ICS 25.080.99; 25.030
CCS J 59

NATIONAL STANDARD OF THE PEOPLE'S REPUBLIC OF CHINA

GB/T 25632-2024

Replacing GB/T 25632-2010

Data interface format for software of additive manufacturing machines

增材制造机床软件数据接口格式

Issued on: April 25, 2024 **Implemented on: November 1, 2024**

Issued by: State Administration for Market Regulation;
Standardization Administration of the PRC

Table of Contents

3 Terms and definitions

The terms defined in GB/T 14896.7 and GB/T 35351 apply, together with seven terms defined here.

3.1 Additive manufacturing machine: a machine that manufactures parts on the principle of layer-by-layer discretisation and accumulation.

3.2 Data interface format: the uniform set of rules for writing electronic files used to transfer data and exchange information between different computer software systems.

3.3 Triangular facet: in a three-dimensional model in STL format, the smallest unit, made up of three non-collinear points in three-dimensional space, the vertices, and one vector identifying the direction.

3.4 Layer: the space between two adjacent cross sections, bounded by the thickness of that region and by a set of contours and hatches.

3.5 Polyline: the line type formed by a series of straight segments joined end to end in a plane.

3.6 Contour: the boundary of the solid at a given height within a specified layer, made up of polylines. A note adds that contours are usually divided into inner and outer contours, and that each polyline is closed and neither crosses another polyline nor intersects itself.

3.7 Hatches: the set of independent line segments used to fill the contour of the solid. A note adds that each segment is defined by a start point coordinate and an end point coordinate.

4 Three-dimensional part geometry transfer file (STL file)

4.1 The STL file defines the geometric features of a three-dimensional part through a series of spatial triangular facets carrying a direction attribute, and generally exists in two coded storage forms, ASCII and binary. The file is an unordered collection of triangular facets, and its format is defined as follows: an STL file consists of triangular facet 1, triangular facet 2 and so on to triangular facet n; a triangular facet consists of a normal vector followed by vertex 1, vertex 2 and vertex 3; a normal vector consists of the components lx, ly and lz; and a vertex consists of x, y and z.

4.2.1 In the ASCII STL format the first line is the description line and the recording of triangular facets starts on the second line: the normal of the facet is recorded first, then the loop, then the coordinates of the three vertices in turn, the order of the vertices agreeing with the facet normal by the right-hand rule. When one facet has been recorded the next one is started, until all the facets of the model have been recorded and the file ends; Annex A gives an example. The keywords used are solid followed by the name, as the description line recording the file name of the STL file; facet normal followed by the three components n1, n2 and n3, recording the normal of the facet up to endfacet; outer loop, recording the coordinates of the three vertices up to endloop, with one vertex line per vertex; endloop and endfacet closing the facet; and endsolid followed by the name, ending the file.

4.2.2 The binary STL coding is stored byte by byte: the first 80 bytes are used as a description, the following 4 bytes hold the total number of triangular facets as a long integer, and the facet information, the normal vector and the three vertices, then follows. The components of the normal vector and the vertex coordinates are floating point numbers occupying 4 bytes each, expressed in little-endian order. The normal vector of a facet shall point outwards from the model. After each facet has been recorded, two bytes are left empty and the next facet is recorded in turn until all the information has been recorded and the file ends. The structure is defined as the STL file entity name, the total number of triangular facets N and the facet information; the entity name occupies 80 bytes, padded with spaces if shorter; N is a 4-byte long integer; the facet information is the facet normal vector followed by the facet vertex coordinates followed by two empty bytes, repeated; the facet normal vector is the three floating point numbers lx, ly and lz occupying 12 bytes in total; the facet vertex coordinates are the nine floating point numbers x1, y1, z1, x2, y2, z2, x3, y3 and z3 occupying 36 bytes in total; and the two empty bytes form a 2-byte unsigned integer.

4.3 Colour in a binary coded STL file is expressed through the initial 80-byte header, which carries the overall colour of the model. If colour is used, the header carries at some point the text string COLOR= followed by 4 bytes of RGBA, the last being transparency, with values from 0 to 255; this is the colour of the whole model and each facet may specify its own. Surface properties, that is the material, may then be specified: after COLOR=RGBA the ASCII string MATERIAL= may be added, followed by 12 bytes, three groups of 4 bytes, expressing three colours of the material, the first 4 bytes being the diffuse reflection colour, the second the specular highlight colour and the third the ambient light. The colour of each triangular facet is stored in the attribute character count of 16 bits held in the two empty bytes after the facet, as follows: bits 0 to 4 give the intensity of red, from 0 to 31; bits 5 to 9 the intensity of green, from 0 to 31; bits 10 to 14 the intensity of blue, from 0 to 31; and bit 15 states whether the facet colour is used, 0 meaning the facet colour and 1 the model colour.

