 40). A Witness Line Entity is a Form Number 40 of a Copious Data Entity 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 81 for examples.

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 81 show the blanking of segments and the order of points stored in the copious data.)

See Section 3.5 for parameters of the Witness Line Entity.

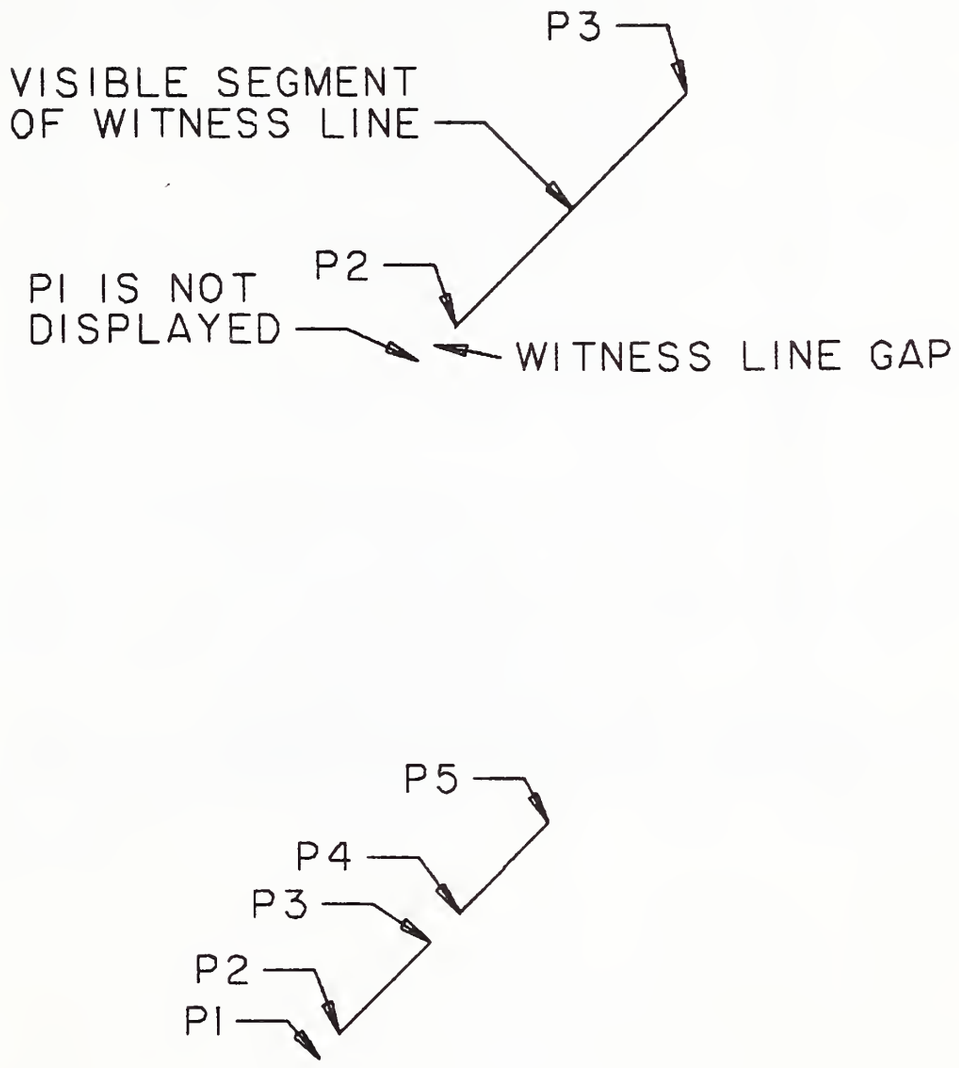


Figure 81. Examples Defined Using the Witness Line Entity

4.3 Structure Entities

4.3.1 Entity Type/Type Number. The following entities are defined in this section:

Entity Type Number	Entity Type
0	Null (see Appendix J)
302	Associativity Definition
304	Line Font Definition
306	MACRO Definition
308	Subfigure Definition
310	Text Font Definition
312	Text Display Template
314	Color Definition
320	Network Subfigure Definition
322	Attribute Table Definition (see Appendix J)
402	Associativity Instance
404	Drawing
406	Property
408	Singular Subfigure Instance
410	View
412	Rectangular Array Subfigure Instance
414	Circular Array Subfigure Instance
416	External Reference
418	Nodal Load/Constraint
420	Network Subfigure Instance
422	Attribute Table Instance (see Appendix J)
600–699	User specified MACRO Instance
10000–99999	User specified MACRO Instance

4.3.2 Associativity Definition Entity (Type 302). 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.

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 indicates if 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.

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 Section 4.3.3.3. (These definitions do not appear in the file). 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 Figure 82 for a complete example of associativity.

4.3.2 ASSOCIATIVITY DEFINITION ENTITY (TYPE 302)

4.3.2.1 Directory Data
Entity Type Number: 302

4.3.2.2 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 ≥ 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.

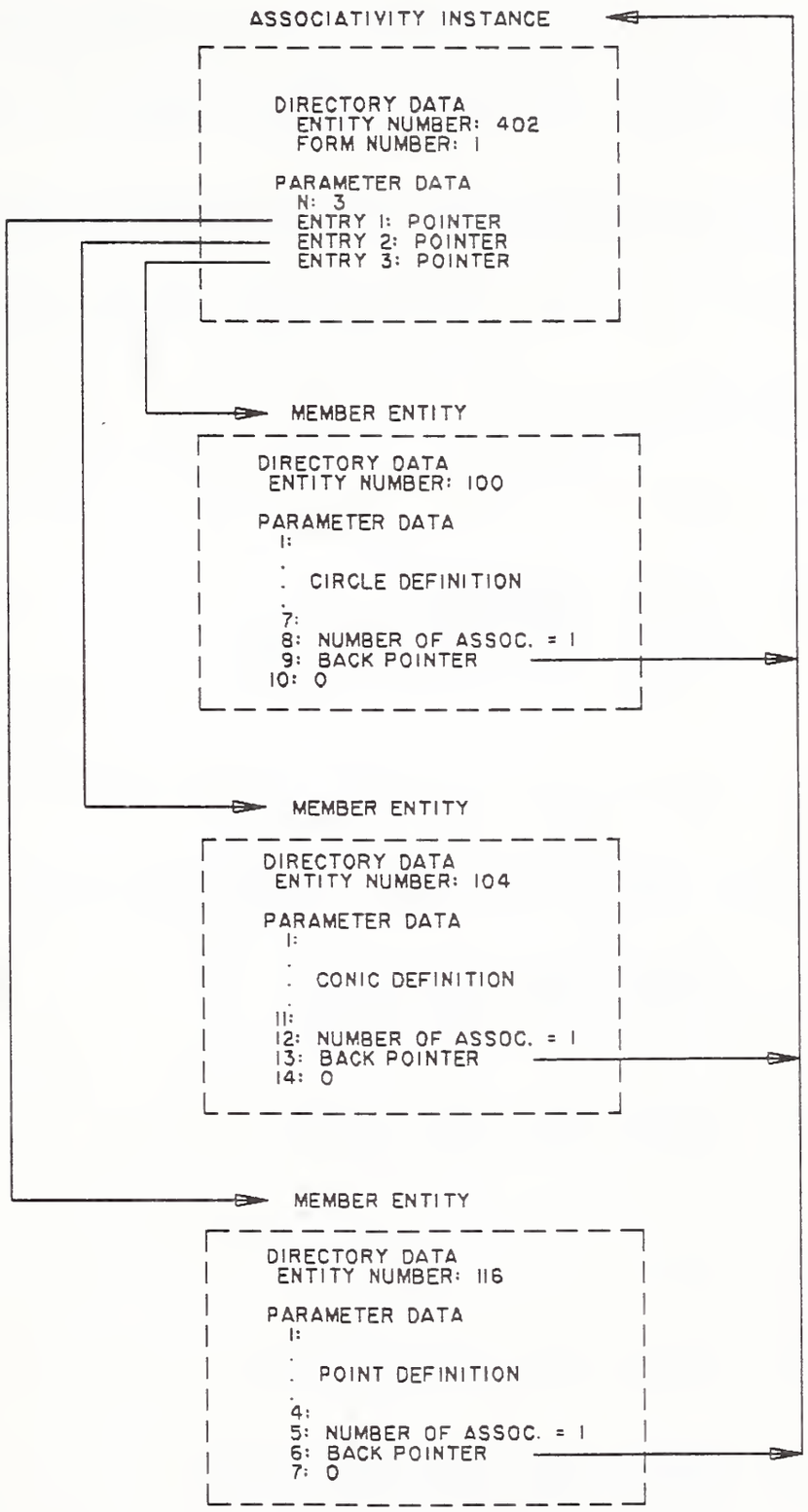


Figure 82. Relationships between Entities in an Associativity

4.3.3 Associativity Instance Entity (Type 402). 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 Section 4.3.2. 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 Section 2.2.4.4.2).

The parameters K and N1, N2, ...NK are specified in the associativity definition (see Section 4.3.2).

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

For K classes with (NE1,...,NEK) entries with (N1, . . . , NK) items per entry

K + 1	I(1,1,1)	Variable	Class 1, Entry 1, Item 1
.	.	Variable	Class 1, Entry 1, Item 2
.	.	.	.
.	.	.	.
.	I(1,1,N1)	Variable	Class 1, Entry 1, Item N1
.	I(1,2,1)	Variable	Class 1, Entry 2, Item 1
.	.	.	.
.	.	.	.
.	I(1,2,N1)	Variable	Class 1, Entry 2, Item N1
.	.	.	.
.	.	.	.
.	I(1,NE1,1)	Variable	Class 1, Entry NE1, Item 1
.	.	.	.
.	.	.	.
.	I(1,NE1,N1)	Variable	Class 1, Entry NE1, Item N1
.	I(2,1,1)	Variable	Class 2, Entry 1, Item 1
.	.	.	.
.	.	.	.
.	I(2,1,N2)	Variable	Class 2, Entry 1, Item N2
.	I(2,2,1)	Variable	Class 2, Entry 2, Item 1
.	.	.	.
.	.	.	.

.	I(2,2,N2)	Variable	Class 2, Entry 2, Item N2
.	.	.	.
.	.	.	.
.	I(2,NE2,N2)	Variable	Class 2, Entry NE2, Item N2
.	.	.	.
.	.	.	.
.	I(K,1,1)	Variable	Class K, Entry 1, Item 1
.	.	.	.
.	.	.	.
X	I(K,NEK,NK)	Variable	Class K, Entry NEK, Item NK

Additional pointers as required (see Section 2.2.4.4.2).

4.3.3.3.1 (TYPE 402, FORM 1) - GROUP

4.3.3.3 Pre-defined Associativities. As defined in Section 4.3.2, the Associativity Definition Entity (Type 302) will only occur in the file 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 Group

The Group Associativity allows a collection of a set of entities to be maintained as a single, logical entity. Figure 82 is an example.

There are four form numbers which specify group associativities:

Form	Meaning
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 Sections 4.3.3.3.6 (Form=7), 4.3.3.3.10 (Form=14), and 4.3.3.3.11 (Form=15), respectively.

DEFINITION

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 Type Number: 402
Form Number: 1

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE1	Pointer	Pointer to entity 1
3	DE2	Pointer	Pointer to entity 2
.	.	.	
.	.	.	
N+1	DEN	Pointer	Pointer to entity N

Additional pointers as required (see Section 2.2.4.4.2).

4.3.3.3.2 (TYPE 402, FORM 2) - EXTERNAL LOGICAL REFERENCE FILE INDEX

4.3.3.3.2 FORM NUMBER: 2 External Logical Reference File Index

The External Logical Reference File Index Entity 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 Type 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
.	.	.	
.	.	.	
2N	NAMEN	String	Last External Reference Entity Symbolic Name
2N+1	PTRN	Pointer	Last Internal Entity DE Pointer

Additional pointers as required (see Section 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 but not all views, 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)

4.3.3.3.3 (TYPE 402, FORM 3) - VIEWS VISIBLE

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 first View Entity
.	.	.	
.	.	.	
N1+2	DEVN1	Pointer	Pointer to last View Entity
N1+3	DE1	Pointer	Pointer to first entity whose display is being specified by this associativity instance
.	.	.	
.	.	.	
N1+N2+2	DEN2	Pointer	Pointer to last entity whose display is being specified by this associativity instance

Additional pointers as required (see Section 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 (Type 410), a line font value or 0, a pointer to a Line Font Definition Entity (Type 304) if the line font value was 0, a color value or pointer to a Color Definition Entity (Type 314), 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 each 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 Entity
6	2	Line Font value
7	1	Pointer to Line Font Definition Entity
8	3	Color Number (value) or pointer
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)

4.3.3.3.4 (TYPE 402, FORM 4) - VIEWS VISIBLE, COLOR, LINE WEIGHT

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 Entity. Otherwise = 0
6	CN1	Integer or Pointer	Color number value 1 or pointer to Color Definition Entity
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 Section 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, Parameter 8 of its DE contains a pointer to an instance of a Form Number 5 associativity.

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 Entity
6	2	XT of text location
7	2	YT of text location
8	2	ZT of text location
9	1	Pointer to Leader Entity
10	2	Entity label level number
11	1	Pointer to entity

4.3.3.3.5 (TYPE 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	DEARR1	Pointer	Pointer to Leader Entity 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 N-th View Entity
.	XTN	Real	XT coordinate of text location in N-th view
.	YTN	Real	YT coordinate of text location in N-th view
.	ZTN	Real	ZT coordinate of text location in N-th view
.	DEARRWN	Pointer	Pointer to Leader Entity in N-th view
.	LLNN	Integer	Entity label level number in N-th view
7*N+1	DEN	Pointer	Pointer to N-th entity being displayed

Additional pointers as required (see Section 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 Section 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

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 is required)
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 Section 2.2.4.4.2).

4.3.3.3.8 FORM NUMBER: 12 External Reference File Index

The External Reference File Index Entity 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 Section 2.2.4.4.2).

4.3.3.3.9 FORM NUMBER: 13 Dimensioned Geometry Associativity

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 83.

DEFINITION

Index	Set Value	Meaning
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

DESCRIPTION

Directory Data
Entity Type Number: 402
Form Number: 13

Parameter Data

Index	Name	Type	Description
1	ND	Integer	Number of dimensions (ND=1 is required)
2	NG	Integer	Number of associated geometry entities
3	DIMPTR	Pointer	Pointer to dimension entity
4	GEOM1	Pointer	Pointer to geometry entity 1
.	.	.	.
.	.	.	.
3+NG	GEOMNG	Pointer	Pointer to geometry entity NG

Additional pointers as required (see Section 2.2.4.4.2).

4.3.3.3.9 (TYPE 402, FORM 13) - DIMENSIONED GEOMETRY ASSOCIATIVITY

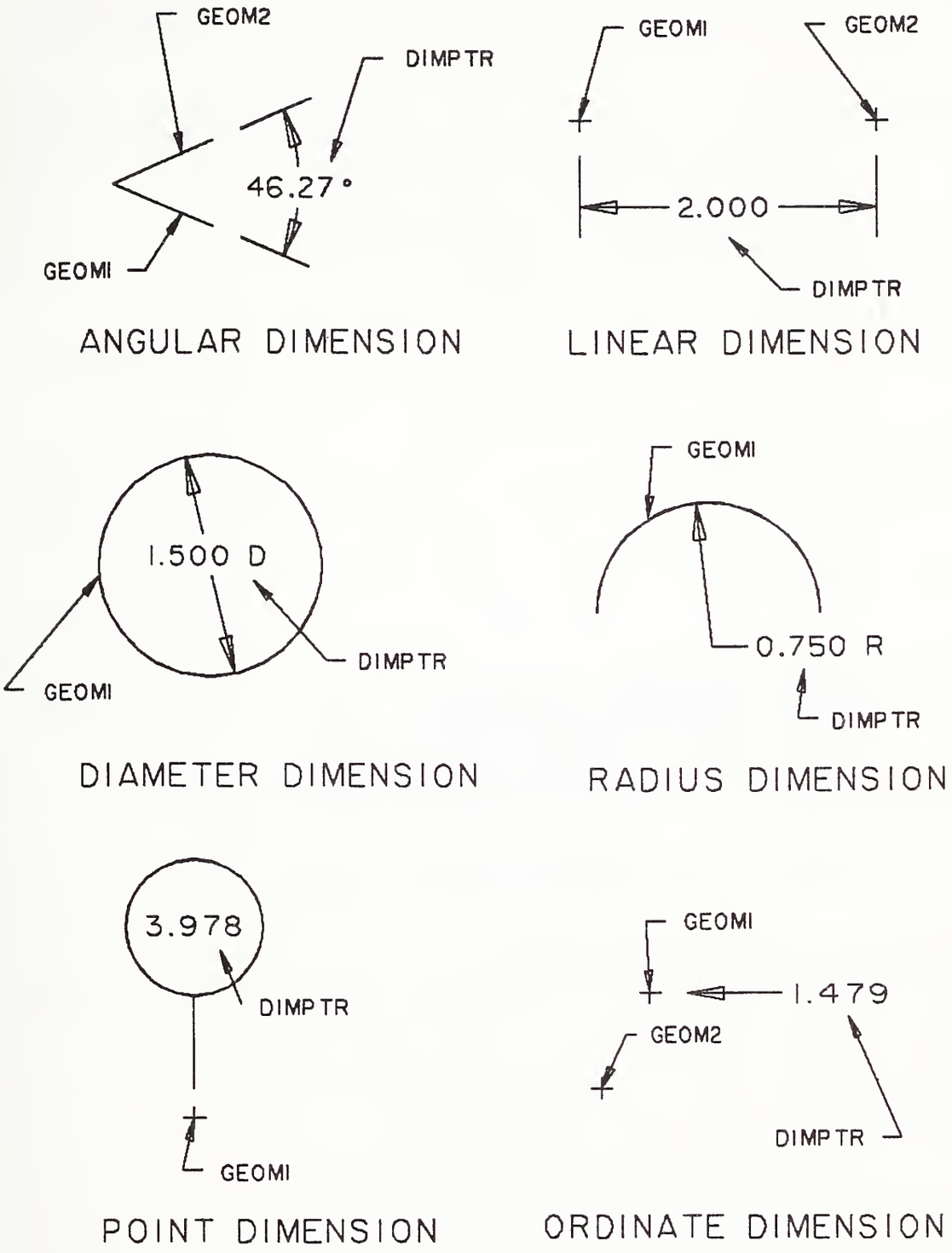


Figure 83. Dimensioned Geometry Associativity

4.3.3.3.10 FORM NUMBER: 14 Ordered Group with Back Pointers

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 Type Number: 402
Form Number: 14

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE1	Pointer	Pointer to entity 1
3	DE2	Pointer	Pointer to entity 2
.	.	.	
.	.	.	
N+1	DEN	Pointer	Pointer to entity N

Additional pointers as required (see Section 2.2.4.4.2).

4.3.3.3.11 FORM NUMBER: 15 Ordered Group without Back Pointers

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 Type Number: 402
Form Number: 15

Parameter Data

Index	Name	Type	Description
1	N	Integer	Number of entries
2	DE1	Pointer	Pointer to entity 1
3	DE2	Pointer	Pointer to entity 2
.	.	.	
.	.	.	
N+1	DEN	Pointer	Pointer to entity N

Additional pointers as required (see Section 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.

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1 (Transformation Matrix)	
2	2	Back pointers not required
3	1	Ordered class
4	1	Number of items per entry
5	1	Pointer
	Class 2 (Coplanar Entities)	
6	2	Back pointers not required
7	2	Unordered class
8	1	Number of items per entry
9	1	Pointer

DESCRIPTION

Directory Data
Entity Type Number: 402
Form Number: 16

Parameter Data

Index	Name	Type	Description
1	NTR	Integer	Number of Transformation Matrices (NTR=1 is required)
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 specified

Additional pointers as required (see Section 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 Point Entities (class three) list 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 Entity
	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

4.3.3.3.13 (TYPE 402, FORM 18) - FLOW

DESCRIPTION

Directory Data
Entity Type Number: 402
Form Number: 18

Parameter Data

Index	Name	Type	Description
1	NCF	Integer	Count of context flags (NCF=2 is required)
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
9	FF	Integer	Function flag: 0 = not specified (Default) 1 = electrical signal 2 = fluid flow path
10	SPTR1	Pointer	Pointer to DE for Flow Associativity 1
.	.	.	
NF+9	SPTRNF	Pointer	Pointer to DE for Flow Associativity NF
NF+10	CPTR1	Pointer	Pointer to DE for Connect Point 1
.	.	.	
NF+NC+9	CPTRNC	Pointer	Pointer to DE for Connect Point NC
NF+NC+10	JPTR1	Pointer	Pointer to Join 1
.	.	.	
NF+NC+NJ+9	JPTRNJ	Pointer	Pointer to Join 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 DE for Text Display Template 1
.	.	.	
NF+NC+NJ+NN+NT+9	GPTRNT	Pointer	Pointer to DE for Text Display 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 Section 2.2.4.4.2).

4.3.4 Drawing Entity (Type 404). 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 a View Entity (Type 410). 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 84, 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:

$$\begin{bmatrix} XD \\ YD \end{bmatrix} = \begin{bmatrix} S & 0 & 0 \\ 0 & S & 0 \end{bmatrix} \begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} + \begin{bmatrix} XORIGIN \\ YORIGIN \end{bmatrix}$$

$$\text{where} \begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix}$$

denotes the view space coordinates

$$\text{and} \begin{bmatrix} XORIGIN \\ YORIGIN \end{bmatrix}$$

denotes the drawing space coordinates of the origin of the transformed view (see Section 4.3.11).

The name of the drawing may be provided by using the Name Property (Type 406, Form 15). If not provided, a receiving system may default the name of the drawing.

The size of the drawing may be specified by using the Drawing Size Property (Type 406, Form 16).

The units for drawing space may be set differently from the model space units specified in the Global Section. This is accomplished by use of the Drawing Units Property (Type 406, Form 17). When this property is not provided, the drawing units will be the same as the model units.

4.3.4 DRAWING ENTITY (TYPE 404)

The following values are given in drawing units:

- view origin drawing locations
- drawing size
- coordinates of annotation entities referenced directly

Refer to Figures 84 and 85 for examples of the use of the Drawing Entity.

4.3.4.1 Directory Data
Entity Type Number: 404

4.3.4.2 Parameter Data

Index	Name	Type	Description
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 Section 2.2.4.4.2).

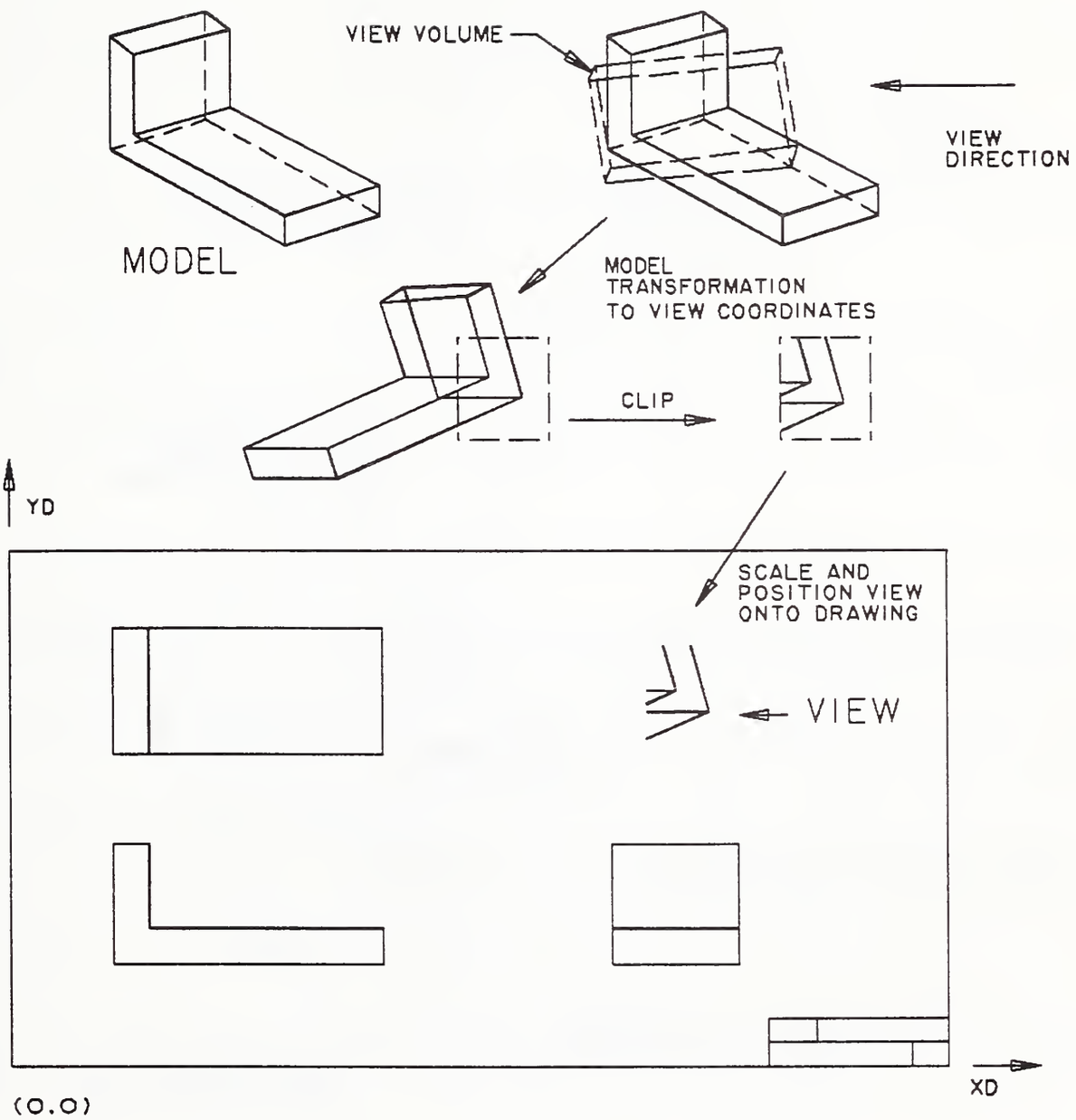


Figure 84. Using Clipping Planes with a View in a Drawing

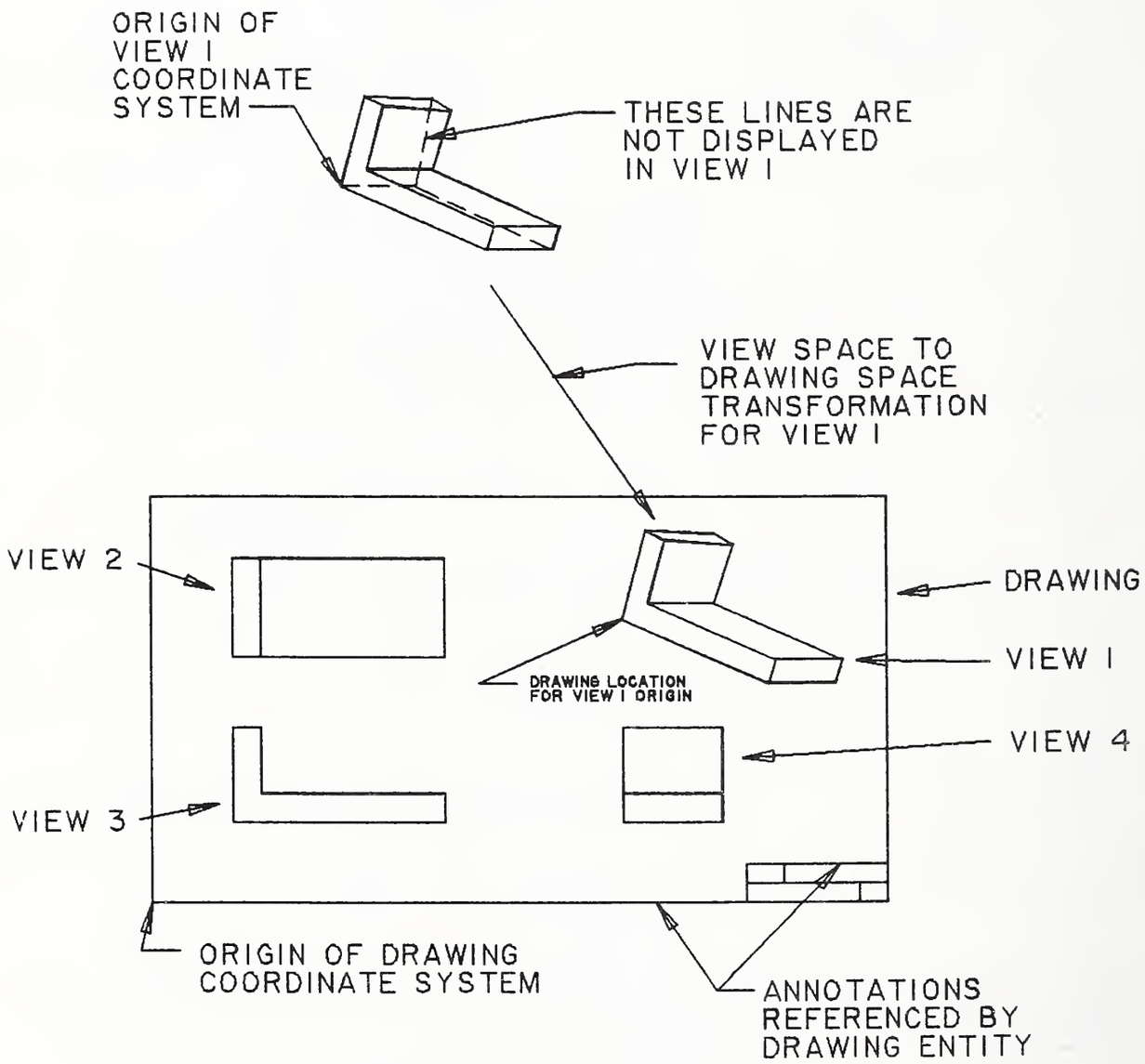


Figure 85. Parameters of the Drawing Entity

4.3.5 Line Font Definition Entity (Type 304) Two types of line fonts may be defined. One type considers a line font as a repetition of a basic pattern of visible-blank (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.

The preprocessor must select one of the line font patterns (see Section 2.2.4.3.4) and place the value in Directory Entry Field 4 of the Line Font Definition Entity. This value should be the closest functional equivalent, or the most visually similar. The value will be used by postprocessors which cannot support 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 (Type 308). In this case, four 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 86 illustrates two examples of a line font with Form Number equal 1. In each case, the anchoring curve is a straight line.

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 "M-th" 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+1 give the lengths of the M segments. Finally, parameter value M+2 is the minimal string of hexadecimal digits whose significance has

been described above.

Figure 87 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-blank 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 Section 2.2.4.4.2).

4.3.5.3 Directory Data
Entity Type Number: 304
Form Number: 2

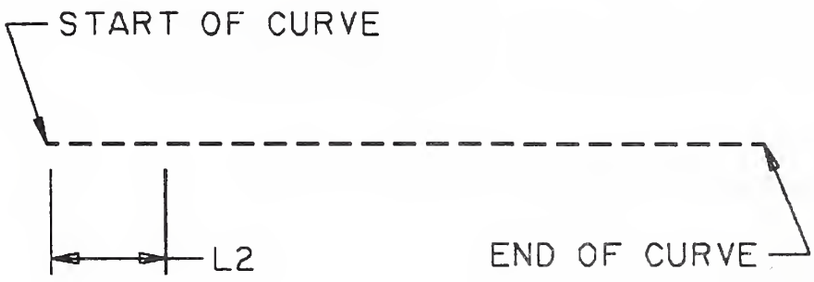
4.3.5.4 Parameter Data

Index	Name	Type	Description
1	M	Integer	Number of segments in the basic pattern of visible-blank 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 M-th segment of the basic pattern
M+2	B	String	(((M-1)/4) + 1) hexadecimal digits indicating which segments of the basic pattern are visible and which are blanked, where the expression represents the greatest integer result. *E.g.*, "5" indicates that segments 1 and 3 are visible. Bits are right justified.

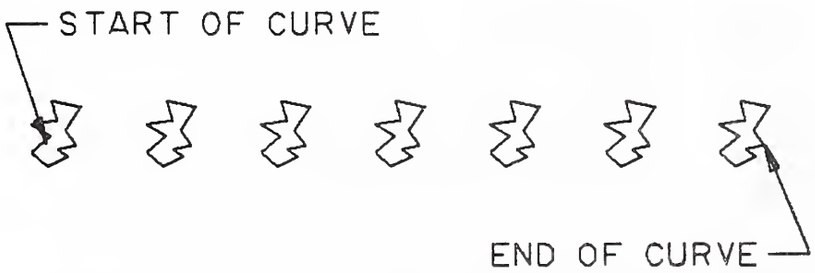
Additional pointers as required (see Section 2.2.4.4.2).

M = 0
LI = POINTER TO SUBFIGURE DEFINITION
L2 = LENGTH OF SEPARATION
L3 = 0.5 (SCALE)

SUBFIGURE DEFINITION



ANCHORING CURVE SHOWN WITH DASHED LINE FONT



CURVE SHOWN WITH RESULTING LINE FONT
FROM SUBFIGURE REFERENCE

Figure 86. Line Font Definition Using Form Number 1 (Template Subfigure)

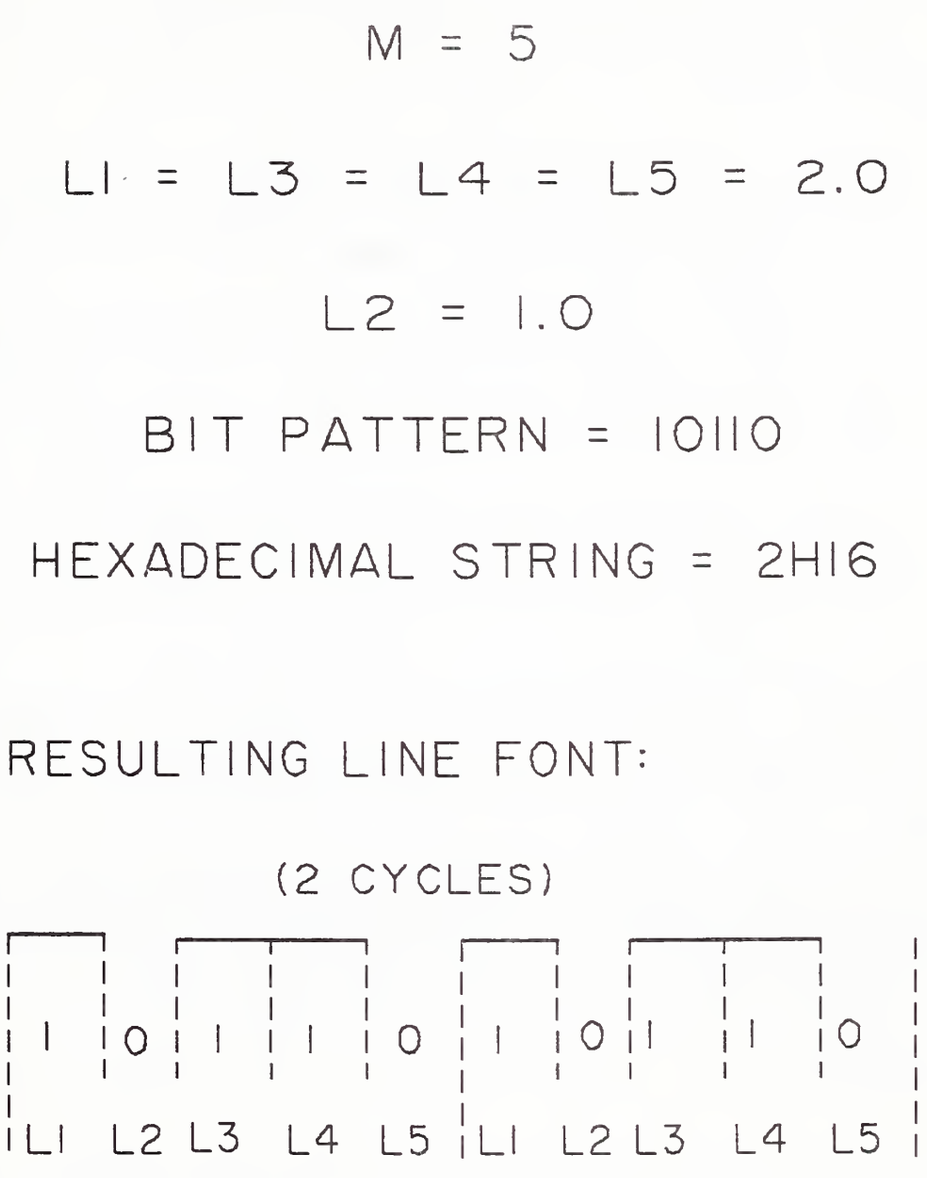


Figure 87. Line Font Definition Using Form Number 2 (Visible-Blank Pattern)

4.3.6 MACRO Capability

4.3.6.1 General. 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 (Type 306). The Parameter Data 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 Section 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 Data 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 Section 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 Section 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 Section 2.2.2).

Variables. The significant part of a variable name is 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 Naming Convention

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.

Variable Type	Valid Names				
Integer	IJK	ICOUNT	K101	NTIMES	max
Real	XYZ	X1	y2	QrsTu1	
Double precision real	!h	!xi	!Y2	!12341	
String	$str	$TITLE	$label		
Pointer	#line	#note	#REF	#XYZ1	

Some examples of invalid variable names are:

$$$$	$ 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:

Function	Value Returned
IABS(i)	Absolute value of i
IDINT(!d)	Integer part of !d
IFIX(x)	Integer part of x
INT(x)	Integer part of x
ISIGN(i)	1 if i is positive, or 0 if it is zero, or -1 if it is negative

4.3.6 MACRO CAPABILITY

Functions Returning Real Values:

Function	Value Returned
`ABS(x)`	Absolute value of `x`
`AINT(x)`	Integer part of `x`, in real form
`ALOG(x)`	Natural logarithm of `x`
`ALOG10(x)`	Common (base 10) logarithm of `x`
`ATAN(x)`	Arctangent of `x`; angle returned in radians
`COS(x)`	Cosine of angle `x`; angle in radians
`EXP(x)`	Natural anti-logarithm of `x` (*i.e.*, e to the `x`)
`FLOAT(i)`	Real (floated) value for `i`, *e.g.*, `FLOAT(2)` returns `2.`
`SIGN(x)`	1. if x is positive, or 0. if x is zero, or -1. if x is negative
`SIN(x)`	Sine of angle `x`; angle in radians
`SNGL(!d)`	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)`	Square root of `x`
`TAN(x)`	Tangent of angle `x`; angle in radians

Functions Returning Double Precision Real Values:

Function	Value Returned
`DABS(!d)`	Absolute value of `!d`
`DATAN(!d)`	Arctangent of `!d`; value returned in radians
`DBLE(x)`	Returns double precision real value of `x`. Note that this is merely conversion, not an extension. Thus, `DBL(.333333333)` will return `.333333333D0`, but not `.33333333333333333333333D0`. Thus, `DBLE(1./3.)` is not necessarily equal to `1D0/3D0`
`DCOS(!d)`	Cosine of angle `!d`; angle in radians
`DEXP(!d)`	Natural anti-logarithm of `!d` (*i.e.*, e to the `!d`)
`DLOG(!d)`	Natural logarithm of `!d`
`DLOG10(!d)`	Common (base 10) logarithm of `!d`
`DSIGN(!d)`	`1D0` if `!d` is positive, or `0D0` if zero, or `-1D0` if negative
`DSIN(!d)`	Sine of angle `!d`; angle in radians
`DSQRT(!d)`	Square root of `!d`
`DTAN(!d)`	Tangent of angle `!d`; angle in radians

Functions Returning String Values:

Function	Value Returned
`STRING(expression, format)`	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	.LE.
less than	.LT.
equal	.EQ.
greater than or equal	.GE.
greater than	.GT.
not equal	.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 (Type 306). 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 Section 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 Section 2.2.4.4.2).

4.3.6.3.3 There are eleven language statements that can be used. They are:

```
BREAK       IF        MREF
CONTINUE    LABEL     REPEAT
ENDM        LET       SET
GOTO        MACRO
```

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, *i.e.*, 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 include:

```
LET /LEV = 1;
LET /VIEW = 3;
LET /LABEL = 6HBottom;
LET /LABEL = $X;
```

The second form allows restoring an attribute to its default value. Examples include:

```
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.

4.3.6.3 MACRO DEFINITION ENTITY (TYPE 306)

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 include:

```
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 are:

```
REPEAT 3;
REPEAT N+1;
REPEAT 0;
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 are:

```
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 are:

```
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:

4.3.6.3 MACRO DEFINITION ENTITY (TYPE 306)

```
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, *i.e.*, `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
M(#DEP)	contains the pointers to properties

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)).

4.3.6.4.2 Parameter Data. The parameter data section for an instance has the following form: 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		Values for the arguments to the MACRO must agree in format and number with the arguments in the MACRO statement of the definition.

Additional pointers as required (See Section 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: Isosceles Triangle

The following MACRO definition creates an isosceles triangle, given a vertex point, a height, a base width 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	X1	Real	X coordinate of vertex
2	Y1	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 88 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:

4.3.6.5 EXAMPLES OF MACRO USAGE

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;
```

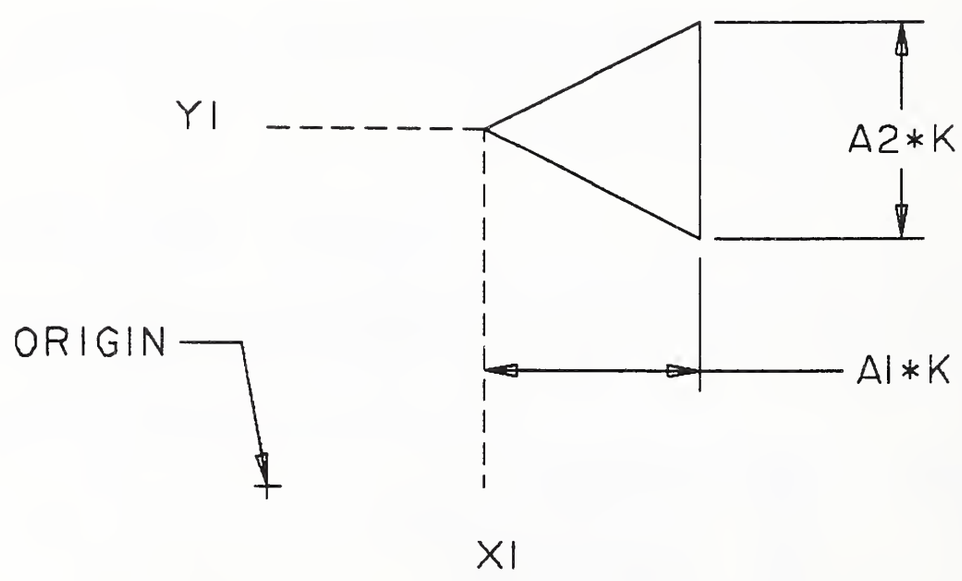


Figure 88. Parameters of the Isoceles Triangle Macro in Example 1 in Text

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 89. 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:

Directory Data
Entity Type Number: 600

Parameter Data

```
600, 1., 1., 2., 5., 5., 2., 1., 3;
```

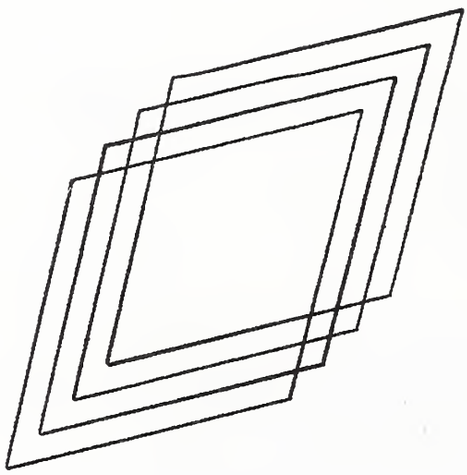


Figure 89. Repeated Parallelograms Created by Macro Example 2 in Text

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 90 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:

Directory Data
Entity Type Number: 601

Parameter Data

```
601, 0., 0., 0., 20., 4;
```

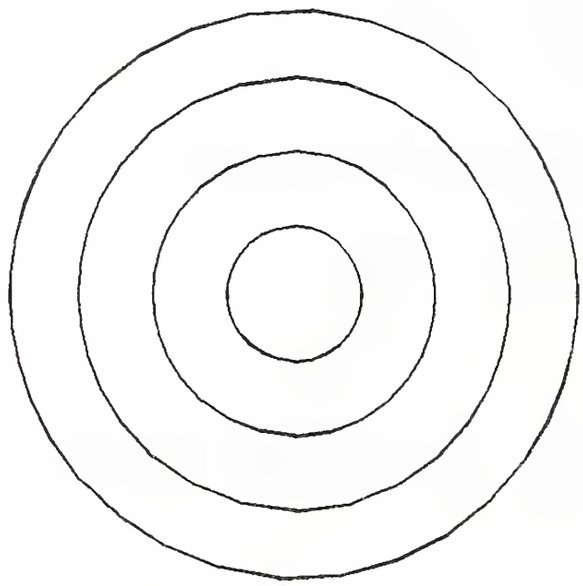


Figure 90. Concentric Circles Created by Macro Example 3 in Text

Example 4: Electrical ground symbol

This MACRO takes a point and a base length and constructs a ground symbol (horizontally) at that point. Figure 91 shows the result.

Directory Data
Entity Type Number: 306

Parameter Data

```
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:

Directory Data
Entity Type Number: 602

Parameter Data

```
602, 1., 6., 2., 1.3;
```

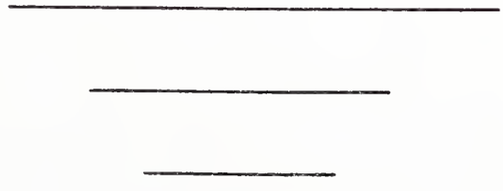


Figure 91. Ground Symbol Created by Macro Example 4 in Text

4.3.6.5 EXAMPLES OF MACRO USAGE

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:

Directory Data
Entity Type Number: 613

Parameter Data

```
613, 4, 5, 0.2, 0.1, 7.85398D-01;
```

4.3.7 Property Entity (Type 406). 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, 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 1.6.1, when an instance is independent it applies to all entities on the same level as the instance.

4.3.7.1 Directory Data
Entity Type Number: 406

4.3.7.2 Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of property values
2	V1	Variable	Property values
.			
.			
NP+1	VNP		

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.1 (TYPE 406, FORM 1) - DEFINITION LEVELS

4.3.7.3 Defined Properties

4.3.7.3.1 FORM NUMBER: 1 Definition levels

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, NP, 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.

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of property values
2	L1	Integer	Level number
.	.	.	
NP+1	LNP	.	.

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.2 FORM NUMBER: 2 Region Restriction

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

Parameter Data

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 Section 2.2.4.4.2).

4.3.7.3.3 (TYPE 406, FORM 3) - LEVEL FUNCTION

4.3.7.3.3 FORM NUMBER: 3 Level Function

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 1.6.1). 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.

Parameter Data

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 Section 2.2.4.4.2).

4.3.7.3.4 FORM NUMBER: 5 Line Widening

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 92 indicates the measurement of the property values.

Parameter Data

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 6
5	JF	Integer	Justification flag 0 = center justified 1 = left justified 2 = right justified
6	E	Real	Extension value, if Parameter 4=2 (Note: this value may be negative)

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.4 (TYPE 406, FORM 5) - LINE WIDENING

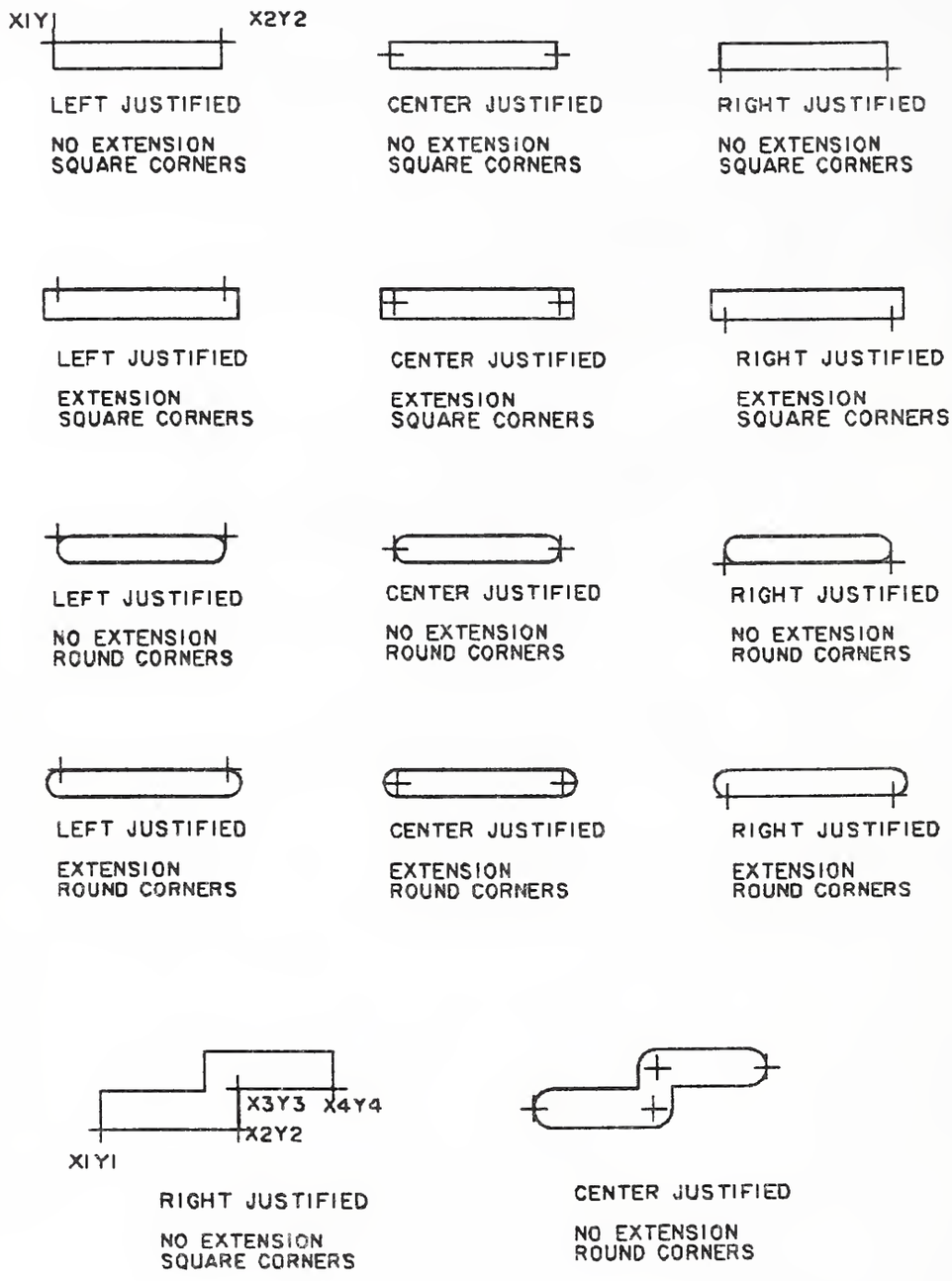


Figure 92. Measurement of the Line Widening Property Values

4.3.7.3.5 FORM NUMBER: 6 Drilled Hole

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.

Parameter Data

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 (see Section 2.2.4.4.2).

4.3.7.3.6 FORM NUMBER: 7 Reference Designator

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 Entity (Type 420) as that value is included in the subfigure parameters.

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=1)
2	RD	String	Reference designator text

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.7 FORM NUMBER: 8 Pin Number

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 (Type 132) as the pin number is included in one of the Connect Point parameters.

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=1)
2	PN	String	Pin Number Value

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.8 (TYPE 406, FORM 9) - PART NUMBER

4.3.7.3.8 FORM NUMBER: 9 Part Number

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. Defaulted strings in any parameter will imply that the missing value is not relevant to the transferred data.

Parameter 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	Military Standard (MIL-STD) part number
4	VPN	String	Vendor part number or name
5	IPN	String	Internal part number

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.9 FORM NUMBER: 10 Hierarchy

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.

Parameter Data

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 Section 2.2.4.4.2).

Acceptable values for Parameters 2 through 7 are 0 and 1. (See definition in Section 2.2.4.3.9.4).

4.3.7.3.10 FORM NUMBER: 11 Tabular Data

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 for this property shall follow the International System of Units (SI) practice for base units and derived units (IEEE76). Typical units are as follows:

Base Units	Unit	Symbol
Length	Meter	m
Mass	Kilogram	kg
Time	Second	s
Electric Current	Ampere	A

Derived Units	Unit	Symbol	Formula
Force	Newton	N	(kg*m/s)/s
Energy	Joule	J	N*m

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number 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

Index	Name	Type	Description
2 cont.	PTYPE	Integer	Property Type 11 = Material Coordinate System ND=3 12 = Nodal Load/Constraint Data ND=Number of Degrees of Freedom 13 = Sectional Properties for Beam Elements ND=8 if the properties are the same at both ends of the beam otherwise, 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
3	ND	Integer	Number of dependent variables
4	NI	Integer	Number of independent variables
5	TYPI1	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
.	.	.	
.	.	.	
5+NI	TYPINI	Integer	Type of the last independent variable
6+NI	NVALI1	Integer	Number of different values of the first independent variable
.	.	.	
.	.	.	
6+2*NI	NVALINI	Integer	Number of different values of the last independent variable
7+2*NI	VALI(1,1)	Real	First value of the first independent variable
.	.	.	
.	.	.	
	VALI(1,NVALI1)	Real	Last value of the first independent variable
.	.	.	
.	.	.	
	VALI(NI,NVALINI)	Real	Last value of the last independent variable
	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

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

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}, 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:

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{bmatrix} = \begin{bmatrix} E_{xx} & 0 & 0 \\ 0 & E_{yy} & 0 \\ 0 & 0 & E_{zz} \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \end{bmatrix}$$

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.*,

$$\nu_{ij} = \epsilon_j / \epsilon_i$$

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_{yz}, ν_{zx}.

The off diagonal matrix values are reciprocals of the principal values, *i.e.*:

$$\nu_{xy} = 1/\nu_{yx}$$

This implies that ND (Number of Dependent Variables) is equal to three.

In matrix form for an orthotropic material:

$$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \end{bmatrix} = \begin{bmatrix} 1/E_{xx} & -\nu_{yx}/E_{yy} & -\nu_{zx}/E_{zz} \\ -\nu_{xy}/E_{xx} & 1/E_{yy} & -\nu_{zy}/E_{zz} \\ -\nu_{xz}/E_{xx} & -\nu_{yz}/E_{yy} & 1/E_{zz} \end{bmatrix} \begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{bmatrix}$$

PTYPE = 3 Shear Modulus

Shear Modulus - The ratio of shear stress to shear strain.

$$G = \tau_s/\gamma$$

Where G is the Shear Modulus
τ_s is the Shear Stress
γ is the Shear Strain

The Shear Modulus is a vector with three principal values:

$$G_{xy},\ G_{yz},\ \text{and}\ G_{zx}.$$

This implies that the ND (Number of Dependent Variables) is equal to three.

In matrix form for orthotropic materials:

$$\begin{bmatrix} \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{bmatrix} = \begin{bmatrix} 1/G_{xy} & 0 & 0 \\ 0 & 1/G_{yz} & 0 \\ 0 & 0 & 1/G_{zx} \end{bmatrix} \begin{bmatrix} \tau_{s_{xy}} \\ \tau_{s_{yz}} \\ \tau_{s_{zx}} \end{bmatrix}$$

PTYPE = 4 Material Matrix

Material matrix defines the tensor qualities of the material. For example:

$$\bar{\sigma} = [C]\ \bar{\epsilon}$$

Where $\bar{\sigma}$ 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:

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & C_{15} & C_{16} \\ C_{21} & C_{22} & C_{23} & C_{24} & C_{25} & C_{26} \\ C_{31} & C_{32} & C_{33} & C_{34} & C_{35} & C_{36} \\ C_{41} & C_{42} & C_{43} & C_{44} & C_{45} & C_{46} \\ C_{51} & C_{52} & C_{53} & C_{54} & C_{55} & C_{56} \\ C_{61} & C_{62} & C_{63} & C_{64} & C_{65} & C_{66} \end{bmatrix} \begin{bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{bmatrix}$$

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

PTYPE = 5 Mass Density

Mass density is the Mass/Unit Volume.

This implies that ND = 1.

Mass Density = ρ.

PTYPE = 6 **Thermal Expansion Coefficient**

The Thermal Expansion Coefficient is a material property that computes the strain given a temperature differential, *i.e.*,

$$\epsilon = \alpha \Delta T \quad \text{or} \quad \alpha = \epsilon / \Delta T$$

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:

$$\alpha_{xx},\ \alpha_{yy} \text{ and } \alpha_{zz}.$$

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 93. 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

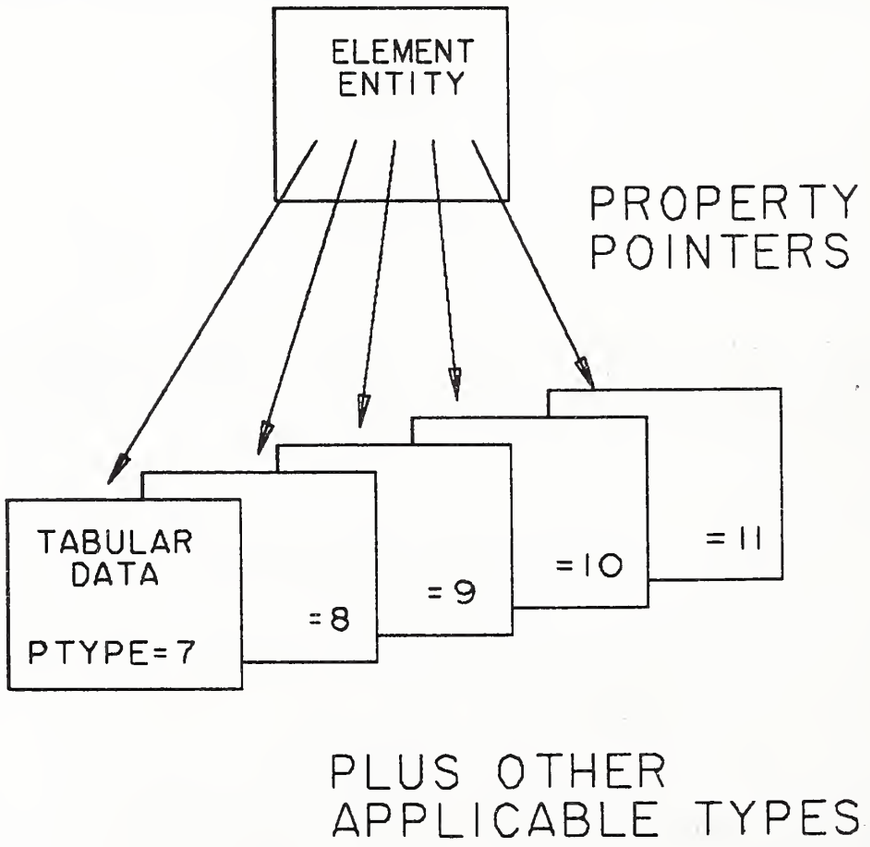


Figure 93. Relationship Between Properties Used to Represent a Composite Material

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 ϵ_x, ϵ_y, ϵ_{xy})
t = Shell thickness - see element property
$[M]$ = Membrane material stiffness matrix

$\bar{f}$ and $\bar{\epsilon}$ are defined in the shell material coordinate system, PTYPE = 11.

Because of symmetry, the elements $M_{ji} = M_{ij}$. 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 plies is:

$$[M_{ij}] = \frac{1}{t} \sum_{n=1}^{m} [G_{ij}]_n \, \Delta t_n$$

Where
Δt_n is thickness of n^{th} ply
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:

$$\overline{m} = \frac{1}{12} t^3 \, [B] \, \bar{\chi}$$

Where $\overline{m}$ = Shell bending moments per unit length (row of elements M_x, M_y, M_{xy})

$\bar{\chi}$ = 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 $\bar{\chi}$ are defined in shell material coordinate system, PTYPE=11

Because of symmetry, the elements $B_{ij} = B_{ji}$. 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 plies is:

$$[B_{ij}] = 12 \, t^{-3} \sum_{n=1}^{m} \left(Z_n^2 \, [G_{ij}]_n \, \Delta t_n \right)$$

Where $[G]_n$ is the stress strain matrix for the n^{th} ply of laminate

Δt_n is thickness of n^{th} ply

t is total thickness of laminate

Z_n is the normal distance from midplane of shell to the centroid of the ply

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:

$$\bar{V} = t_s [S] \bar{\gamma}$$

Where $\bar{V}$ = Transverse shear force per unit length (row of elements V_x, V_y)

$\bar{\gamma}$ = Transverse shear strains, dimensionless (row of elements γ_x, γ_y)

t_s = 5/6 of effective transverse shear thickness

$[S]$ = Transverse shear material stiffness matrix

$\bar{V}$ and $\bar{\gamma}$ are defined in the material coordinate system

Because of symmetry, the elements $S_{ij} = S_{ji}$. 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.

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

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

$$\overline{m} = t^2 [C]^T \bar{\epsilon}$$

Where
- $\bar{f}$ = Forces per unit length (row of elements f_x, f_y, f_{xy})
- $\overline{m}$ = Bending moments per unit length (row of elements m_x, m_y, m_{xy})
- $\bar{\chi}$ = Curvature (measurement of bending strain, (meter)$^{-1}$) (row of elements χ_x, χ_y, χ_{xy})
- $\bar{\epsilon}$ = Midplane strain (row of elements ϵ_x, ϵ_y, ϵ_{xy})
- t = shell thickness

$\bar{f}$, $\overline{m}$, $\bar{\chi}$, 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}

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 plies is:

$$[C_{ij}] = t^{-2} \sum_{n=1}^{m} (Z_n [G_{ij}]_n \Delta t_n)$$

Where
- $[G]_n$ = stress strain matrix for n^{th} ply, defined in shell material coordinate system
- Δt_n = thickness of n^{th} ply
- Z_n = the normal distance from midplane of the shell to centroid of n^{th} ply
- t = 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 $\overline{D}$. The use of the vector $\overline{D}$ depends upon the element type.

For Element Topology Type 1 and 33 of the Finite Element Entity (Type 136), Figure 94 illustrates the use of the vector $\overline{D}$ to define the element material coordinate system. For Topology Type 33 the vector $\overline{D}$ is defined by the Reference Node 3.

The cosines for vector $\overline{D}$ are translated to the location of the shear center offset to establish the reference planes for material property definition (vector $\overline{DT}$).

For Element Topology Types 2 through 26 of the Finite Element Entity (Type 136), the following paragraphs discuss the use of vector $\overline{D}$ to define the material coordinate system.

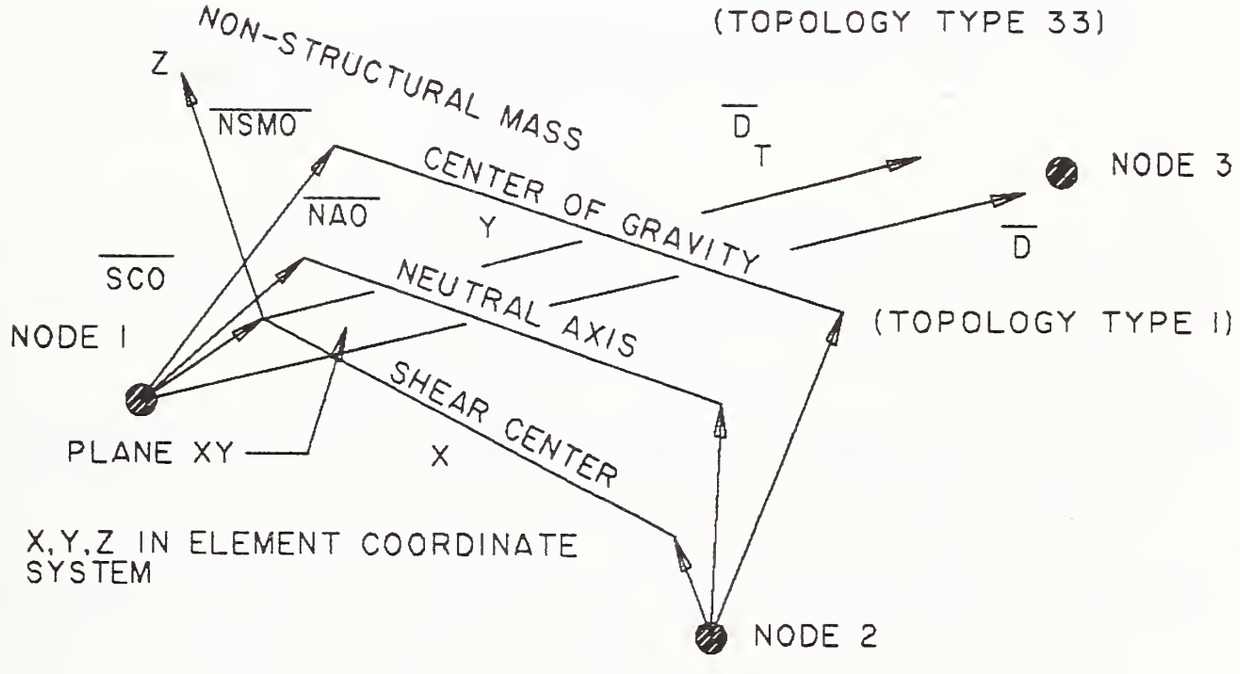


Figure 94. Use of the Vector $\overline{D}$ to Define the Element Material Coordinate System

The projection of $\overline{D}$ on the plane of the element face (F1) (outward normal $\overline{N}$) defines a vector in the X direction of the material coordinate system.

$$\overline{X} = \overline{N} \times \overline{D} \times \overline{N}$$

$\overline{N}$ = positive outward normal of the element face (F1) and is defined by nodal connection of the element, *i.e.:*

$$\overline{N} = \overline{S}_1 \times \overline{S}_2$$

$\overline{S}_1$ = vector from first to second corner of element face (F1)

$\overline{S}_2$ = vector from second to third corner of element face (F1)

Three direction cosines are required to define the Vector $\overline{D}$ in the global coordinate system (therefore, ND = 3):

$$\begin{matrix} D_1 \\ D_2 \\ D_3 \end{matrix}$$

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 95.

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.

$\begin{Bmatrix} f_x \\ f_y \\ f_{xy} \end{Bmatrix} = \overline{f}$ forces per unit length

$\begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_{xy} \end{Bmatrix} = \epsilon$ midplane strains

$\begin{Bmatrix} M_x \\ M_y \\ M_{xy} \end{Bmatrix} = \overline{M}$ moments per unit length

$\begin{Bmatrix} \chi_x \\ \chi_y \\ \chi_{xy} \end{Bmatrix} = \chi$ bending curvatures

$\begin{Bmatrix} V_x \\ V_y \end{Bmatrix} = \overline{V}$ transverse shear forces unit length

$\begin{Bmatrix} \gamma_x \\ \gamma_y \end{Bmatrix} = \gamma$ transverse shear strains

Strain Displacement Relationships:

$$\epsilon_x = \frac{\partial u}{\partial x} \qquad \epsilon_y = \frac{\partial \nu}{\partial y} \qquad \epsilon_{xy} = \frac{\partial u}{\partial y} + \frac{\partial \nu}{\partial x}$$

$$\chi_x = \frac{\partial^2 u}{\partial^2 x} \qquad \chi_y = \frac{\partial^2 \nu}{\partial^2 w} \qquad \chi_{xy} = 2\frac{\partial^2 w}{\partial x \partial y}$$

$$\gamma_x \approx \frac{\partial w}{\partial x} \qquad \gamma_y \approx \frac{\partial w}{\partial y}$$

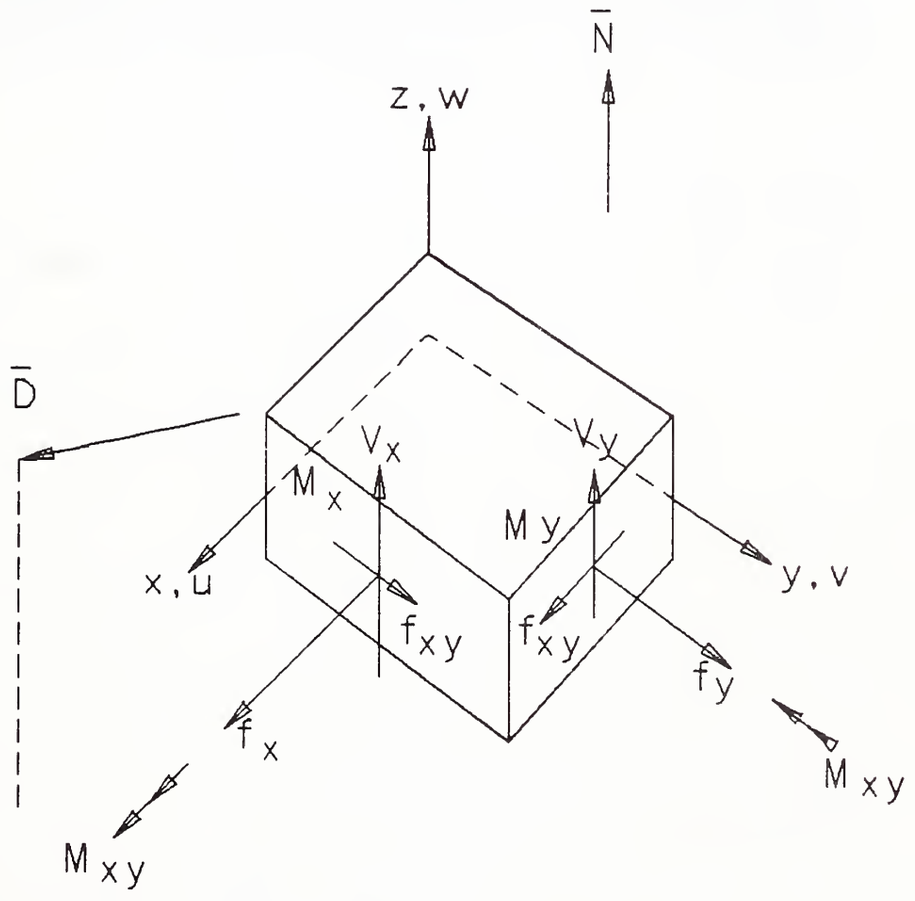


Figure 95. Internal Load and Strain Sign Convention

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

PTYPE = 12 **Nodal Loads/Constraints Data**

The nodal load/constraint data will be stored in the Tabular Data Form of the Property Entity in the following manner:

PTYPE = 12

ND = Number of degrees of freedom.

For example, if the load vector has X, Y, Z, M_x, M_y, and M_z components, then ND=6. (Note M_x, M_y, M_z refer to moments). If the load vector has X and Y components, then 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 with the 4th degree for the rotation X component.

The constraint vector has X, Y, Z, M_x, M_y, and M_z constraints. These constraints are represented by 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 Name	Units	Description
AREA	M^2	Area of section
IX	M^4	Area moment of inertia about the element x-axis
IY	M^4	Area moment of inertia about the element y-axis
IXY	M^4	Product of inertia
J	M^4	Torsional stiffness parameter
SRXY	Unitless	Shear stiffness ratio
SRXZ	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.

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

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 Name	Description
For Each End	X	X direction translation freedom/constraint
	Y	Y direction translation freedom/constraint
	Z	Z direction translation freedom/constraint
	MX	X direction rotational freedom/constraint
	MY	Y direction rotational freedom/constraint
	MZ	Z direction rotational freedom/constraint

The value for each property X, Y, Z, M_x, M_y, and M_z is 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 94 shows the Shear Center Offset, SCO, the Neutral Axis Offset, NAO, and the Non-Structural Mass Offset, NSMO.

These offsets are vectors in the global coordinate system (model space) relative to the end of the beam.

	Property Name	Description
For Each End	SCOX	Shear Center Offset in global x direction
	SCOY	Shear Center Offset in global y direction
	SCOZ	Shear Center Offset in global z direction
	NAOX	Neutral Axis Offset in global x direction
	NAOY	Neutral Axis Offset in global y direction
	NAOZ	Neutral Axis Offset in global z direction
	NSMOX	Non – Structural Mass Offset in global x direction
	NSMOY	Non – Structural Mass Offset in global y direction
	NSMOZ	Non – Structural Mass Offset in global z direction

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.

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

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 the beam element axis. These points occur in pairs from one end of the beam to the other.

	Property Name	Description
END1 first pair offsets	SRI1X1	Stress Recovery Information beam end 1 x – direction pair 1
	SRI1Y1	Stress Recovery Information beam end 1 y – direction pair 1
	SRI1Z1	Stress Recovery Information beam end 1 z – direction pair 1
END2 first pair offsets	SRI2X1	Stress Recovery Information beam end 2 x – direction pair 1
	SRI2Y1	Stress Recovery Information beam end 2 y – direction pair 1
	SRI2Z1	Stress Recovery Information beam end 2 z – direction pair 1 (*for the first pair of offsets*)
⋮	⋮	⋮
END1 fourth pair offsets	SRI1X4	Stress Recovery Information beam end 1 x – direction pair 4
	SRI1Y4	Stress Recovery Information beam end 1 y – direction pair 4
	SRI1Z4	Stress Recovery Information beam end 1 z – direction pair 4
END2 fourth pair offsets	SRI2X4	Stress Recovery Information beam end 2 x – direction pair 4
	SRI2Y4	Stress Recovery Information beam end 2 y – direction pair 4
	SRI2Z4	Stress Recovery Information beam end 2 z – direction pair 4 (*for the fourth pair of offsets*)

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	Description
1	Scalar thickness of element
n	Thickness of the n^{th} plate of a sandwich or laminate

The thickness is ordered from 1 to n.

The thicknesses are measured in the positive z direction in order of increasing z in local element coordinate system.

Property Name	Description
T1	Thickness for homogeneous plate or the thickness for the first lamina of the plate
.	.
.	.
.	.
Tn	Thickness for the nth lamina of the plate

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

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. Thus, ND=1.

Property Name	Description
NSM	Mass/unit length or Mass/unit area or Mass/unit volume

Where the description depends on the 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:

$$\frac{\partial}{\partial x}\left[k_x \frac{\partial T}{\partial x}\right] + \frac{\partial}{\partial y}\left[k_y \frac{\partial T}{\partial y}\right] + \frac{\partial}{\partial z}\left[k_z \frac{\partial T}{\partial z}\right] = \rho\, C_p \frac{\partial T}{\partial t} - \dot{Q}_I$$

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
- $\dot{Q}_I$ = Rate that energy is converted to internal heat

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.*,

$$\frac{\partial}{\partial n}\left[k_n \frac{\partial T}{\partial n}\right] = k_n \frac{\partial^2 T}{\partial n^2}$$

Where n is x, y, or z.

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. In matrix form the heat balance equation is:

$$[k_x\, k_y\, k_z] \begin{bmatrix} \frac{\partial^2 T}{\partial x^2} \\ \frac{\partial^2 T}{\partial y^2} \\ \frac{\partial^2 T}{\partial z^2} \end{bmatrix} = \rho\, C_p \frac{\partial T}{\partial t} - \dot{Q}_I$$

Now assume that there are no internal heat sources $\dot{Q}_I$ and the heat flow is steady state, ($\partial T/\partial t = 0$). Then, integrating the above equation in one dimension yields

$$\dot{Q}_x = k_x\, A \frac{\partial T}{\partial x}$$

where $\dot{Q}_x$ 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 directions.

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

Property Name	Description
KX	Thermal Conductivity coefficient x direction
KY	Thermal Conductivity coefficient y direction
KZ	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.

$$\frac{\partial}{\partial x}\left[k_x \frac{\partial T}{\partial x}\right] + \frac{\partial}{\partial y}\left[k_y \frac{\partial T}{\partial y}\right] + \frac{\partial}{\partial z}\left[k_z \frac{\partial T}{\partial z}\right] = \rho C_p \frac{\partial T}{\partial t} - \dot{Q}_I$$

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
$\dot{Q}_I$ = Rate that energy is converted to internal heat

If we consider constant pressure, the heat capacity can be considered a constant.

Consequently, ND=1 to define the constant.

Property Name	Description
CP	Heat capacity at constant pressure

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

PTYPE = 21 **Convective Film Coefficient**

Convective film coefficient relates to the amount of heat flux that is convected to adjacent materials at the interface boundary of a heat source.

$$\dot{Q} = -h_c A \Delta T$$

Where $\dot{Q}$ = 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 Name	Description
HC	Convective Film Coefficient

PTYPE = 22 **Electromagnetic Radiation Parameters**

Properties for Absorptivity, Transmissivity, Reflectivity, and Emissivity are defined for structural elements using four values. Thus, ND=4.

Property Name	Description
A	Absorptivity Constant
T	Transmissivity Constant
R	Reflectivity Constant
E	Emissivity Constant

4.3.7.3.10 (TYPE 406, FORM 11) - TABULAR DATA

Examples of the use of the Tabular Data Form of the Property Entity:

Consider the representation of the mass density (PTYPE = 5) as a function of pressure. In this case, there is one independent variable. Suppose the density is known for two values of pressure. The Parameter Data Section would contain:

Index	Name	Recorded Value
1	NP	9
2	PTYPE	5
3	ND	1
4	NI	1
5	TYPI	2
6	NVALI	2
7	VALI1	50
8	VALI2	25
9	VALD(1,1)	33
10	VALD(1,2)	46

as well as additional pointers as required (see Section 2.2.4.4.2).

Consider the representation of Young's modulus (PTYPE = 1) for a linear, static, independent case.In this case, there is no independent variable. The Parameter Data Section would contain:

Index	Name	Recorded Value
1	NP	6
2	PTYPE	1
3	ND	3
4	NI	0
5	Exx	30.0E6
6	Eyy	30.0E6
7	Ezz	30.0E6

As well as additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.11 FORM NUMBER: 12 External Reference File List

Description

The External Reference File List appears in a file which references definitions that reside in another file. It contains a list of the names of the files directly referenced by entities within this file. See Section 2.5.4 and the External Reference Entity (Type 416) for more detail.

Index	Name	Type	Description
1	NP	Integer	Number of List Entries
2	NAME1	String	First External Reference File Name
.	.	.	.
NP+1	NAMENP	String	Last External Reference File Name

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.12 (TYPE 406, FORM 13) - NOMINAL SIZE

4.3.7.3.12 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 names have predefined meanings:

Nominal Size Name	Predefined 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	Nominal size name
4	SP	String	Name of relevant engineering standard (optional)

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.13 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	Primary flow line specification name
3	L2	String	Modifier (optional)
.	.	.	.
.	.	.	.
.	.	.	.
NP+1	LNP	String	Modifier (optional)

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.14 (TYPE 406, FORM 15) - NAME

4.3.7.3.14 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 explicitly specified in the parameter data for the entity.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=1)
2	NAME	String	Entity Name

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.15 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	Number 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 Section 2.2.4.4.2).

4.3.7.3.16 (TYPE 406, FORM 17) - DRAWING UNITS

4.3.7.3.16 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 Section 2.2.4.2.15).

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP=2)
2	FLAG	Integer	Units Flag
3	UNIT	String	Units Name

Additional pointers as required (see Section 2.2.4.4.2).

4.3.7.3.17 FORM NUMBER: 18 Intercharacter Spacing

Description

The definition of this property can be found in Appendix J (see Section J.10).

4.3.8 Subfigure Definitions. The following definition entities are used to provide a "template" for the instances of subfigures (see Section 2.5.1).

4.3.8.1 Subfigure Definition Entity (Type 308). 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	DE1	Pointer	Pointer to the first Directory Entry for the associated entities
.	.	.	
.	.	.	
.	.	.	
N+3	DEN	Pointer	Pointer to the last Directory Entry for the associated entities

Additional pointers as required (see Section 2.2.4.4.2).

4.3.8.2 Network Subfigure Definition Entity (Type 320). 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 following) 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 in the Subfigure Definition Entity (Type 308).

Note: The depth of the subfigure is inclusive of both the Network Subfigure Definition Entity (Type 320) and the ordinary Subfigure Definition 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
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 Directory Entry for associated Connect Point 1
.	.	.	
.	.	.	
.	.	.	
NA+NC+7	CPTRNC	Pointer	Pointer to the Directory Entry for associated Connect Point NC

Additional pointers as required (see Section 2.2.4.4.2).

4.3.9 Subfigure Instances

4.3.9.1 Singular Subfigure Instance Entity (Type 408). This entity defines the occurrence of a single instance of the defined subfigure (Type 308). See Figure 96 and Section 2.5.1.

4.3.9.1.1 Directory Data

Entity Type Number: 408

4.3.9.1.2 Parameter Data

Index	Name	Type	Description
1	DE	Pointer	Pointer to Subfigure Definition Entity
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 Section 2.2.4.4.2).

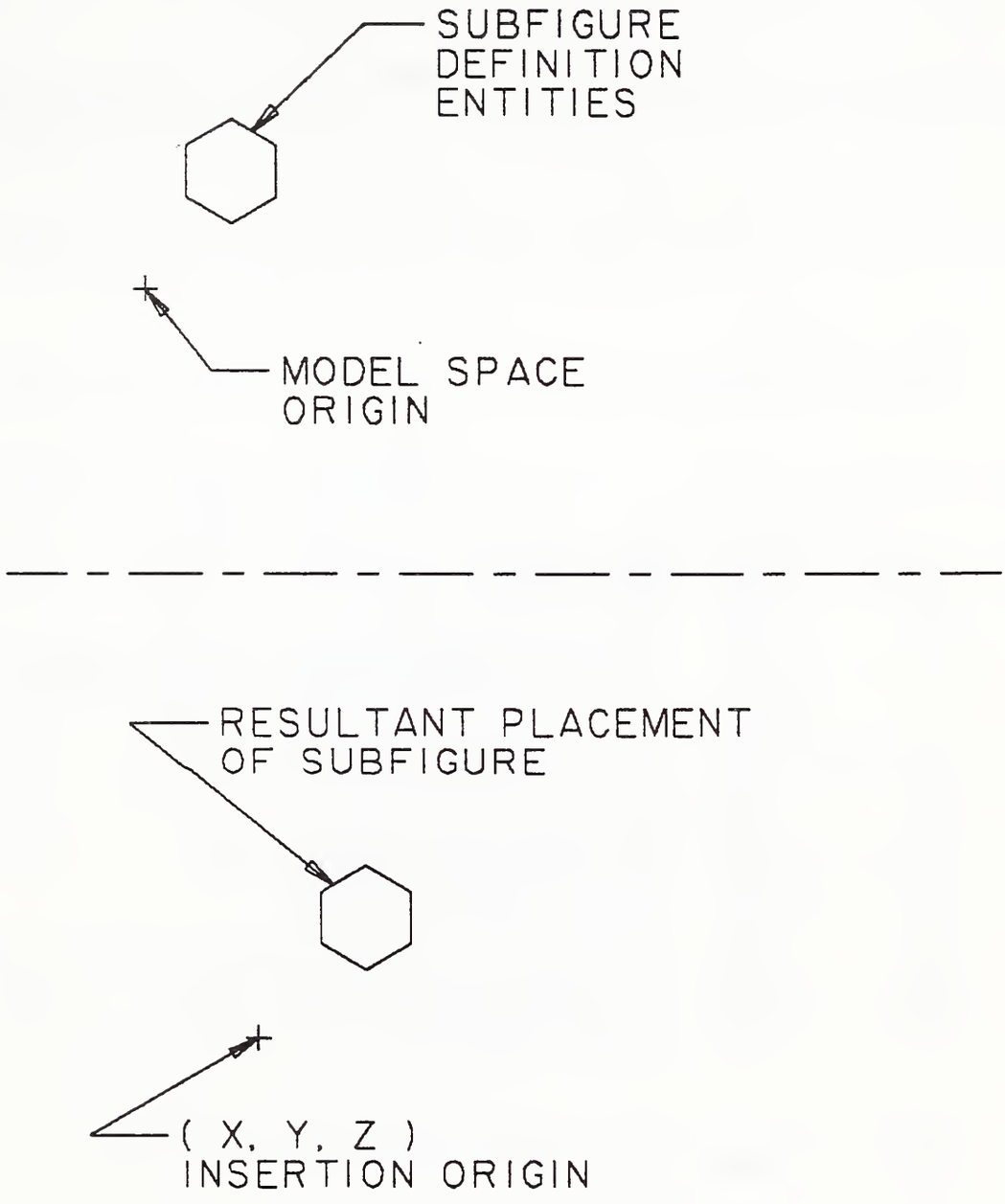


Figure 96. Relationship Between Subfigure Definition and Subfigure Instance

4.3.9.2 Rectangular Array Subfigure Instance Entity (Type 412). The Rectangular Array Subfigure Instance Entity 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 are given. This is where the first entity in the reproduction process is placed and is called position Number 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 value is chosen, half or fewer of the elements of the rectangular array are to be defined. If the DON'T value 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 to be used as lower left corner of array
4	Y	Real	
5	Z	Real	
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 (LC=0 indicates all to be displayed.)
12	DDF	Integer	DO-DON'T flag (DO=0; DON'T=1)
13	N1	Integer	Number of first position to be processed (DO), or not to be processed (DON'T)
.	.	.	.
.	.	.	.
12+LC	NLC	Integer	Number of last position

Additional pointers as required (see Section 2.2.4.4.2).

4.3.9.3 Circular Array Subfigure Instance Entity (Type 414). The Circular Array Entity 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 Subfigure 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 value is chosen, half or fewer of the elements of the circular array are to be defined. If the DON'T value 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 of imaginary circle
4	Y	Real	
5	Z	Real	
6	R	Real	Radius of imaginary circle
7	AS	Real	Start angle in radians
8	AD	Real	Delta angle in radians
9	LC	Integer	DO-DON'T list count (LC=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 Section 2.2.4.4.2).

4.3.9.4 Network Subfigure Instance Entity (Type 420). Each instance of a Network Subfigure Definition Entity (Type 320) 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, and is required if both are present in the file.

The Network Subfigure Instance Entity allows different scale factors in the x, y, and z directions. 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 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	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 (default) 1 - logical 2 - physical
9	PRD	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 Section 2.2.4.4.2).

4.3.10 Text Font Definition Entity (Type 310). 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 (AC). 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, if negative, is 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 up/down 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 or Pointer	Number of the font which this definition supersedes

4.3.10 TEXT FONT DEFINITION ENTITY (TYPE 310)

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
8	NY1	Integer	"
9	NM1	Integer	Number of pen motions for first character
10	PF1(1)	Integer	Pen up/down flag: 0 = Down (default), 1 = Up
11	X1(1)	Integer	Grid location to which the pen is to move
12	Y1(1)	Integer	"
.	.	.	
8+NM1*3	Y1(NM1)	Integer	Last grid location of first character
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+ $3*\sum$NMi	YN(NMN)	Integer	Last grid location of last character

Additional pointers as required (see Section 2.2.4.4.2).