5 Two-dimensional layer process information transfer file (CLI file)

5.1 The CLI file is a general layer file suited to layered manufacturing technology and generally exists in two coded storage forms, ASCII and binary.

5.2.1.1 An ASCII CLI file is divided into several parts, each marked by a start and an end identifier. Only the characters A to Z, a to z, 0 to 9, the dollar sign, the separators, that is the slash, the comma and the double slash, and the decimal point are interpreted; all other characters are ignored by the computer. Every file shall have a header, the part between HEADERSTART and HEADEREND, and a geometry section, the part between GEOMETRYSTART and GEOMETRYEND; the other parts are optional. The start of the header serves as the start of the file and the end of the geometry section as the end of the file, as shown in Annex B. A note adds that the file also includes the parts before the header and after the geometry section, but that this content is ignored by the computer.

5.2.1.2 Apart from comments, all commands take the general form keyword followed by a slash and the parameters. The keyword and the parameters are separated by the slash; where there is no parameter there shall be no separator, the only exception being the command formed by a double slash. All keywords shall be in upper case and each keyword shall begin with the double dollar characters. Parameters are numbers or ASCII strings separated by commas. The separators are the slash, the comma and the double slash. An ASCII string is made up of legal printable ASCII characters of any length between double quotation marks.

5.2.1.5 Table 1 gives the meanings of the common commands of an ASCII CLI file, grouped as non-geometry commands, header information, optional header information, geometry commands and other commands. The non-geometry comment command places the text between double slashes; the text is a set of printable characters and shall contain no double slash. The non-geometry structural commands are HEADERSTART, which marks the start of the header and serves as the start of the data; HEADEREND, which marks the end of the header; GEOMETRYSTART, which starts the geometry section; and GEOMETRYEND, which ends the geometry section and serves as the end of the data; none of them takes a parameter. The header information commands are BINARY, stating that the data of the geometry section are binary, and ASCII, stating that the data format of the geometry section is ASCII, neither taking a parameter; UNITS followed by u, where u is a real number giving the coordinate unit in millimetres; and VERSION followed by v, an integer whose value divided by 100 is the version number. The optional header information commands are DATE followed by d, an integer interpreted in the order day, month, year; DIMENSION followed by the six real numbers x1, y1, z1, x2, y2 and z2, describing the coordinate limits and dimensional range of the solid in the chosen coordinate system in millimetres, subject to x1 less than x2, y1 less than y2 and z1 less than z2; LAYERS followed by i, an integer giving the number of layers in the file; ALIGN, which aligns the data of the geometry section to 32 bits, used only for a binary geometry section, the data starting at a 32-bit boundary and the header ending at a 32-bit boundary; LABEL followed by an integer id and a text, the id defining one of several models within a file and appearing at the start of polylines and hatches, and the text, an ASCII string, explaining the part; and USERDATA followed by uid, len and user-data, where uid is an ASCII string user identifier, len a long integer giving the length of the user data, and user-data the content, binary or ASCII, of len bytes.

5.2.1.5 The geometry commands of an ASCII CLI file are LAYER followed by z, a real number giving the height of the upper surface of the layer as z multiplied by the length unit in millimetres, all layers being arranged in ascending order of z, the layer thickness being the difference in height between the current and the previous layer, and the thickness of the first layer being definable by including a zero layer that has a z value but no polyline; POLYLINE followed by id, dir, n and the coordinate pairs p1x, p1y to pnx, pny, where id is an integer defining one of several models within a file as for the label command, dir is an integer giving the direction of the segments seen from the negative z direction, 0 being clockwise and internal, 1 anticlockwise and external and 2 an open segment that is not solid, n is an integer giving the number of points, that is the number of vertices on the polyline, and the coordinate pairs are real numbers for points 1 to n, with inner contour polylines running clockwise and outer contour polylines anticlockwise, the parameter dir and the order of the points agreeing with one another and the point order being rewritten to match dir in case of error, and with p1x equal to pnx and p1y equal to pny for a closed polyline; and HATCHES followed by id, n and the end point data p1sx, p1sy, p1ex, p1ey to pnex, pney, where id is an integer defining one of several models within a file, n is an integer giving the number of hatch segments, n multiplied by 4 being the amount of data, and the end point data are real numbers for hatches 1 to n, each hatch having the four parameters start x, start y, end x and end y. The other command is FEATURE followed by n and the parameters p1, p2 to pn; it is inserted by the user before a polyline or hatch command as an identifier of the geometry feature type, stating that the data that follow