Examples of character definitions are shown in Figures 97 and 98. The parameters for the first example are:

```
FC        1
FNAME     8HSTANDARD
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 Data Section of the file, this definition would look like:

```
1,8HSTANDARD,,8,60,65,11,0,4,,4,8,,8,0,1,2,4,,6,4....
```

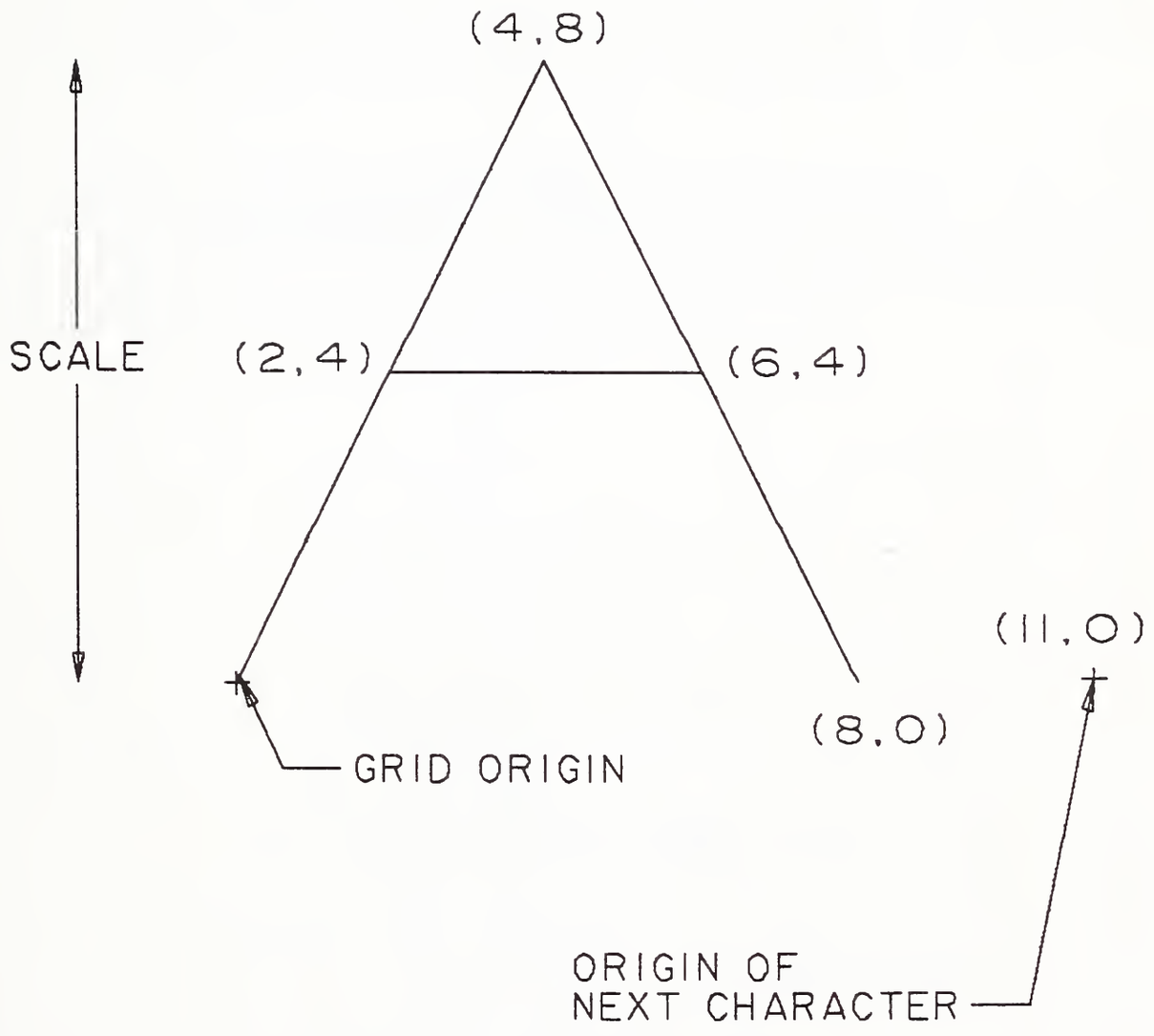


Figure 97. Example of a Character Definition

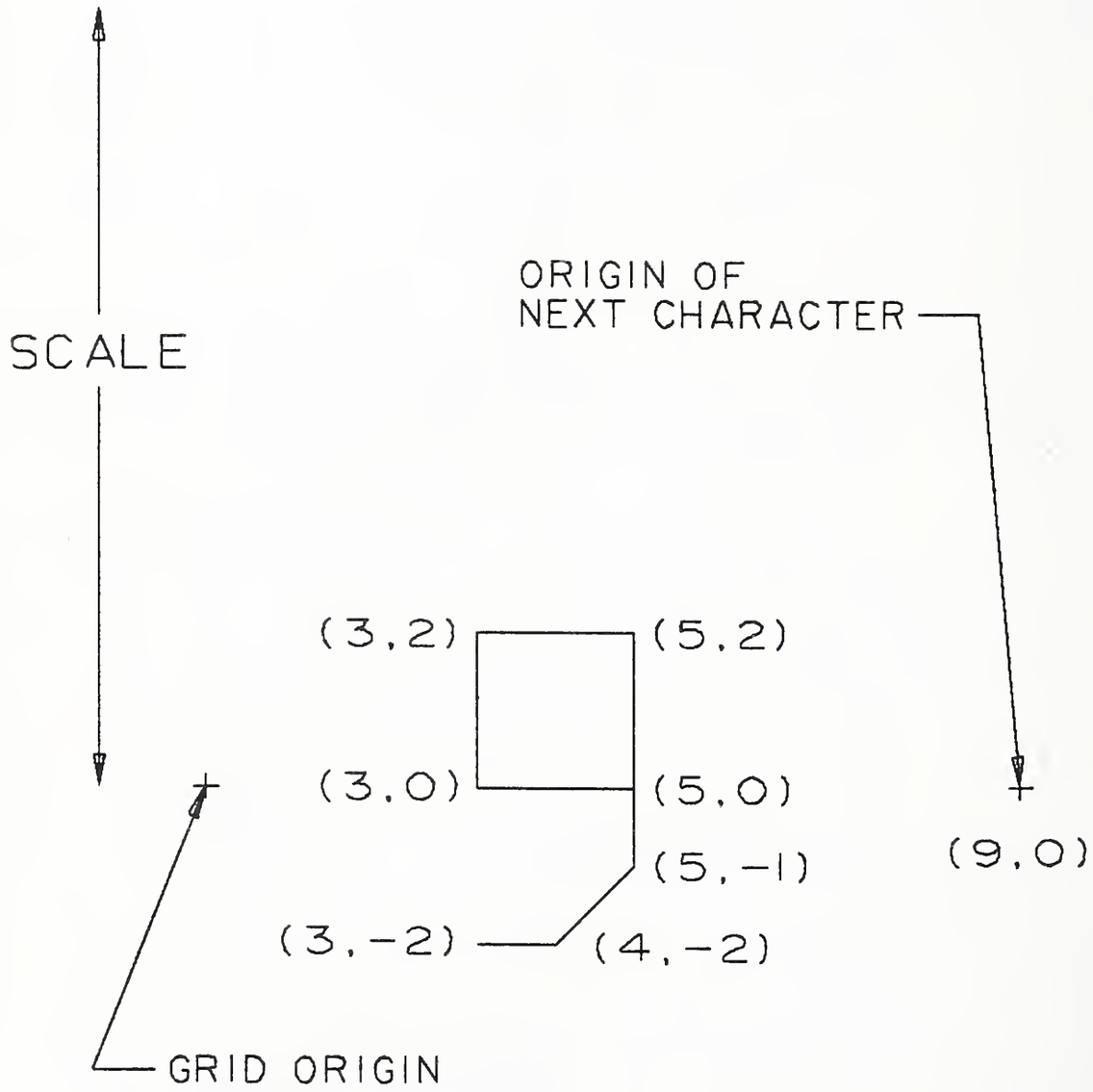


Figure 98. Example of a Character Definition Including Descenders

4.3.11 View Entity (Type 410). View Entity (Type 410) 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.

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 99.

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 100 is called the view origin. Thus, a complete viewing orientation is specified by a view coordinate system.

View Coordinates Obtained from Model Coordinates. View coordinates are obtained from model coordinates through translation and rotation. There are several ways that systems 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 Section 3.14).

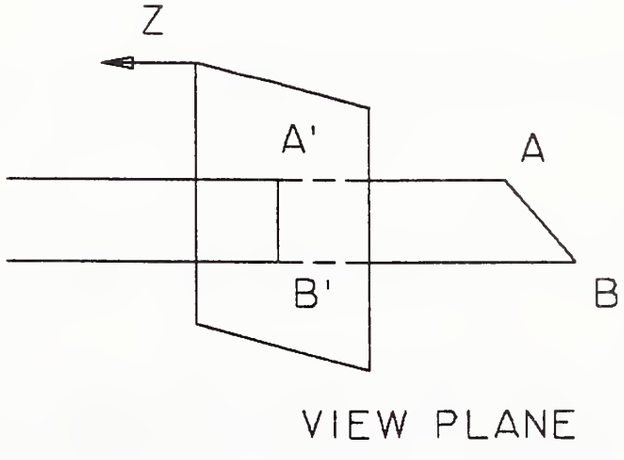


Figure 99. Orthographic Parallel Projection of AB on a View Plane

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$

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 DE Field 7. In the special case when the view matrix is the identity matrix and there is zero translation, a zero value in DE 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:

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix} \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} XO \\ YO \\ ZO \end{bmatrix} \right)$$

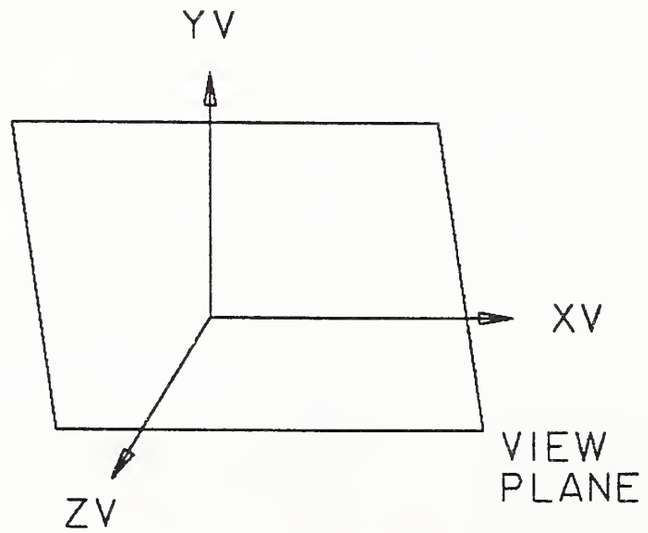


Figure 100. View Coordinate System (View Origin at XV=YV=ZV=0)

or,

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix} \begin{bmatrix} XO \\ YO \\ ZO \end{bmatrix}.$$

Therefore, the rotation matrix is the view matrix, and in the Transformation Matrix Entity,

$$R = \begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix}, \qquad T = -\begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix} \begin{bmatrix} XO \\ YO \\ ZO \end{bmatrix}$$

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

$$\begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} = \begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} XL \\ YL \\ ZL \end{bmatrix}$$

Therefore, the rotation matrix is the view matrix, and in the Transformation Matrix Entity,

$$R = \begin{bmatrix} 3 \times 3 \\ \text{Rotation} \\ \text{Matrix} \end{bmatrix}, \qquad T = -\begin{bmatrix} XL \\ YL \\ ZL \end{bmatrix}.$$

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 DE Field 7), and a view number. The Transformation Matrix Entity specifies a view matrix R and a translation vector T as given in the preceding section.

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 Section 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 indicated by setting the pointer for the appropriate Plane Entity equal to zero.

The Plane Entities used to define the view volume shall not be arbitrary planar definitions (see Figure 101). After the transformation from model coordinates to view coordinates, each plane must be perpendicular to the appropriate view coordinate system axis (*e.g.*, the left side of the view volume

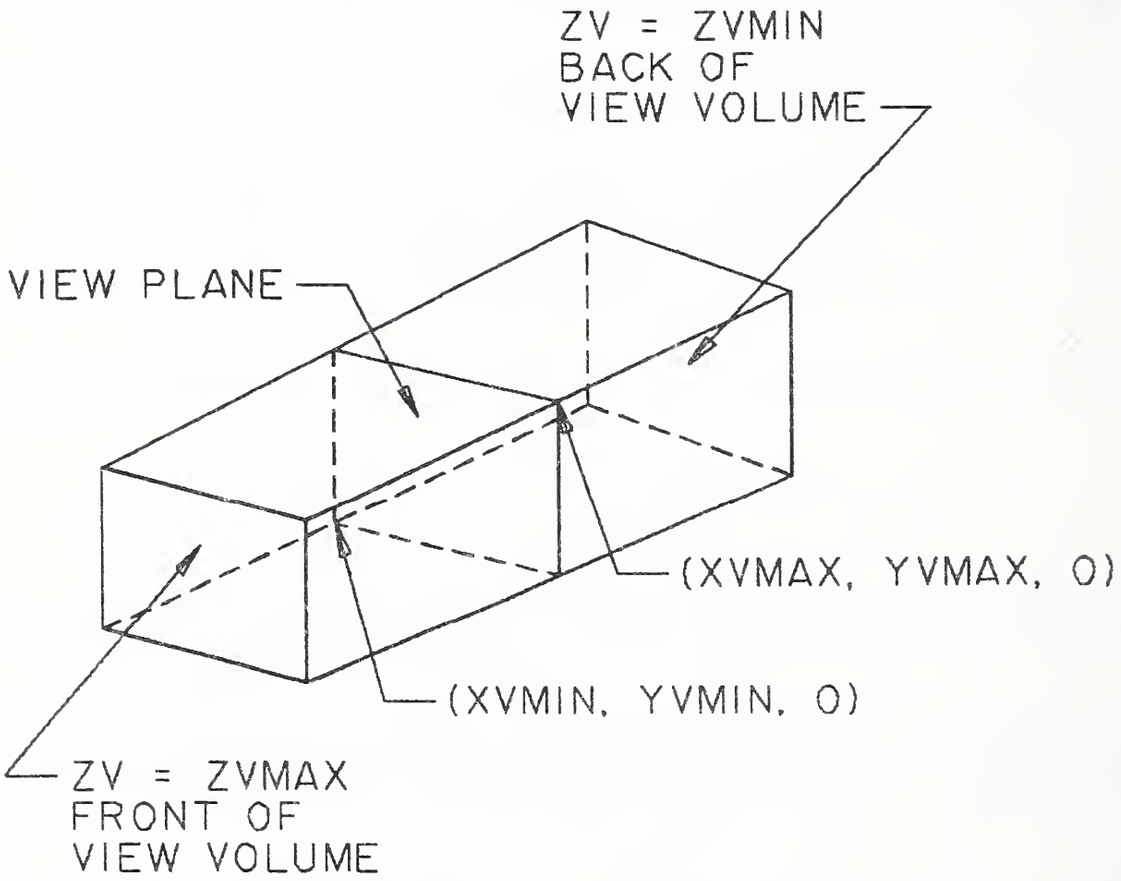


Figure 101. Planes Defining the View Volume

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.

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 Section 4.3.4).

$$\begin{bmatrix} XD \\ YD \end{bmatrix} = \begin{bmatrix} S & 0 & 0 \\ 0 & S & 0 \end{bmatrix} \begin{bmatrix} XV \\ YV \\ ZV \end{bmatrix} + \begin{bmatrix} XORIGIN \\ YORIGIN \end{bmatrix}$$

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 402, Form 3 or 4). The view value for the entity then is a pointer to this associativity instead of to a View Entity.

4.3.11.1 Directory Data

Entity Type Number: 410

4.3.11.2 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 Section 2.2.4.4.2).

4.3.12 External Reference Entity (Type 416). The External Reference Entity provides a link between an entity in a referencing file and the definition of a logically related entity in a 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 Type Number	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 Def.	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 Section 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 Section 2.2.4.4.2).

4.3.13 Nodal Load/Constraint Entity (Type 418). 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 102 shows the relationship or linkage between the Nodal Load/Constraint Entity and the Tabular Data Property which carries the load or constraint vector. The relationship or linkage is also shown on the General Note Entity which describes 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 Property containing the load case magnitudes (see also Section 2.5.6).

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	Pointer to Node
4	PTR1	Pointer	Pointer to definition of each load case stored in General Note Entity
.	.	.	.
.	.	.	.
NC+3	PTRNC	Pointer	

Additional Pointers as required (see Section 2.2.4.4.2).

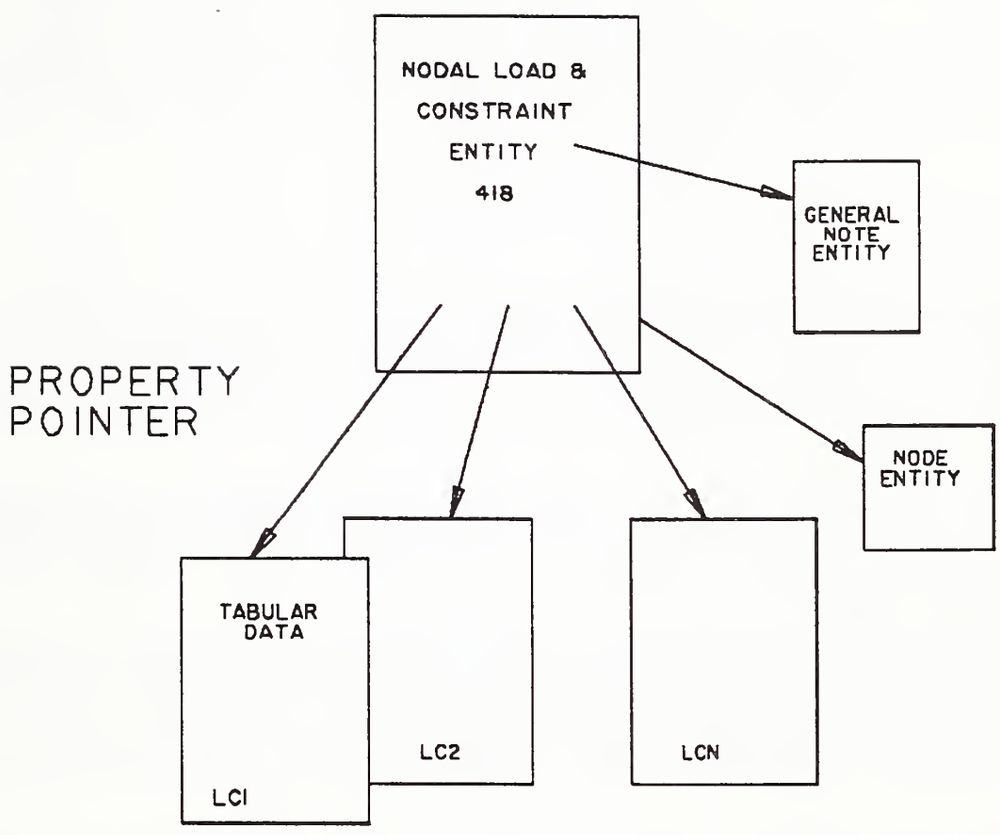


Figure 102. Relationship Between the Nodal Load/Constraint Entity and Tabular Data Properties

4.3.14 Text Display Template Entity (Type 312). The Text Display Template is used to set parameters for display of information which has been logically included in another entity as a parameter value. The text to be displayed is derived from the indicated parameter value of the entity which is pointing to the Text Display Template. In addition to string constants, the parameter values to be displayed may be integer, real or logical constants. The parameter value shall be processed into text as defined in Section 2.2.2 and according to the processing rules presented in the following. Furthermore, the pointer to the Text Display Template may be explicitly defined in the pointing entity description or it may be an implicitly defined additional pointer available with all entities (see Section 2.2.4.4.2). When the pointer is explicitly defined (*e.g.*, Class 6 of the Flow Associativity (Type 402, Form 18)), the specified explicit parameter shall be processed for display. (In the example cited, the parameter to be displayed is the first flow name listed in Class 5.) When, on the other hand, the pointer to the template is one of the additional pointers (as defined in Section 2.2.4.4.2) the parameter to be processed for display is the first information value in the pointing entity. (The information value to be displayed shall be data as opposed to meta-data. For example, if the first parameter of the pointing entity is the number of property values, that should be skipped and the first actual property value should be processed and displayed.) For a more detailed description of the parameters, see the General Note Entity (Type 212).

Processing Rules for Text Display

The following rules are provided for the sake of uniformity in case postprocessed files are to be tested for identical textual presentation of Integer, Real, and Logical values. Strict application of these rules may lead to overwriting other entities, impaired legibility, and reduced visual association with pertinent nearby structures. Consequently, local site standards and conventions may supersede these rules whenever identical textual presentation is not required.

Integer Values. All integers shall be processed such that the resultant text string contains only the valid decimal digits (0–9) and a sign. A leading minus sign shall denote negative values. A leading plus sign shall be provided for positive values.

Real Values. All reals shall be processed so that the resulting string represents a valid approximation of the number in scientific notation. The decimal point shall appear immediately to the left of the most significant digit. The decimal point shall be preceded by a zero. A leading minus sign shall denote negative values. A leading plus sign shall be provided for positive values.

Logical Values. The logical value .TRUE. (indicated by a 1 in the file) shall be processed as if the string constant "4HTRUE" was to be displayed. The logical value .FALSE. (indicated by a 0 in the file) shall be processed as if the string constant "5HFALSE" was to be displayed.

4.3.14.1 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.

4.3.14.1.1 Directory Data
Entity Type Number: 312
Form Number: 0

4.3.14.1.2 Parameter Data

Index	Name	Type	Description
1	CBW	Real	Character box width
2	CBH	Real	Character box height
3	FC	Integer	Font characteristic (Default = 1)
		or	or
		Pointer	Pointer to Text Font Definition if negative
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
9	YS	Real	
10	ZS	Real	

Additional pointers as required (see Section 2.2.4.4.2).

4.3.14.2 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.

4.3.14.2.1 Directory Data
Entity Type Number: 312
Form Number: 1

4.3.14.2.2 Parameter Data

Index	Name	Type	Description
1	CBW	Real	Character box width
2	CBH	Real	Character box height
3	FC	Integer	Font characteristic (Default = 1)
		or	or
		Pointer	Pointer if negative
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 Section 2.2.4.4.2).

4.3.15 Color Definition Entity (Type 314). 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 G.

The preprocessor must select one of the Color Numbers (see Section 2.2.4.3.13) and place the value in Directory Entry Field 13 of the Color Definition Entity. This value should be the closest functional equivalent, or the most visually similar. The value will be used by postprocessors which cannot support the Color Definition Entity.

4.3.15.1 Directory Data

Entity Type Number: 314

4.3.15.2 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 Section 2.2.4.4.2).

4.3.16 Attribute Table Definition Entity (Type 322). The definition of this entity can be found in Appendix J (see Section J.8).

4.3.17 Attribute Table Instance Entity (Type 422). The definition of this entity can be found in Appendix J (see Section J.9).

Appendix A. Part File Examples

This appendix contains three sample parts encoded in the ASCII Form. These files are included 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 two-dimensional drawing of a mechanical part with dimensioning, and a three-dimensional part with two-dimensional drawing views defined.

Example file 1 is an integrated circuit (IC) cell. The IC application was selected because of the predominance of two-dimensional geometry used in electrical designs. The geometry used in the cell in Figure A1 consists of simple closed area, linear path entities and line widening property. The structure entities are nested subfigures using a Network Subfigure Definition Entity and Array Subfigure Instance Entities. A Connect Point Entity 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 contains (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 [HON80] with kind permission from the author.

Example file 2 is a two-dimensional drawing of a 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 A2 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 A3 shows the example drawing. In this way, model geometry and viewing parameters are logically separate. A three-dimensional 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.

APPENDIX A. ELECTRICAL PART EXAMPLE

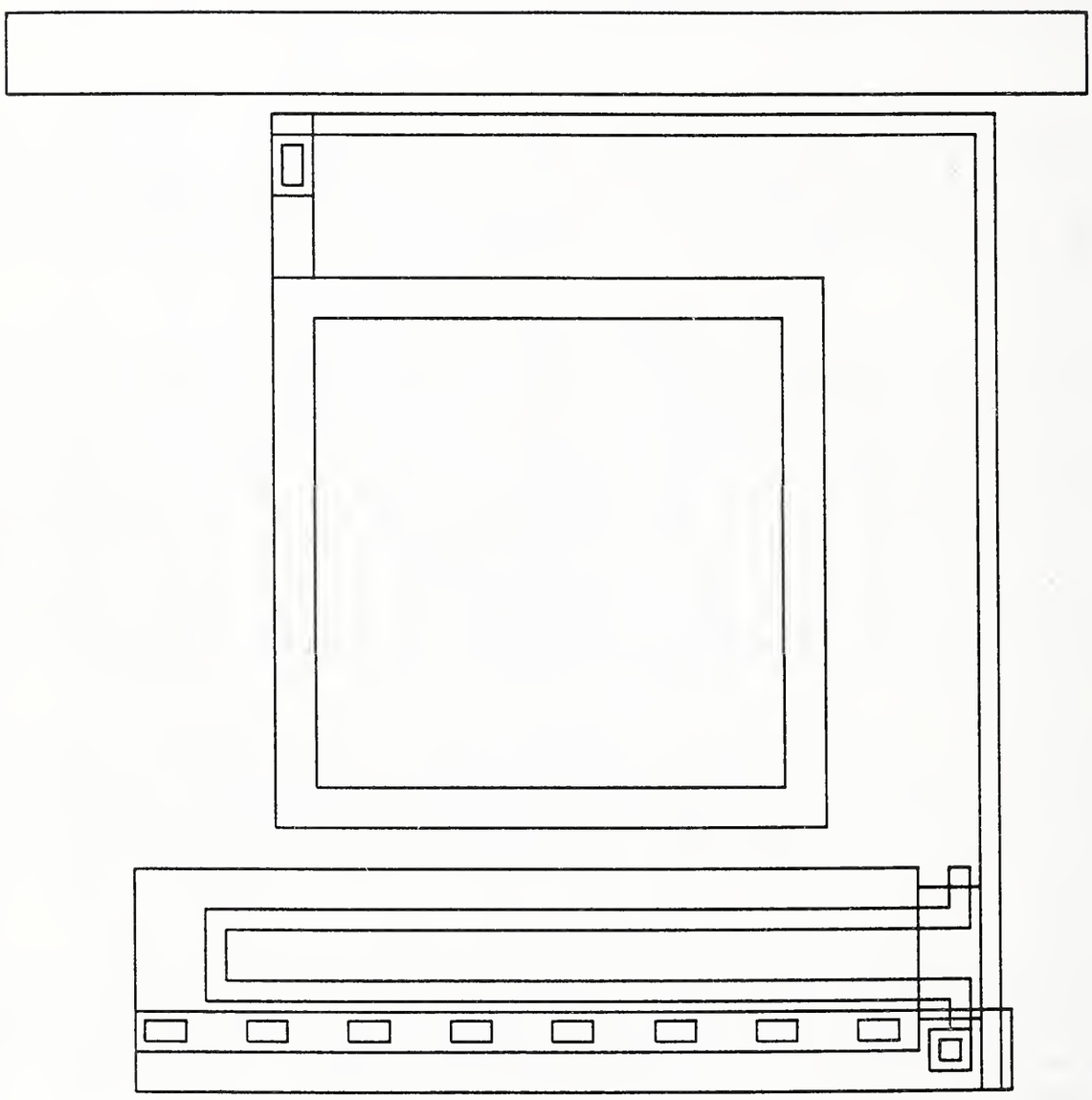


Figure A1. Electrical Part Example

Example 1 Electrical Part

```
INTEGRATED CIRCUIT SEMICUSTOM CELL (ONE PART OF A LIBRARY FILE)          S      1
USED IN APPENDIX A OF IGES VERSION 3.0 AND MODIFIED FOR VERSION 4.0      S      2
1H,,1H;,10H5MICRONLIB,5HPADIN,9HEXAMPLE 1,4HIGES,16,38,06,38,13,         G      1
10HIC.LIBRARY,1.0,9,2HUM,1,,13H880616.090000,0.01,265.0,9HBILL LOYE,     G      2
33HELECTRICAL APPLICATIONS COMMITTEE,5,0;                                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        00030200D     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
```

APPENDIX A. ELECTRICAL PART EXAMPLE

```
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,11,13,15,17,19,21,25,27,33,35,37,39,2,,,1,23;            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       2G      3D      42P     27                                      T       1
```

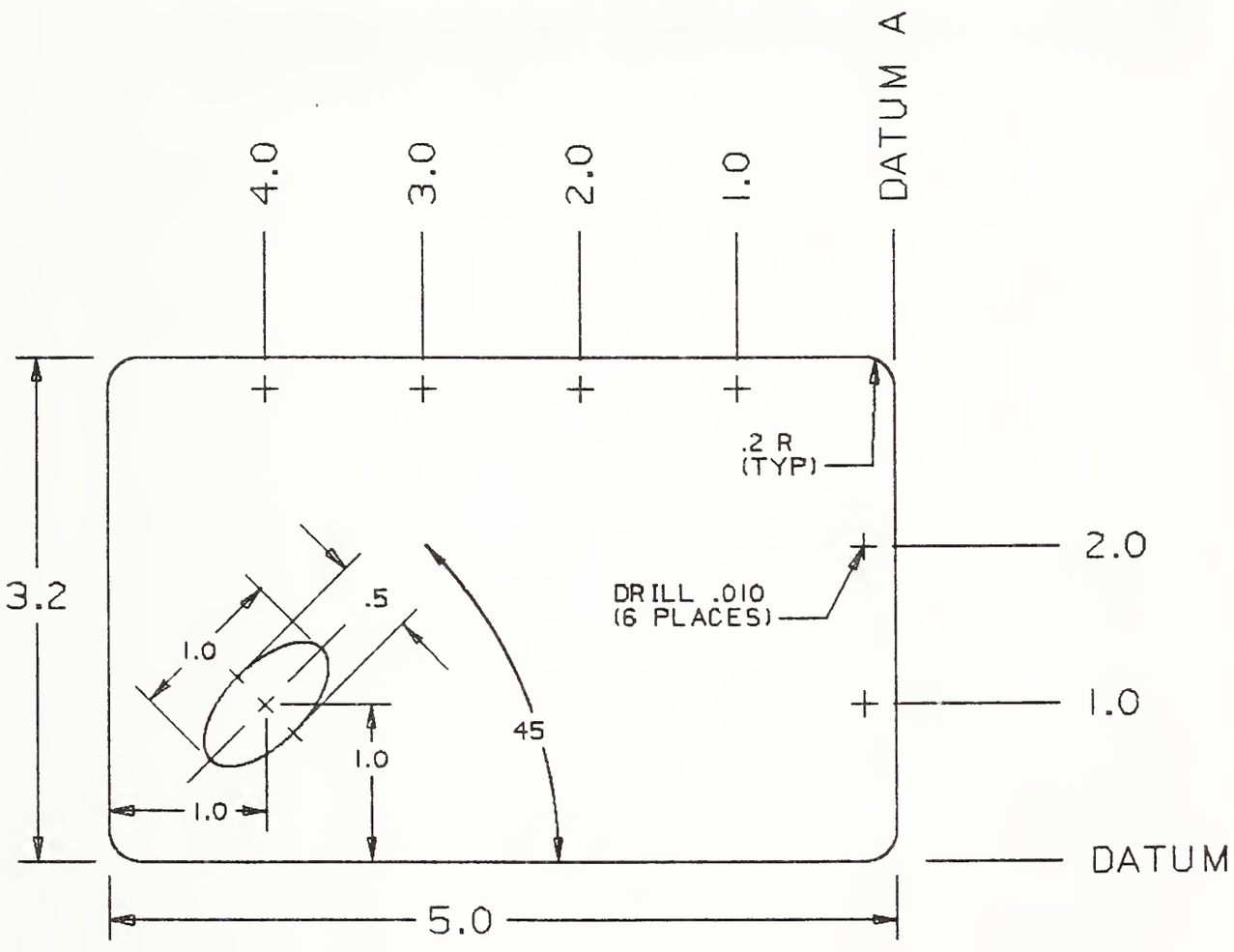


Figure A2. Mechanical Part Example

Example 2 Mechanical Part Example

```
PLATE.001    SAMPLE MECHANICAL PART WITH ANNOTATION                          S      1
  USED AT AUTOFACT - OCTOBER 1982 AND IN VERSION 3.0 APPENDIX A              S      2
     AUTHOR:  Dave Briggs, Boeing                                            S      3
ENTITY CONTENT:  POINT, LINE, ARC and CONIC                                  S      4
                 LINEAR, ANGULAR, RADIUS, POINT and ORDINATE DIMENSION       S      5
                 GENERAL NOTE, GENERAL LABEL                                 S      6
CHANGE LOG:                                                                  S      7
,,8HPANEL123,10HPANEL.IGES,4HEX 2,4HIGES,16,38,7,38,14,8HPANEL123,1.0,       G      1
1,4HINCH,1,0.028,13H880516.144004,0.0005,100.0,9HD. BRIGGS,6HBOEING,4,0;     G      2
     124       1       1       1       0       0       0       0       0D      1
     124       0       0       2       0                               0D      2
     212       3       1       1       5       0       0       0   10100D      3
     212       0       0       2       0                               0D      4
     214       5       1       1       5       0       0       0   10100D      5
     214       0       0       1       2                               0D      6
     210       6       1       1       5       0       0       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       0       0       0D     17
     100       0       0       1       0                               0D     18
     100      12       1       1       1       0       0       0       0D     19
     100       0       0       1       0                               0D     20
     100      13       1       1       1       0       0       0       0D     21
     100       0       0       1       0                               0D     22
     100      14       1       1       1       0       0       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	0	0	10100D	39
212	0	0	1	0				0D	40
214	24	1	1	4	0	0	0	10100D	41
214	0	0	1	2				0D	42
214	25	1	1	4	0	0	0	10100D	43
214	0	0	1	2				0D	44
106	26	1	1	4	0	0	0	10100D	45
106	0	0	1	40				0D	46
106	27	1	1	4	0	0	0	10100D	47
106	0	0	1	40				0D	48
216	28	1	1	4	0	0	0	100D	49
216	0	0	1	0				0D	50
212	29	1	1	4	0	0	0	10100D	51
212	0	0	1	0				0D	52
214	30	1	1	4	0	0	0	10100D	53
214	0	0	1	2				0D	54
214	31	1	1	4	0	0	0	10100D	55
214	0	0	1	2				0D	56
106	32	1	1	4	0	0	0	10100D	57
106	0	0	1	40				0D	58
106	33	1	1	4	0	0	0	10100D	59
106	0	0	1	40				0D	60
216	34	1	1	4	0	0	0	100D	61
216	0	0	1	0				0D	62
212	35	1	1	5	0	0	0	10100D	63
212	0	0	1	0				0D	64
106	36	1	1	5	0	0	0	10100D	65
106	0	0	1	40				0D	66
218	37	1	1	5	0	0	0	100D	67
218	0	0	1	0				0D	68
212	38	1	1	5	0	0	0	10100D	69
212	0	0	1	0				0D	70
106	39	1	1	5	0	0	0	10100D	71
106	0	0	1	40				0D	72
218	40	1	1	5	0	0	0	100D	73
218	0	0	1	0				0D	74
212	41	1	1	5	0	0	0	10100D	75
212	0	0	1	0				0D	76
106	42	1	1	5	0	0	0	10100D	77
106	0	0	1	40				0D	78
218	43	1	1	5	0	0	0	100D	79
218	0	0	1	0				0D	80
212	44	1	1	5	0	0	0	10100D	81
212	0	0	1	0				0D	82
106	45	1	1	5	0	0	0	10100D	83
106	0	0	1	40				0D	84
218	46	1	1	5	0	0	0	100D	85
218	0	0	1	0				0D	86
212	47	1	1	5	0	0	0	10100D	87
212	0	0	1	0				0D	88
106	48	1	1	5	0	0	0	10100D	89

106	0	0	1	40				0D	90
218	49	1	1	5	0	0	0	100D	91
218	0	0	1	0				0D	92
212	50	1	1	5	0	0	0	10100D	93
212	0	0	1	0				0D	94
106	51	1	1	5	0	0	0	10100D	95
106	0	0	1	40				0D	96
218	52	1	1	5	0	0	0	100D	97
218	0	0	1	0				0D	98
212	53	1	1	5	0	0	0	10100D	99
212	0	0	2	0				0D	100
106	55	1	1	5	0	0	0	10100D	101
106	0	0	1	40				0D	102
218	56	1	1	5	0	0	0	100D	103
218	0	0	1	0				0D	104
212	57	1	1	5	0	0	0	10100D	105
212	0	0	2	0				0D	106
106	59	1	1	5	0	0	0	10100D	107
106	0	0	1	40				0D	108
218	60	1	1	5	0	0	0	100D	109
218	0	0	1	0				0D	110
212	61	1	1	5	0	0	0	10100D	111
212	0	0	2	0				0D	112
214	63	1	1	5	0	0	0	10100D	113
214	0	0	1	2				0D	114
222	64	1	1	5	0	0	0	100D	115
222	0	0	1	0				0D	116
212	65	1	1	6	0	0	0	10100D	117
212	0	0	1	0				0D	118
214	66	1	1	6	0	0	0	10100D	119
214	0	0	1	2				0D	120
214	67	1	1	6	0	0	0	10100D	121
214	0	0	1	2				0D	122
106	68	1	1	6	0	0	0	1010100D	123
106	0	0	1	40				0D	124
106	69	1	1	6	0	0	0	10100D	125
106	0	0	1	40				0D	126
216	70	1	1	6	0	0	0	100D	127
216	0	0	1	0				0D	128
212	71	1	1	6	0	0	0	10100D	129
212	0	0	1	0				0D	130
214	72	1	1	6	0	0	0	10100D	131
214	0	0	1	2				0D	132
214	73	1	1	6	0	0	0	10100D	133
214	0	0	1	2				0D	134
106	74	1	1	6	0	0	0	10100D	135
106	0	0	1	40				0D	136
106	75	1	1	6	0	0	0	1010100D	137
106	0	0	1	40				0D	138
216	76	1	1	6	0	0	0	100D	139
216	0	0	1	0				0D	140

```
     212      77       1       1       6       0       0       0   10100D     141
     212       0       0       1       0                               0D     142
     214      78       1       1       6       0       0       0   10100D     143
     214       0       0       1       2                               0D     144
     214      79       1       1       6       0       0       0   10100D     145
     214       0       0       1       2                               0D     146
     106      80       1       1       6       0       0       0   10100D     147
     106       0       0       1      40                               0D     148
     106      81       1       1       6       0       0       0   10100D     149
     106       0       0       1      40                               0D     150
     216      82       1       1       6       0       0       0    100D     151
     216       0       0       1       0                               0D     152
     212      83       1       1       6       0       0       0   10100D     153
     212       0       0       1       0                               0D     154
     214      84       1       1       6       0       0       0   10100D     155
     214       0       0       1       2                               0D     156
     214      85       1       1       6       0       0       0   10100D     157
     214       0       0       1       2                               0D     158
     106      86       1       1       6       0       0       0   10100D     159
     106       0       0       1      40                               0D     160
     106      87       1       1       6       0       0       0   10100D     161
     106       0       0       1      40                               0D     162
     216      88       1       1       6       0       0       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       0       0   10100D     169
     212       0       0       1       0                               0D     170
     214      92       1       1       6       0       0       0   10100D     171
     214       0       0       1       2                               0D     172
     214      93       1       1       6       0       0       0   10100D     173
     214       0       0       1       2                               0D     174
     106      94       1       1       6       0       0       0 1010100D     175
     106       0       0       1      40                               0D     176
     106      95       1       1       6       0       0       0 1010100D     177
     106       0       0       1      40                               0D     178
     202      96       1       1       6       0       0       0    100D     179
     202       0       0       1       0                               0D     180
124,0.70710678,-0.70710678,0.0,1.0,0.70710678,0.70710678,0.0,                 1P       1
1.0,0.0,0.0,1.0,0.0,0,0;                                                      1P       2
212,2,10,0.98,0.1,1,1.571,0.0,0,0,3.21,1.656,0.0,10HDRILL .010,               3P       3
10,1.02,0.1,1,1.571,0.0,0,0,3.210,1.506,0.0,10H(6 PLACES),0,0;                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
116,4.000,3.000,0.0,0,0,0;                                              25P     15
116,3.000,3.000,0.0,0,0,0;                                              27P     16
116,2.000,3.000,0.0,0,0,0;                                              29P     17
116,1.000,3.000,0.0,0,0,0;                                              31P     18
104,4.000,0.0,16.000,0.0,0.0,-1.000,0.0,0.500,0.0,0.500,0.0,            33P     19
0,0;                                                                    33P     20
116,4.800,1.000,0.0,0,0,0;                                              35P     21
116,4.800,2.000,0.0,0,0,0;                                              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
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      7G      2D    180P     96                                        T      1
```

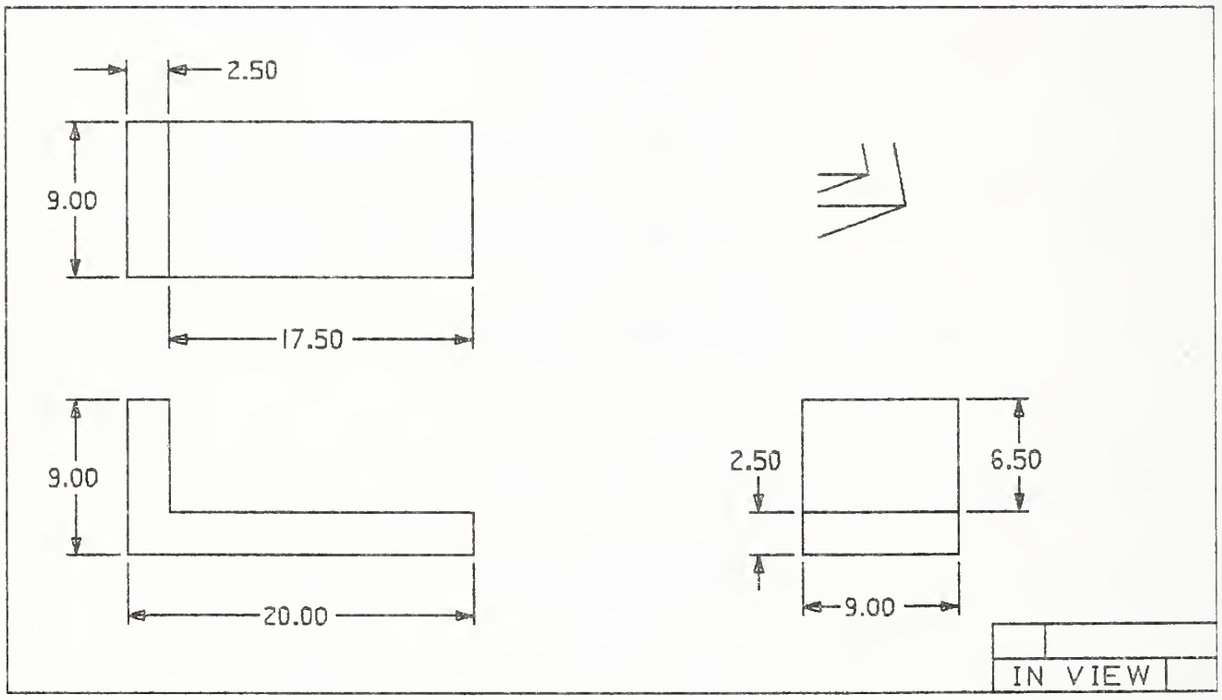


Figure A3. Drawing and View Example

Example 3 Drawing and View Example

```
Test file of model with VIEW (410) and DRAWING (404) entities                S      1
                                                                             S      2
This file demonstrates annotation attached to the VIEWS,                     S      3
i.e., the dimension entities are flagged as INDEPENDENT,                     S      4
and their DE field 6 points to a VIEW entity. The coordinates                S      5
of the dimensions are in MODEL space, and they have a                        S      6
transformation matrix which is the inverse of the VIEW matrix.               S      7
                                                                             S      8
This file was submitted by:                                                  S      9
                                                                             S     10
       Dennette A. Harrod, Jr.                                               S     11
       Computervision Corporation                                            S     12
       14 Crosby Drive / Bldg. 5-1                                           S     13
       Bedford, MA  01730                                                    S     14
       617-275-1800 ext.5172                                                 S     15
                                                                             S     16
1H,,1H;,8HVIEWDWG2,12HVIEWDWG2.IGS,13H<unspecified>,13H<unspecified>,32,G      1
38,6,38,15,8HVIEWDWG2,1.,1,2HIN,8,1.,13H870930.151125,1.E-06,71.,             G      2
13H<unspecified>,13H<unspecified>,4,0;                                       G      3
     406       1       1       0       0       0       0       0   10300D      1
     406       0       0       1      15                               0D      2
     124       2       1       0       0       0       0       0   10300D      3
     124       0       0       2       0                               0D      4
     108       4       1       0       0       0       0       0   10201D      5
     108       0       0       2       1                               0D      6
     108       6       1       0       0       0       0       0   10201D      7
     108       0       0       2       1                               0D      8
     108       8       1       0       0       0       0       0   10201D      9
     108       0       0       2       1                               0D     10
     108      10       1       0       0       0       0       0   10201D     11
     108       0       0       2       1                               0D     12
     410      12       1       0       0       0       3       0    201D     13
     410       0       0       1       0                               0D     14
     406      13       1       0       0       0       0       0   10300D     15
     406       0       0       1      15                               0D     16
     124      14       1       0       0       0       0       0   10300D     17
     124       0       0       1       0                               0D     18
     108      15       1       0       0       0       0       0   10201D     19
     108       0       0       1       1                               0D     20
     108      16       1       0       0       0       0       0   10201D     21
     108       0       0       1       1                               0D     22
     108      17       1       0       0       0       0       0   10201D     23
     108       0       0       1       1                               0D     24
     108      18       1       0       0       0       0       0   10201D     25
     108       0       0       1       1                               0D     26
     410      19       1       0       0       0      17       0    201D     27
     410       0       0       1       0                               0D     28
```

```
406     20      1       0       0       0       0       0     10300D    29
406      0      0       1      15                               0D      30
124     21      1       0       0       0       0       0     10300D    31
124      0      0       1       0                               0D      32
108     22      1       0       0       0       0       0     10201D    33
108      0      0       1       1                               0D      34
108     23      1       0       0       0       0       0     10201D    35
108      0      0       1       1                               0D      36
108     24      1       0       0       0       0       0     10201D    37
108      0      0       1       1                               0D      38
108     25      1       0       0       0       0       0     10201D    39
108      0      0       1       1                               0D      40
410     26      1       0       0       0      31       0      201D     41
410      0      0       1       0                               0D      42
406     27      1       0       0       0       0       0     10300D    43
406      0      0       1      15                               0D      44
108     28      1       0       0       0       0       0     10201D    45
108      0      0       1       1                               0D      46
108     29      1       0       0       0       0       0     10201D    47
108      0      0       1       1                               0D      48
108     30      1       0       0       0       0       0     10201D    49
108      0      0       1       1                               0D      50
108     31      1       0       0       0       0       0     10201D    51
108      0      0       1       1                               0D      52
410     32      1       0       0       0       0       0      201D     53
410      0      0       1       0                               0D      54
110     33      1       1       5       0       0       0     10100D    55
110      1      4       1       0                               0D      56
110     34      1       1       5       0       0       0     10100D    57
110      1      4       1       0                               0D      58
110     35      1       1       5       0       0       0     10100D    59
110      1      4       1       0                               0D      60
110     36      1       1       5       0       0       0     10100D    61
110      1      4       1       0                               0D      62
110     37      1       1       5       0       0       0     10100D    63
110      1      4       1       0                               0D      64
110     38      1       1       6       0       0       0     10100D    65
110      1      4       1       0                               0D      66
110     39      1       1       6       0       0       0     10100D    67
110      1      4       1       0                               0D      68
110     40      1       1       6       0       0       0     10100D    69
110      1      4       1       0                               0D      70
110     41      1       1       6       0       0       0     10100D    71
110      1      4       1       0                               0D      72
406     42      1       1       0       0       0       0     10300D    73
406      0      0       1      15                               0D      74
404     43      1       1       0       0       0       0      201D     75
404      0      0       2       0                               0D      76
110     45      1       1       5       0       0       0        0D    77
110      1      2       1       0                               0D      78
```

110	46	1	1	5	0	0	0	0D	79
110	1	2	1	0				0D	80
110	47	1	1	5	0	0	0	0D	81
110	1	2	1	0				0D	82
110	48	1	1	5	0	0	0	0D	83
110	1	2	1	0				0D	84
110	49	1	1	5	0	0	0	0D	85
110	1	2	1	0				0D	86
110	50	1	1	5	0	0	0	0D	87
110	1	2	1	0				0D	88
110	51	1	1	5	0	0	0	0D	89
110	1	2	1	0				0D	90
110	52	1	1	5	0	0	0	0D	91
110	1	2	1	0				0D	92
110	53	1	1	5	0	0	0	0D	93
110	1	2	1	0				0D	94
110	54	1	1	5	0	0	0	0D	95
110	1	2	1	0				0D	96
110	55	1	1	5	0	0	0	0D	97
110	1	2	1	0				0D	98
110	56	1	1	5	0	0	0	0D	99
110	1	2	1	0				0D	100
110	57	1	1	5	0	0	0	0D	101
110	1	2	1	0				0D	102
110	58	1	1	5	0	0	0	0D	103
110	1	2	1	0				0D	104
110	59	1	1	5	0	0	0	0D	105
110	1	2	1	0				0D	106
110	60	1	1	5	0	0	0	0D	107
110	1	2	1	0				0D	108
110	61	1	1	5	0	0	0	0D	109
110	1	2	1	0				0D	110
110	62	1	1	5	0	0	0	0D	111
110	1	2	1	0				0D	112
106	63	1	1	5	53	0	0	10101D	113
106	1	3	1	40				0D	114
214	64	1	1	5	53	0	0	10101D	115
214	1	3	1	2				0D	116
212	65	1	1	5	53	0	0	10101D	117
212	1	3	1	0				0D	118
214	66	1	1	5	53	0	0	10101D	119
214	1	3	1	2				0D	120
106	67	1	1	5	53	0	0	10101D	121
106	1	3	1	40				0D	122
216	68	1	1	5	53	0	0	100D	123
216	1	3	1	0				0D	124
106	69	1	1	5	27	0	0	10101D	125
106	1	3	1	40				0D	126
214	70	1	1	5	27	0	0	10101D	127
214	1	3	1	2				0D	128

212	71	1	1	5	27	0	0	10101D	129
212	1	3	1	0				0D	130
214	72	1	1	5	27	0	0	10101D	131
214	1	3	1	2				0D	132
106	73	1	1	5	27	0	0	10101D	133
106	1	3	1	40				0D	134
216	74	1	1	5	27	211	0	100D	135
216	1	3	1	0				0D	136
106	75	1	1	5	41	0	0	10101D	137
106	1	3	1	40				0D	138
214	76	1	1	5	41	0	0	10101D	139
214	1	3	1	2				0D	140
212	77	1	1	5	41	0	0	10101D	141
212	1	3	1	0				0D	142
214	78	1	1	5	41	0	0	10101D	143
214	1	3	1	2				0D	144
106	79	1	1	5	41	0	0	10101D	145
106	1	3	1	40				0D	146
216	80	1	1	5	41	213	0	100D	147
216	1	3	1	0				0D	148
106	81	1	1	5	41	0	0	10101D	149
106	1	3	1	40				0D	150
214	82	1	1	5	41	0	0	10101D	151
214	1	3	1	2				0D	152
212	83	1	1	5	41	0	0	10101D	153
212	1	3	1	0				0D	154
214	84	1	1	5	41	0	0	10101D	155
214	1	3	1	2				0D	156
106	85	1	1	5	41	0	0	10101D	157
106	1	3	1	40				0D	158
216	86	1	1	5	41	213	0	100D	159
216	1	3	1	0				0D	160
106	87	1	1	5	53	0	0	10101D	161
106	1	3	1	40				0D	162
214	88	1	1	5	53	0	0	10101D	163
214	1	3	1	2				0D	164
212	89	1	1	5	53	0	0	10101D	165
212	1	3	1	0				0D	166
214	90	1	1	5	53	0	0	10101D	167
214	1	3	1	2				0D	168
106	91	1	1	5	53	0	0	10101D	169
106	1	3	1	40				0D	170
216	92	1	1	5	53	0	0	100D	171
216	1	3	1	0				0D	172
106	93	1	1	5	41	0	0	10101D	173
106	1	3	1	40				0D	174
214	94	1	1	5	41	0	0	10101D	175
214	1	3	1	2				0D	176
212	95	1	1	5	41	0	0	10101D	177
212	1	3	1	0				0D	178

```
     214      96       1       1       5      41       0       0   10101D     179
     214       1       3       1       2                                0D     180
     106      97       1       1       5      41       0       0   10101D     181
     106       1       3       1      40                                0D     182
     216      98       1       1       5      41     213       0    100D     183
     216       1       3       1       0                                0D     184
     106      99       1       1       5      27       0       0   10101D     185
     106       1       3       1      40                                0D     186
     214     100       1       1       5      27       0       0   10101D     187
     214       1       3       1       2                                0D     188
     212     101       1       1       5      27       0       0   10101D     189
     212       1       3       1       0                                0D     190
     214     102       1       1       5      27       0       0   10101D     191
     214       1       3       1       2                                0D     192
     106     103       1       1       5      27       0       0   10101D     193
     106       1       3       1      40                                0D     194
     216     104       1       1       5      27     211       0    100D     195
     216       1       3       1       0                                0D     196
     106     105       1       1       5      27       0       0   10101D     197
     106       1       3       1      40                                0D     198
     214     106       1       1       5      27       0       0   10101D     199
     214       1       3       1       2                                0D     200
     212     107       1       1       5      27       0       0   10101D     201
     212       1       3       1       0                                0D     202
     214     108       1       1       5      27       0       0   10101D     203
     214       1       3       1       2                                0D     204
     106     109       1       1       5      27       0       0   10101D     205
     106       1       3       1      40                                0D     206
     216     110       1       1       5      27     211       0    100D     207
     216       1       3       1       0                                0D     208
     212     111       1       1       5       0       0       0   10101D     209
     212       1       3       2       0                                0D     210
     124     113       1       0       0       0       0       0   10300D     211
     124       0       0       1       0                                0D     212
     124     114       1       0       0       0       0       0   10300D     213
     124       0       0       1       0                                0D     214
     406     115       1       1       0       0       0       0   10300D     215
     406       0       0       1      16                                0D     216
406,1,7HCLIPPED,0,0;                                                     1P       1
124,-0.67499,-0.171,0.71774,1.00728,-0.24401,0.96977,0.00157,            3P       2
-0.821391,-0.69631,-0.17408,-0.69631,4.613057,0,0;                       3P       3
108,0.67499,0.171,-0.71774,5.0055,0,6.390347,2.455541,-0.379235,         5P       4
0.,0,0;                                                                  5P       5
108,-0.24401,0.96977,0.00157,3.55308,0,3.025032,4.420965,                7P       6
2.494731,0.,0,0;                                                         7P       7
108,-0.67499,-0.171,0.71774,2.99094,0,0.992841,1.088152,                 9P       8
5.360119,0.,0,0;                                                         9P       9
108,0.24401,-0.96977,-0.00157,1.9103,0,4.358156,-0.877273,              11P      10
2.486153,0.,0,0;                                                        11P      11
410,4,1.,5,7,9,11,0,0,0,1,1;                                            13P      12
```

```
406,1,5HRIGHT,0,0;                                                    15P     13
124,0.,0.,-1.,0.,0.,1.,0.,0.,1.,0.,0.,0.,0,0;                         17P     14
108,0.,0.,1.,36.,0,0.,0.,36.,0.,0,0;                                  19P     15
108,0.,1.,0.,24.59627,0,0.,24.59627,0.,0.,0,0;                        21P     16
108,0.,0.,-1.,36.,0,0.,0.,-36.,0.,0,0;                                23P     17
108,0.,-1.,0.,24.59627,0,0.,-24.59627,0.,0.,0,0;                      25P     18
410,3,1.,19,21,23,25,0,0,0,1,15;                                      27P     19
406,1,3HTOP,0,0;                                                      29P     20
124,1.,0.,0.,0.,0.,0.,-1.,0.,0.,1.,0.,0.,0,0;                         31P     21
108,-1.,0.,0.,36.,0,-36.,0.,0.,0.,0,0;                                33P     22
108,0.,0.,-1.,24.59627,0,0.,0.,-24.59627,0.,0,0;                      35P     23
108,1.,0.,0.,36.,0,36.,0.,0.,0.,0,0;                                  37P     24
108,0.,0.,1.,24.59627,0,0.,0.,24.59627,0.,0,0;                        39P     25
410,2,1.,33,35,37,39,0,0,0,1,29;                                      41P     26
406,1,5HFRONT,0,0;                                                    43P     27
108,-1.,0.,0.,36.,0,-36.,0.,0.,0.,0,0;                                45P     28
108,0.,1.,0.,24.59627,0,0.,24.59627,0.,0.,0,0;                        47P     29
108,1.,0.,0.,36.,0,36.,0.,0.,0.,0,0;                                  49P     30
108,0.,-1.,0.,24.59627,0,0.,-24.59627,0.,0.,0,0;                      51P     31
410,1,1.,45,47,49,51,0,0,0,1,43;                                      53P     32
110,67.,0.,0.,67.,2.,0.,0,0;                                          55P     33
110,60.,4.,0.,60.,2.,0.,0,0;                                          57P     34
110,57.,4.,0.,57.,0.,0.,0,0;                                          59P     35
110,70.,4.,0.,57.,4.,0.,0,0;                                          61P     36
110,70.,2.,0.,57.,2.,0.,0,0;                                          63P     37
110,0.,40.,0.,0.,0.,0.,0,0;                                           65P     38
110,70.,40.,0.,0.,40.,0.,0,0;                                         67P     39
110,70.,0.,0.,70.,40.,0.,0,0;                                         69P     40
110,0.,0.,0.,70.,0.,0.,0,0;                                           71P     41
406,1,7HDRAWING,0,0;                                                  73P     42
404,4,13,51.,29.,27,46.,8.,41,7.,24.,53,7.,8.,10,55,57,59,61,63,      75P     43
65,67,69,71,209,0,2,73,215;                                           75P     44
110,0.,9.,0.,0.,9.,-9.,0,0;                                           77P     45
110,2.5,9.,0.,2.5,9.,-9.,0,0;                                         79P     46
110,2.5,2.5,0.,2.5,2.5,-9.,0,0;                                       81P     47
110,20.,2.5,0.,20.,2.5,-9.,0,0;                                       83P     48
110,20.,0.,0.,20.,0.,-9.,0,0;                                         85P     49
110,0.,0.,0.,0.,0.,-9.,0,0;                                           87P     50
110,0.,9.,-9.,0.,0.,-9.,0,0;                                          89P     51
110,2.5,9.,-9.,0.,9.,-9.,0,0;                                         91P     52
110,2.5,2.5,-9.,2.5,9.,-9.,0,0;                                       93P     53
110,20.,2.5,-9.,2.5,2.5,-9.,0,0;                                      95P     54
110,20.,0.,-9.,20.,2.5,-9.,0,0;                                       97P     55
110,0.,0.,-9.,20.,0.,-9.,0,0;                                         99P     56
110,0.,9.,0.,0.,0.,0.,0,0;                                           101P     57
110,2.5,9.,0.,0.,9.,0.,0,0;                                          103P     58
110,2.5,2.5,0.,2.5,9.,0.,0,0;                                        105P     59
110,20.,2.5,0.,2.5,2.5,0.,0,0;                                       107P     60
110,20.,0.,0.,20.,2.5,0.,0,0;                                        109P     61
110,0.,0.,0.,20.,0.,0.,0,0;                                          111P     62
```

```
106,1,3,0.,0.,0.,0.,-0.5,0.,-4.,0,0;                                113P    63
214,1,1.,0.5,0.,0.,-3.5,7.625,-3.5,0,0;                             115P    64
212,1,5,3.75,1.,1,1.57079632679489,0.,0,0,8.,-4.,0.,5H20.00,0,0;    117P    65
214,1,1.,0.5,0.,20.,-3.5,12.125,-3.5,0,0;                           119P    66
106,1,3,0.,20.,0.,20.,-0.5,20.,-4.,0,0;                             121P    67
216,117,115,119,113,121,0,0;                                        123P    68
106,1,3,0.,0.,0.,0.,-0.5,0.,-3.5,0,0;                               125P    69
214,1,1.,0.5,0.,0.,-3.,2.125,-3.,0,0;                               127P    70
212,1,4,3.,1.,1,1.57079632679489,0.,0,0,2.5,-3.5,0.,4H9.00,0,0;     129P    71
214,1,1.,0.5,0.,9.,-3.,5.875,-3.,0,0;                               131P    72
106,1,3,0.,9.,0.,9.,-0.5,9.,-3.5,0,0;                               133P    73
216,129,127,131,125,133;0,0;                                        135P    74
106,1,3,0.,2.5,0.,2.5,-0.5,2.5,-4.,0,0;                             137P    75
214,1,1.,0.5,0.,2.5,-3.5,8.625,-3.5,0,0;                            139P    76
212,1,5,3.75,1.,1,1.57079632679489,0.,0,0,9.,-4.,0.,5H17.50,0,0;    141P    77
214,1,1.,0.5,0.,20.,-3.5,13.125,-3.5,0,0;                           143P    78
106,1,3,0.,20.,0.,20.,-0.5,20.,-4.,0,0;                             145P    79
216,141,139,143,137,145,0,0;                                        147P    80
106,1,3,0.,0.,9.,0.,9.5,0.,12.5,0,0;                                149P    81
214,1,1.,0.5,0.,0.,12.,-3.,12.,0,0;                                 151P    82
212,1,4,3.,1.,1,1.57079632679489,0.,0,0,6.,11.5,0.,4H2.50,0,0;      153P    83
214,1,1.,0.5,0.,2.5,12.,5.5,12.,0,0;                                155P    84
106,1,3,0.,2.5,9.,2.5,9.5,2.5,12.5,0,0;                             157P    85
216,153,151,155,149,157,0,0;                                        159P    86
106,1,3,0.,0.,0.,-0.5,0.,-3.5,0.,0,0;                               161P    87
214,1,1.,0.5,0.,-3.,0.,-3.,3.5,0,0;                                 163P    88
212,1,4,3.,1.,1,1.57079632679489,0.,0,0,-4.5,4.,0.,4H9.00,0,0;      165P    89
214,1,1.,0.5,0.,-3.,9.,-3.,5.5,0,0;                                 167P    90
106,1,3,0.,0.,9.,-0.5,9.,-3.5,9.,0,0;                               169P    91
216,165,163,167,161,169,0,0;                                        171P    92
106,1,3,0.,0.,0.,-0.5,0.,-3.5,0.,0,0;                               173P    93
214,1,1.,0.5,0.,-3.,0.,-3.,3.5,0,0;                                 175P    94
212,1,4,3.,1.,1,1.57079632679489,0.,0,0,-4.5,4.,0.,4H9.00,0,0;      177P    95
214,1,1.,0.5,0.,-3.,9.,-3.,5.5,0,0;                                 179P    96
106,1,3,0.,0.,9.,-0.5,9.,-3.5,9.,0,0;                               181P    97
216,177,175,179,173,181,0,0;                                        183P    98
106,1,3,0.,0.,0.,-0.5,0.,-3.,0.,0,0;                                185P    99
214,1,1.,0.5,0.,-2.5,0.,-2.5,-2.,0,0;                               187P   100
212,1,4,3.,1.,1,1.57079632679489,0.,0,0,-4.,5.,0.,4H2.50,0,0;       189P   101
214,1,1.,0.5,0.,-2.5,2.5,-2.5,4.5,0,0;                              191P   102
106,1,3,0.,0.,2.5,-0.5,2.5,-3.,2.5,0,0;                             193P   103
216,189,187,191,185,193,0,0;                                        195P   104
106,1,3,0.,9.,2.5,9.5,2.5,13.,2.5,0,0;                              197P   105
214,1,1.,0.5,0.,12.5,2.5,12.5,4.5,0,0;                              199P   106
212,1,4,3.,1.,1,1.57079632679489,0.,0,0,11.,5.,0.,4H6.50,0,0;       201P   107
214,1,1.,0.5,0.,12.5,9.,12.5,6.5,0,0;                               203P   108
106,1,3,0.,9.,9.,9.5,9.,13.,9.,0,0;                                 205P   109
216,201,199,203,197,205,0,0;                                        207P   110
212,1,7,8.25,1.,1,1.57079632679489,0.,0,0,58.,0.5,0.,7HIN VIEW,     209P   111
0,0;                                                                209P   112
```

```
124,0.,0.,1.,0.,0.,1.,0.,0.,-1.,0.,0.,0.,0,0;                    211P    113
124,1.,0.,0.,0.,0.,0.,1.,0.,0.,-1.,0.,0.,0,0;                    213P    114
406,2,70.,40.,0,0;                                               215P    115
S      16G       3D     216P     115                                T       1
```

Appendix B. Electrical/Electronic Product Representation

B.1 Introduction

B.1.1 Purpose. The purpose of this appendix is to provide implementors and users with a roadmap to the representation of electrical/electronic product designs using this Specification. The topics of discussion will include (but are not limited to) design, engineering, manufacturing, testing, and inspection.

B.1.2 Assumptions. The reader should have a basic understanding of electrical/electronic product design using CAD/CAM and Computer-Aided Engineering (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 on the subject.

B.2 Connectivity

B.2.1 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), and
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).

B.2.2 Representation. 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 Definition and Instance represent electrical components which participate in signals. A number of Property Entities will also be used.

B.2.2.1 Network Subfigure Construction. A component is constructed using the Network Subfigure Definition Entity. The entities representing the component geometry are referenced in the same manner as 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.

B.2.2.2 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 association needed 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.

B.2.2.3 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 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 (see Section 2.2.4.4.2).

B.2.2.4 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.

B.2.2.5 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 would represent the physical connection. The two associativities would be related by the pointers provided in the Flow Associativity.

B.2.2.6 Figures. The following figures illustrate certain aspects of the above discussions. Figure B1 illustrates the basic entity relationships. Figure B2 and Table B1 illustrate the usage of the Text Display Template. Figure B3 illustrates an actual implementation. Figure B4 shows an example of logical and physical signals and their relationships in the same file.

Table B1. Text Display Template Values for Sample Schematic

TEMPLATE	TLEFT	TRIGHT	TTOP	TBOT	
WIDTH	.10	.10	.10	.10	
HEIGHT	.13	.13	.13	.13	
SLANT ANGLE	(default)	(default)	(default)	(default)	
ROT'N. ANGLE	0.	0.	0.	0.	
MIRROR FLAG	0	0	0	0	
VRT./HORIZ.	0	0	0	0	
DE FORM NO.	1	1	1	1	
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

B.3 Electrical Entity Descriptions

The following entities (in entity number order) are the subset of 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 Pield.

102 Composite Curve 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.

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 printed circuit (PC) board or integrated circuit (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 Entity 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 three-dimensional 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 Entity is used to display constant text. Design notes would require a General Note Entity.

APPENDIX B.3 ELECTRICAL ENTITY DESCRIPTIONS

302 Associativity Definition Entity

When the originating system provides a relationship not included among the predefined associativities, this entity is required. Possible uses are to relate subfigures to entities in other databases (*i.e.*, 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 (*i.e.*, a pin number).

320 Network Subfigure Definition Entity

For printed circuit 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 are identified by form numbers (*i.e.*, Form 18: Flow). The user defined associativities are defined by 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.

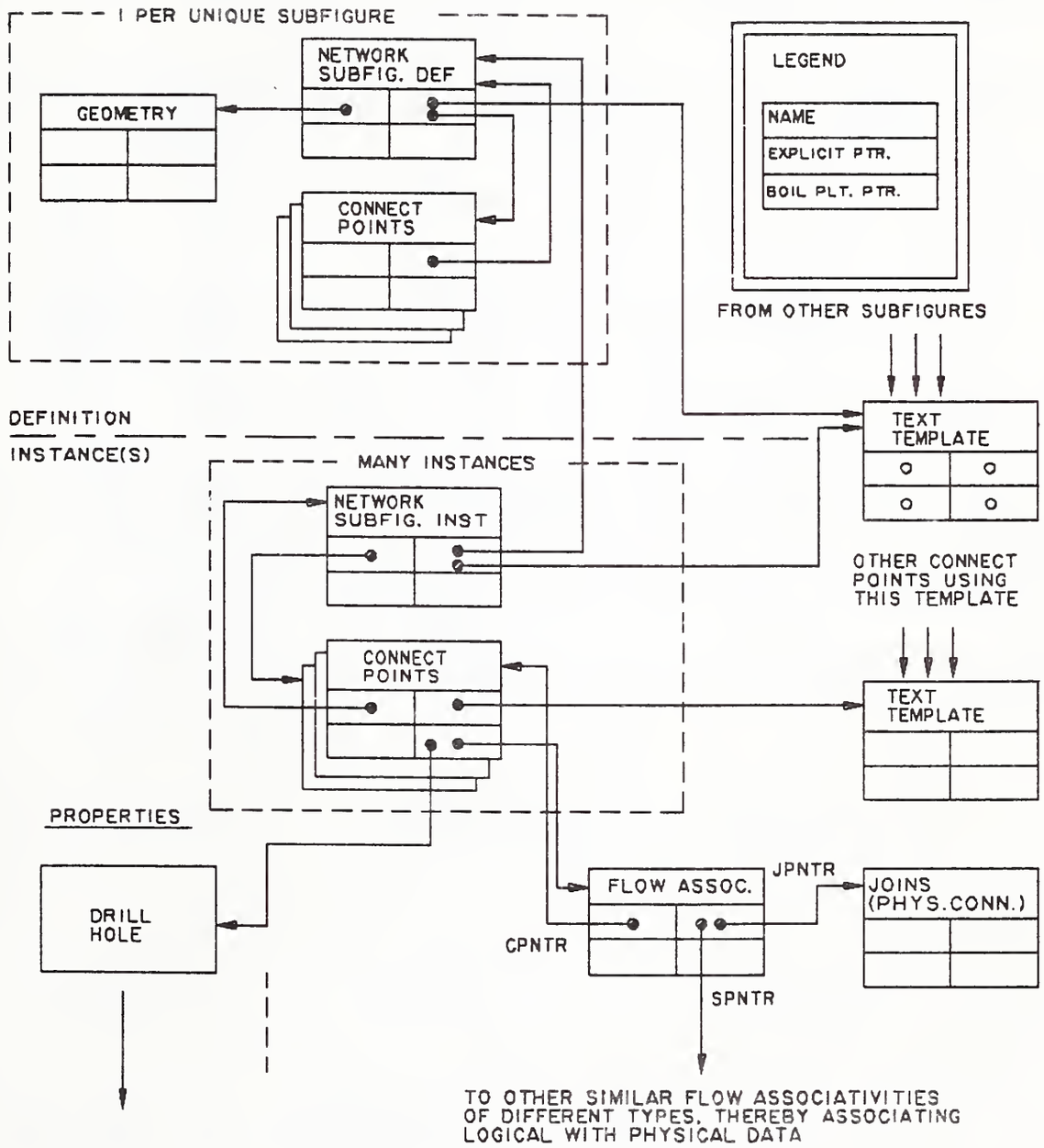


Figure B1. General Pointer and Entity Model

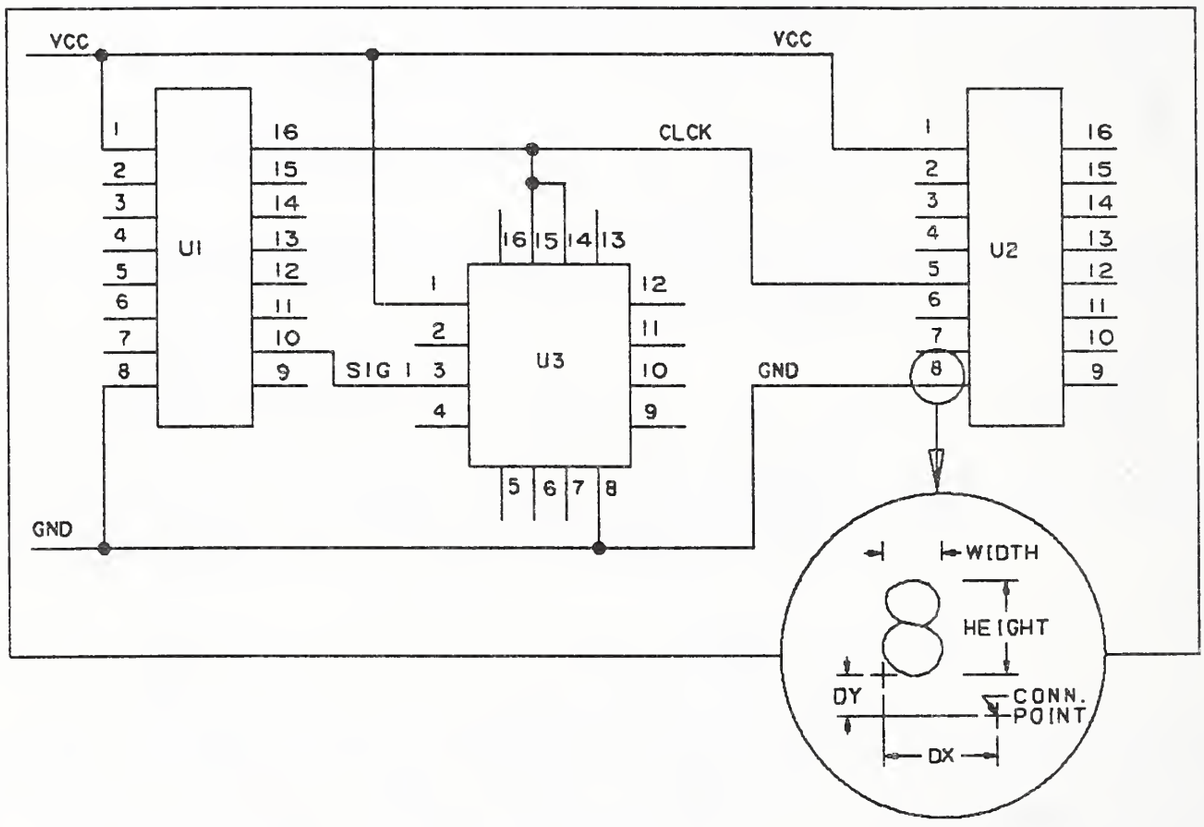


Figure B2. Sample Schematic

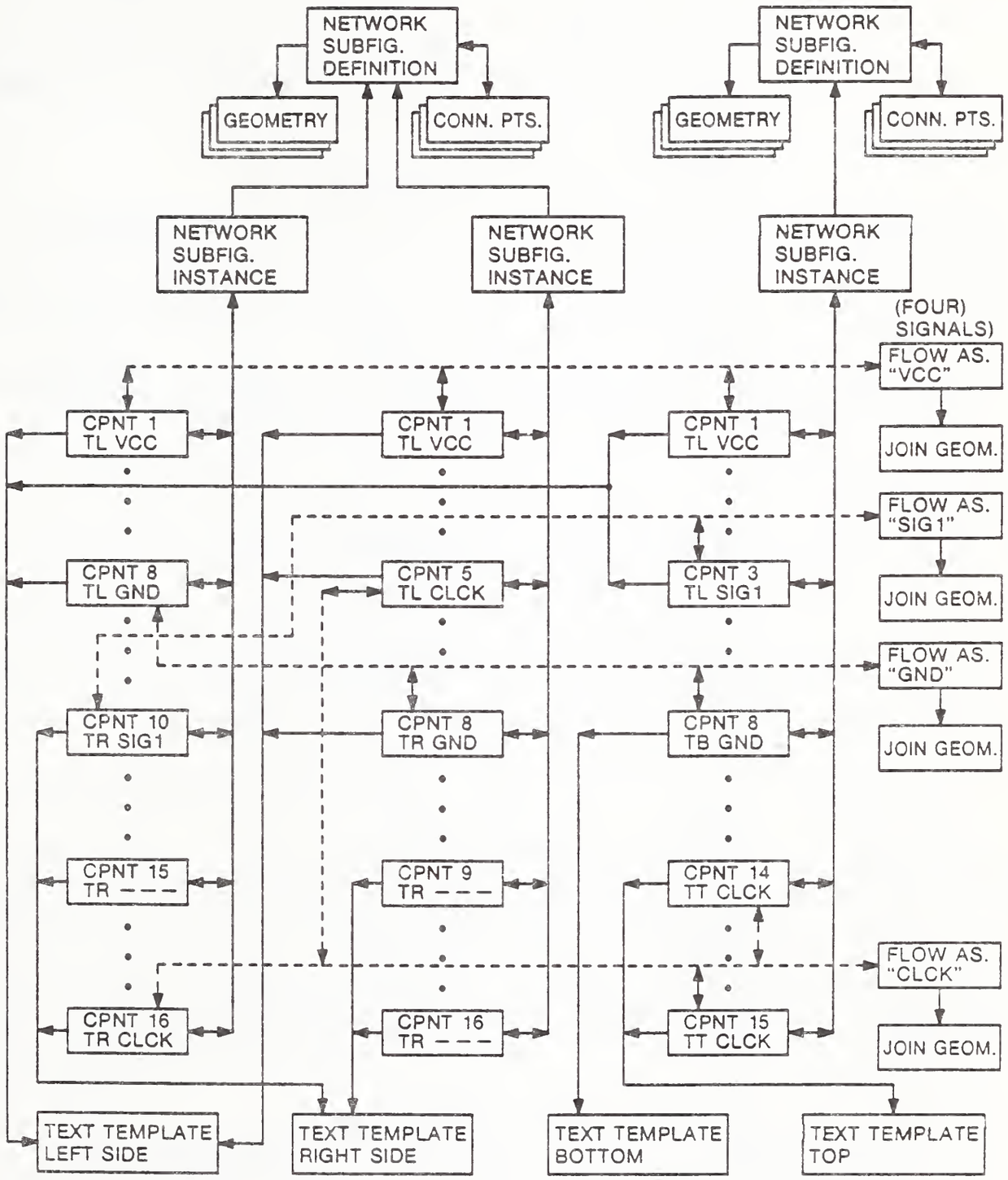


Figure B3. Entity Relations Chart for Sample Schematic

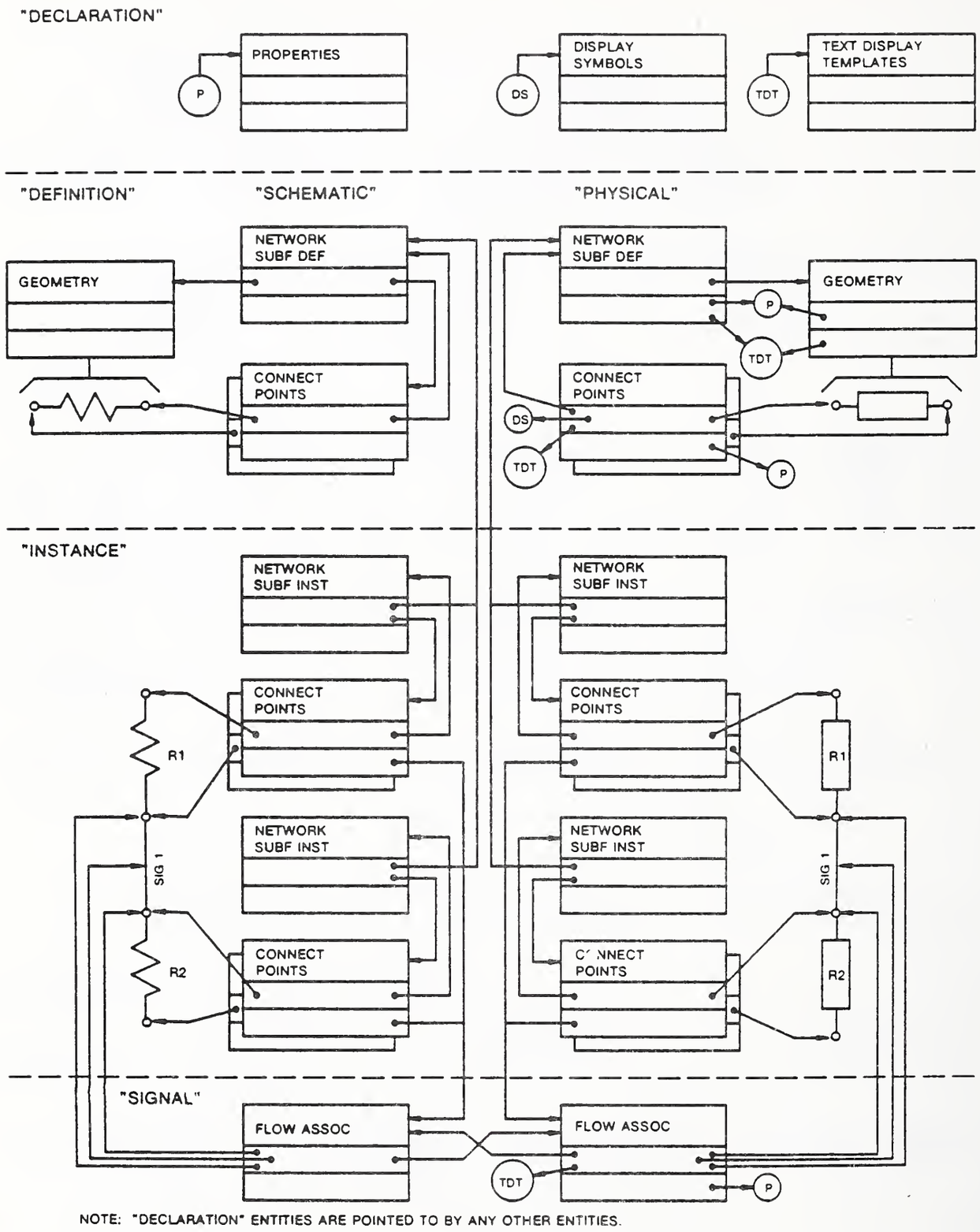


Figure B4. Schematic/Physical Diagram for Sample Schematic

Appendix C. Plant Flowsheet Representation

C.1 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, specification, 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, tanks, 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 a line that is too large for a single sheet and is continued on another flowsheet.

C.2 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 definitions) 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 (i.e., points to their Connect Point Entities) and then lists the in-line components. Properties can be attached to any entity and Nominal Size and Flow Line Specification Property Entities would be attached to pipe line path entities.

C.3 Plant Flowsheet Example

A simple flowsheet is shown in Figures C1 and C2. The way the flowsheet would appear on paper is shown in Figure C1. The recommended assignment of connection points is shown in Figure C2.

A view of the flowsheet showing the conceptual entities to be used to describe the flowsheet is given in Figure C3. 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 lines 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 C4. 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 Associativity 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.

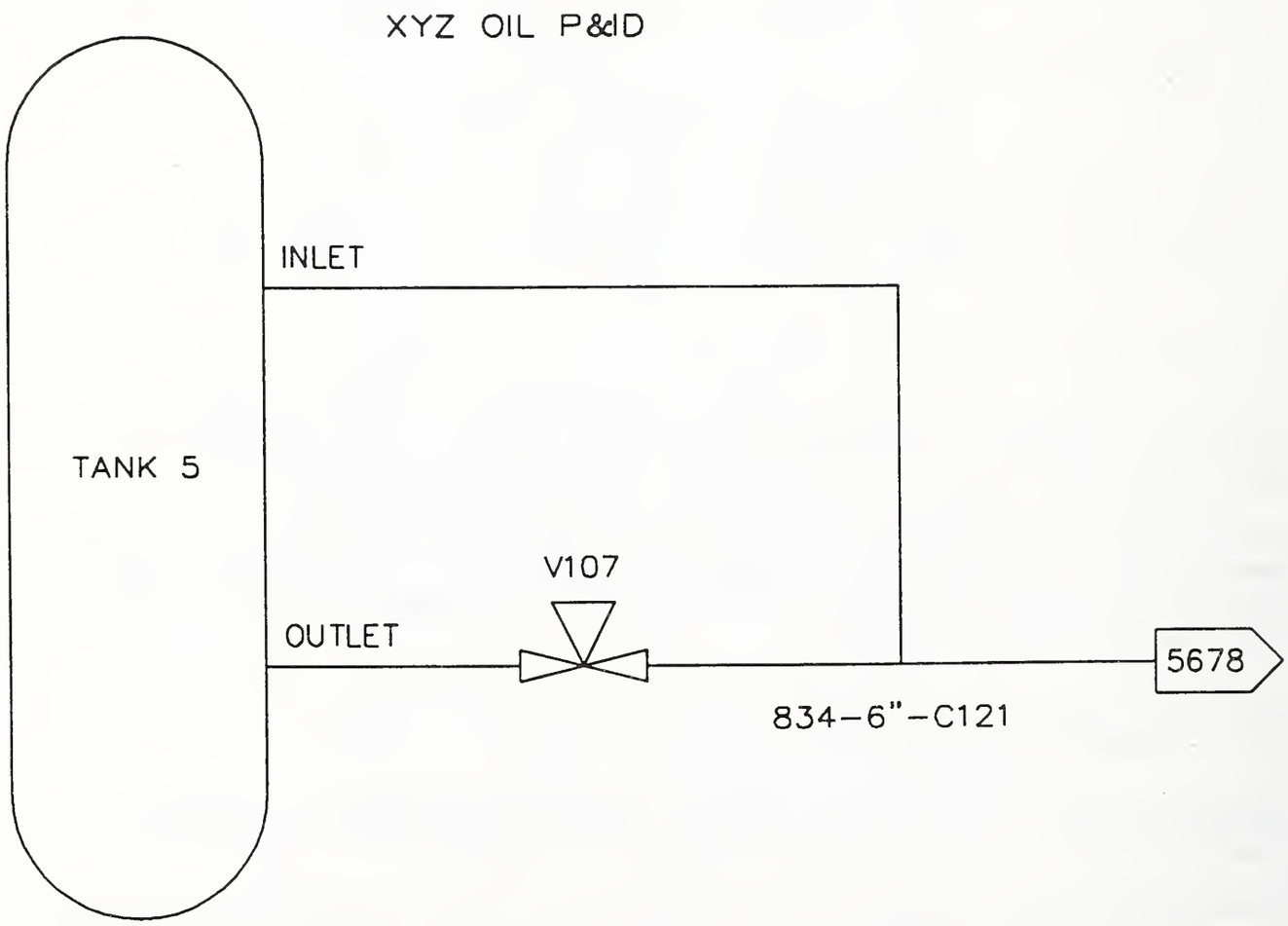


Figure C1. Flowsheet Appearance

The entities which encode this flowsheet are given in Table C1. The entity numbers given in the table are the sequence number in field 10 of the Directory Entry for each entity in the example file shown in Table C2.

C.4 Network Subfigure Definitions

The definitions of the tank and valve subfigures are shown in Figures C5 and C6. Each subfigure consists of the Network Subfigure Definition Entity, geometry entities, and Connect Point Entities. The tank is defined by entities D1, D5, D7, D9, D11, D13, and D15. The Network Subfigure Definition Entity D1 points to the geometry entities (D9, D11, D13, and D15) and to the Connect Point Entities D5 and D7. This is merely a definition which is "instanced" when actually needed.

The valve has a similar form. The Network Subfigure Definition Entity D3 points to geometric entities D21 through D33 and to Connect Point Entities D17 and D19.

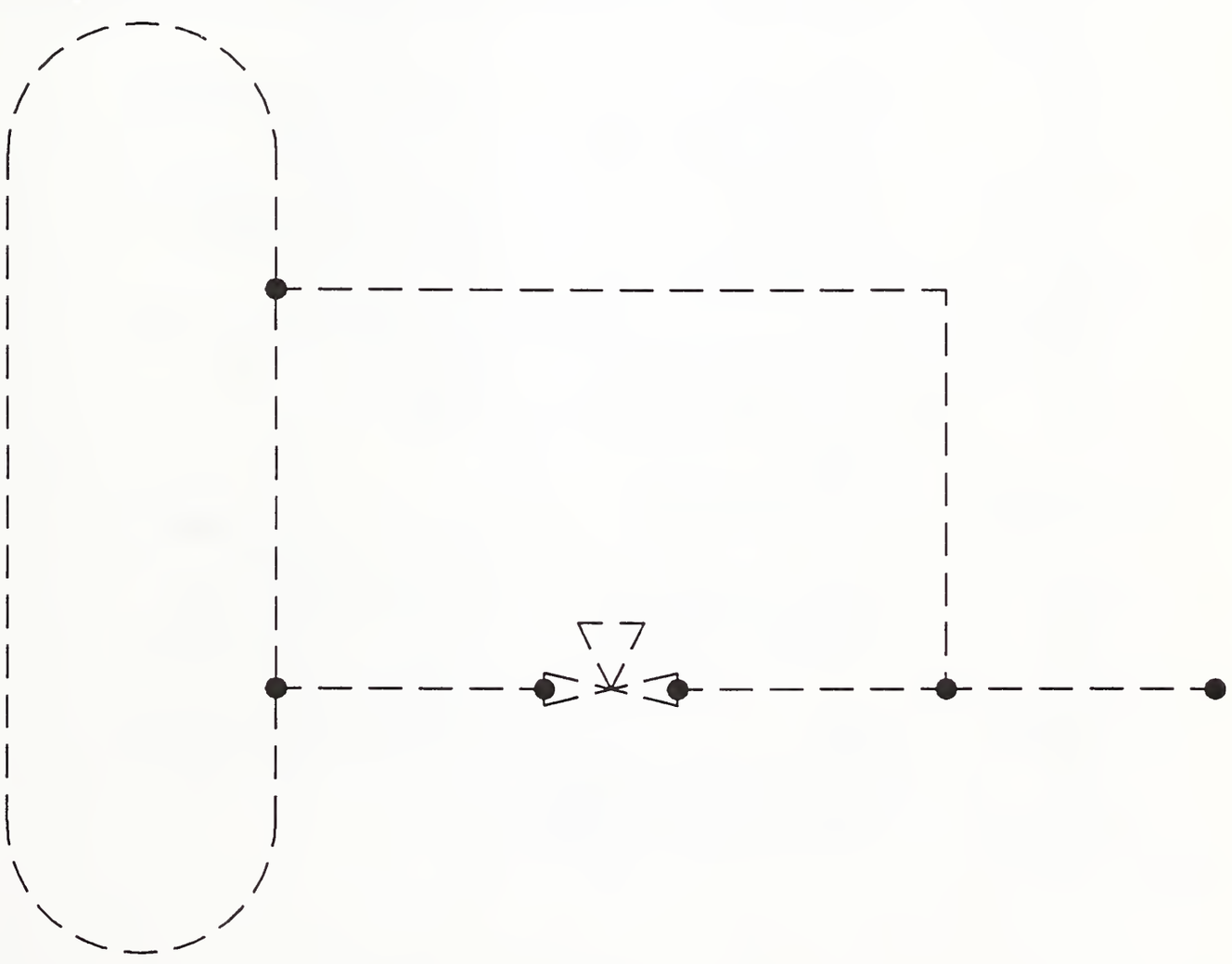


Figure C2. Required Connection Points

Table C1. Entities in Flowsheet Example

Entity	Type	Remarks
D1	Network Subfigure Def	Tank
D3	Network Subfigure Def	Valve
D5, D7	Connect Point	Tank nozzle
D9, D13	Line	Tank geometry
D11, D15	Circular Arc	Tank geometry
D17, D19	Connect Point	Valve nozzle
D21-D33	Line	Valve geometry
D35	General Note	Miscellaneous drawing text
D37	Network Subfigure Inst	Instance of column
D39	Network Subfigure Inst	Instance of valve
D41, D43	Connect Point	Instance of tank nozzles
D45, D47	Connect Point	Instance of valve nozzles
D49	Connect Point	Branch point
D51	Connect Point	Flow path end
D53-D61	Line	Geometry of connecting paths
D63, D65, D67	Nominal Size Property	12 IPS, *etc.*
D69	Part Number Property	"98021"
D71	Flow Line Specification Property	"A100"
D73, D75	Flow Associativity	
D77, D79	Text Display Template	Display the subfigure name reference designators
D81	Text Display Template	Displays the nozzle names
D83	Text Display Template	Displays the flow path name
D85	Text Display Template	Displays the connect point function designator
D87-D91	Line	Continuation symbol geometry
D97	External Reference	Continuation of flow path
D99	External Reference File List	
D101	External Reference File Index	

C.5 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 C7 shows the subfigures as instanced in the flowsheet with instance entities D37 ("tank") and D39 ("valve"). Note that the subfigures may be scaled when they are instanced. Independent x and y axis scaling is often used by CAD vendors to "fit" their symbols to the drawing. Network Subfigure Instance Entity D37 points to Subfigure Definition Entity D1 and to the new duplicated Connect Point Entities D41 and D43. Similarly, Network Subfigure Instance Entity D39 points to Subfigure Definition Entity D3 and to the duplicated Connect Point Entitiess D45 and D47.

C.6 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 Associativity Instance Entities. The geometric entities are lines D53 through D61.

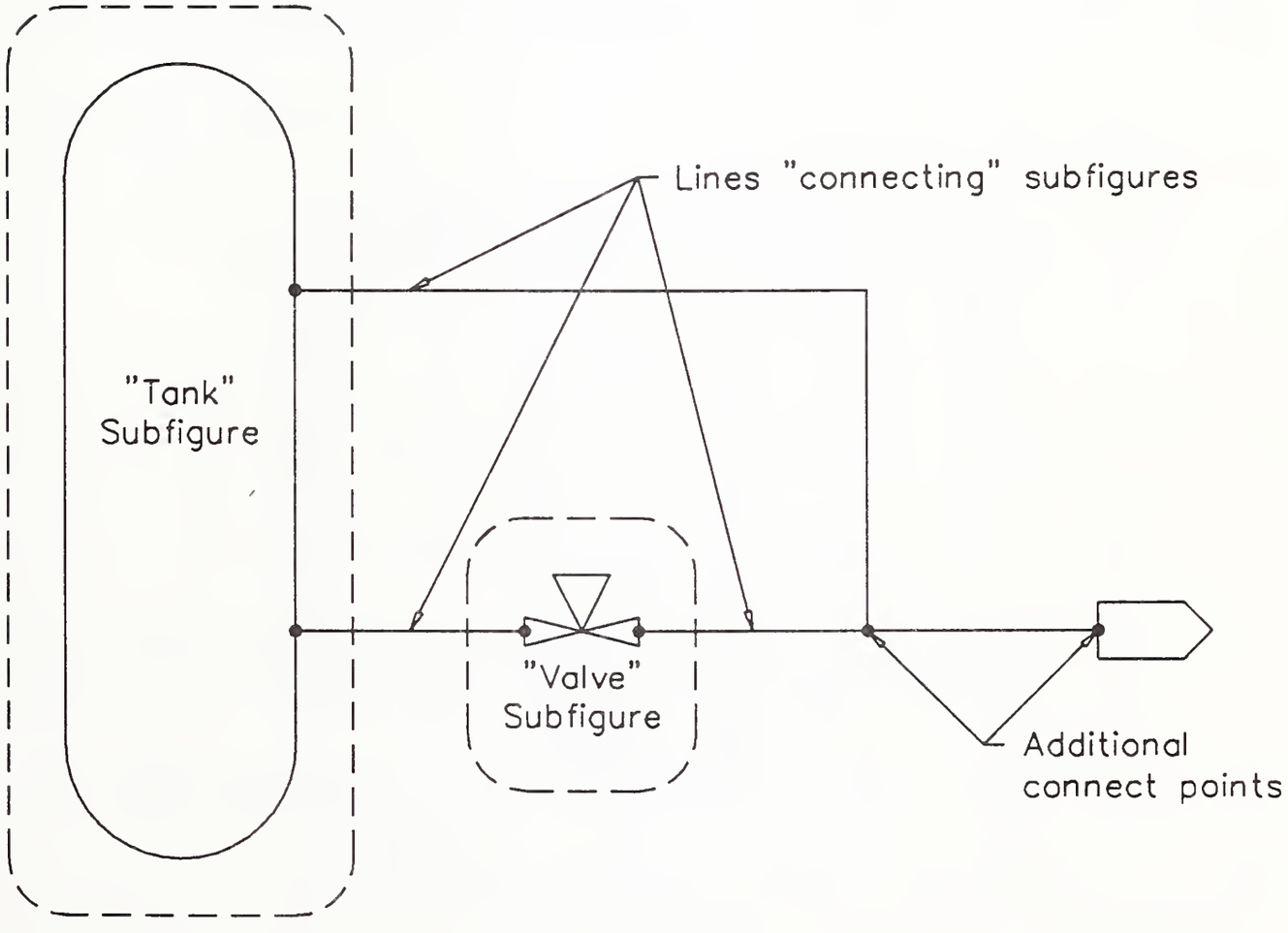


Figure C3. Conceptual Entities in the Flowsheet Example

C.7 Attributes

Three specific properties (attributes) are provided for flowsheets. These are part number, nominal size, and (flow) line specification. Figure C8 shows the properties used for the flowsheet example. These are implemented by three forms of the Property Entity. The entities in the example are D63 through D71.

C.8 Flow Paths

Figure C9 shows an assignment of flow paths. Two Flow Associativity Entities are required, D73 and D75. Each Flow 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 Associativity Entity D73 lists Flow Associativity Entity D75 as a continuation. These points are bi-directional, so Flow Associativity Entity D75 contains a list of the entities (D73) which point to it. The Flow Associativity is also the entity to which the Line Specification Property Entity is attached. Note that a rather arbitrary

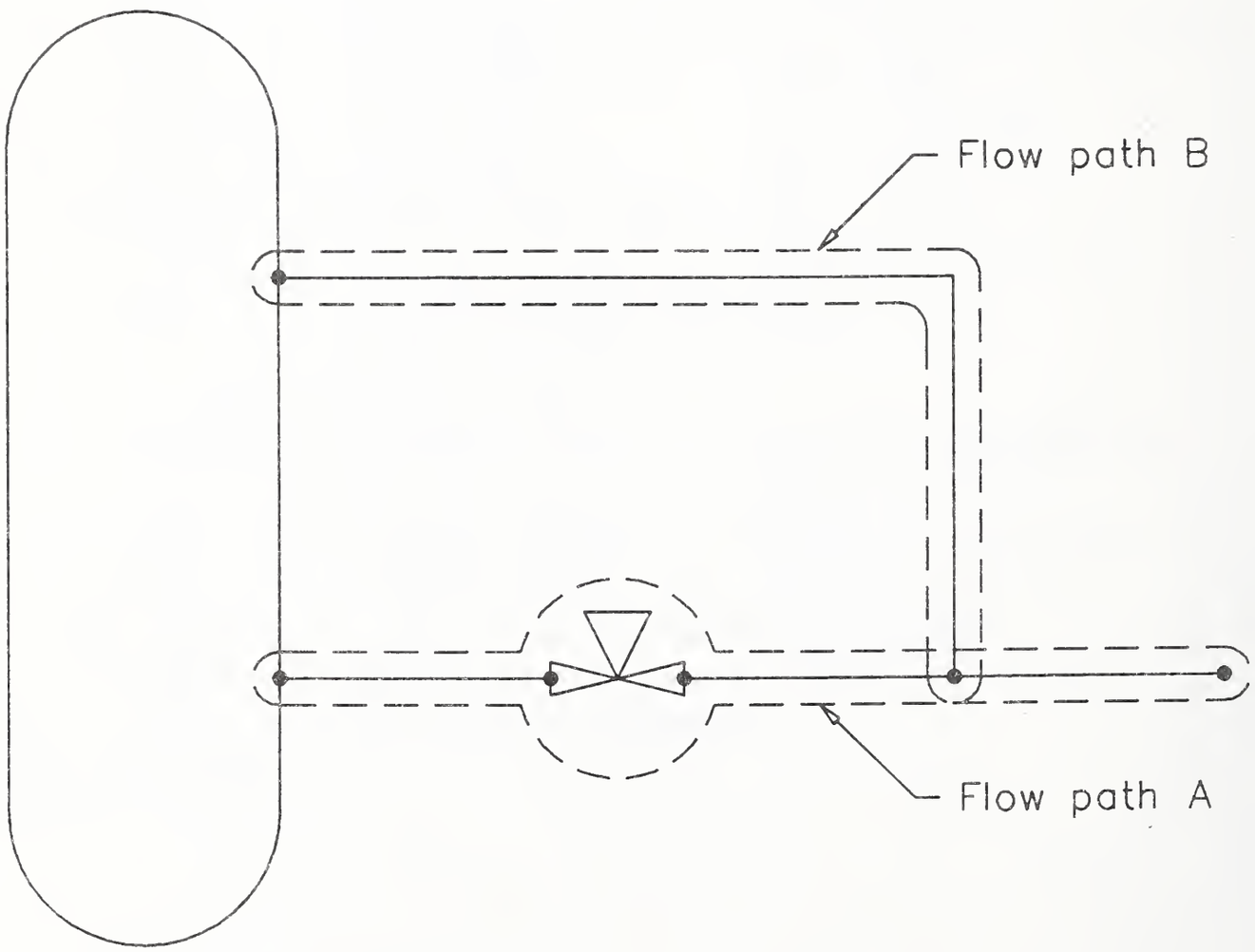


Figure C4. Logical Flow Paths for Streams

flow path assignment was made by the process engineer. Other assignments are possible and could be encoded.

C.9 Text Display Templates

Text templates are used to control display of the text which is actually contained in the Subfigure Instance Entities and Connect Point Entities. There are two uses of Text Display 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 as well as the name of the primary flow path. Absolute Text Display template Entities D77, D79, and D85 are used for this. The tank nozzles (Connect Point Entities D41 and D43) 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 Display Template Entity D81. In addition, the Connect Point Entity for the flow path end, D51, points to an Incremental Text Display Template Entity D83, for control of the display of its name.

C.10 External References

Flow path D73 continues offpage, i.e., the path is logically continued on another flowsheet. Three pieces of information must be matched to make this continuation:

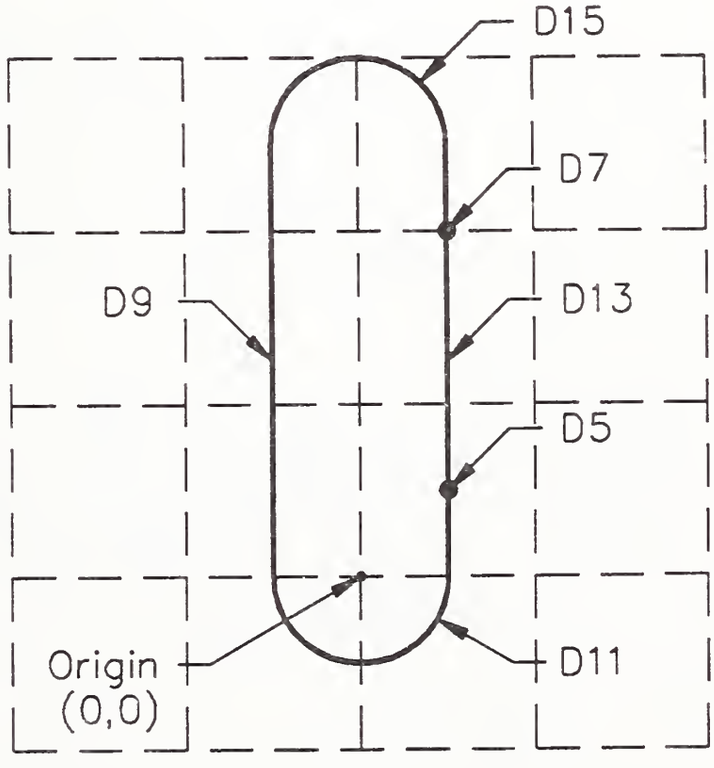


Figure C5. Definition of Tank Subfigure Entity D1

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 Entity of the path upon which continuation is made, and to the file in which the drawing is contained. The Function Flag parameter of the Connect Point Entity is used to carry the drawing name and a Text Display Template Entity is used to display it. File names are recommended to be constructed from the drawing name by the preprocessor.
2. The position of the Connect Point Entity within a Flow Associativity Entity 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 C10).

Flow path D73 ends at External Reference Entity D97. 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 preprocessor using 'I' for incoming and '834-6"-C121', the line number, size,

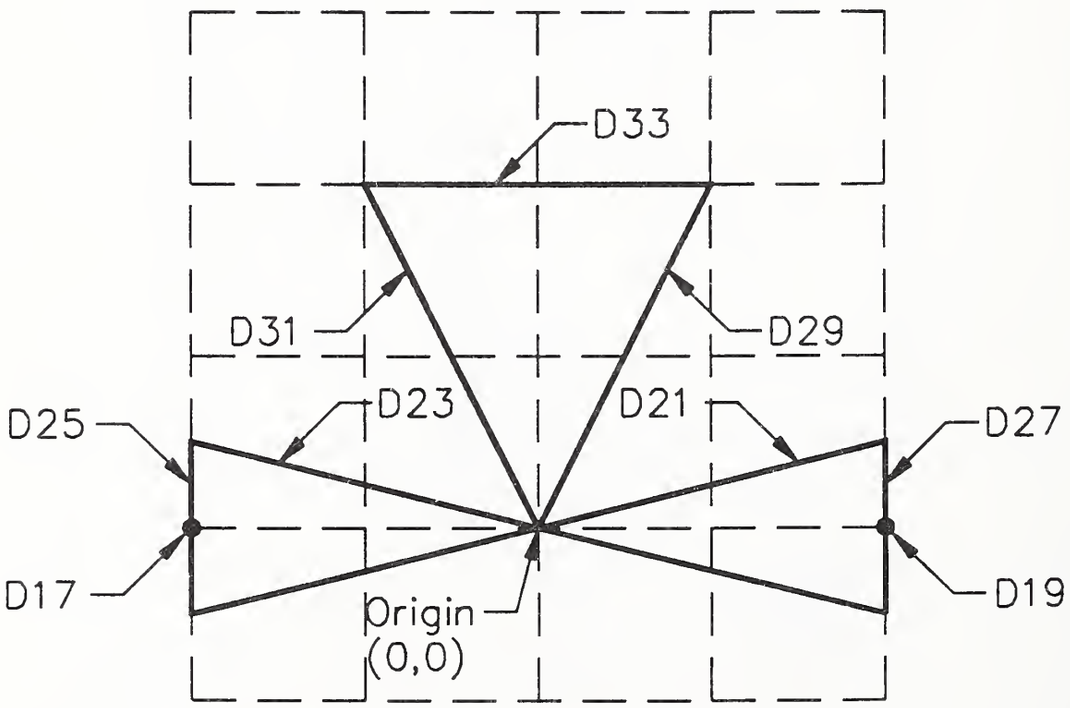


Figure C6. Definition of Valve Subfigure Entity D3

and specification. 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 D99 carries this file name in the referencing file also.

In the external flowsheet, an External Reference File Index Entity will list the symbolic name 'I834-6"-C121', and reciprocal entities will also exist to allow the external flowsheet to continue back into the referencing flowsheet. The flow path begins with an External Reference Entity 'O834-6"-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 containing the symbolic name 'I834-6"-C121', a pointer to connect point D51.

C.11 Encoded Flowsheet

The encoded flowsheet is shown in Table C2.

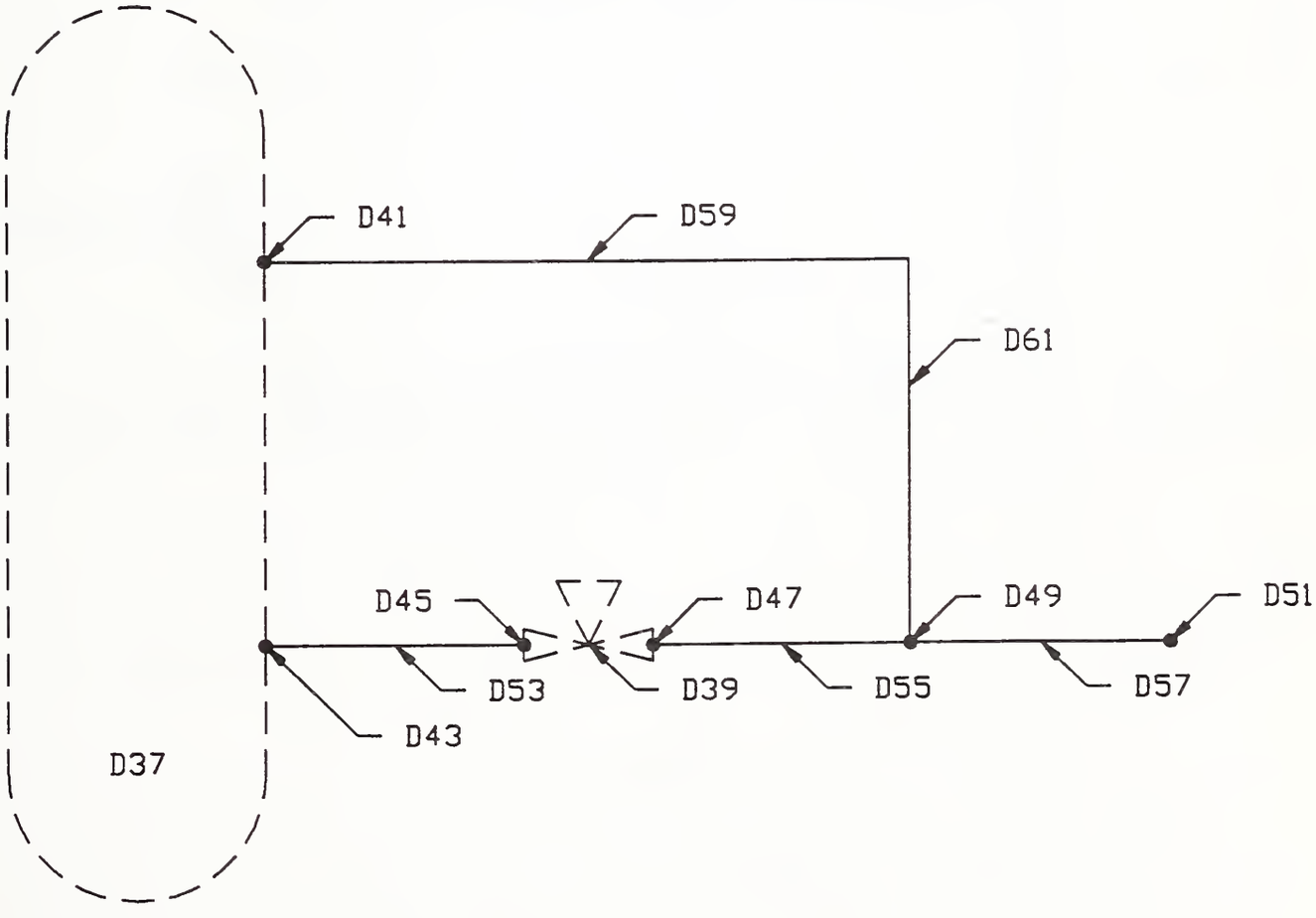


Figure C7. Geometry Entities

APPENDIX C.11 ENCODED FLOWSHEET

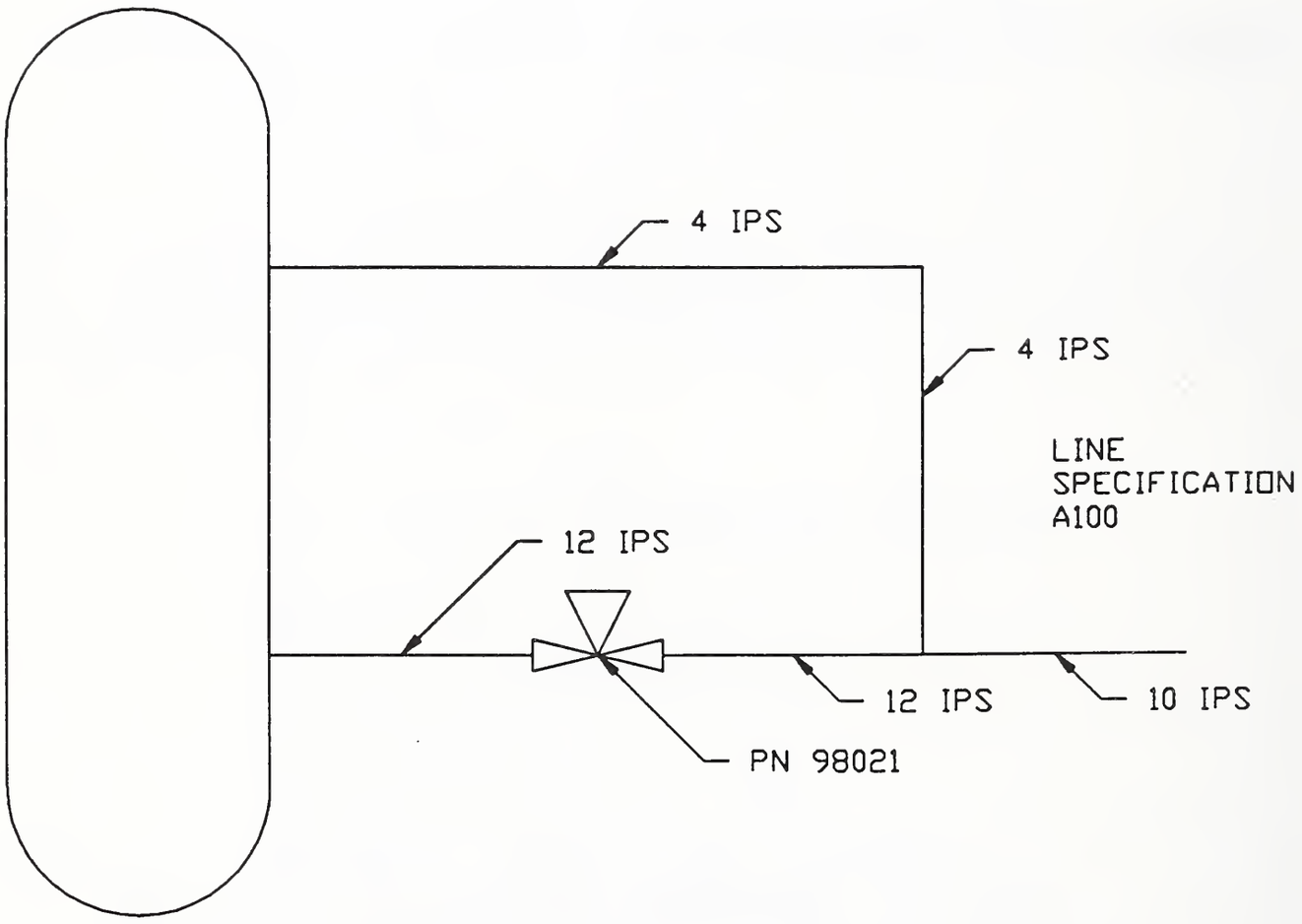


Figure C8. Attributes

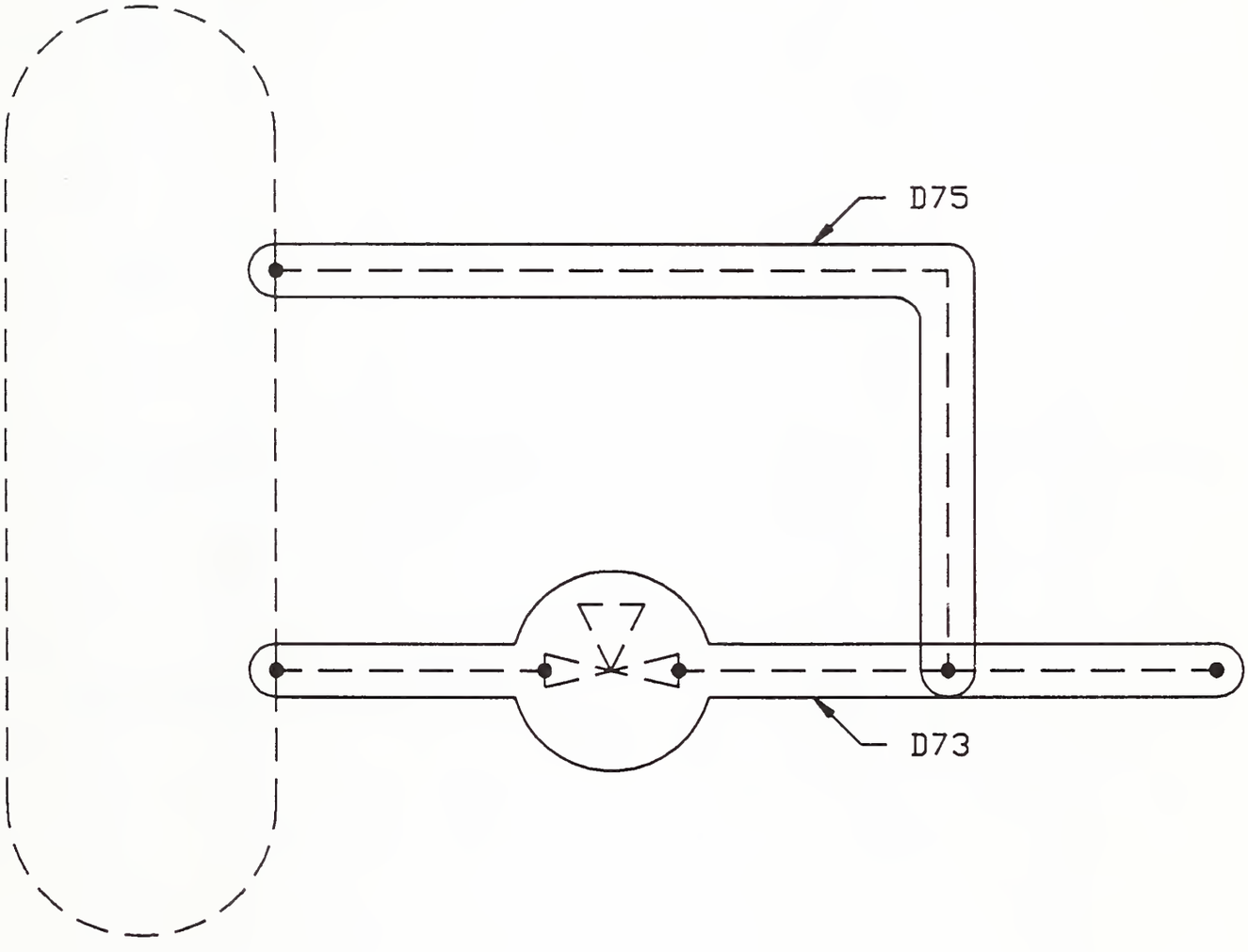


Figure C9. Flow Paths

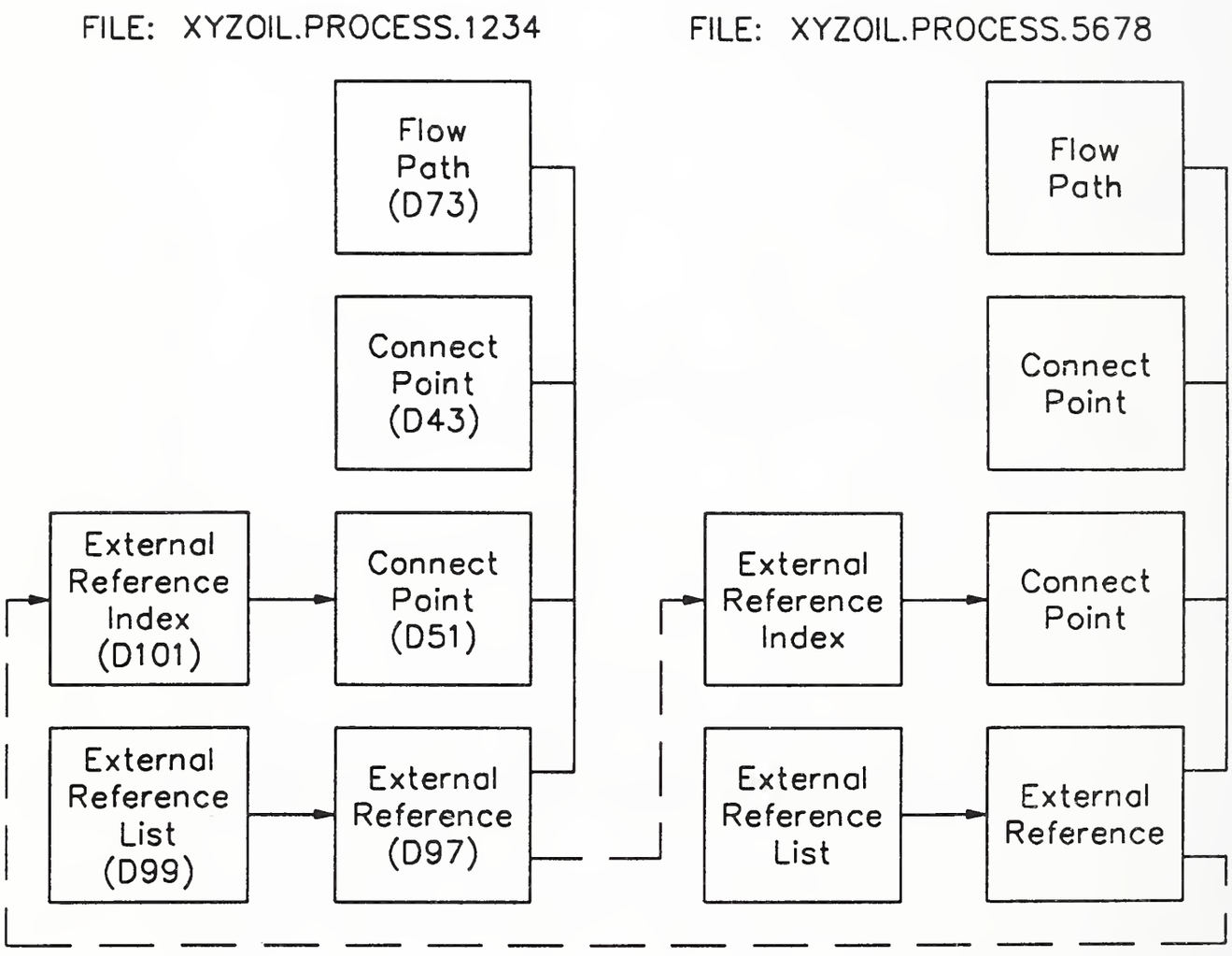


Figure C10. External References for the Flowsheet Example

Table C2. Encoded Flowsheet Example for Piping and Instrumentation Diagram

```
FLOWSHEET EXAMPLE -- PIPING & INSTRUMENTATION DIAGRAM                    S      1
1H,,1H;,12HXYZ OIL P&ID,19HXYZOIL.PROCESS.1234,10HHYBRID 1.0,1H2,32,38, G      1
7,308,15,12HXYZ OIL P&ID,0.100000E+01,1,4HINCH,3,0.06,13H880308.235959, G      2
0.0001,10000.,19HP.W.ROURKE/K.A.REED,3HORG,4,0;                          G      3
     320       1       1       0       1       0       0        00010200D      1
     320       0       1       1       0                TANK DEF       1D      2
     320       2       1       0       1       0       0        00010200D      3
     320       0       1       1       0                VALVEDEF       3D      4
     132       3       1       1       1       0       0        00020400D      5
     132       2       1       1       0                  TANK-CP      5D      6
     132       4       1       1       1       0       0        00020400D      7
     132       2       1       1       0                  TANK-CP      7D      8
     110       5       1       1       1       0       0        00010000D      9
     110       2       1       1       0                TANKLINE       9D     10
     100       6       1       1       1       0       0        00010000D     11
     100       2       1       1       0                TANK ARC      11D     12
     110       7       1       1       1       0       0        00010000D     13
     110       2       1       1       0                TANKLINE      13D     14
     100       8       1       1       1       0       0        00010000D     15
     100       2       1       1       0                TANK-ARC      15D     16
     132       9       1       1       1       0       0        00020400D     17
     132       2       1       1       0                VALVE-CP      17D     18
     132      10       1       1       1       0       0        00020400D     19
     132       2       1       1       0                VALVE-CP      19D     20
     110      11       1       1       1       0       0        00010000D     21
     110       2       1       1       0                VALVLINE      21D     22
     110      12       1       1       1       0       0        00010000D     23
     110       2       1       1       0                VALVLINE      23D     24
     110      13       1       1       1       0       0        00010000D     25
     110       2       1       1       0                VALVLINE      25D     26
     110      14       1       1       1       0       0        00010000D     27
     110       2       1       1       0                VALVLINE      27D     28
     110      15       1       1       1       0       0        00010000D     29
     110       2       1       1       0                VALVLINE      29D     30
     110      16       1       1       1       0       0        00010000D     31
     110       2       1       1       0                VALVLINE      31D     32
     110      17       1       1       1       0       0        00010000D     33
     110       2       1       1       0                VALVLINE      33D     34
     212      18       1       1       1       0       0        00000100D     35
     212       2       1       2       0                GEN-NOTE      35D     36
     420      20       1       1       1       0       0        00000300D     37
     420       2       1       1       0                TANKINST      37D     38
     420      21       1       1       1       0       0        00000300D     39
     420       2       1       1       0                VALVINST      39D     40
     132      22       1       1       1       0       0        00020400D     41
     132       2       1       1       0                CONECTPT      41D     42
     132      23       1       1       1       0       0        00020400D     43
     132       2       1       1       0                CONECTPT      43D     44
```

132	24	1	1	1	0	0	00020400D	45
132	2	1	1	0		CONECTPT	45D	46
132	25	1	1	1	0	0	00020400D	47
132	2	1	1	0		CONECTPT	47D	48
132	26	1	1	1	0	0	00020400D	49
132	2	1	1	0		CONECTPT	49D	50
132	27	1	1	1	0	0	00020400D	51
132	2	1	1	0		CONECTPT	51D	52
110	28	1	1	1	0	0	00000000D	53
110	2	1	1	0		PATHLINE	53D	54
110	29	1	1	1	0	0	00000000D	55
110	2	1	1	0		PATHLINE	55D	56
110	30	1	1	1	0	0	00000000D	57
110	2	1	1	0		PATHLINE	57D	58
110	31	1	1	1	0	0	00000000D	59
110	2	1	1	0		PATHLINE	59D	60
110	32	1	1	1	0	0	00000000D	61
110	2	1	1	0		PATHLINE	61D	62
406	33	1	1	1	0	0	01000000D	63
406	2	1	1	13		SIZEPROP	63D	64
406	34	1	1	1	0	0	01000000D	65
406	2	1	1	13		SIZEPROP	65D	66
406	35	1	1	1	0	0	01000000D	67
406	2	1	1	13		SIZEPROP	67D	68
406	36	1	1	1	0	0	01000000D	69
406	2	1	1	9		PN-PROP	69D	70
406	37	1	1	1	0	0	01000000D	71
406	2	1	1	14		SPECPROP	71D	72
402	38	1	1	1	0	0	01000300D	73
402	2	1	2	18		FLOWASSO	73D	74
402	40	1	1	1	0	0	01000300D	75
402	2	1	1	18		FLOWASSO	75D	76
312	41	1	1	1	0	0	00010201D	77
312	2	1	1	0		TEXTTEMP	77D	78
312	42	1	1	1	0	0	00010201D	79
312	2	1	1	0		TEXTTEMP	79D	80
312	43	1	1	1	0	0	00010201D	81
312	2	1	1	1		TEXTTEMP	81D	82
312	44	1	1	1	0	0	00010201D	83
312	2	1	1	1		TEXTTEMP	83D	84
312	45	1	1	1	0	0	00010201D	85
312	2	1	1	0		TEXTTEMP	85D	86
110	46	1	1	1	0	0	00000000D	87
110	2	1	1	0		SYM-LINE	87D	88
110	47	1	1	1	0	0	00000000D	89
110	2	1	1	0		SYM-LINE	89D	90
110	48	1	1	1	0	0	00000000D	91
110	2	1	1	0		SYM-LINE	91D	92
110	49	1	1	1	0	0	00000000D	93
110	2	1	1	0		SYM-LINE	93D	94

```
    110      50       1       1       1       0       0        00000000D     95
    110       2       1       1       0                SYM-LINE      95D     96
    416      51       1       1       1       0       0        01000000D     97
    416       2       1       1       2                  EXT-REF     97D     98
    406      52       1       1       1       0       0        01000000D     99
    406       2       1       1      12                XREFLIST     99D    100
    402      53       1       1       1       0       0        01000000D    101
    402       2       1       1      12                XREFINDX    101D    102
320,0,4HTANK,4,9,11,13,15,1,,,2,5,7;                                    1P      1
320,0,5HVALVE,7,21,23,25,27,29,31,33,1,,,2,17,19;                       3P      2
132, 0.5, 0.5, 0.0,,1,2,1HA,,,,,,,1;                                   5P      3
132, 0.5, 2.0, 0.0,,1,2,1HB,,,,,,,1;                                   7P      4
110,-0.5, 2.5, 0.0,-0.5, 0.0, 0.0;                                      9P      5
100, 0.0, 0.0, 0.0,-0.5, 0.0, 0.5, 0.0;                                11P      6
110, 0.5, 0.0, 0.0, 0.5, 2.5, 0.0;                                     13P      7
100, 0.0, 0.0, 2.5, 0.5, 2.5,-0.5, 2.5;                                15P      8
132,-2.0, 0.0, 0.0,,1,2,1HA,,,,,,,3;                                   17P      9
132, 2.0, 0.0, 0.0,,1,2,1HB,,,,,,,3;                                   19P     10
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,-2.0, 0.5, 0.0,-2.0,-0.5, 0.0;                                     25P     13
110, 2.0, 0.5, 0.0, 2.0,-0.5, 0.0;                                     27P     14
110, 0.0, 0.0, 0.0, 1.0, 2.0, 0.0;                                     29P     15
110, 0.0, 0.0, 0.0,-1.0, 2.0, 0.0;                                     31P     16
110, 1.0, 2.0, 0.0,-1.0, 2.0, 0.0;                                     33P     17
212,1,12, 0.200000E+01, 0.164063,1,, 0.0,0,0,                          35P     18
0.400000E+01, 0.800000E+01, 0.0,12HXYZ OIL P&ID;                       35P     19
420,1, 2.0, 2.0, 0.0, 2.0,,,1,6HTANK 5,77,2,41,43;                     37P     20
420,3, 5.5, 3.0, 0.0, 0.25,,,1,4HV107,79,2,45,47,1,73,2,63,69;         39P     21
132, 3.0, 6.0, 0.0,,1,2,5HINLET,81,,,,,,37;                            41P     22
132, 3.0, 3.0, 0.0,,1,2,6HOUTLET,81,,,,,,37;                           43P     23
132, 5.0, 3.0, 0.0,,1,2,,,,,,,,39;                                     45P     24
132, 6.0, 3.0, 0.0,,1,2,,,,,,,,39;                                     47P     25
132, 8.0, 3.0, 0.0,,1,2;                                               49P     26
132, 10.0, 3.0, 0.0,,1,2,4H5678,83;                                    51P     27
110, 3.0, 3.0, 0.0, 5.0, 3.0, 0.0, 1,73, 1,63;                         53P     28
110, 6.0, 3.0, 0.0, 8.0, 3.0, 0.0, 1,73, 1,63;                         55P     29
110, 8.0, 3.0, 0.0, 10.0, 3.0, 0.0, 1,73, 1,65;                        57P     30
110, 3.0, 6.0, 0.0, 8.0, 6.0, 0.0, 1,75, 1,67;                         59P     31
110, 8.0, 3.0, 0.0, 8.0, 6.0, 0.0, 1,75, 1,67;                         61P     32
406,2,12.,3HIPS;                                                       63P     33
406,2,10.,3HIPS;                                                       65P     34
406,2,4.,3HIPS;                                                        67P     35
406,4,5H98021,,,;                                                      69P     36
406,1,4HA100;                                                          71P     37
402,2,0,5,4,1,1,1,1,2,43,45,47,49,51,53,39,55,57,                      73P     38
11H834-6"-C121,85,75,0,1,71;                                           73P     39
402,2,0,2,2,0,0,0,1,2,49,41,59,61,1,73,1,71;                           75P     40
312, 1.0, 0.164063,,, 0.0,0,0, 1.5, 4.5, 0.0;                          77P     41
312, 0.6, 0.164063,,, 0.0,0,0, 5.2, 3.7, 0.0;                          79P     42
```

APPENDIX C.11 ENCODED FLOWSHEET

```
 312, 1.0, 0.164063,,, 0.0,0,0, 0.164063, 0.164063, 0.0;                81P     43
 312, 0.6, 0.164063,,, 0.0,0,0, 0.1, 0.1675, 0.0;                       83P     44
 312, 1.6, 0.164063,,, 0.0,0,0, 7.2, 2.670, 0.0;                        85P     45
 110, 10.0, 2.75, 0.0, 10.75, 2.75, 0.0;                                87P     46
 110, 10.75, 2.75, 0.0, 11.0, 3.0, 0.0;                                 89P     47
 110, 11.0, 3.0, 0.0, 10.75, 3.25, 0.0;                                 91P     48
 110, 10.75, 3.25, 0.0, 10.0, 3.25, 0.0;                                93P     49
 110, 10.0, 3.25, 0.0, 10.0, 2.75, 0.0;                                 95P     50
 416,19HXYZOIL.PROCESS.5678,12HO834-6"-C121;                             97P     51
 406,1,19HXYZOIL.PROCESS.5678;                                          99P     52
 402,1,12HI834-6"-C121,51;                                             101P     53
S      1G      3D    102P     53                                          T      1
```

Appendix D. Piping Model Example

D.1 Piping Model Characteristics

Process piping models carry several important types of information which determine what data structures must be present. Process piping models carry:

1. The identity of the lines in the piping model.
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 and piping components which control, mix, pump, and process the stream.
4. The association of lines, nozzles, equipment, and piping components with one another in the piping model.
5. Indication of a line that extends beyond the piping model and is continued elsewhere (*i.e.*, in another model).
6. Geometric and coordinate information for lines, nozzles, equipment, and piping components.

D.2 Entity Support for Piping Models

Figure D1 illustrates the general entity structure for representing piping and tubing models.

Pipes and tubes are represented by a Composite Curve Entity defining their centerline and a Property Entity defining their diameter. Component types are defined once (Network Subfigure Definition Entity) using whatever geometric representation is appropriate and then are referenced as many times as needed (Network Subfigure Instance Entity).

Logical connectivity is communicated by Connect Point and Composite Curve Entities. Every pipe or tube starts and ends at connect points and has intermediate connect points if there are branches off of the pipe or tube. Every component instance has a connect point at each port. The logical connection of pipes to pipes, pipes to components and components to components is communicated through null Composite Curve Entities which point to pairs of Connect Point Entities. This mechanism is used instead of shared connect points so that user defined end preparation and joint type properties could be associated with the component or pipe ends and with the joints respectively.

If the Connect Point Entity points to a Transformation Matrix Entity, the Z axis of that matrix defines the outward normal vector of the pipe end/nozzle.

A pipe line is described by a Flow Associativity Entity which lists the from- and to- nozzles (points to their Connect Point Entities) and then lists the in-line components.

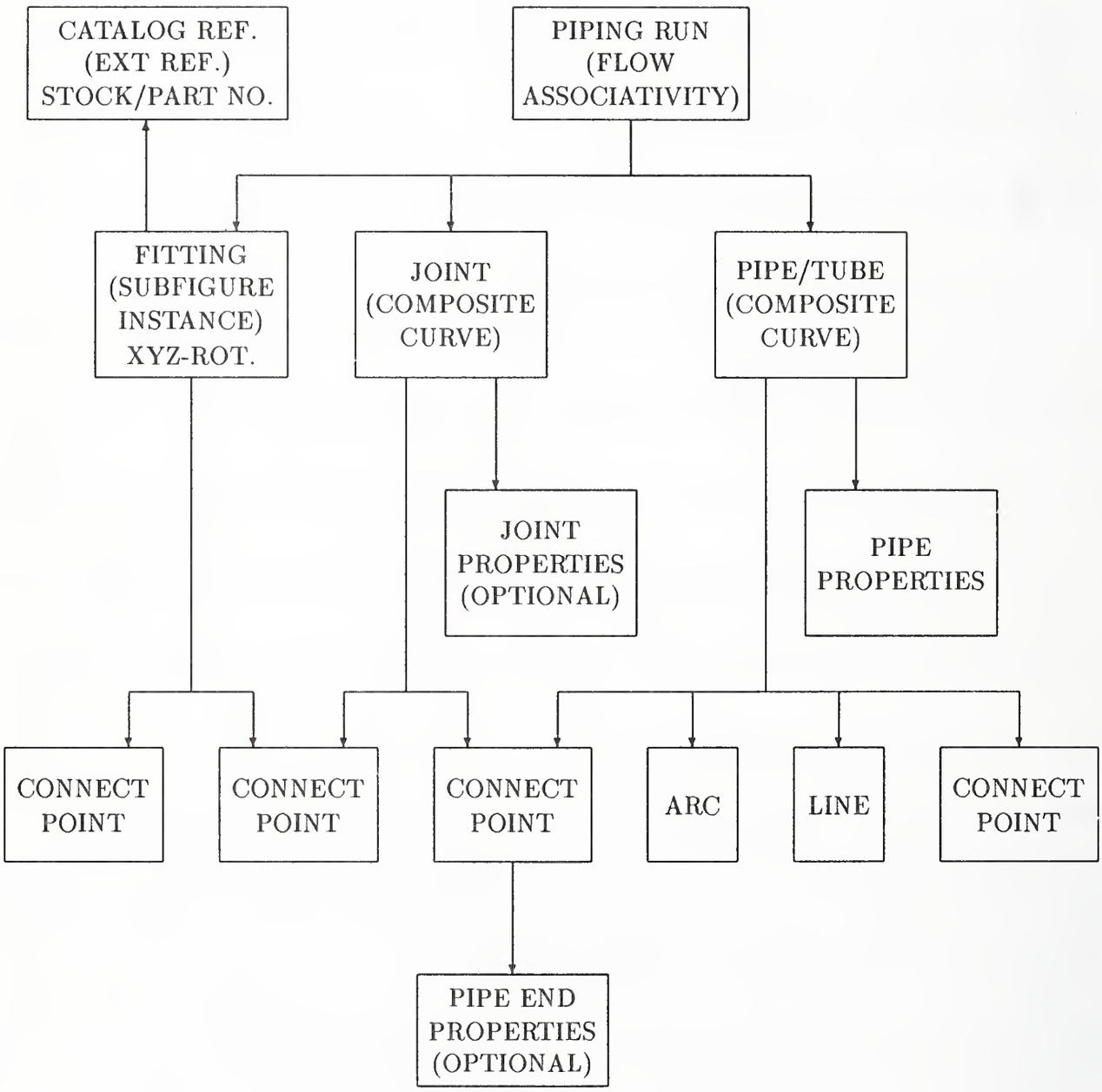


Figure D1. Piping Model Structure

Process plant piping is typically broken into multiple interconnected models, each of which would be transmittable by a separate file. The External Reference Entity is then used to tie the files together.

The geometry of process plant components is commonly maintained in a separate parts catalog. A parts catalog either may be copied into files that need it or may be transmitted as a separate file and referenced with the External Reference Entity.

D.3 Piping Model Example

A simple piping model is shown in Figure D2. Figure D3 indicates where network subfigure instances are used for the tank, tank nozzles, a valve, and two flanges. Figure D4 identifies the connect points required for the model, and Figure D5 shows the logical flow paths that the piping designer has assigned.

D.4 Component Definitions

Each instance of a component or piece of equipment is recorded in the file with a Network Subfigure Instance Entity. The geometry of components and equipment is defined once with a Network Subfigure Definition Entity. These Network Subfigure Definition Entities may be in the same file as the Network Subfigure Instance Entities or may be in a separate file. The latter case is illustrated in this appendix.

The level of completeness of component definition depends on the intended use of the piping model. A 3-D wire frame representation may be used for arrangement design for example.

Piping component catalogs typically contain tens of thousands of components. Plant interference analysis requires solid geometry models for components. CAD systems supporting 3-D plant design routinely use parametric representation to condense the catalog and ease user input. This specification supports the exchange of parametric component models by having Network Subfigure Definition Entities point to Macro Instance Entities, as shown in Figure D6.

In the example catalog file, the tank is defined as a subfigure which generates a (non-parametric) constructive solid geometry object consisting of a cylinder with a cone on top. The remaining parts required are tank nozzles, flanges, and a valve. Three parametric macro families are defined in the catalog file: nozzle, flange, and valve. A normal catalog file would have thousands of subfigure definitions, each with an appropriate part number. For this example, only the pieces required by the example model are listed, i.e., a 2 inch tank nozzle, a 4 inch tank nozzle, a 4 inch flange, and a 4 inch valve. Note that the 2 and 4 inch nozzles referenced the same macro definition but use different dimensions in the macro instances. Each subfigure definition points to a companion Macro Instance Entity which contains the part macro dimensions and points to the macro definition.

D.5 Attributes

Three specific properties (attributes) are supported for piping models. These are Part Number, Nominal Size, and Flow Line Specification Property Entities.

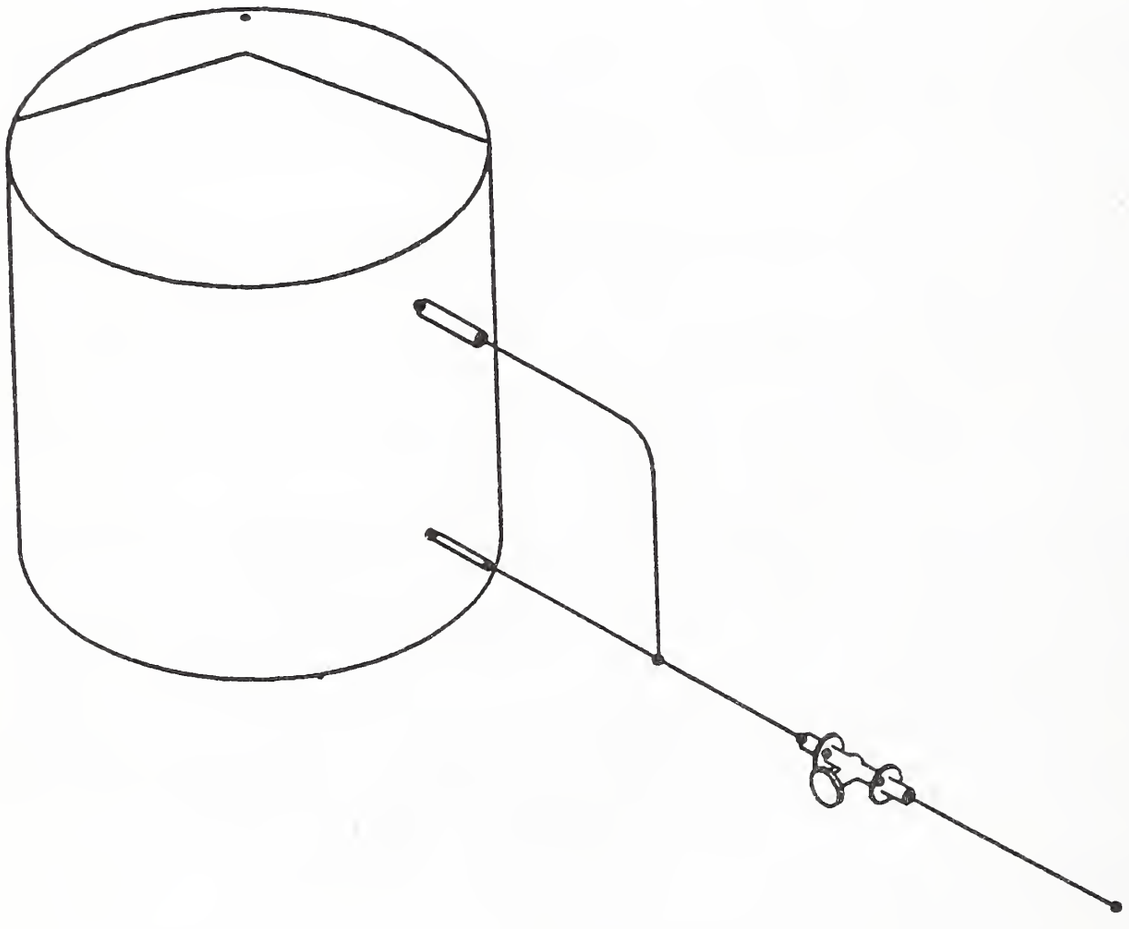


Figure D2. Simple Piping Model

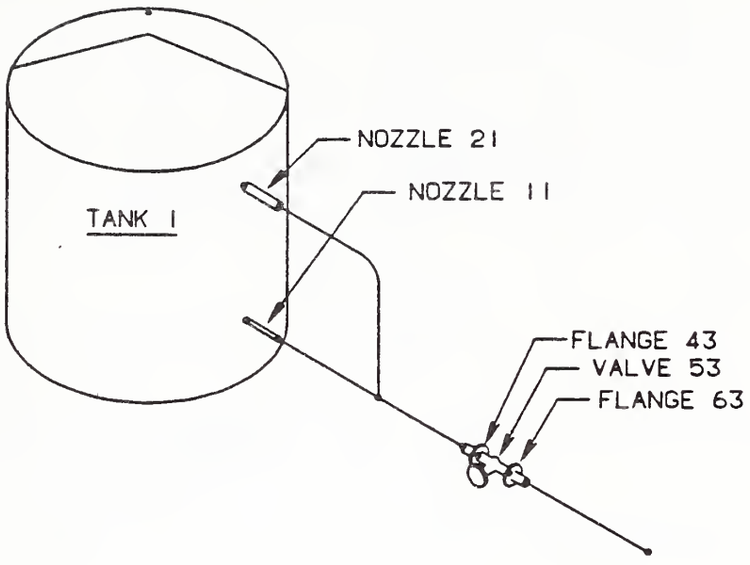


Figure D3. Subfigure Instances

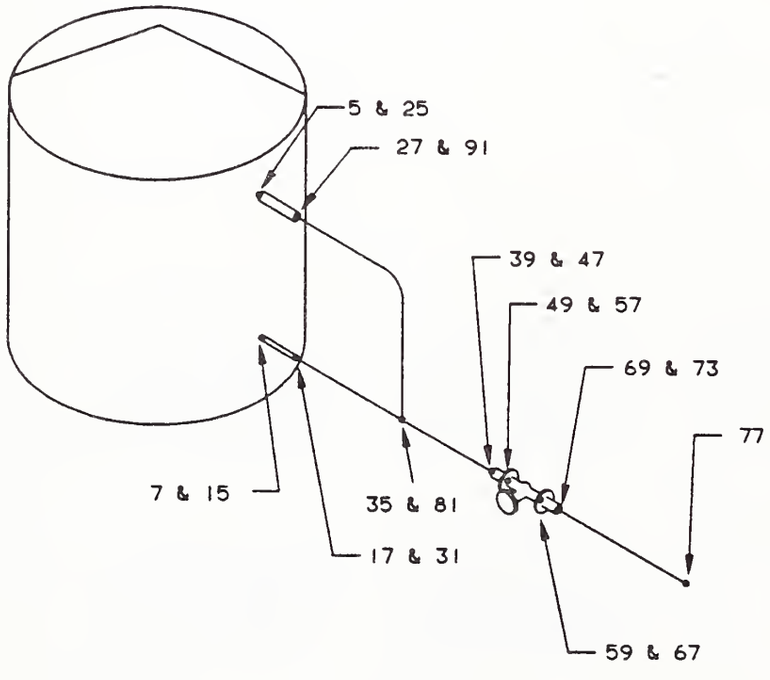


Figure D4. Required Connection Points

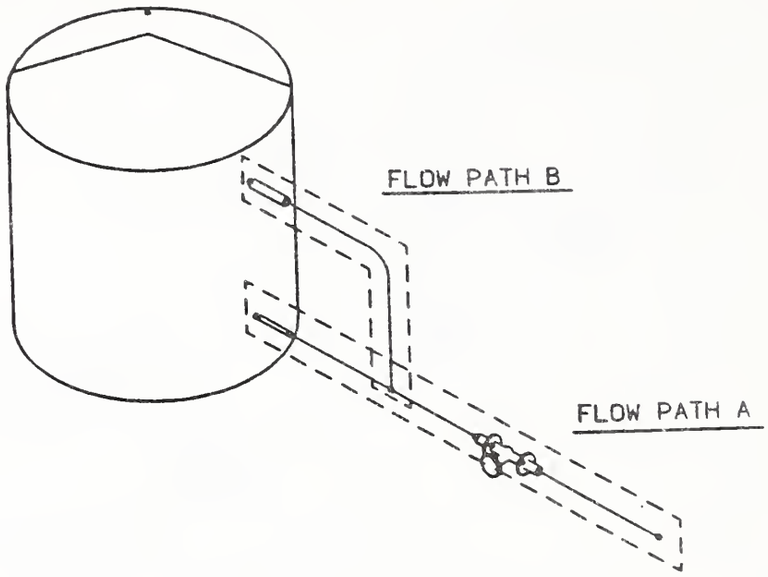


Figure D5. Logical Flow Path for Fluids

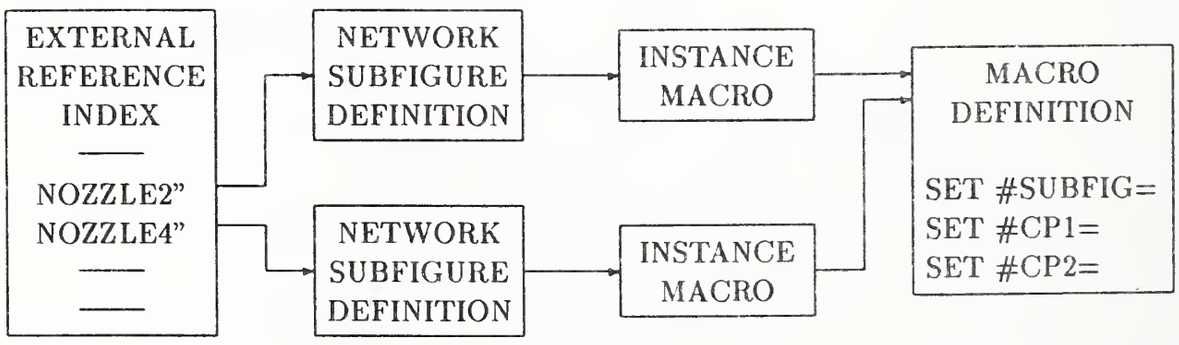


Figure D6. Macro Parametric Catalog Structure

D.6 Flow Paths

Each Flow Associativity Entity contains an ordered list of the Connect Point Entities along the path and a separate list of the geometry entities representing the path. Where a branch occurs, one path may point to another path. In the example (see Figure D5), the Flow Associativity Entity defining the flow path containing connect points 1 through 4 lists the Flow Associativity Entity containing connect points 15 through 12 as a continuation. This Flow Associativity Entity is also the entity to which the Flow Line Specification Property Entity is attached. Note that a rather arbitrary flow path assignment was made by the process engineer. Other assignments are possible and could be encoded.

D.7 Connectivity Across Piping Models

The flow path from connect point 77 continues in another piping model, in this case connect node 0834-4"C121 in model XYZOIL.PROCESS.5678. Both model files must contain external reference pointers to each other and must contain external reference indices.

The logical structure is the same as that used for two-dimensional piping diagrams and flow sheets as described in Appendix C.

D.8 Encoded Piping Model Example

The encoded piping model example is shown in Table D1 and Table D2.

APPENDIX D. PIPING MODEL EXAMPLE

Table D1. Encoded Piping Model Example

(This example file contains constructs which have not been implemented in any IGES translators. It may contain both semantic and syntactic errors in addition to known violations of the Specification. It is included in this version of the Specification because it illustrates the general entity support for piping and tubing models.)

```
3D PLANT PIPING MODEL EXAMPLE                                                S      1
1H,,1H;,12HPIPE EXAMPLE,19HXYZOIL.PROCESS.1234,9H<unknown>,3H5.0,32,38,      G      1
6,308,15,12HPIPE EXAMPLE,0.100000E+01,2,2HMM,3,0.06,13H870623.112300,        G      2
0.0001,10000.,9HG.E.SCOTT,3HORG,5,0;                                         G      3
     420       1                        0       0       3       000000300D      1
     420                  1            0                               0D      2
     124       2                                                   00000000D      3
     124                  3            0                               0D      4
     132       5       1  1            0       0       0       00020400D      5
     132                  1            0                               D      6
     132       6       1  1            0       0       0       00020400D      7
     132                  1            0                               D      8
     416       7       1  1            0       0       0       00000200D      9
     416                  1            0                               D     10
     420       8                       0       0      13       000000300D     11
     420                  1            0                               0D     12
     124       9                                           0       00000000D     13
     124                  3            0                               0D     14
     132      12       1  1            0       0       0       00020400D     15
     132                  1            0                               D     16
     132      13       1  1            0       0       0       00020400D     17
     132                  1            0                               D     18
     416      14       1  1            0       0       0       00000200D     19
     416                  1            0                               D     20
     420      15                       0       0      23       000000300D     21
     420                  1            0                               0D     22
     124      16                                           0       00000000D     23
     124                  3            0                               0D     24
     132      19       1  1            0       0       0       00020400D     25
     132                  1            0                               D     26
     132      20       1  1            0       0       0       00020400D     27
     132                  1            0                               D     28
     416      21       1  1            0       0       0       00000200D     29
     416                  1            0                               D     30
     132      22       1  1            0       0       0       00020400D     31
     132                  1            0                               D     32
     110      23          1            0       0       0       000010000D     33
     110       0       0  2            0                               0D     34
     132      25       1  1            0       0       0       00020400D     35
     132                  1            0                               D     36
     110      26          1            0       0       0       000010000D     37
     110       0       0  2            0                               0D     38
     132      28       1  1            0       0       0       00020400D     39
     132                  1            0                               D     40
```

102	29		1	0	0	0	00000000D	41
102	0	0	1	0			0D	42
420	30			0	0	45	000000300D	43
420			1	0			0D	44
124	31					0	00000000D	45
124			3	0			0D	46
132	34	1	1	0	0	0	00020400D	47
132			1	0			D	48
132	35	1	1	0	0	0	00020400D	49
132			1	0			D	50
416	36	1	1	0	0	0	00000200D	51
416			1	0			D	52
420	37			0	0	55	000000300D	53
420			1	0			0D	54
124	38						00000000D	55
124			3	0			D	56
132	41	1	1	0	0	0	00020400D	57
132			1	0			D	58
132	42	1	1	0	0	0	00020400D	59
132			1	0			D	60
416	43	1	1	0	0	0	00000200D	61
416			1	0			D	62
420	44			0	0	65	000000300D	63
420			1	0			0D	64
124	45					0	00000000D	65
124			3	0			0D	66
132	48	1	1	0	0	0	00020400D	67
132			1	0			D	68
132	49	1	1	0	0	0	00020400D	69
132			1	0			D	70
416	50	1	1	0	0	0	00000200D	71
416			1	0			D	72
132	51	1	1	0	0	0	00020400D	73
132			1	0			D	74
110	52		1	0	0	0	000010000D	75
110	0	0	2	0			0D	76
132	54	1	1	0	0	0	00020400D	77
132			1	0			D	78
102	55		1	0	0	0	00000000D	79
102	0	0	1	0			0D	80
132	56	1	1	0	0	0	00020400D	81
132			1	0			D	82
110	57		1	0	0	0	000010000D	83
110	0	0	2	0			0D	84
124	59					0	00000000D	85
124			3	0			0D	86
100	62		1	0	0	85	000010000D	87
100	0	0	1	0			0D	88
110	63		1	0	0	0	000010000D	89
110	0	0	2	0			0D	90

```
    132      65       1       1       0       0       0        00020400D     91
    132                       1       0                                D     92
    102      66               1       0       0       0        00000000D     93
    102       0       0       1       0                                0D     94
    102      67               1       0       0       0        00000000D     95
    102       0       0       1       0                                0D     96
    102      68               1       0       0       0        00000000D     97
    102       0       0       1       0                                0D     98
    102      69               1       0       0       0        00000000D     99
    102       0       0       1       0                                0D    100
    102      70               1       0       0       0        00000000D    101
    102       0       0       1       0                                0D    102
    102      71               1       0       0       0        00000000D    103
    102       0       0       1       0                                0D    104
    102      72               1       0       0       0        00000000D    105
    102       0       0       1       0                                0D    106
    102      73               1       0       0       0        00000000D    107
    102       0       0       1       0                                0D    108
    102      74               1       0       0       0        00000000D    109
    102       0       0       1       0                                0D    110
    102      75               1       0       0       0        00000000D    111
    102       0       0       1       0                                0D    112
    406      76                       1                        00020000D    113
    406                       1      13                                0D    114
    406      77                       2                        00020000D    115
    406                       1      13                                0D    116
    402      78       1                                        00000300D    117
    402                       2      18                                D    118
    402      80       1                                        00020300D    119
    402                       1      18                                D    120
    416      81       1       1       0       0       0        01000000D    121
    416                       1       2                                D    122
    406      82                       2                        00020000D    123
    406                       1      12                                0D    124
    402      83       1                                        00020000D    125
    402                       1      12                                0D    126
420,9,,,,1.0,1.0,1.0,,,,2,5,7;                                          1P      1
124, 0.10000E+01, 0.00000E+00, 0.00000E+00, 0.20000E+05,                 3P      2
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,                 3P      3
     0.00000E+00, 0.00000E+00, 0.10000E+01, 0.00000E+00;                 3P      4
132,0.18500E+05,0.0E+00,0.25000E+04,,2,2,5HINLET,,,,,,,1,1,119;         5P      5
132,0.18500E+05,0.0E+00,0.77500E+03,,2,2,6HOUTLET,,,,,,,1,1,117;        7P      6
416,7HCATALOG,4HTANK;                                                   9P      7
420,19,,,,1.0,1.0,1.0,,,,2,15,17,1,117;                                11P      8
124, 0.10000E+01, 0.00000E+00, 0.00000E+00, 0.18300E+05,                13P      9
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,                13P     10
     0.00000E+00, 0.00000E+00, 0.10000E+01, 0.77500E+03;                13P     11
132,.18500E+05,0.0E+00,0.77500E+03,,2,2,6HOUTLET,,,,,,,11,1,117;        15P     12
132,.18300E+05,0.0E+00,0.77500E+03,,2,2,6HOUTLET,,,,,,,11,1,117;        17P     13
416,7HCATALOG,7HNOZZLE2;                                                19P     14
420,29,,,,1.0,1.0,1.0,,,,2,25,27,1,119;                                 21P     15
```

```
124, 0.10000E+01, 0.00000E+00, 0.00000E+00, 0.18300E+05,                23P     16
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,                23P     17
     0.00000E+00, 0.00000E+00, 0.10000E+01, 0.25000E+04;                23P     18
132,0.18500E+05,0.0E+00,0.250E+04,,2,2,5HINLET,,,,,,,21,1,119;          25P     19
132,0.18300E+05,0.0E+00,0.250E+04,,2,2,5HINLET,,,,,,,21,1,119;          27P     20
416,7HCATALOG,7HNOZZLE4;                                                29P     21
132,0.18300E+05,0.00000E+00,0.77500E+03,,2,2,,,,,,,,1,117;              31P     22
110, 0.18300E+05, 0.00000E+00, 0.77500E+03,                             33P     23
     0.16500E+05, 0.00000E+00, 0.77500E+03;                             33P     24
132,0.1650E+05,0.0E+00,0.7750E+03,,2,2,,,,,,,,1,117;                    35P     25
110, 0.16500E+05, 0.00000E+00, 0.77500E+03,                             37P     26
     0.15200E+05, 0.00000E+00, 0.77500E+03;                             37P     27
132,0.15200E+05,0.00000E+00,0.77500E+03,,2,2,,,,,,,,1,117;              39P     28
102,5,31,33,35,37,39,1,117,1,113;                                       41P     29
420,51,,,,1.0,1.0,1.0,,,,2,47,49,1,117;                                 43P     30
124, -.10000E+01, 0.00000E+00, 0.00000E+00, 0.15000E+05,                45P     31
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,                45P     32
     0.00000E+00, 0.00000E+00, -.10000E+01, 0.77500E+03;                45P     33
132,0.15200E+05,0.0E+00,0.77500E+03,,2,2,1HA,,,,,,,43,1,117;            47P     34
132,0.15000E+05,0.0E+00,0.77500E+03,,2,2,1HB,,,,,,,43,1,117;            49P     35
416,7HCATALOG,6HFLANGE;                                                 51P     36
420,61,,,,1.0,1.0,1.0,,,,2,57,59,1,117;                                 53P     37
124, -.10000E+01, 0.00000E+00, 0.00000E+00, 0.14750E+05,                55P     38
     0.00000E+00, 0.31398E-06, 0.10000E+01, 0.00000E+00,                55P     39
     0.00000E+00, 0.10000E+01, -.31398E-06, 0.77500E+03;                55P     40
132,0.15000E+05,0.00000E+00,0.77500E+03,,2,2,1HA,,,,,,,53,1,117;        57P     41
132,0.14500E+05,0.00000E+00,0.77500E+03,,2,2,1HB,,,,,,,53,1,117;        59P     42
416,7HCATALOG,5HVALVE;                                                  61P     43
420,71,,,,1.0,1.0,1.0,,,,2,67,69,1,117;                                 63P     44
124, 0.10000E+01, 0.00000E+00, 0.00000E+00, 0.14500E+05,                65P     45
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,                65P     46
     0.00000E+00, 0.00000E+00, 0.10000E+01, 0.77500E+03;                65P     47
132,0.14500E+05,0.00000E+00,0.77500E+03,,2,2,1HB,,,,,,,63,1,117;        67P     48
132,0.14300E+05,0.00000E+00,0.77500E+03,,2,2,1HA,,,,,,,63,1,117;        69P     49
416,7HCATALOG,6HFLANGE;                                                 71P     50
132,0.14300E+05,0.00000E+00,0.77500E+03,,2,2,,,,,,,,1,117;              73P     51
110, 0.14300E+05, 0.00000E+00, 0.77500E+03,                             75P     52
     0.12500E+05, 0.00000E+00, 0.77500E+03;                             75P     53
132,0.12500E+05,0.00000E+00,0.77500E+03,,2,2,,,,,,,,1,117;              77P     54
102,    3,     73,     75,     77,1,117,1,113;                          79P     55
132,0.16500E+05,0.00000E+00,0.77500E+03,,2,2,,,,,,,,1,119;              81P     56
110, 0.16500E+05, 0.00000E+00, 0.77500E+03,                             83P     57
     0.16500E+05, 0.00000E+00, 0.22500E+04;                             83P     58
124, -.10000E+01, 0.00000E+00, 0.00000E+00, 0.16750E+05,                85P     59
     0.00000E+00, 0.00000E+00, 0.10000E+01, 0.00000E+00,                85P     60
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.22500E+04;                85P     61
100,0.,0.,0.,0.25000E+03,0.,0.,0.25000E+03;                             87P     62
110, 0.16750E+05, 0.00000E+00, 0.25000E+04,                             89P     63
     0.18300E+05, 0.00000E+00, 0.25000E+04;                             89P     64
132,0.18300E+05,0.00000E+00,0.25000E+04,,2,2,,,,,,,,1,119;              91P     65
102,5,81,83,87,89,91,1,119,1,115;                                       93P     66
```

```
102,    2,      5,     25;                                                    95P     67
102,    2,     27,     91;                                                    97P     68
102,    2,     39,     47;                                                    99P     69
102,    2,     49,     57;                                                   101P     70
102,    2,     59,     67;                                                   103P     71
102,    2,     69,     73;                                                   105P     72
102,    2,      7,     15;                                                   107P     73
102,    2,     17,     31;                                                   109P     74
102,    2,     81,     35;                                                   111P     75
 406,2,4.,3HIPS;                                                             113P     76
 406,2,2.,3HIPS;                                                             115P     77
 402,2,0,15,6,1,0,1,2,2,7,15,17,31,35,39,47,49,57,59,67,69,                  117P     78
     73,77,121,11,41,43,53,63,79,11H834-4"-C121,119;                         117P     79
 402,2,0,5,2,1,0,0,2,2,81,91,27,25,5,93,21,11H834-2"-C121,1,117;             119P     80
 416,19HXYZOIL.PROCESS.5678,11H0834-4"C121;                                  121P     81
 406,2,19HXYZOIL.PROCESS.5678,7HCATALOG;                                     123P     82
 402,1,11H0834-2"C121,77;                                                    125P     83
S      1G      3D    126P     83                                                T0000001
```

Table D2. Encoded Piping Catalog for Piping Model Example

(This example file contains constructs which have not been implemented in any IGES translators. It may contain both semantic and syntactic errors in addition to known violations of the Specification. It is included in this version of the Specification because it illustrates the general entity support for piping and tubing models.)

```
3D PLANT PIPING CATALOG FOR PIPING MODEL EXAMPLE                          S      1
1H,,1H;,7HCATALOG,7HCATALOG,9H<unknown>,3H5.0,32,38,                      G      1
6,308,15,7HCATALOG,0.100000E+01,2,2HMM,3,.06,13H870621.091200,            G      2
0.0001,10000.,9HG.E.SCOTT,3HORG,5,0;                                      G      3
     320       1              0                        0       000000200D      1
     320       0       0      1       0                              0D      2
     154       2       1      0       1       0       0        00010200D      3
     154                      1       0                              D      4
     156       3       1      0       1       0       7        00010200D      5
     156                      1       0                              D      6
     124       4                                       0        00000000D      7
     124                      3       0                              0D      8
     320       7              0                        0       000000200D      9
     320       0       0      1       0                              0D     10
   10001       8     -25      0       1       0       0        00010200D     11
   10001                      1       0                              D     12
     320       9              0                        0       000000200D     13
     320       0       0      1       0                              0D     14
   10001      10     -25      0       1       0       0        00010200D     15
   10001                      1       0                              D     16
     320      11              0                        0       000000200D     17
     320       0       0      1       0                              0D     18
   10002      12     -27      0       1       0       0        00010200D     19
   10002                      1       0                              D     20
     320      13              0                        0       000000200D     21
     320       0       0      1       0                              0D     22
   10003      14     -29      0       1       0       0        00010200D     23
   10003                      1       0                              D     24
     306      15                                                00000200D     25
     306                      6       0                              0D     26
     306      21                                                00000200D     27
     306                     18       0                              0D     28
     306      39                                                00000200D     29
     306                     47       0                              0D     30
     402      86       1                                        00000000D     31
     402                      1      12                              0D     32
320,0,4HTANK,2,3,5,2;                                                     1P      1
154,3000.,1500.;                                                          3P      2
156,1000.,1500.,25.;                                                      5P      3
124, 0.10000E+01, 0.00000E+00, 0.00000E+00, 0.00000E+00,                  7P      4
     0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,                  7P      5
     0.00000E+00, 0.00000E+00, 0.10000E+01, 0.30000E+04;                  7P      6
320,0,7HNOZZLE4,1,11,2;                                                   9P      7
```

```
10001,101.6,200;                                                    11P      8
320,0,7HNOZZLE2,1,15,2;                                             13P      9
 10001,50.8,200;                                                    15P     10
 320,0,6HFLANGE,1,19,2;                                             17P     11
 10002,100.,100.,190.;                                              19P     12
 320,0,5HVALVE,1,23,2;                                              21P     13
 10003,100.,100.,250.,250.,250.,40.,250;                            23P     14
306,MACRO,10001,RAD,HT;                                             25P     15
 SET #SUBFIG=320,0,6HNMACRO,1,#CYL,2,,,2,#CP1,#CP2;                 25P     16
 SET #CP1   =132,0.,0.,0.,,2,0,,,,,,,,#SUBFIG;                      25P     17
 SET #CP2   =132,HT,0.,0.,,2,0,,,,,,,,#SUBFIG;                      25P     18
 SET #CYL   =154,HT,RAD,0.,0.,0.,1.,0.,0.;                          25P     19
 ENDM;                                                              25P     20
306,MACRO,10002,P1,P2,P6;                                           27P     21
  SET #SUBFIG=320,0,6HFMACRO,2,#CYL,#CONE,2,,,2,#CP1,#CP2;          27P     22
  SET #CP1   =132,-P6-10.,0.,0.,,2,0,,,,,,,,#SUBFIG;               27P     23
  SET #CP2   =132,0.,0.,0.,,2,0,,,,,,,,#SUBFIG;                    27P     24
  SET #XFM1  =                                                      27P     25
  124, 0.00000E+00, 0.00000E+00, 0.10000E+01, 0.00000E+00,          27P     26
       0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,          27P     27
      -0.10000E+01, 0.00000E+00, 0.00000E+00, 0.00000E+00;          27P     28
  LET /MTX   =#XFM1;                                                27P     29
  SET #CYL   =154,10.,1.25*P1,0.,0.,0.,0.,0.,-1.;                   27P     30
  LET /MTX   =/HDR;                                                 27P     31
  SET #XFM2  =                                                      27P     32
  124, 0.00000E+00, 0.00000E+00, 0.10000E+01,-0.10000E+02,          27P     33
       0.00000E+00, 0.10000E+01, 0.00000E+00, 0.00000E+00,          27P     34
      -0.10000E+01, 0.00000E+00, 0.00000E+00, 0.00000E+00;          27P     35
  LET /MTX   =#XFM2;                                                27P     36
  SET #CONE  =156,P6,.5*P1,.5*P2,0.,0.,0.,0.,0.,-1.;                27P     37
 ENDM;                                                              27P     38
 306,MACRO,10003,P1,P2,P6,P7,P8,P9,P10;                             29P     39
  SET #SUBFIG=320,,6HVMACRO,6,#CYL1,#CYL2,#CYL3,#SP,#CYL4,#CYL5,    29P     40
              2,,,2,#CP1,#CP2;                                      29P     41
  SET #CP1   =132,-P6,0.,0.,,2,0,,,,,,,,#SUBFIG;                    29P     42
  SET #CP2   =132,P7,0.,0.,,2,0,,,,,,,,#SUBFIG;                     29P     43
  SET #XFM1  =                                                      29P     44
  124, 0.,   0.,   1.,   -P6,                                       29P     45
       0.,   1.,   0.,    0.,                                       29P     46
      -1.,   0.,   0.,    0.;                                       29P     47
  LET /MTX   =#XFM1;                                                29P     48
  SET #CYL1  =154,P7+P6,.5*P1;                                      29P     49
  LET /MTX   =/HDR;                                                 29P     50
  SET #XFM2  =                                                      29P     51
  124, 1.,   0.,   0.,    0.,                                       29P     52
       0.,   0.,   1.,    P8,                                       29P     53
       0.,  -1.,   0.,    0.;                                       29P     54
  LET /MTX   =#XFM2;                                                29P     55
  SET #CYL2  =154,P9,.5*P10;                                        29P     56
  LET /MTX   =/HDR;                                                 29P     57
```

```
  SET #XFM3  =                                                    29P     58
  124, 1.,   0.,   0.,    0.,                                     29P     59
       0.,   0.,   1.,    0.,                                     29P     60
       0.,  -1.,   0.,    0.;                                     29P     61
  LET /MTX   =#XFM3;                                             29P     62
  SET #CYL3  =154,P8,.5*P1;                                      29P     63
  LET /MTX   =/HDR;                                              29P     64
  SET #XFM4  =                                                   29P     65
  124, 1.,   0.,   0.,    .5*P1,                                  29P     66
       0.,   1.,   0.,    -.25*P1,                                29P     67
       0.,   0.,   1.,    0.;                                     29P     68
  LET /MTX   =#XFM4;                                             29P     69
  SET #SP  =158,.6*P1;                                           29P     70
  LET /MTX   =/HDR;                                              29P     71
  SET #XFM5  =                                                   29P     72
  124, 0.,   0.,   1.,    -P6,                                    29P     73
       0.,   1.,   0.,    0.,                                     29P     74
      -1.,   0.,   0.,    0.,                                     29P     75
  LET /MTX   =#XFM5;                                             29P     76
  SET #CYL4  =154,10.,1.25*P1;                                   29P     77
  LET /MTX   =/HDR;                                              29P     78
  SET #XFM6  =                                                   29P     79
  124, 0.,   0.,   1.,    P7-10,                                  29P     80
       0.,   1.,   0.,    0.,                                     29P     81
      -1.,   0.,   0.,    0.;                                     29P     82
  LET /MTX   =#XFM6;                                             29P     83
  SET #CYL5  =154,10.,1.25*P1;                                   29P     84
 ENDM;                                                           29P     85
 402,5,4HTANK,1,7HNOZZLE4,9,7HNOZZLE2,13,6HFLANGE,17,5HVALVE,21;  31P     86
S      1G      3D     32P     86                                   T0000001
```

Appendix E. Spline Curves and Surfaces

E.1 Introduction

Chapter 3 of this Specification includes four different types of spline representations:

1. A parametric piecewise cubic polynomial curve,
2. A rational B-spline curve,
3. A grid of bicubic patches (for surfaces), and
4. 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 Chapter 3 include parametric cubics, piecewise linear, Wilson-Fowler, modified Wilson-Fowler, rational and nonrational B-splines, and rational and nonrational Cartesian product B-spline surfaces. 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.

E.2 Spline Functions

In Section 3.8, 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:

1. N: The number of segments,
2. $T(1), \ldots, T(N+1)$: The endpoints and the breakpoints separating the cubic polynomial segments,
3. $A(i), B(i), C(i), D(i), i = 1, \ldots, N$: The coefficients of the polynomials representing the spline in each of the N segments,
4. 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.
5. H: Degree of continuity. See Section 3.8.

To evaluate the spline at a point "u", first determine the segment containing "u", *i.e.*, the segment "i" such that $T(i) \leq u \leq T(i+1)$, 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.*, $S(T(i)) = A(i), i = 1, \ldots, N$, and $S(T(N+1)) = TP0$). 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:

1. End points $E1, E2$ and end slopes $S1, S2$: The spline can be evaluated using the "Hermite" basis (see [DEBO78], p. 59).
2. Values at four points: The spline value can be computed from the Lagrange or Newton interpolation formulae (see [DEBO78]).
3. 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 (see [GORD74])) and indirectly in various spline calculations (*e.g.*, computing a cubic spline interpolate). For every set of breakpoints $T(1), \ldots, T(N+1)$ and degree of continuity H, a set of B-spline functions $B(1,u)$, $B(2,u), \ldots, B(n^*,u)$ can be constructed (see [DEBO78]). Then, for any piecewise polynomial $S(u)$ with these breakpoints and continuity there is a set of B-spline coefficients $a(1), \ldots, a(n^*)$ such that $S(u)$ can be represented as a linear combination of these B-splines,

$$S(u) = a(1) * B(1,u) + a(2) * B(2,u) + \ldots + a(n^*) * B(n^*,u)$$

where $n^* = (N-1) * (3-H) + 4$.

B-splines can be computed from piecewise polynomials and *vice versa* (see [DEBO78], p. 116).

E.3 Spline Curves

The comments in this section pertain primarily to Section 3.8.

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),

$$\begin{aligned} X(u) &= S_x(u), \\ Y(u) &= S_y(u), \text{ and} \\ Z(u) &= S_z(u) \end{aligned}$$

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 (see [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.

E.4 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 (positive real numbers).

The $P(i)$ are the control points (points in R^3).

The b_i are the B-spline basis functions.

These are defined as soon as their degree, M, and underlying knot sequence, 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; $T(-M), \ldots, T(0), \ldots, T(N), \ldots, T(N+M)$.

The curve itself is parameterized for $V(0) \le t \le V(1)$ where $T(0) \le V(0) < V(1) \le T(N)$.

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

$$\sum_{i=0}^{K} b_i(t) \;=\; 1.$$

As 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 $N(t|t_{i-M},\ldots,t_{i+1})$ denote the B-spline basis function of degree M supported by the interval $[t_{i-M}, t_{i+1}]$.

With this notation, the degree 0 functions are simply characteristic functions of a half-open interval.

$$N(t|\,a,b) \;=\; \begin{array}{l} 1 \text{ if } a \le t < b \\ 0 \text{ otherwise} \end{array}$$

The degree k functions are defined in terms of those of degree $k-1$.

$$N(t|s_0,\ldots,s_k) \;=\; \frac{(t-s_0)\,N(t|s_0,\ldots,s_{k-1})}{s_{k-1}\;-\;s_0} \;+\; \frac{(s_k-t)N(t|s_1,\ldots,s_k)}{s_k\;-\;s_1}$$

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 can be expressed exactly as rational B-spline curves. An unpublished paper by Fuhr on this subject is available from the IGES Office at the National Bureau of Standards.

E.5 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:

1. M: The number of grid lines in u,
2. $TU(1),\ldots,TU(M+1)$: The breakpoints in u (u values of grid lines),
3. N: The number of grid lines in v,
4. $TV(1),\ldots,TV(N+1)$: The breakpoints in v (v values of grid lines),
5. $Ax(i,j), Bx(i,j),\ldots, Ay(i,j),\ldots, Az(i,j),\ldots$, for $i = 1,\ldots,M; j = 1,\ldots,N$:
   The $M * N$ sets of $3 * 16$ coefficients defining the bicubic polynomial for each of the three coordinates of the patch,
6. CTYPE: The spline type. (1=linear, 2=quadratic, 3=cubic, 4=Wilson- Fowler, 5=Modified Wilson-Fowler, 6 = B-spline), and
7. 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 $TU(i) \le u \le TU(i+1)$ and $TV(j) \le v \le TV(j+1)$, then evaluate the bicubic polynomial in that patch, *i.e.*, compute:

$$
\begin{aligned}
X(u,v) \;=\; & Ax(i,j) * vv^0 * uu^0 \;+\; Bx(i,j) * vv^0 * uu^1 \;+\; Cx(i,j) * vv^0 * uu^2 + Dx(i,j) * vv^0 * uu^3 \\
& + Ex(i,j) * vv^1 * uu^0 + Fx(i,j) * vv^1 * uu^1 + Gx(i,j) * vv^1 * uu^2 + Hx(i,j) * vv^1 * uu^3 \\
& + Kx(i,j) * vv^2 * uu^0 + Lx(i,j) * vv^2 * uu^1 + Mx(i,j) * vv^2 * uu^2 + Nx(i,j) * vv^2 * uu^3 \\
& + Px(i,j) * vv^3 * uu^0 + Qx(i,j) * vv^3 * uu^1 + Rx(i,j) * vv^3 * uu^2 + Sx(i,j) * vv^3 * uu^3
\end{aligned}
$$

$Y(u,v) = Ay(i,j)$ . . .

$Z(u,v) = Az(i,j)$ . . .

where $uu = u - TU(i)$ and $vv = v - TV(j)$.

The patches in the spline surface are equivalent to the bicubic surface patch (see [ROGE76], p. 170) for the conversion details). The parameters of the bicubic surface 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 (*i.e.*, 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.

E.6 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,

$$G(s,t) \;=\; \frac{\sum_{i=0}^{K1} \sum_{j=0}^{K2} W(i,j)P(i,j)b_i(s)b_j(t)}{\sum_{i=0}^{K1} \sum_{j=0}^{K2} W(i,j)b_i(s)b_j(t)}$$

where the notation is analogous to that used for rational B-spline curves.

The $W(i,j)$ are the weights (positive 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 $U(0) \leq s \leq U(1)$ and for $V(0) \leq t \leq V(1)$ where $S(0) \leq U(0) < U(1) \leq S(N1)$ and $T(0) \leq V(0) < V(1) \leq T(N2)$. Rational Bezier surfaces can be expressed exactly as rational B-spline surfaces. An unpublished paper by Fuhr on this subject is available from the IGES Office at the National Bureau of Standards.

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. The periodic flags are to be interpreted as purely informational. The surfaces which are flagged to be periodic are to be evaluated exactly the same as in the non-periodic case.

APPENDIX E.6 RATIONAL B-SPLINE SURFACES

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.

Appendix F. Conic Arcs

Conic arcs as specified are extremely sensitive to the data in two distinct ways:

Accuracy. It is numerically sensitive; small changes in the coefficients can cause large changes in the locations of the points satisfying the conic equation.

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 Conic Arc Entity (Type 104) 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 Conic Arc Entity in their own databases, the preprocessor may put the data into 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.

The following are suggested sets of values for the cases of an ellipse, a hyperbola, and a parabola:

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).	

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).		

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 or Y axis and centered about the origin. A Transformation Matrix (Type 124) 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 G. Color Space Mappings

It is often more convenient to operate in some color space other than RGB. The relationship between RGB (Red, Green, Blue) and CMY (Cyan, Magenta, Yellow) is given by:

R = 100.0 - C		R = red	C = cyan
G = 100.0 - M	where:	G = green	M = magenta
B = 100.0 - Y		B = blue	Y = yellow

The HSL (Hue, Saturation, Lightness) color space can be defined in terms of RGB in several ways with subtle variations. A typical approach is given by the following transformation:

$$H = \frac{1}{2\pi} tan^{-1} \left(\frac{2R-G-B}{\sqrt{3}\,(G-B)} \right)$$

$$S = \sqrt{R^2 + G^2 + B^2 - RG - RB - BG}$$

$$L = \frac{1}{3}(R + G + B)$$

where:

H = Hue

S = Saturation

L = Lightness

Variations on this transformation are given in [JOBL78] and [SMIT78].

Appendix H. ASCII Form Conversion Utility

This appendix gives details of a utility program to convert a file in the ASCII Form from the regular (fixed line length) ASCII Format to the Compressed ASCII Format and back again. The program is written in FORTRAN 77 code. The program is known to fail to convert a file from Compressed ASCII Format to regular ASCII Format if the file contains any string constant compressed such that an ASCII character "D" falls into column one. The source code is available from the IGES Office at the National Bureau of Standards.

```
C*******************************************************************************
C     THIS PROGRAM IS WRITTEN IN VAX 4.2 FORTRAN 77 SOURCE.  ITS PURPOSE IS
C     TO CONVERT BETWEEN REGULAR ASCII FORMAT AND COMPRESSED ASCII FORMAT.
C
      PROGRAM IGES
C
C     PROGRAM ORIGINALLY 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     REVISED BY LEE KLEIN 7-28-86
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY LEE KLEIN 8-1-86
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY LEE KLEIN 8-7-86
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY ROBERT COLSHER 22 AUG 1986
C                                   IGES DATA ANALYSIS COMPANY
C
C     PURPOSE:
C              TO CONVERT NEW FORM OF IGES OUTPUT TO OLD FORM AND
C              OLD FORM TO NEW.
C
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*******************************************************************************
C     SPECIAL NOTES:
C
C     1.  THE DOLLAR SIGN IN I/O FORMAT STATEMENTS IS THERE TO SUPPRESS
C         THE CARRIAGE RETURN AT THE END OF THE PROMPT LINE.
```

```
C     2.  IN COMPILERS THAT DO NOT ACCEPT A VARIABLE LENGTH OUTPUT FORMAT,
C         SOME MEANS OF COMPRESSING BLANK PADDED LINES MUST BE USED.
C     3.  SEE CHANGE NOTES THROUGHOUT THE CODE
C
C*******************************************************************************
      CHARACTER * 80 LINE1
      CHARACTER * 60 INFILE,OUTFIL
C
C     PROMPT AND GET FILE NAMES...
C
      GOTO 1010
1000  WRITE(*,1900)
1010  WRITE(*,1910)
      WRITE(*,1920)
      READ(*,1930)INFILE
      WRITE(*,1940)
      READ(*,1930)OUTFIL
C
C     READ THE FIRST LINE...
C
      OPEN(UNIT=10,FILE=INFILE,STATUS='OLD',ERR=1000)
      READ(10,1950)LINE1
      CLOSE(10)
C
C     CHECK TO SEE WHICH FORM THE INPUT IS IN...
C
C     ONLY A 'C'OMPRESS RECORD CAN OCCUR AT BEGINNING OF 'NEW' FILE
C     ONLY A 'S'TART RECORD CAN OCCUR AT BEGINNING OF 'OLD' FILE
C
      IF (LINE1(73:73).EQ.'C') THEN
       CALL OLDFRM(INFILE,OUTFIL)
      ELSE IF (LINE1(73:73).EQ.'S') THEN
        CALL NEWFRM(INFILE,OUTFIL)
       ELSE
        WRITE(*,1960)' File contains ILLEGAL record format'
        CLOSE(10)
        STOP
       ENDIF
      WRITE(*,1960)' '
      WRITE(*,1960)' IGES conversion complete'
      WRITE(*,1960)' '
C
C     FORMATS
C
1900  FORMAT(/1X,'Error in filename. Try again.')
1910  FORMAT(/1X,'*** IGES FILE CONVERSION PROGRAM ***'/)
1920  FORMAT($,' Enter input file name: ')
1930  FORMAT(A60)
1940  FORMAT($,' Enter output file name: ')
1950  FORMAT(A80)
```

```
1960  FORMAT(A)
C
      END

C*********************************************************************
C
      SUBROUTINE OLDFRM(INFILE,OUTFIL)
C
C     OLD FORM CONVERSION
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     REVISED BY ROBERT COLSHER 22 AUG 1986
C                                  IGES DATA ANALYSIS COMPANY
C
C     PURPOSE:
C              TO CONVERT NEW FORM OF IGES OUTPUT TO OLD FORM...
C
C     VARIABLE DECLARATIONS...
C
      CHARACTER * (*) INFILE,OUTFIL
      CHARACTER * 8   BLNK,INARR(20),NEWARR(20)
      CHARACTER * 80  INLINE
      CHARACTER * 160 OUTARR
      INTEGER         ICNT1,ICNT2,ICNT3,IA,IB,IC
      INTEGER         IJ,IK,IL,IT
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=OUTFIL,STATUS='NEW',CARRIAGECONTROL='LIST')
      OPEN(UNIT=10,FILE=INFILE,STATUS='OLD')
C
C     INITIALIZE COUNTERS...
C
      ICNT1 = -1
      ICNT2 = 1
      ICNT3 = 0
C
C     READ THE FILE AND SEPARATE INTO PARTS...
C
```

```
2000  READ(10,2900,END=2090)INLINE
      IF ((INLINE(73:73).EQ.'S').OR.(INLINE(73:73).EQ.'G')) GOTO 2010
      IF (INLINE(73:73).EQ.'T') GOTO 2020
      IF ((INLINE(1:1).EQ.'@').OR.(INLINE(1:1).EQ.'D')) GOTO 2040
C
C     IF IT IS AN C THAN DELETE IT OFF BY READING THE NEXT LINE
C
      IF (INLINE(73:73).EQ.'C') GOTO 2000
C
C     PUT THE PARAMETER DATA INTO FILE4.TMP...
C
      WRITE(4,2910) (INLINE(1:64),ICNT1,'P',ICNT2)
      ICNT2 = ICNT2 + 1
      GOTO 2000
C
C     WRITE HEADER LINES INTO FILE1.TMP...
C
2010  WRITE(1,2900)INLINE
      GOTO 2000
C
C     WRITE TERMINATION LINE INTO FILE2.TMP...
C
2020  WRITE(2,2900)INLINE
      GOTO 2000
C
C     WRITE DIRECTORY ENTRY LINES INTO FILE3.TMP...
C
2030  WRITE(3,2920) (OUTARR(1:8),ICNT2,OUTARR(17:80))
      WRITE(3,2900) OUTARR(81:160)
      ICNT1 = ICNT1 + 2
      GOTO 2000
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
2040  IL = 1
2050  IF (IL.GT.80) GOTO 2000
         IF (INLINE(IL:IL).EQ.';') GOTO 2030
         IF (INLINE(IL:IL).NE.'@') GOTO 2080
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

2060  IK=IK+1
       IF (INLINE(IL+IK:IL+IK).EQ.'@') GOTO 2070
       IF (INLINE(IL+IK:IL+IK).EQ.' ') THEN
        IF (INLINE(IL+IK+1:IL+IK+1).EQ.' ') GOTO 2070
       ENDIF
       IF (INLINE(IL+IK:IL+IK).EQ.';') GOTO 2070
       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 2060
2070  IL=IL+IT
2080  IL = IL + 1
      GOTO 2050
C
C     REWIND ALL FILES BEFORE WE WRITE THEM TO OUTPUT...
C
2090  REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
C
C     WRITE START AND GLOBAL RECORDS TO OUTPUT FILE...
C
2100  READ(1,2900,END=2110) INLINE
      WRITE(9,2900) INLINE
      GOTO 2100
C
```

```
C     WRITE THE DE RECORDS TO OUTPUT FILE.  THEY NOW BECOME RE-FORMATTED...
C
2110  READ(3,2930,END=2140) (INARR(I),I=1,10)
      READ(3,2930) (INARR(I),I=11,20)
      DO 2130 IP=1,20
       IF ((IP.NE.9).AND.(IP.NE.18)) GOTO 2120
       NEWARR(IP)=INARR(IP)
       GOTO 2130
C
C     CHANGE THOSE FIELDS THAT MUST BE RIGHT JUSTIFIED
C
2120   NEWARR(IP)=BLNK(INARR(IP))
2130  CONTINUE
      ICNT3=ICNT3+1
      WRITE(9,2940) ((NEWARR(J),J=1,9),'D',ICNT3)
      ICNT3=ICNT3+1
      WRITE(9,2940) ((NEWARR(J),J=11,19),'D',ICNT3)
      GOTO 2110
C
C     WRITE THE PD LINES TO THE OUTPUT FILE...
C
2140  READ(4,2900,END=2150) INLINE
      WRITE(9,2900) INLINE
      GOTO 2140
C
C     WRITE THE TERMINATE LINES TO THE OUTPUT FILE...
C
2150  READ(2,2900,END=2160) INLINE
      WRITE(9,2900) INLINE
      GOTO 2150
C
C     NOW CLOSE THE FILES AND DELETE THE TEMP ONES...
C
2160  CONTINUE
      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
C
C     FORMATS
C
2900  FORMAT(A80)
2910  FORMAT(A64,1I8,1A1,1I7)
2920  FORMAT(A8,I8,A64)
2930  FORMAT(10A8)
2940  FORMAT(9A8,A,I7)
```

```
C
      END
C*********************************************************************
C
      CHARACTER*(*) FUNCTION BLNK(BUF)
C
C     START FUNCTION BLNK HERE
C
C     WRITTEN BY P. R. KENNICOTT 9-29-83.
C                                GENERAL ELECTRIC CORP. RE. & DEV.
C     RE-WRITTEN BY LEE KLEIN 9-2-84.
C                                GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY LEE KLEIN 8-7-86
C                                GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY ROBERT COLSHER 22 AUG 1986
C                                IGES DATA ANALYSIS COMPANY
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
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*(*) 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 3000 I=1,N
3000   IBUF(M-I+1:M-I+1)=BUF(N-I+1:N-I+1)
      IBUF(1:M-N)=' '
      BLNK=IBUF
      RETURN
      END

C*********************************************************************
C
      SUBROUTINE NEWFRM(INFILE,OUTFIL)
C
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     REVISED BY LEE KLEIN 8-7-86
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY ROBERT COLSHER 22 AUG 1986
C                                   IGES DATA ANALYSIS COMPANY
C
C     PURPOSE:
C              TO CONVERT OLD FORM OF IGES OUTPUT TO NEW FORM...
C
C     VARIABLE DECLARATIONS...
C
      INTEGER PDRCD
      CHARACTER * 80 INLINE
      CHARACTER * (*) INFILE,OUTFIL
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',RECL=80,
     +     ACCESS='DIRECT',FORM='FORMATTED')
      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(1,4900)'C',1
C
C     SEPERATE THE PD AND DE RECORDS, WHILE WRITING G,S, &T LINES
C     TO THE OUTPUT FILE...
C
4000  READ(10,4910,END=4040) INLINE
      IF (INLINE(73:73).EQ.'D') GOTO 4010
      IF (INLINE(73:73).EQ.'P') GOTO 4020
      IF (INLINE(73:73).EQ.'T') GOTO 4030
C
C     WRITE HEADER LINES INTO A TEST.TMP...
C
      WRITE (1,4910) INLINE
      GOTO 4000
C
C     WRITE DIRECTORY LINES INTO FILE3.TMP...
C
4010  WRITE (3,4910) INLINE
      GOTO 4000
C
C     WRITE PARAMETER DATA INTO FILE2.TMP...
C
4020  READ (INLINE(74:80),4920) PDRCD
      WRITE (2,REC=PDRCD,FMT=4910) INLINE
      GOTO 4000
C
C     WRITE TERMINATE RECORD INTO FILE5.TMP...
C
4030  WRITE (7,4910) INLINE

4040  CALL XPD
      REWIND 7

4050  READ(7,4910,END=4060) INLINE
      WRITE (1,4910) INLINE
      GOTO 4050
4060  CALL CMPRES(OUTFIL)
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
C
C     FORMATS
C
```

```
4900  FORMAT(72X,A,I7)
4910  FORMAT(A80)
4920  FORMAT(I7)
C
      END
C*******************************************************************************
C
      SUBROUTINE XPD
C
C     PROGRAM ORIGINALLY WRITTEN BY J. M. SPAETH 7-24-84
C                                  GENERAL ELECTRIC CORP. RE. & DEV.
C     REVISED BY LEE KLEIN 8-7-86
C                                  GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY ROBERT COLSHER 22 AUG 1986
C                                  IGES DATA ANALYSIS COMPANY
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),NUDAT
      CHARACTER * 80  PDLINE,DELIN1,DELIN2
      CHARACTER * 160 NEWDE
      INTEGER         LFLD(20),FLDNUM,FLDBEG,FLDEND
      INTEGER         CHRPTR,NEWPTR,PDPTR,PDCNT
C
C     REWIND THE FILES...
C
      REWIND 3
C
C     INITIALIZE LAST DATA SO AS NOT TO EQUAL NEXT DATA...
C
      DO 5000 FLDNUM=1,20
       LSTDAT(FLDNUM) = 'XXXXXXXX'
5000  CONTINUE
C
C     GET NEW DE RECORD
C
5010  READ(3,5900,END=5100) DELIN1
      READ(3,5900) DELIN2
      READ(DELIN1(9:16),5910) PDPTR
      READ(DELIN2(25:32),5910) PDCNT
      DELIN1(73:73)=' '
      DELIN2(73:73)=' '
C
C     CLEAR OUT THE DE RECORD BUFFER...
C
      NEWDE(1:160) = ' '
```

```
C
C     GET THE DATA FROM EACH FIELD OF THE DE RECORD SET...
C
      FLDBEG = -7
      FLDEND = 0
      DO 5020 FLDNUM=1,10
       FLDBEG=FLDBEG + 8
       FLDEND=FLDEND + 8
       READ(DELIN1(FLDBEG:FLDEND),5920) NEWDAT(FLDNUM)
       READ(DELIN2(FLDBEG:FLDEND),5920) NEWDAT(FLDNUM+10)
5020  CONTINUE
C
C     FIELD 9 MUST BE ZERO FILLED
C
      DO 5030 I = 1,8
      IF (NEWDAT(9)(I:I).EQ.' ')NEWDAT(9)(I:I) = '0'
5030  CONTINUE
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 5050 FLDNUM=1,20
       DO 5040 I=1,8
        IF (NEWDAT(FLDNUM)(I:I).NE.' ') THEN
         LFLD(FLDNUM)= 9 - I
         GOTO 5050
        ENDIF
5040   CONTINUE
5050  CONTINUE
C
C     WRITE THE DE SEQUENCE NUMBER AT THE BEGINNING OF THE OUTPUT DE RECORD
C
      NUDAT=NEWDAT(10)
      ENCODE(LFLD(10)+1,5930,NEWDE(1:LFLD(10)+1)) NUDAT(9-LFLD(10):8)
      CHRPTR = LFLD(10) + 2
C
C     SEARCH NEW DE RECORD SET FOR CHANGED DATA; WHEN FOUND WRITE CHANGED
C     DATA TO OUTPUT DE RECORD...
C
      DO 5060 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 5060
```

```
        IF (NEWDAT(FLDNUM).NE.LSTDAT(FLDNUM)) THEN
         IF (FLDNUM.GT.9) THEN
          ENCODE(4,5940,NEWDE(CHRPTR:CHRPTR+3)) FLDNUM
          CHRPTR=CHRPTR+4
         ELSE
          ENCODE(3,5950,NEWDE(CHRPTR:CHRPTR+2)) FLDNUM
          CHRPTR=CHRPTR+3
         ENDIF
         IF (LFLD(FLDNUM).NE.0) THEN
          IF (FLDNUM.NE.9) THEN
           READ(NEWDAT(FLDNUM)(9-LFLD(FLDNUM):8),5960)
     +          NEWDE(CHRPTR:CHRPTR-1+LFLD(FLDNUM))
           CHRPTR=CHRPTR+LFLD(FLDNUM)
          ELSE
C
C     FIELD 9 IS A SPECIAL CASE...
C
           READ(NEWDAT(9)(1:8),5960) NEWDE(CHRPTR:CHRPTR+7)
           CHRPTR=CHRPTR+8
          ENDIF
         ENDIF
        ENDIF
C
C     STORE DATA FROM CURRENT DE RECORD SET TO COMPARE WITH NEXT SET
C
        LSTDAT(FLDNUM)=NEWDAT(FLDNUM)
5060   CONTINUE
       NEWDE(CHRPTR:CHRPTR) = SCOLN
C
C     IF OUTPUT DE RECORD > 80 CHAR'S, WRITE 2 LINES...
C
       IF (CHRPTR.GT.80) THEN
        DO 5070 I=1,11
         IF (NEWDE(82-I:82-I).EQ.'@') GOTO 5080
5070    CONTINUE
5080    WRITE(1,5970)NEWDE(1:81-I)
        NEWDE(1:80)=NEWDE(82-I:161-I)
       ENDIF
       WRITE(1,5900) NEWDE(1:80)
C
C     ERASE UNNECESSARY DATA FROM PD RECORD AND WRITE TO OUTPUT FILE;
C
       DO 5090 IL=1,PDCNT
       READ (2,5900,REC=PDPTR) PDLINE
       PDPTR = PDPTR+1
       PDLINE(65:80) = ' '
       WRITE(1,5900) PDLINE
5090   PDLINE(1:80) = ' '
C
C     END OF LOOP GET NEXT DE RECORD...
```

```
C
      GOTO 5010
5100  CONTINUE
      RETURN
C
C     FORMATS
C
5900  FORMAT(A80)
5910  FORMAT(I8)
5920  FORMAT(A8)
5930  FORMAT ('D',A)
5940  FORMAT('@',I2,'_')
5950  FORMAT('@',I1,'_')
5960  FORMAT(A)
5970  FORMAT(A<81-I>)
C
      END

C*****************************************************************************
C
      SUBROUTINE CMPRES(OUTFIL)
C
C     START OF SUBROUTINE
C
C     PROGRAM ORIGINALLY WRITTEN BY J. M. SPAETH 7-24-84
C                                   GENERAL ELECTRIC CORP. RE. & DEV.
C     REVISED BY LEE KLEIN 8-7-86
C                                   GENERAL DYNAMICS CAD/CAM POMONA DIV.
C     REVISED BY ROBERT COLSHER 22 AUG 1986
C                                   IGES DATA ANALYSIS COMPANY
C
C     PURPOSE:
C             TO CLEAR AWAY ALL TRAILING BLANKS FROM THE OUTPUT FILE
C
C     VARIABLE DECLARATIONS...
C
      CHARACTER * 80 TEXT
      CHARACTER * (*) OUTFIL
      INTEGER        LENGTH
C
C     REWIND THE INPUT FILE AND OPEN THE OUTPUT FILE
C
      REWIND 1
      OPEN (UNIT=4,NAME=OUTFIL,STATUS='NEW',CARRIAGECONTROL='LIST',
     +      ERR=6000)
      GOTO 6010
C
C     GETS HERE IF THERE IS AN ERROR IN THE OUTPUT FILE NAME...
C
```

```
6000  WRITE(*,6900)'Error in OUTPUT file name. Output written to file',
     +       ' IGES.OUT'
      OPEN (UNIT=4,NAME='IGES.OUT',STATUS='NEW',CARRIAGECONTROL='LIST')
C
C     READ RECORD LINES INTO BUFFER ONE AT A TIME...
C
6010  READ(1,6910,END=6999) TEXT
      LENGTH = 80
C
C     GO THRU EACH LINE DELETING TRAILING BLANKS
C
6020  IF (TEXT(LENGTH:LENGTH).NE.' ') GOTO 6030
      LENGTH=LENGTH-1
      IF (LENGTH.GT.1) GOTO 6020
C
C     WRITE PROCESSED LINES TO THE OUTPUT FILE
C
6030  WRITE(4,6920) TEXT(1:LENGTH)
C
      GOTO 6010
C
6999  CONTINUE
      RETURN
C
C     FORMATS
C
6900  FORMAT(1X,A,A)
6910  FORMAT(A80)
6920  FORMAT(A<LENGTH>)
C
      END
```

Appendix I. Obsolete Entities

I.1 General

The addition of new entities and forms which greatly increase the capability for transfer of specific data constructs has given cause to deprecate other entities and forms published in previous versions of this Specification. The parameter lists for these entities and forms are included herein to provide for interpretation of files created under an earlier version. The new entities or forms which are valid for the previous forms are as follows:

Obsolete Entity/Form		Valid Entity/Form	
402/6	View List	402/3	Views Visible
402/8	Signal String	402/18	Flow
402/10	Text Node	312	Text Template
402/11	Connect Node	132	Connect Point
406/4	Region Fill Property	230	Sectioned Area

In addition, Font Zero for the General Note Entity (Type 212) is obsolete.

The parameter lists for these following entities may have additional parameters as specified in Section 2.2.4.4.2.

I.2 Associativity Instance Entity (Type 402)

(See Section 4.3.3 for additional details about this entity.)

FORM NUMBER: 6 View List

This associativity has two classes. The first class has only one entry which is a pointer to the directory entry of a specific view. The second class is a list of entities (pointers to their respective directory entries) which are visible in the view referenced in Class 1. Back pointers are required in both classes; the view as well as all entities visible in the view must have pointers to this associativity instance.

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1 (View)	
2	1	Back pointers required
3	2	Unordered
4	1	One item per entry
5	1	Pointer to view Directory Entry
	Class 2 (Entities)	
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	

APPENDIX I.2 ASSOCIATIVITY INSTANCE ENTITY (TYPE 402, FORM 8)

FORM NUMBER: 8 Signal String

This associativity has four classes and is intended to represent a single signal string. Class one provides all names of the signal in an order that should be preserved. Class two collects together a set of connection nodes in the string and thus can be considered as specifying the connections for the signal. Class three relates the signal string to a set of geometric entities on a schematic drawing, while class four accomplishes the same thing with respect to the implemented board or chip.

The geometric entities which may be members of classes 2 and 3 include composite, copious (Forms 11 or 12), or any of the entities which may be members of composite.

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

APPENDIX I.2 ASSOCIATIVITY INSTANCE ENTITY (TYPE 402, FORM 8)

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	String	Signal name
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	

APPENDIX I.2 ASSOCIATIVITY INSTANCE ENTITY (TYPE 402, FORM 10)

FORM NUMBER: 10 Text Node

The purpose of the text node is to act as a template for future addition of text. It is defined as an associativity to allow it to refer to multiple instances of itself in those cases in which it is instanced as part of a subfigure definition.

In accordance with the general rule of multiple-instanced entities, digits 5-6 of Directory Entry Field 9 have the value 04, and Class 1 consists of a pointer to a point representing its original location followed by pointers to multiple instances, if these exist.

Class 2 consists of those parameters of the General Note which are pertinent to the definition of a text template, as opposed to text itself. In general, these consist of all parameters but the text string. The location is omitted because it is included in Class 1 as a pointer to a point representing the geometric location of the text node.

An instance of a text node consists of this Associativity, a point indicating the position of the instance, and one or more General Notes attached to the node through the text pointers of the geometric entities. If parameters in the General Notes are null, the value of the same parameter in Class 2 of the associativity is taken as the default; non-null parameters over-ride the defaults. In the cases of multiple instances from a subfigure, the General Notes representing text will be attached to the instance point (pointers 2, 3,... in Class 1).

As a text-type entity, the Text Node can be pointed to by the back pointer/text pointer field in each entity.

Note that the associativity definition has an unusual value for Parameter 11 (Font Characteristic). The value 3 implies either a pointer or a data item. A positive value implies a data item; a negative value implies the absolute value is to be taken as a pointer.

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

APPENDIX I.2 ASSOCIATIVITY INSTANCE ENTITY (TYPE 402, FORM 10)

DESCRIPTION

Directory Data
Entity Type Number: 402
Form Number: 10

Parameter Data

Index	Name	Type	Description
1	NP	Integer	Number of geometry pointers
2	NTD	Integer	Number of Text Descriptions (NTD=1)
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	Font characteristic (default = 1) or pointer
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)

APPENDIX I.2 ASSOCIATIVITY INSTANCE ENTITY (TYPE 402, FORM 11)

FORM NUMBER: 11 Connect Node

The purpose of the Connect Node is to imply a logical connection between one or more entities. In the case of an electrical application, this logical connectivity would mean an electrical connection, but the Connect Node has applicability in other applications such as piping.

The Connect Node is defined as a 2-class associativity with the second class undefined.

In accordance with the general rule of multiple-instanced entities, digits 5-6 of directory entry field 9 have the value 04, and class 1 consists of a pointer to the geometry representing the original location of the Connect Node, followed by pointers to multiple instances, if these exist. Each of the geometry entities is the Point Entity. In the case of a singly-instanced Connect Node, the point represents the position of the Connect Node. In the case of a multiply-instanced Connect Node (*i.e.*, a Connect Node in a Subfigure Definition), the first point in the class represents the defining location (in the Subfigure Definition), while the remaining points represent instance locations of the Connect Node.

The second class is intended to describe the properties of the Connect Node such as physical connection constraints. Its definition will be developed in the future when these requirements become more clear.

The name of a Connect Node is found in its entity label. If the name is longer than 8 characters, the entity label is blank, and the name is found in a Name Property attached to the entity. In the case of multiply-instanced Connect Nodes, separate names can be attached to the instance points by the same means.

DEFINITION

Index	Set Value	Meaning
1	2	Two classes
	Class 1 (Geometry Pointers)	
2	1	Back pointers required
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

APPENDIX I.2 ASSOCIATIVITY INSTANCE ENTITY (TYPE 402, FORM 11)

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

I.3 Property Entity (Type 406)

(See Section 4.3.7 for additional details about this entity.)

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.

Parameter Data

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 Use of Fill Code = 2 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 the Type 230 Sectioned Area Entity.
3	0	Pointer	Obsolete. Note: a previous erroneous 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.

I.4 General Note Font Zero

Font Zero is an obsolete symbol font for the General Note Entity (Type 212) and should not be used. It is included here (see Figure I1) for reference in processing files written in accordance with Version 1.0 of the Specification.

0	⧖	30	λ	60	0	110	H	140	`	170	⍋
1	÷	31	α	61	1	111	I	141	∠	171	⧖
2	≤	32	δ	62	2	112	J	142	⌽	172	Y
3	≥	33	μ	63	3	113	K	143	▱	173	{
4	△	34	π	64	4	114	L	144	⌓	174	|
5	√	35	¯	65	5	115	M	145	○	175	}
6	×	36	±	66	6	116	N	146	//	176	~
7	≡	37	°	67	7	117	O	147	⌀	177	Z
10	≠	40		70	8	120	P	150	↗		
11	∫	41	!	71	9	121	Q	151	⫩		
12	⊃	42	"	72	:	122	R	152	⊕		
13	∨	43	#	73	;	123	S	153	⌒		
14	∧	44	$	74	<	124	T	154	⊥		
15	≈	45	%	75	=	125	U	155	Ⓜ		
16	∑	46	&	76	>	126	V	156	∅		
17	↑	47	´	77	?	127	W	157	○		
20	↓	50	(	100	@	130	W	160	Ⓟ		
21	→	51	)	101	A	131	Y	161	₵		
22	←	52	*	102	B	132	Z	162	◎		
23	ϕ	53	+	103	C	133	[	163	Ⓢ		
24	θ	54	,	104	D	134	\	164	⍞		
25	γ	55	-	105	E	135	]	165	⏲		
26	ψ	56	.	106	F	136	^	166	△		
27	ω	57	/	107	G	137	_	167	⌔		

Figure I1. Obsolete General Note Font Zero

Appendix J. Untested Entities

J.1 Introduction

This appendix contains extensions to Version 3.0 of this Specification [NBS86] which have been approved by the IGES/PDES Organization but have not been sufficiently tested to be included in the main body of the Specification. Implementors should be cautioned that the entities may not work and may be significantly changed based on implementation experience.

When the IGES/PDES Organization deems that sufficient implementation experience has been obtained on a specific extension, it will be moved to its appropriate place in the main body of the Specification.

This appendix serves the same function as "gray pages" in other specifications.

The following entities or entity forms are included in this appendix:

Entity Type Number	Form	Entity Type
0	0	Null
146	0-34	Nodal Results
148	0-34	Element Results
212	All	General Note (OCR-B Text Font) (Drafting Font)
222	1	Radius Dimension (Multiple Leader)
228	1-3	General Symbol (Form Numbers)
322	0-2	Attribute Table Definition
422	0-1	Attribute Table Instance
406	18	Intercharacter Spacing Property

J.2 Null Entity (Type 0)

The Null Entity (Type 0) is intended to be ignored by a processor. It may contain an arbitrary amount of data in its PD data. When encountered by a processor, this entity should be skipped over and not processed.

This entity is useful when editing a file. By changing the entity type number of an entity in a file to 0, one ensures that the entity will not be processed. Thus, the replacement of an entity in a file can easily be done by adding the replacement entity to the end of the DE and PD Sections and changing the replaced entity type number to 0.

When editing a file to create a Null Entity, care should be taken to change both Entity Type Number Fields in the DE Section, as well as the first field of the first PD line.

J.3 Nodal Results Entity (Type 146)

The number of analysis results data values per FEM node and their physical interpretation depends upon specified values of the form number (TYPE) and NV (see Table J1). Also, the node number identifier is equivalent to the node number in the directory entry subscript field of the node entity.

J.3.1 Directory Data
Entity Type Number: 146

```
Entity Label:        Analysis Label (Optional)
Entity Subscript:    Analysis Case Number (Required)
Form Number:         TYPE
```

The value of TYPE (see Table J1) indicates the physical interpretation of the finite element analysis result data. For a specific TYPE of data, multiple values are positioned within the Parameter Data record in the order in which they appear in the parenthetical expression in the description column of the table.

J.3.2 Parameter Data

Index	Name	Type	Description
1	GNOTE	Pointer	Pointer to a General Note that describes the analysis case.
2	SCN	Integer	Analysis Subcase number. If there is no subcase, then the value of this parameter should be zero.
3	TIME	Real	Analysis time value used for this subcase. (This time value is not the time that the analysis was executed, nor does it have anything to do with the amount of time that a computer took to execute the job. It is the time at which transient analysis results occur in the mathematical FEM model.)
4	NV	Integer	Number of real values in array V for a FEM node. (The value of NV must agree with the form number specified in the Directory Data, see Table J1.)
5	NN	Integer	Number of FEM nodes for which data is to be read.
6	NODE(1)	Integer	FEM node number identifier for first node.
7	NP(1)	Pointer	Pointer to FEM node Directory Entry.
8	V(i)	Real	Values of the finite element analysis result data array for the first FEM node. There are NV data values in array V.
	.		
	.		(loop over number of nodes, NN)

In subsequent index equations, let NNV = (NV+2)*(NN-1)

6+NNV	NODE(NN)	Integer	FEM node number identifier for last node.
7+NNV	NP(NN)	Pointer	Pointer to FEM node Directory Entry.
8+NNV	V(i)	Real	Values of the finite element analysis result data array for the last FEM node. There are NV data values in array V.

Additional pointers as required (see Section 2.2.4.4.2).

Table J1. Description of TYPE Numbers for the Nodal and Element Results Entities

Type	NV	Description
0	nv	Unknown/Miscellaneous (The number of values, nv, is not predefined for form type 0. The value of nv must always be positive.)
1	1	Temperature
2	1	Pressure
3	3	Total Displacement (xx, yy, zz - consistent with the Nodal Displacement Coordinate System)
4	6	Total Displacement and Rotation (Dxx, Dyy, Dzz, Rxx, Ryy, Rzz - consistent with the Nodal Displacement Coordinate System)
5	3	Velocity
6	3	Velocity Gradient
7	3	Acceleration
8	3	Flux
9	3	Elemental Force
10	1	Strain Energy
11	1	Strain Energy Density
12	3	Reaction Force
13	1	Kinetic Energy
14	1	Kinetic Energy Density
15	3	Hydrostatic Pressure
16	1	Coefficient of Pressure
17	3	Symmetric 2-Dimensional Elastic Stress Tensor (xx, yy, xy)
18	3	Symmetric 2-Dimensional Total Stress Tensor (xx, yy, xy)
19	3	Symmetric 2-Dimensional Elastic Strain Tensor (xx, yy, xy)
20	3	Symmetric 2-Dimensional Plastic Strain Tensor (xx, yy, xy)
21	3	Symmetric 2-Dimensional Total Strain Tensor (xx, yy, xy)
22	3	Symmetric 2-Dimensional Thermal Strain (xx, yy, xy)
23	6	Symmetric 3-Dimensional Elastic Stress Tensor (xx, yy, zz, xy, yz, zx)
24	6	Symmetric 3-Dimensional Total Stress Tensor (xx, yy, zz, xy, yz, zx)
25	6	Symmetric 3-Dimensional Elastic Strain Tensor (xx, yy, zz, xy, yz, zx)
26	6	Symmetric 3-Dimensional Plastic Strain Tensor (xx, yy, zz, xy, yz, zx)
27	6	Symmetric 3-Dimensional Total Strain Tensor (xx, yy, zz, xy, yz, zx)
28	6	Symmetric 3-Dimensional Thermal Strain (xx, yy, zz, xy, yz, zx)
29	9	General Elastic Stress Tensor (xx, yx, zx, xy, yy, zy, xz, yz, zz)
30	9	General Total Stress Tensor (xx, yx, zx, xy, yy, zy, xz, yz, zz)
31	9	General Elastic Strain Tensor (xx, yx, zx, xy, yy, zy, xz, yz, zz)
32	9	General Plastic Strain Tensor (xx, yx, zx, xy, yy, zy, xz, yz, zz)
33	9	General Total Strain Tensor (xx, yx, zx, xy, yy, zy, xz, yz, zz)
34	9	General Thermal Strain (xx, yx, zx, xy, yy, zy, xz, yz, zz)

J.4 Element Results Entity (Type 148)

The number of result data values depends upon: (1) NV, the number of results data values per reporting location; (2) NRL, the number of result data reporting locations in a FEM element per layer; and (3) NL, the number of layers in the FEM element. The physical interpretation and location of the results data depends upon: (1) TYPE, the type of results data which is specified by using the form number in the Directory Data section (see Table J1); (2) RRF, the results reporting flag which associates results data with FEM element location; and (3) DLF, the data layer flag which specifies the FEM element layer location of the results data.

J.4.1 Directory Data
Entity Type Number: 148

```
Entity Label:        Analysis Label (Optional)
Entity Subscript:    Analysis Case Number (Required)
Form Number:         TYPE
```

The value of TYPE (see Table J1) indicates the physical interpretation of the finite element analysis result data. For a specific TYPE of data, multiple values are positioned within the Parameter Data record in the order in which they appear in the parenthetical expression in the description column of the table.

J.4.2 Parameter Data

Index	Name	Type	Description
1	GNOTE	Pointer	Pointer to a General Note that describes the analysis case.
2	SCN	Integer	Analysis Subcase number. If there is no subcase, then the value of this parameter should be zero.
3	TIME	Real	Analysis time value used for this subcase. (This time value is not the time that the analysis was executed, nor does it have anything to do with the amount of time that a computer took to execute the job. It is the time at which transient analysis results occur in the mathematical model.)
4	NV	Integer	Number of result values per FEM element reporting location. (The value of NV must agree with the form number specified in the Directory Data, see Table J1.)
5	RRF	Integer	Results Reporting Flag. This flag is used to associate the data with a FEM location. The following values are possible: 0 - Indicates that the results data pertains to the FEM element's nodes. 1 - Indicates that the results data pertains to the FEM element's centroid. 2 - Indicates that the results data is constant on all faces and throughout the entire volume of the FEM element. 3 - Indicates that the results data pertains to the FEM element's Gauss points (reserved for future definition).
6	NE	Integer	Number of FEM elements defined in this entity.
7	EN(1)	Integer	FEM element number identifier for first element.
8	EP(1)	Pointer	Pointer to FEM element Directory Entry.
9	ITOP(1)	Integer	Element Topology type of first FEM element.

10	NL(1)	Integer	Number of layers per result data report location. This parameter along with the form number indicates the total number of result values to be read for a particular FEM element.
11	DLF(1)	Integer	Data Layer Flag. This flag indicates other information necessary to interpret the actual layer position of the data. Five values are possible. They are: 0 - Indicates that a layer is not special. (NL must be 1 for this case.) 1 - Indicates the layer is the top surface of a FEM plate element. (NL must be 1 for this case.) 2 - Indicates the layer is the middle surface of a FEM plate element. (NL must be 1 for this case.) 3 - Indicates the layer is the bottom surface of a FEM plate element. (NL must be 1 for this case.) 4 - Indicates the layers are an ordered set of values from the top to the bottom surface of a FEM element. There are NL individual layers.
12	NRL(1)	Integer	Number of result data report locations for first FEM element.
13	RDRL(I)	Integers	The result data report locations for the FEM element. The values of RDRL depends on the results reporting flag, RRF. If RRF is: 0 - These are the node numbers for this FEM element at which result values are reported. There are NRL of them. 1 - This is FEM element centroidal results data. NRL should be 1 and this value should be zero. 2 - This is FEM element constant results data. NRL should be 1 and this value should be zero. 3 - These are a topologically ordered list of gauss points (reserved for future definition). There are NRL values for RDRL.
13+NRL	NUMV(1)	Integer	This value represents the total number of results contained in the following V array. It is the product of NV, NL, and NRL for this FEM element; *i.e.*, for FEM element number one. NUMV(1) = NV*NL(1)*NRL(1)
14+NRL	V(J,K,L)	Reals	The result data values of the FEM analysis for the first FEM element. The result data values are arranged in column major order; *i.e.*, the leftmost subscript changes most rapidly. The subscripts are: (1) J is the value number that is incremented from 1 to NV (see Table J1); (2) K is the layer number that is incremented from 1 to NL(I); and (3) L is the results data report location index that is incremented from 1 to NRL(I). (The subscript I indicates that these values are dependent upon a particular FEM element.) The loop through the V array is done by using the following FORTRAN code fragment.

```
DO 10 L = 1, NRL(I)
    DO 20 K = 1, NL(I)
        DO 30 J = 1, NV
            READ(unit,*)  V(J,K,L)
        30 CONTINUE
    20 CONTINUE
10 CONTINUE
```

There are NUMV values for array V.

.
. (loop over number of elements)
.

In subsequent index equations, let NLS = $\sum$(7+(NL*NV+1)*NRL(I)); where I = 1 to NE-1 and NE represents the number of elements. Also, let NLSE = NLS + NRL(NE).

7+NLS	EN(NE)	Integer	FEM element number identifier for last element.
8+NLS	EP(NE)	Pointer	Pointer to FEM element Directory Entry.
9+NLS	ITOP(NE)	Integer	Element Topology type of last FEM element.
10+NLS	NL(NE)	Integer	Number of layers per result data report location for last FEM element.
11+NLS	DLF(NE)	Integer	Data Layer Flag of last FEM element.
12+NLS	NRL(NE)	Integer	Number of result data report locations for the last FEM element.
13+NLS	RDRL(I)	Integers	The result data location list for the last FEM element.
13+NLSE	NUMV(NE)	Integer	This value represents the total number of results contained in the V array for the last FEM element.
14+NLSE	V(J,K,L)	Reals	The result data values of the FEM element analysis for the last FEM element.

Additional pointers as required (see Section 2.2.4.4.2).

J.5 Additional General Note Fonts

Two additional fonts—Font 19 (OCR-B) and Font 1003 (Drafting Font)—are defined for use by the General Note Entity (Type 212). See Section 4.2.9 for more detail.

Font 19 (OCR-B) [ISO1073] is defined in Figure J1.

Font 1003 (Drafting Font) is defined in Figure J2 and Table J2.

BL		0	0	@	@	P	P	‘	`	p	p
!	!	1	1	A	A	Q	Q	a	a	q	q
"	"	2	2	B	B	R	R	b	b	r	r
#	#	3	3	C	C	S	S	c	c	s	s
$	$	4	4	D	D	T	T	d	d	t	t
%	%	5	5	E	E	U	U	e	e	u	u
&	&	6	6	F	F	V	V	f	f	v	v
’	’	7	7	G	G	W	W	g	g	w	w
(	(	8	8	H	H	X	X	h	h	x	x
)	)	9	9	I	I	Y	Y	i	i	y	y
*	*	:	:	J	J	Z	Z	j	j	z	z
+	+	;	;	K	K	[	[	k	k	{	{
,	,	<	<	L	L	\	\	l	l	|	|
-	-	=	=	M	M	]	]	m	m	}	}
.	.	>	>	N	N	^	^	n	n	~	~
/	/	?	?	O	O	_	_	o	o		

Figure J1. General Note Font 19 (OCR-B)

BL		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	V
(	(	8	8	H	H	X	X	h	↗	x	⊤
)	)	9	9	I	I	Y	Y	i	≒	y	⊳
*	*	:	:	J	J	Z	Z	j	⊕	z	◺
+	+	;	;	K	K	[	[	k	⌒	{	{
,	,	<	<	L	L	\	\	l	Ⓛ	|	|
-	-	=	=	M	M	]	]	m	Ⓜ	}	}
.	.	>	>	N	N	^	⌒	n	∅	~	°
/	/	?	?	O	O	_	_	o	□		

Figure J2. General Note Font 1003 (Drafting Font)

Table J2. Character Names for the Drafting Font

Name	Symbol	Font† 1003
Space		32
Exclamation mark	!	33
Quotation marks	"	34
Pound sign	#	35
Dollar sign	$	36
Percent sign	%	37
Ampersand	&	38
Apostrophe	'	39
Left parenthesis	(	40
Right parenthesis	)	41
Asterisk	*	42
Plus sign	+	43
Comma	,	44
Minus sign/hyphen	-	45
Period	.	46
Slash	/	47
Numeric 0	0	48
Numeric 1	1	49
Numeric 2	2	50
Numeric 3	3	51
Numeric 4	4	52
Numeric 5	5	53
Numeric 6	6	54
Numeric 7	7	55
Numeric 8	8	56
Numeric 9	9	57
Colon	:	58
Semi-colon	;	59
Less than	<	60
Equal sign	=	61
Greater than	>	62
Question mark	?	63
Commercial at	@	64
Upper case letter A	A	65
Upper case letter B	B	66
Upper case letter C	C	67
Upper case letter D	D	68
Upper case letter E	E	69
Upper case letter F	F	70
Upper case letter G	G	71
Upper case letter H	H	72
Upper case letter I	I	73
Upper case letter J	J	74

†Entries are decimal ASCII equivalent

Table J2. Character Names for the Drafting Font (continued)

Name	Symbol	Font† 1003
Upper case letter K	K	75
Upper case letter L	L	76
Upper case letter M	M	77
Upper case letter N	N	78
Upper case letter O	O	79
Upper case letter P	P	80
Upper case letter Q	Q	81
Upper case letter R	R	82
Upper case letter S	S	83
Upper case letter T	T	84
Upper case letter U	U	85
Upper case letter V	V	86
Upper case letter W	W	87
Upper case letter X	X	88
Upper case letter Y	Y	89
Upper case letter Z	Z	90
Left bracket	[	91
Backward slash	\	92
Right bracket	]	93
Arc length	⌒	94
Underscore	_	95
Plus/minus	±	96
Angularity	∠	97
Perpendicularity	⊥	98
Flatness	⏥	99
Profile of a surface	⌓	100
Circularity	○	101
Parallelism	//	102
Cylindricity	⌭	103
Runout	↗	104
Symmetry	⌯	105
Position	⌖	106
Profile of a line	⌒	107
Least material condition	Ⓛ	108
Maximum material condition	Ⓜ	109
Diameter	⌀	110
Square	□	111
Projected tolerance zone	Ⓟ	112
Centerline	℄	113
Concentricity	◎	114
Regardless of feature side	Ⓢ	115

†Entries are decimal ASCII equivalent

Table J2. Character Names for the Drafting Font (continued)

Name	Symbol	Font† 1003
Total runout	⌰	116
Straightness	—	117
Counterbore	⌴	118
Countersink	∨	119
Depth	↧	120
Conical taper	⌲	121
Slope	◺	122
Left brace	{	123
Vertical bar	|	124
Right brace	}	125
Degree symbol	°	126

†Entries are decimal ASCII equivalent

J.6 Radius Dimension Entity (Type 222, Form 1)

This second form of the existing Radius Dimension Entity (Type 222) accounts for the occasional need to have two Leader (Arrow) Entities referenced. An example is shown in Figure J3. See Section 4.2.14 for more detail.

J.6.1 Directory Data

Entity Type Number: 222
Form Number: 1 Multiple Leader Format

J.6.2 Parameter Data

Index	Name	Type	Description
1	DENOTE	Pointer	Pointer to General Note Directory Entry
2	DEARRW1	Pointer	Pointer to first Leader (Arrow) Directory Entry; the arrow head should touch the arc
3	XT	Real	Arc center coordinates
4	YT	Real	
5	DEARRW2	Pointer	Pointer to second Leader (Arrow) Directory Entry or zero

Additional pointers as required (see Section 2.2.4.4.2).

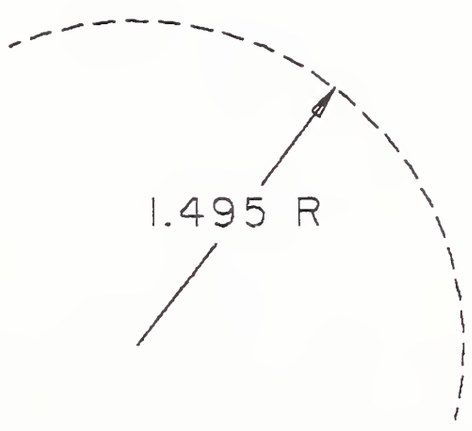


Figure J3. Example Defined Using the Radius Dimension Entity

J.7 General Symbol Entity (Type 228, Forms 1,2,3)

Three additional forms—Form Numbers 1,2, and 3—of the General Symbol Entity are defined below. See Section 4.2.16 for more detail.

Form	Meaning (see [ANSI82])
0	General Symbol - Originated as a symbol which was not necessarily a standard symbol.
1	Datum Feature Symbol - the included data originated as a datum feature symbol consisting of a frame containing the datum identifying letter preceded and followed by a dash. The identifying letter is a letter of the alphabet (except I, O, and Q). Where datum features are so numerous as to exhaust the single alpha series, the double alpha series is used - AA through AZ, BA through BZ, *etc.*
2	Datum Target Symbol - The included data originated as a datum target symbol consisting of a circle divided horizontally into two halves. The lower half contains a letter identifying the associated datum, followed by the target number assigned sequentially starting with one for each datum. Where the target is an area, the area size may be entered into the upper half of the symbol; otherwise, the upper half is blank. A radial line attached to the symbol is directed to the target point, line, or area, as applicable.
3	Feature Control Frame - The included data originated as a feature control frame consisting of a frame divided into compartments containing the geometric characteristic symbol followed by the tolerance. The tolerance may be preceded by a diameter symbol or followed by a material condition symbol.

J.8 Attribute Table Definition Entity (Type 322)

The Attribute Table Definition Entity is designed to support the concept of a well-defined collection of attributes (Section 2.5.8), whether it is a table or a single row of attributes. The entity provides a template for the instance of attribute tables (see Section J.9), or for the combination of template and instance (see Section J.8.2 and Section J.8.3). The entity includes a table name (NAME), and for each attribute, an attribute type (AT), data type (AVDT), and a count (AVC).

Definitions. The following definitions and abbreviations are used in the entity description.

Attribute List Type (ALT). The designated attribute list contains the names (or descriptors) of each attribute type appearing in the attribute table. Within each attribute list the integer numbers representing the attributes must be unique. As an aid to implementors and users, the attribute list also may contain useful supporting information such as suggested units, suggested data types, a footnote for reference, or a range of acceptable values.

Value	Designated List
0	See Property Entity Form 15 for the name of the specific engineering standard that defines the attribute list
1	General attribute list
2	Electrical attribute list (see Table J3)
3	AEC attribute list (see Table J4)
4	Process plant attribute list (see Table J5)
5–5000	other application areas
5001–9999	user defined lists

Attribute Type (AT). Each integer number designates an attribute type defined in the designated attribute list. The number must exist in the list.

Attribute Value Data Type (AVDT). Each attribute has one or more associated value types which may be presented in this entity using one of the following data type indicators (there is no default - a value must be specified):

Value	Data Type
0	No value
1	Integer
2	Real
3	Character string
4	Pointer
5	Not used
6	Logical

Note that these are the same types and are in the same order as the constants described in Section 2.2.2.

Attribute Value Count (AVC). The number of values (Form=0,1) or pairs of values and pointers (Form=2) which follow. The default count is 1. A count of zero implies that values exist and will be recorded at some future time but are currently unknown. In this special case, no values or pairs of values and pointers are required.

Attribute Value (AV). Each attribute contains zero, one or more values as counted by the AVC field. Each AV is specified in the data type field indicated by the current AVDT.

Attribute Value Pointer (AVP). A pointer to a Text Display Template Entity (Type 312) can be associated with each AV. If the pointer contains a non-null value, the AV is displayed by the Text Display Template at either the absolute location given (Form 0), or by combining the increment given (Form 1) with the location of the parent entity (to which this entity is attached, if it is dependent).

J.8.1 Attribute Table Definition (Form 0). This form of the entity is for the definition only of a group of attributes (name, type, and count). It is to be used for the one-to-many case where there will be many instances of a single attribute table definition (the file will contain one or more attribute table instance entities referencing the Attribute Table Definition Entity).

J.8.1.1 Directory Data

Entity Type Number: 322
Form Number: 0

J.8.1.2 Parameter Data

Index	Name	Type	Description
1	NAME	String	Attribute Table name, or comment (default = blank, no name)
2	ALT	Integer	Attribute list type
3	NA	Integer	Number of attributes
	(first attribute definition)		
4	AT(1)	Integer	First attribute type
5	AVDT(1)	Integer	First attribute value data type
6	AVC(1)	Integer	First attribute value count
⋮	⋮		

Let M = 3*NA

	(last attribute definition)		
M+1	AT(NA)	Integer	Last attribute type
M+2	AVDT(NA)	Integer	Last attribute value data type
M+3	AVC(NA)	Integer	Last attribute value count

Additional pointers as required (see Section 2.2.4.4.2).

APPENDIX J.8 ATTRIBUTE TABLE DEFINITION ENTITY (TYPE 322)

J.8.2 Attribute Table Definition (Form 1). This form of the entity is to be used for the one-to-one case where there will be few, or only one, instance of the group of attributes (the attribute values will follow immediately after the attribute definition).

J.8.2.1 Directory Data
Entity Type Number: 322
Form Number: 1

J.8.2.1.1 Parameter Data

Index	Name	Type	Description
1	NAME	String	Attribute Table name, or comment (default=blank, no name)
2	ALT	Integer	Attribute list type
3	NA	Integer	Number of attributes
	(first attribute definition and values)		
4	AT(1)	Integer	First attribute type
5	AVDT(1)	Integer	First attribute value data type
6	AVC(1)	Integer	First attribute value count
7	AV(1,1)	Variable	First attribute value
⋮	⋮		
6+AVC(1)	AV(1,AVC(1))	Variable	Last attribute value
⋮	⋮		

Let M = 3*NA + AVC(1) + ··· + AVC(NA-1)

	(last attribute definition and values)		
M+1	AT(NA)	Integer	Last attribute type
M+2	AVDT(NA)	Integer	Last attribute value data type
M+3	AVC(NA)	Integer	Last attribute value count
M+4	AV(NA,1)	Variable	First attribute value
⋮	⋮		
M+4+AVC(NA)	AV(NA,AVC(NA))	Variable	Last attribute value

Additional pointers as required (see Section 2.2.4.4.2).

J.8.3 Attribute Table Definition (Form 2). This form is similar to Form 1 with the addition of a pointer to a Text Display Template Entity following each attribute value.

J.8.3.1 Directory Data
Entity Type Number: 322
Form Number: 2

J.8.3.2 Parameter Data

Index	Name	Type	Description
1	NAME	String	Attribute Table name, or comment (default=blank, no name)
2	ALT	Integer	Attribute list type
3	NA	Integer	Number of attributes
	(first attribute definition)		
4	AT(1)	Integer	First attribute type
5	AVDT(1)	Integer	First attribute value data type
6	AVC(1)	Integer	First attribute value count
7	AV(1,1)	Variable	First attribute value
8	AVP(1,1)	Pointer	Pointer to Text Display Template
⋮	⋮		
6+AVC(1)	AV(1,AVC(1))	Variable	Last attribute value
7+AVC(1)	AVP(1,AVC(1))	Variable	Pointer to Text Display Template
⋮	⋮		
Let M = 3*NA + 2*(AVC(1) + ··· + AVC(NA-1))			
	(last attribute definition)		
M+1	AT(NA)	Integer	Last attribute type
M+2	AVDT(NA)	Integer	Last attribute value data type
M+3	AVC(NA)	Integer	Last attribute value count
M+4	AV(NA,1)	Variable	First attribute value
M+5	AVP(NA,1)	Pointer	Pointer to Text Display Template
⋮	⋮		
M+4+AVC(NA)	AV(NA,AVC(NA))	Variable	Last attribute value
M+5+AVC(NA)	AVP(NA,AVC(NA))	Variable	Pointer to Text Display Template

Additional pointers as required (see Section 2.2.4.4.2).

J.9 Attribute Table Instance Entity (Type 422)

Each occurrence of an Attribute Table (Type 322, Form 0) is represented by an Attribute Table Instance Entity (Type 422). Directory Entry Field 3 (Structure) of each instance contains a negated pointer to the Directory Entry of its corresponding Attribute Table Definition Entity. All forms of this entity may be independent (not necessarily attached to another entity), or dependent ("pointed at" by other entities as a property would be) See Section 2.2.4.4.2 for more details.

J.9.1 Attribute Table Instance (Form 0). This form of the entity is for a an instance of a single row or tuple.

J.9.1.1 Directory Data

Entity Type Number: 422
Form Number: 0

J.9.1.2 Parameter Data

Index	Name	Type	Description
	(first attribute instance)		
1	AV(1,1)	Variable	First attribute value
2	AV(1,2)	Variable	Second attribute value
⋮	⋮		
AVC(1)	AV(1,AVC(1))	Variable	Last attribute value
Let M = AVC(1) + ··· + AVC(NA-1)			
	(last attribute instance)		
M+1	AV(NA,1)	Variable	First attribute value
M+2	AV(NA,2)	Variable	Second attribute value
⋮	⋮		
M+AVC(NA)	AV(NA,AVC(NA))	Variable	Last attribute value

Additional pointers as required (see Section 2.2.4.4.2).

J.9.2 Attribute Table Instance (Form 1). This form of the entity is for a table of attributes in row-major order; that is, the values for the first attribute through the last attribute for the first row are followed by the values for the first attribute through the last attribute for the second row, *etc.*

J.9.2.1 Directory Data

Entity Type Number: 422
Form Number: 1

J.9.2.2 Parameter Data

Index	Name	Type	Description
	(attributes, first row, first column)		
1	NR	Integer	Number of rows
2	AV(1,1,1)	Variable	First attribute value
3	AV(1,1,2)	Variable	Second attribute value
⋮	⋮		
1+AVC(1)	AV(1,1,AVC(1))	Variable	Last attribute value
Let M = 1+ AVC(1) + ··· + AVC(NA-1))			
	(attributes, first row, last column)		
M+1	AV(1,NA,1)	Variable	First attribute value
M+2	AV(1,NA,2)	Variable	Second attribute value
⋮	⋮		
M+AVC(NA)	AV(1,NA,AVC(NA))	Variable	Last attribute value
Let M = 1 + NR*(AVC(1) + ··· + AVC(NA)) - AVC(NA)			
	(attributes, last row, last column)		
M+1	AV(NR,NA,1)	Variable	First attribute value
M+2	AV(NR,NA,2)	Variable	Second attribute value
⋮	⋮		
M+AVC(NA)	AV(NR,NA,AVC(NA))	Variable	Last attribute value

Additional pointers as required (see Section 2.2.4.4.2).

Table J3. Electrical Attribute List (ALT=2)

No.	Definition	Symbol	Unit	Type	Ref.
1	Access time, memory		second	Real	[IEEE84]
2	Accuracy (percent of full scale)			Real	[IEEE84]
3	Ambient air temperature	T_a	C	Real	[MIL195]
4	Amplification	A		Real	[IEEE84]
5	Amplification Factor	A_f		Real	[IEEE84]
6	Angle (of a waveform)		degree	Real	[IEEE84]
7	Anode voltage	V_a	volt	Real	[IEEE84]
8	Async. input pulse width, minimum	A_{PW}	second	Real	[MIL133]
9	Automatic gain control	AGC		Real	[MIL133]
10	Average forward-current rating		ampere	Real	[MIL195]
11	Average gate power dissipation	W_{gpd}	watt	Real	[MIL195]
12	Average gate-power-dissipation rating		watt	Real	[MIL195]
13	Backlash		degree	Real	[IEEE84]
14	Bandwidth	BW	hertz	Real	[MIL133]
15	Base resistance	R_B	ohm	Real	[IEEE84]
16	Base current	I_B	ampere	Real	[MIL195]
17	Base current, instantaneous total	i_B	ampere	Real	[MIL195]
18	Base spreading resistance	r_b	ohm	Real	[MIL195]
19	Base supply voltage	V_{BB}	volt	Real	[MIL195]
20	Base to emitter voltage	V_B	volt	Real	[MIL195]
21	Board thickness		inch	Real	[IPCT85]
22	Breakdown voltage	V_{BR}	volt	Real	[MIL133]
23	Capacitance	C	farad	Real	[IEEE84]
24	Carrier frequency	C_f	hertz	Real	[IEEE84]
25	Case temperature	T_C	C	Real	[MIL195]
26	Cathode current	I_c	ampere	Real	[IEEE84]
27	Circuit commutated turn-off time		second	Real	[MIL195]
28	Clearance		inch	Real	[IEEE84]
29	Clock level transition time	t_{TC}	second	Real	[MIL133]
30	Clock pulse width, minimum	C_{PW}	second	Real	[MIL133]
31	Clock repetition rate	C_{RR}	hertz	Real	[MIL133]
32	Collector current	I_C	ampere	Real	[MIL195]
33	Collector current, instantaneous total	i_C	ampere	Real	[MIL195]
34	Collector cut-off current	I_{CEX}	ampere	Real	[MIL133]
35	Collector efficiency (percent)			Real	[MIL195]
36	Collector power dissipation	P_C	watt	Real	[MIL195]
37	Collector to base voltage	V_{CB}	volt	Real	[MIL195]
38	Collector to emitter voltage	V_{CE}	volt	Real	[MIL195]
39	Common-mode input voltage	V_{ICR}	volt	Real	[MIL133]
40	Common-mode output voltage	V_{OC}	volt	Real	[MIL133]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
41	Common-mode rejection ratio	C_{MRR}		Real	[MIL133]
42	Common-mode voltage amplification	A_{VC}		Real	[MIL133]
43	Conductance	G	siemens	Real	[IEEE84]
44	Conductor spacing		inch	Real	[IPCT85]
45	Conductor width		inch	Real	[IPCT85]
46	Conversion efficiency (percent)			Real	[MIL195]
47	Conversion time, analog-to-digital		second	Real	[IEEE84]
48	Coupling coefficient			Real	[IEEE84]
49	Critical anode voltage		volt	Real	[IEEE84]
50	Current	I	ampere	Real	[IEEE84]
51	Current capacity	I_c	ampere	Real	[IEEE84]
52	Current limit	I_l	ampere	Real	[IEEE84]
53	Current rating	I_r	ampere	Real	[IEEE84]
54	Cut-off grid voltage	$V_{gc(off)}$	volt	Real	[IEEE84]
55	Cutoff current	I_c	ampere	Real	[MIL195]
56	Data rate		hertz	Real	[IEEE84]
57	Delay time	t_d	second	Real	[MIL195]
58	Dielectric constant	K		Real	[IEEE84]
59	Dielectric strength		volt/inch	Real	[IEEE84]
60	Differential control voltage		volt	Real	[IEEE84]
61	Differential input impedance	Z_{id}	ohm	Real	[MIL133]
62	Differential output impedance	Z_{od}	ohm	Real	[MIL133]
63	Differential output voltage		volt	Real	[IEEE84]
64	Differential voltage amplification	A_{vd}		Real	[MIL133]
65	Distortion (percent)			Real	[IEEE84]
66	Drain currrent	I_D	ampere	Real	[MIL195]
67	Drain cutoff current	$I_{D(off)}$	ampere	Real	[MIL195]
68	Drain supply voltage	V_{DD}	volt	Real	[MIL195]
69	Drain to gate voltage	V_{DG}	volt	Real	[MIL195]
70	Drain to source voltage	V_{DS}	volt	Real	[MIL195]
71	Drain to substrate voltage	V_{Ds}	volt	Real	[MIL195]
72	Drift (percent of full scale)			Real	[IEEE84]
73	Driving point impedance	Z_{dp}	ohm	Real	[IEEE84]
74	Dynamic impedance	Z_D	ohm	Real	[MIL195]
75	Efficiency (percent)			Real	[IEEE84]
76	Emission current	I_e	ampere	Real	[IEEE84]
77	Emission efficiency (percent)			Real	[IEEE84]
78	Emitter current	I_E	ampere	Real	[MIL195]
79	Emitter current, instantaneous total	i_E	ampere	Real	[MIL195]
80	Emitter supply voltage	V_{EE}	volt	Real	[MIL195]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
81	Emitter to base voltage	V_{EB}	volt	Real	[MIL195]
82	Emitter to collector voltage	V_{EC}	volt	Real	[MIL195]
83	External base resistance	R_B	ohm	Real	[MIL195]
84	External collector resistance	R_C	ohm	Real	[MIL195]
85	External emitter resistance	R_E	ohm	Real	[MIL195]
86	Extrapolated unity gain frequency	f_T	hertz	Real	[MIL195]
87	Failure rate			Real	[IEEE84]
88	Fall time	t_f	second	Real	[MIL195]
89	Field emission				[IEEE84]
90	Figure of merit				[IEEE84]
91	Filament voltage	V_F	volt	Real	[IEEE84]
92	Floating potential	V_{FP}	volt	Real	[MIL195]
93	Forward blocking voltage	V_{FBO}	volt	Real	[MIL195]
94	Forward breakover current	I_{FBR}	ampere	Real	[MIL195]
95	Forward breakover voltage	V_{FBR}	volt	Real	[MIL195]
96	Forward current	I_F	ampere	Real	[MIL195]
97	Forward current, overload	$I_{F(OV)}$	ampere	Real	[MIL195]
98	Forward current, surge peak	I_{FSM}	ampere	Real	[MIL195]
99	Forward gate current	I_{GF}	ampere	Real	[MIL195]
100	Forward gate-to-source breakdown voltage	V_{FBRGS}	volt	Real	[MIL195]
101	Forward gate-to-source voltage	V_{FGS}	volt	Real	[MIL195]
102	Forward power dissipation	P_F	watt	Real	[MIL195]
103	Forward recovery time	t_{fr}	second	Real	[MIL195]
104	Forward voltage	V_F	volt	Real	[MIL195]
105	Frequency	F	hertz	Real	[IEEE84]
106	Gain			Real	[IEEE84]
107	Gate controlled turn-off time	$t_{gc(off)}$	second	Real	[MIL195]
108	Gate current	I_G	ampere	Real	[MIL195]
109	Gate supply voltage	V_{GG}	volt	Real	[MIL195]
110	Gate to source voltage	V_{GS}	volt	Real	[MIL195]
111	Gate trigger current	I_{gt}	ampere	Real	[MIL195]
112	Gate trigger voltage	V_{gt}	volt	Real	[MIL195]
113	Gate turn-off current	$I_{gt(off)}$	ampere	Real	[MIL195]
114	Gate turn-off voltage	$V_{gt(off)}$	volt	Real	[MIL195]
115	Gate voltage	V_g	volt	Real	[MIL195]
116	Gate-to-source cutoff voltage	$V_{gs(off)}$	volt	Real	[MIL195]
117	Gate-to-source threshold voltage	V_{gst}	volt	Real	[MIL195]
118	Gate-to-source voltage	V_{gs}	volt	Real	[MIL195]
119	Grid control ratio			Real	[IEEE84]
120	Grid current	I_g	ampere	Real	[IEEE84]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
121	Grid voltage	V_g	volt	Real	[IEEE84]
122	High clock level	V_{CH}	volt	Real	[MIL133]
123	High level input current	I_{IH}	ampere	Real	[MIL133]
124	High level node input current	I_{INH}	ampere	Real	[MIL133]
125	High level node input voltage	V_{INH}	volt	Real	[MIL133]
126	High level output current	I_{OH}	ampere	Real	[MIL133]
127	High level output voltage	V_{OH}	volt	Real	[MIL133]
128	High level supply current drain	I_{CCH}	ampere	Real	[MIL133]
129	Holding current	I_H	ampere	Real	[MIL195]
130	Impedance	Z	ohm	Real	[IEEE84]
131	Incremental resistance	R_{inc}	ohm	Real	[IEEE84]
132	Inductance	L	henry	Real	[IEEE84]
133	Input bias current	I_{IB}	ampere	Real	[MIL133]
134	Input bias current temp. sensitivity	$I_{IB/T}$	amp/deg C	Real	[MIL133]
135	Input impedance	Z_i	ohm	Real	[IEEE84]
136	Input offset current	I_{IO}	ampere	Real	[MIL133]
137	Input offset current temp. sensitivity	$I_{IO/T}$	amp/deg C	Real	[MIL133]
138	Input offset voltage	V_{IO}	volt	Real	[MIL133]
139	Input offset voltage temp. sensitivity	$V_{IO/T}$	volt/deg C	Real	[MIL133]
140	Input signals timing relationships	I_{TR}		Real	[MIL133]
141	Insulation resistance	R_i	ohm	Real	[IEEE84]
142	Interelectrode capacitance	C_{IE}	farad	Real	[IEEE84]
143	Interrupting current	I_i	ampere	Real	[IEEE84]
144	Ionization time		second	Real	[IEEE84]
145	Junction temperature	T_J	deg C	Real	[MIL195]
146	Knee impedance	Z_k	ohm	Real	[MIL195]
147	Knee voltage	V_K	volt	Real	[MIL195]
148	Latching current	I_l	ampere	Real	[MIL195]
149	Leakage current	I_{lk}	ampere	Real	[IEEE84]
150	Limiting current	I_L	ampere	Real	[MIL195]
151	Limiting voltage	V_L	volt	Real	[MIL195]
152	Load immittance			Real	[MIL195]
153	Logic level high	H		Real	[IEEE84]
154	Logic level low	L		Real	[IEEE84]
155	Low clock level	V_{CL}	volt	Real	[MIL133]
156	Low level input current	I_{IL}	ampere	Real	[MIL133]
157	Low level node input current	I_{INL}	ampere	Real	[MIL133]
158	Low level node input voltage	V_{INL}	volt	Real	[MIL133]
159	Low level output current	I_{OL}	ampere	Real	[MIL133]
160	Low level output voltage	V_{OL}	volt	Real	[MIL133]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
161	Low level supply current drain	I_{CCL}	ampere	Real	[MIL133]
162	Luminous energy	c_d		Real	[MIL195]
163	Material			String	[IEEE84]
164	Maximum frequency of oscillation	f_{max}	hertz	Real	[MIL195]
165	Maximum surge current, nonrepetitive	I_{smax}	ampere	Real	[MIL195]
166	Max. surge-current rating, non-repetitive		ampere	Real	[MIL195]
167	Minimum on-voltage	V_{min}	volt	Real	[MIL195]
168	Mode of operation			String	[IEEE84]
169	Moisture-resistant			Logical	[IEEE84]
170	Mutual impedance	Z_m	ohm	Real	[IEEE84]
171	Mutual conductance	G_m	siemens	Real	[IEEE84]
172	N-channel field-effect transistor	N_{fet}		Real	[MIL195]
173	Negative logic			Logical	[IEEE84]
174	Noise figure	N_F	dB	Real	[MIL133]
175	Noise margin	V_N	volt	Real	[MIL133]
176	Noise temperature	T_N	deg C	Real	[MIL195]
177	Nonlinearity (percent of full scale)			Real	[IEEE84]
178	Nonrepetitive peak off-state voltage	V_{DSM}	volt	Real	[MIL195]
179	Nonrepetitive peak reverse current	I_{RSM}	ampere	Real	[MIL195]
180	Nonrepetitive peak reverse voltage	V_{RSM}	volt	Real	[MIL195]
181	Off-state current	I_D	ampere	Real	[MIL195]
182	Off-state voltage	V_D	volt	Real	[MIL195]
183	Offset error (percent of full range)			Real	[IEEE84]
184	On-state current	I_T	ampere	Real	[MIL195]
185	On-state drain current	$I_{D(on)}$	ampere	Real	[MIL195]
186	On-state drain-to-source voltage	$V_{DS(on)}$	volt	Real	[MIL195]
187	On-state voltage	V_T	volt	Real	[MIL195]
188	Open loop gain	A_v		Real	[IEEE84]
189	Operating temperature	T_{op}	deg C	Real	[MIL195]
190	Operator (the person's name or initials)			String	[IEEE84]
191	Orientation			String	[IEEE84]
192	Output impedance	Z_o	ohm	Real	[IEEE84]
193	Output leakage current high	i_{ozh}	ampere	Real	[IEEE84]
194	Output current		ampere	Real	[IEEE84]
195	Output frequency	F_o	hertz	Real	[IEEE84]
196	Output frequency high	F_{oh}	hertz	Real	[IEEE84]
197	Output frequency low	F_{ol}	hertz	Real	[IEEE84]
198	Output leakage current low	i_{ozl}	ampere	Real	[IEEE84]
199	Output offset voltage	V_{OO}	volt	Real	[MIL133]
200	Output polarity			String	[IEEE84]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
201	Output pulse duration	t_{op}	second	Real	[IEEE84]
202	Output pulse duration high	t_{oph}	second	Real	[IEEE84]
203	Output pulse duration low	t_{opl}	second	Real	[IEEE84]
204	Output short circuit current	I_{OS}	ampere	Real	[MIL133]
205	Output swing bandwidth, maximum	B_{OM}	hertz	Real	[MIL133]
206	Output type			String	[IEEE84]
207	Output voltage	V_o	volt	Real	[IEEE84]
208	Output voltage high	V_{oh}	volt	Real	[IEEE84]
209	Output voltage low	V_{ol}	volt	Real	[IEEE84]
210	Output voltage regulation ((high-low)/nominal, percent)			Real	[IEEE84]
211	Output voltage sensitivity to temperature		$(degC)^{-1}$	Real	[IEEE84]
212	Output voltage swing, maximum	V_{OPP}	volt	Real	[MIL133]
213	Outputs			String	[IEEE84]
214	Over-voltage sense			String	[IEEE84]
215	Overall average noise figure	N_{foa}		Real	[MIL195]
216	Overload recovery time	t_{or}	second	Real	[MIL133]
217	P-channel field-effect transistor	P_{fet}		Real	[MIL195]
218	Parallel entry			String	[IEEE84]
219	Parametric electrical test required			Logical	[IEEE84]
220	Peak current rating		ampere	Real	[IEEE84]
221	Peak forward-blocking voltage rating		volt	Real	[MIL195]
222	Peak forward-current rating, repetitive		volt	Real	[MIL195]
223	Peak forward-voltage rating		volt	Real	[MIL195]
224	Peak gate current	I_{GP}	ampere	Real	[MIL195]
225	Peak gate power dissipation	W_{GP}	watt	Real	[MIL195]
226	Peak gate voltage	V_{GP}	volt	Real	[MIL195]
227	Peak gate-current rating		ampere	Real	[MIL195]
228	Peak gate-power-dissipation rating		watt	Real	[MIL195]
229	Peak gate-voltage rating		volt	Real	[MIL195]
230	Peak repetitive on-state current	I_p	ampere	Real	[MIL195]
231	Phase margin	ϕ_m	degree	Real	[MIL133]
232	Phase shift	ϕ_s	degree	Real	[IEEE84]
233	Photoelectric current	I_λ	ampere	Real	[IEEE84]
234	Pin size		inch	Real	[IEEE84]
235	Plate efficiency (percent)			Real	[IEEE84]
236	Polarity			String	[IEEE84]
237	Polarity on stud			String	[IEEE84]
238	Positive logic			Logical	[IEEE84]
239	Power	W	watt	Real	[IEEE84]
240	Power dissipation	P_D	watt	Real	[MIL133]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
241	Power gain	G_p	dB	Real	[MIL133]
242	Power per diode	W_D	watt	Real	[IEEE84]
243	Power rating		watt	Real	[IEEE84]
244	Power supply high	W_{sh}	volt	Real	[IEEE84]
245	Power supply low	W_{sl}	volt	Real	[IEEE84]
246	Power supply rejection ratio	P_{SRR}		Real	[MIL133]
247	Procurement spec # 1 MIL-M-38510/xxx			String	[IEEE84]
248	Procurement spec # 2 desc xxxxx			String	[IEEE84]
249	Programmable			Logical	[IEEE84]
250	Programmable gain			Real	[IEEE84]
251	Propagation delay time		second	Real	[IEEE84]
252	Propagation delay time, high to low level	t_{PHL}	second	Real	[MIL133]
253	Propagation delay time, low to high level	t_{PLH}	second	Real	[MIL133]
254	Propagation delay to output high	t_{pdh}	second	Real	[IEEE84]
255	Propagation delay to output low	t_{pdl}	second	Real	[IEEE84]
256	Protocol			String	[IEEE84]
257	Pulse storage time	t_{ps}	second	Real	[MIL195]
258	Pulse time	t_p	second	Real	[MIL195]
259	Pulse width	t_w	second	Real	[IEEE84]
260	Quiescent input voltage	V_I	volt	Real	[MIL133]
261	Quiescent output voltage	V_O	volt	Real	[MIL133]
262	Radiant energy	J_ω	joule	Real	[MIL195]
263	Radiation hardened			Logical	[IEEE84]
264	Reach-through voltage		volt	Real	[MIL195]
265	Reactance	X	ohm	Real	[IEEE84]
266	Receiver input impedance	Z_i	ohm	Real	[IEEE84]
267	Rectification efficiency (percent)			Real	[MIL195]
268	Rectified voltage	V_{rect}	volt	Real	[IEEE84]
269	Register number			String	[IEEE84]
270	Register types			String	[IEEE84]
271	Regulator current	I_S	ampere	Real	[MIL195]
272	Regulator voltage	V_S	volt	Real	[MIL195]
273	Regulator impedance	Z_s	ohm	Real	[MIL195]
274	Repetitive peak reverse voltage	V_{RRM}	volt	Real	[MIL195]
275	Repetitive peak off-state current	I_{DRM}	ampere	Real	[MIL195]
276	Repetitive peak off-state voltage	V_{DRM}	volt	Real	[MIL195]
277	Repetitive peak on-state current	I_{TRM}	ampere	Real	[MIL195]
278	Repetitive peak reverse voltage	V_{RRM}	volt	Real	[MIL195]
279	Repetitive peak-reverse voltage rating		volt	Real	[MIL195]
280	Resettable			Logical	[IEEE84]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
281	Resistance	R	ohm	Real	[IEEE84]
282	Resistance tolerance, high (percent)			Real	[IEEE84]
283	Resistance tolerance, low (percent)			Real	[IEEE84]
284	Resistance value (measured)		ohm	Real	[IEEE84]
285	Resistivity (in ohms/square)	Ω	ohm	Real	[IEEE84]
286	Resistor construction class			String	[IEEE84]
287	Resolution in bits			Integer	[IEEE84]
288	Response time	t_r	second	Real	[IEEE84]
289	Resultant carry			String	[IEEE84]
290	Reverse blocking current	I_{RBO}	ampere	Real	[MIL195]
291	Reverse breakdown current	I_{rb}	ampere	Real	[MIL195]
292	Reverse breakdown voltage	V_{rb}	volt	Real	[MIL195]
293	Reverse current	I_R	ampere	Real	[MIL195]
294	Reverse current, repetitive peak	I_{RRM}	ampere	Real	[MIL195]
295	Reverse current, surge peak	I_{RSM}	ampere	Real	[MIL195]
296	Reverse gate current	I_{GR}	ampere	Real	[MIL195]
297	Reverse gate-to-source voltage	V_{SG}	volt	Real	[MIL195]
298	Reverse power dissipation	P_R	watt	Real	[MIL195]
299	Reverse recovery current	$I_{RM(REC)}$	ampere	Real	[MIL195]
300	Reverse recovery time	t_{rr}	second	Real	[MIL195]
301	Reverse voltage	V_R	volt	Real	[MIL195]
302	Rise time	t_r	second	Real	[MIL195]
303	Saturation current	I_{sat}	ampere	Real	[IEEE84]
304	Saturation resistance	R_{sat}	ohm	Real	[MIL195]
305	Saturation voltage	V_{sat}	volt	Real	[MIL195]
306	Screening test			String	[IEEE84]
307	Sealed			String	[IEEE84]
308	Settling time	t_s	second	Real	[IEEE84]
309	Settling time high	t_{sh}	second	Real	[IEEE84]
310	Settling time low	t_{sl}	second	Real	[IEEE84]
311	Settling time to p-percent of final max.	t_{pmax}	second	Real	[IEEE84]
312	Settling time to p-percent of final min.	t_{pmin}	second	Real	[IEEE84]
313	Shaft diameter		inch	Real	[IEEE84]
314	Shift direction (e.g. 4Hleft)			String	[IEEE84]
315	Shift out clock frequency	F_{shift}	hertz	Real	[IEEE84]
316	Short circuit			String	[MIL195]
317	Signal to noise ratio	SNR	dB	Real	[IEEE84]
318	Single-ended input impedance	Z_{is}	ohm	Real	[MIL133]
319	Single-ended input voltage	V_{ISR}	volt	Real	[MIL133]
320	Single-ended output impedance	Z_{os}	ohm	Real	[MIL133]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
321	Single-ended voltage amplification	A_{vs}		Real	[MIL133]
322	Slew rate	S_R	volt/second	Real	[MIL133]
323	Small-signal breakdown impedance	z_{BR}	ohm	Real	[MIL195]
324	Small-signal forward impedance	z_f	ohm	Real	[MIL195]
325	Small-signal resistance	r	ohm	Real	[MIL195]
326	Soft start			String	[IEEE84]
327	Solderability			Logical	[IEEE84]
328	Source current	I_S	ampere	Real	[MIL195]
329	Source cutoff current	$I_{S(off)}$	ampere	Real	[MIL195]
330	Source supply voltage	V_{SS}	volt	Real	[MIL195]
331	Source terminal			Real	[MIL195]
332	Source-to-substrate voltage	V_{ss}	volt	Real	[MIL195]
333	Space charge density			Real	[IEEE84]
334	Standard reference			Real	[MIL195]
335	Standing wave ratio	SWR		Real	[MIL195]
336	Static drain-to-source on-state resistance	$R_{sds(on)}$	ohm	Real	[MIL195]
337	Static forward current transfer-ratio	I_{sft}		Real	[MIL195]
338	Static input resistance	R_{in}	ohm	Real	[MIL195]
339	Static transconductance	G_M	siemens	Real	[MIL195]
340	Storage temperature	T_{STG}	deg C	Real	[MIL195]
341	Storage time	t_s	second	Real	[MIL195]
342	Stored charge	Q_S	coulomb	Real	[MIL195]
343	Strip force		lbf	Real	[IEEE84]
344	Strobing input currents	I_{st}	ampere	Real	[IEEE84]
345	Strobing pulse width	t_{st}	hertz	Real	[IEEE84]
346	Stud torque		inch-lbf	Real	[IEEE84]
347	Subcarrier			Real	[IEEE84]
348	Substrate size length		inch	Real	[IEEE84]
349	Substrate size width		inch	Real	[IEEE84]
350	Substrate terminal			Real	[MIL195]
351	Supply voltage	V_{CC}	volt	Real	[MIL133]
352	Surge current	I_S	ampere	Real	[MIL195]
353	Surge, on-state current	I_{TSM}	ampere	Real	[MIL195]
354	Symbol function			String	[IEEE84]
355	Temperature	T	deg C	Real	[IEEE84]
356	Temperature coefficient			Real	[IEEE84]
357	Terminal type			String	[IEEE84]
358	Thermal equilibrium			Real	[MIL195]
359	Thermal noise			Real	
360	Thermal resistance	R_θ	ohm	Real	[MIL195]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
361	Thermal resistance, case to ambient	$R_{\theta CA}$	ohm	Real	[MIL195]
362	Thermal resistance, junction to ambient	$R_{\theta JA}$	ohm	Real	[MIL195]
363	Thermal resistance, junction to case	$R_{\theta JC}$	ohm	Real	[MIL195]
364	Thermal resistance, junction to lead	$R_{\theta jl}$	ohm	Real	[MIL195]
365	Thermal resistance, junction to reference	$R_{\theta JR}$	ohm	Real	[MIL195]
366	Threshold current	I_{th}	ampere	Real	[IEEE84]
367	Threshold current high	I_{thh}	ampere	Real	[IEEE84]
368	Threshold current low	I_{thl}	ampere	Real	[IEEE84]
369	Threshold voltage	V_{th}	volt	Real	[IEEE84]
370	Threshold voltage high	V_{thh}	volt	Real	[IEEE84]
371	Threshold voltage low	V_{thl}	volt	Real	[IEEE84]
372	Tolerance			Real	[IEEE84]
373	Tolerance at d deg C (percent)			Real	[IEEE84]
374	Total duration	t_{td}	second	Real	[IEEE84]
375	Total harmonic distortion (percent)	THD		Real	[MIL133]
376	Total power dissipation, all terminals	P_T	watt	Real	[MIL195]
377	Transducer power gain	G_T	dB	Real	[MIL133]
378	Transient response	T_R	second	Real	[MIL133]
379	Transition duration		second	Real	[IEEE84]
380	Transition time, high to low level	t_{THL}	second	Real	[MIL133]
381	Transition time, low to high level	t_{TLH}	second	Real	[MIL133]
382	Turn off time	t_{off}	second	Real	[MIL195]
383	Turn on time	t_{on}	second	Real	[MIL195]
384	Turn-off delay time	$t_{d(off)}$	second	Real	[MIL195]
385	Turn-on delay time	$t_{d(on)}$	second	Real	[MIL195]
386	Unbalance voltage	V_{OU}	volt	Real	[MIL133]
387	Under-voltage protection			String	[IEEE84]
388	Under-voltage release			String	[IEEE84]
389	Units			String	[IEEE84]
390	Unity gain bandwidth	G_b	hertz	Real	[IEEE84]
391	Unity gain bandwidth, maximum	G_{bmax}	hertz	Real	[IEEE84]
392	Unity gain bandwidth, minimum	G_{bmin}	hertz	Real	[IEEE84]
393	Update time			String	[IEEE84]
394	Utilization factor (maximum demand/rated capacity)			Real	[IEEE84]
395	Value			String	[IEEE84]
396	Voltage	V	volt	Real	[IEEE84]
397	Voltage control	V_c	volt	Real	[IEEE84]
398	Voltage control oscillator frequency	F_{vco}	hertz	Real	[IEEE84]
399	Voltage control oscillator frequency, high	F_{vcoh}	hertz	Real	[IEEE84]
400	Voltage control oscillator frequency, low	F_{vcol}	hertz	Real	[IEEE84]

Table J3. Electrical Attribute List (ALT=2) (continued)

No.	Definition	Symbol	Unit	Type	Ref.
401	Voltage control, maximum	V_{cmax}	volt	Real	[IEEE84]
402	Voltage control, minimum	V_{cmin}	volt	Real	[IEEE84]
403	Voltage rating		volt	Real	[IEEE84]
404	Voltage reference	V_{ref}	volt	Real	[IEEE84]
405	Voltage-temperature coefficient		volt/deg C	Real	[MIL195]
406	Wavelength		hertz	Real	[IEEE84]
407	Width		inch	Real	[IEEE84]
408	Wire diameter		inch	Real	[IEEE84]
409	Working peak off-state voltage	$V_{wp(off)}$	volt	Real	[MIL195]
410	Working peak reverse voltage	V_{RWM}	volt	Real	[MIL195]
411	Working peak off-state voltage	V_{DWM}	volt	Real	[MIL195]
412	Working peak-reverse voltage rating		volt	Real	[MIL195]
413	Write pulse width	T_w	second	Real	[IEEE84]
414	Write pulse width, maximum	T_{wmax}	ampere	Real	[IEEE84]
415	Write pulse width, minimum	T_{wmin}	ampere	Real	[IEEE84]
416	Zero gate voltage drain current	I_{DDS}	ampere	Real	[MIL195]
417	Zero gate voltage source current	I_{SDS}	ampere	Real	[MIL195]
418	Zero power resistance at T deg C	r_T	ohm	Real	[IEEE84]

APPENDIX J.9 ATTRIBUTE TABLE INSTANCE ENTITY (TYPE 422)

Table J4. AEC Attribute List (ALT=3)

No.	Definition	Unit†	Data Type
1	Air changes per hour	$hour^{-1}$	Real
2	Area of glazing	$foot^2$	Real
3	Bearing wall		Logical
4	Bearing wall capacity	pound/foot	Real
5	Building name		String
6	Building occupancy type		String
7	Capacity per unit of egress width		Integer
8	Ceiling cavity depth	inch	Real
9	Ceiling type		String
10	Combustible		Logical
11	Concentrated dead load	pound	Real
12	Concentrated live load	pound	Real
13	Cooled		Logical
14	Cost per square foot	dollar(US)	Real
15	Finish color		String
16	Finish type		String
17	Finished floor elevation	foot	Real
18	Finished opening height	inch	Real
19	Finished opening width	inch	Real
20	Fire door		Logical
21	Fire protection		Logical
22	Fire rating	hour	Real
23	Fire suppression system		Logical
24	Fire wall		Logical
25	Floor name		String
26	Floor to ceiling height	foot	Real
27	Floor to floor height	foot	Real
28	Floor type		String
29	Frame type		String
30	Gross area	$foot^2$	Real
31	Gross floor area per occupant	$foot^2$	Real
32	Hardware type		String
33	Heated		Logical
34	Hydrostatic pressure	$pound/foot^2$	Real
35	Illumination level	foot-candle	Real
36	Infiltration	$foot^3$/minute	Real
37	Latent heat gain	BTU/hour	Real
38	Light reflectance (percent)		Real
39	Lintel height	inch	Real
40	Live load reduction (percent)		Real
41	Means of egress		Logical

†The use of English rather than SI units follows the current practice of the AEC industry.

APPENDIX J.9 ATTRIBUTE TABLE INSTANCE ENTITY (TYPE 422)

Table J4. AEC Attribute list (ALT=3)

No.	Definition	Unit†	Data Type
42	Net area	foot^2	Real
43	Net floor area per occupant	foot^2	Real
44	Number of stories		Integer
45	Occupant load		Integer
46	Opening type		String
47	Operable		Logical
48	Relative humidity (percent)		Real
49	Riser height	inch	Real
50	Riser height, maximum	inch	Real
51	Riser height, minimum	inch	Real
52	Room activity		String
53	Room name		String
54	Rough opening height	inch	Real
55	Rough opening width	inch	Real
56	Sensible heat gain	BTU/hour	Real
57	Shading coefficient (percent)		Real
58	Shear wall		Logical
59	Shear wall capacity	pound/foot	Real
60	Sill height	inch	Real
61	Slope (percent)		Real
62	Smoke door		Logical
63	Smoke rating	hour	Real
64	Smoke wall		Logical
65	Snow load	pound/foot^2	Real
66	Soil bearing capacity	pound/foot^2	Real
67	Soil density	pound/foot^3	Real
68	Soil type		String
69	Sound level	dB	Real
70	Sound reflectance (percent)		Real
71	Sound transmission class		Integer
72	Temperature	deg F	Real
73	Thermal transmittance	BTU/hour/foot	Real
74	Tread width	inch	Real
75	Tread width, maximum	inch	Real
76	Tread width, minimum	inch	Real
77	Uniform dead load	pound/foot^2	Real
78	Uniform live load	pound/foot^2	Real
79	Unit of egress width	inch	Real
80	Ventilation	foot^3/minute	Real
81	Wall type		String
82	Wind pressure	pound/foot^2	Real
83	Work plane height	inch	Real

†The use of English rather than SI units follows the current practice of the AEC industry.

APPENDIX J.9 ATTRIBUTE TABLE INSTANCE ENTITY (TYPE 422)

Table J5. Process Plant Attribute list (ALT=4)

No.	Definition	Data Type
1	Nominal IPS	Real
2	Material name	String
3	End preparation	String
4	Wall thickness	Real
5	User part number	String
6	Joint identification number	String
7	Configuration non-deviation code	Integer
8	Material non-deviation code	Integer
9	Specification body	String
10	Material control level	String
11	Criticality class	String
12	Joint type	String
13	Weight/mass value	Real
14	Weight/mass units	String
15	Pipe spool/detail name	String
16	Functional group code	String
17	Part type	String

J.10 Intercharacter Spacing Property

This additional form of the Property Entity (Type 406) is defined below. See Section 4.3.7 for more details.

FORM NUMBER: 18 Intercharacter Spacing

The intercharacter spacing is used to specify the gap between letters when fixed-pitch spacing is used. It is applicable to text generated by the General Note and Text Template Entities. The gap is specified as a percentage of the text height.

Index	Name	Type	Description
1	NP	Integer	Number of property values (NP = 1)
2	ISPACE	Real	Intercharacter Space in percent of text height (Range 0. to 100.)

Appendix K. List of References

[ANSI68] *Code for Information Interchange* (X3.4-1968), American National Standards Institute, 1968.

[ANSI74] *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), American National Standards Institute, 1974.

[ANSI77] *Code for Information Interchange* (X3.4-1977), American National Standards Institute, 1977.

[ANSI78] *Programming Language FORTRAN* (X3.9-1978), American National Standards Institute, 1978.

[ANSI79] *Line Conventions and Lettering* (Y14.2-1979), American National Standards Institute, 1979.

[ANSI81] *Digital Representation for Communication of Product Definition Data*, Parts 1,2, and 3, (Y14.26M-1981), American National Standards Institute, 1981. Permanently out of print.

[ANSI82] *Dimensioning and Tolerancing*, (ANSI Y14.5M-1982), American National Standards Institute, 1982.

[ANSI88] *Digital Representation for Communication of Product Definition Data*, (Y14.26M-1988), American National Standards Institute, in publication.

[DEBO78] deBoor, C., *A Practical Guide to Splines*, Springer-Verlag, 1978.

[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.

[GORD74] 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.

[HON80] Hon, R. W., and C. H. Sequin, *A Guide to LSI Implementation*, SSL 79-7, Xerox Palo Alto Research Center, January 1980.

[IEEE75] *Reference Designators for Electrical and Electronics Parts and Equipment*, (IEEE Std 200-1975), Institute of Electrical and Electronics Engineers, 1975.

APPENDIX K. LIST OF REFERENCES

[IEEE76] *An American National Standard ASTM/IEEE Standard Metric Practice* (IEEE Std 268-1976), Institute of Electrical and Electronics Engineers, 1976.

[IEEE84] *Standard Dictionary of Electrical and Electronics Terms*, (ANSI/IEEE Standard 100-1984), Institute of Electrical and Electronics Engineers, 1984.

[IEEE85] *Standard for Binary Floating-Point Arithmetic* (ANSI/IEEE Std 754-1985), Institute of Electrical and Electronics Engineers, 1985.

[IEEE260] *IEEE Standard Letter Symbols for Units of Measurement* (ANSI/IEEE Std 260), Institute of Electrical and Electronics Engineers, 1978.

[IITR68] *APT Computer System Manual: Volume 2 - Subroutine Library*, Illinois Institute of Technology Research Institute, 1968.

[IPCT85] *Terms and Definitions for Interconnecting and Packaging Electronic Circuits* (ANSI/IPC-T-50C), Institute for Interconnecting and Packaging Electronic Circuits, Revision C, March 1985.

[ISO1073] *Alphanumeric Character Sets for Optical Recognition – Part II: Character Set OCR-B – Shapes and Dimensions of the Printed Image*, (ISO1073/II), International Organization for Standardization, 1976.

[JOBL78] Joblove, G. H. and D. Greenberg, "Color Spaces for Computer Graphics", *SIGGRAPH Proceedings*, 1978.

[KAPL52] Kaplan, W., *Advanced Calculus*, Addison-Wesley, 1952.

[MIL12] *Abbreviations for Use on Drawings, Specifications, Standards, and in Technical Documents* (MIL-STD-12D), U.S. Department of Defense, May 1981.

[MIL133] *Parameters to be Controlled for the Specification of Microcircuits* (MIL-STD-1331), U.S. Department of Defense, August 1970.

[MIL195] *General Specification for Semiconductors* (MIL-STD-19500G), U.S. Department of Defense, August 1987.

[NBS80] *Initial Graphics Exchange Specification (IGES), Version 1.0*, NBSIR 80-1978 (R), U.S. National Bureau of Standards, 1980. Out of print.

[NBS83] *Initial Graphics Exchange Specification (IGES), Version 2.0*, NBSIR 82-2631 (AF), U.S. National Bureau of Standards, 1982. Available from the National Technical Information Service (NTIS) as PB83-137448.

[NBS86] *Initial Graphics Exchange Specification (IGES), Version 3.0*, NBSIR 86-3359, U.S. National Bureau of Standards, 1986. Available from the National Technical Information Service (NTIS) as PB86-199759.

[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.

[TILO80] Tilove, R. B., and Requicha, A. A. G., "Closure of Boolean Operations on Geometric Entities", *Computer Aided Design*, Vol. 12, No. 5, September 1980.

Appendix L. 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 ([IEEE75])

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 ANNOTATION

In electrical engineering, the practice of changing the unique identifier for components noted by symbols on a schematic to match those assigned actual components when the circuit is packaged.

BACK POINTER

A pointer in the parameter data section of an entity pointing to an associativity instance of which it is a member.

APPENDIX L. GLOSSARY

BASIC PART ([IEEE75])

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 E 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, nonintersecting 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 one 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.

CONNECT 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. As the weights must all be 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 Appendix E for more details.

COONS PATCH

A three dimensional surface.

COPIOUS DATA ENTITY

A geometric entity sometimes used as an annotation entity, containing arrays of tuples of real numbers to which a specific meaning has been assigned. Each 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.

DIRECTORY ENTRY SECTION

The section of an IGES file, consisting of fixed field data items, that forms 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 subfigure instance.

EDGE 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 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 (u_i, v_j) where u_i and v_j 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 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 about an axis.

MODEL

A particular collection of data in an IGES file that 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 physical 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.

NULL ENTITY

The Null Entity (Type 0) is intended to be ignored by a processor. A processor should skip over all DE and PD data associated with this entity.

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 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 can be viewed as blendings of 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 consisting of all or a portion of a plane.

PLATED-THROUGH HOLE ([IPCT85])

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 defined by this Specification 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 defined by this Specification.

PRINTED BOARD ([IPCT85])

The general term for completely processed printed circuit or printed wiring configurations. It includes rigid or flexible, single, double, or multilayer boards.

PRINTED CIRCUIT ([IPCT85])

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 ([IPCT85])

A part manufactured from rigid base material upon which a completely processed printed circuit has been formed.

PRINTED WIRING ([IPCT85])

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.

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 himan-readable file prologue.

SUBASSEMBLY ([IEEE75])

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 ([IEEE75])

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 dependent and independent variable.

TABULATED CYLINDER ENTITY

A geometric entity which is a surface generated by moving a line segment called the generatrix parallel to itself along a space curve called the directrix.

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 ([IEEE75])

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 ([IPCT85])

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 positive 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 Appendix E 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.

Appendix M. Index of Topics

Appendix N. Numerical Index of Entities

Entity Type Number	Form Number	Entity Name	Page
0		Null	450
100		Circular Arc	72
102		Composite Curve	74
104	0–3	Conic Arc	78
106		Copious Data	83
	1–3	Data Points	83
	11–13	Piecewise Linear Curve	83
	20–21	Centerline	178
	31–38	Section	208
	40	Witness Line	216
	63	Simple Closed Area	83
108	-1, 0, +1	Plane	86
110		Line	89
112		Parametric Spline Curve	91
114		Parametric Spline Surface	95
116		Point	99
118		Ruled Surface	100
120		Surface of Revolution	104
122		Tabulated Cylinder	1107
124	0–1, 10–12	Transformation Martrix	109
125		Flash	116
126	0–5	Rational B-Spline Curve	118
128	0–9	Rational B-Spline Surface	120
130		Offset Curve	122
132		Connect Point	124
134		Node	125
136		Finite Element	128
138		Nodal Displacement and Rotation	142
140		Offset Surface	144
142		Curve on a Parametric Surface	146
144		Trimmed (Parametric) Surface	148
146	TYPE	Nodal Results	150, 451
148	TYPE	Element Results	151, 453
150		Block	154
152		Right Angular Wedge	156
154		Right Circular Cylinder	158
156		Right Circular Cone Frustum	159
158		Sphere	161

APPENDIX N. NUMERICAL INDEX OF ENTITIES

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. **BIBLIOGRAPHIC DATA SHEET** *(See instructions)*	1. PUBLICATION OR REPORT NO. NBSIR 88-3813	2. Performing Organ. Report No.	3. Publication Date June 1988

4. TITLE AND SUBTITLE

Initial Graphics Exchange Specification (IGES) Version 4.0

5. AUTHOR(S)

Bradford Smith, Gaylen R. Rinaudot, Kent A. Reed, Thomas Wright

6. PERFORMING ORGANIZATION *(If joint or other than NBS, see instructions)*

NATIONAL BUREAU OF STANDARDS
U.S. DEPARTMENT OF COMMERCE
GAITHERSBURG, MD 20899

7. Contract/Grant No.

8. Type of Report & Period Covered
Final
1986 - 1988

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS *(Street, City, State, ZIP)*

National Bureau of Standards

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

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 4.0 of the Initial Graphics Exchange Specification, a defined format for the representation and exchange of data found in today's commercially available CAD/CAM systems.

Newly added capability includes constructive solid geometry and more complete support for finite element, electrical and drafting applications.

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

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES
547

15. Price
$42.95

USCOMM-DC 6043-P80
☆ U.S. GOVERNMENT PRINTING OFFICE: 1988—201-597/92528


AMERICAN BINDERY-EAST
1993