INCITS L1 ASPRS
LAS Standard
DRAFT - Version 2.0

August 17, 2007

TABLE OF CONTENTS

Annex

Tables

1 LAS DATA EXCHANGE FORMAT VERSION 2.0

1.1 Scope, purpose, and application

Laser-based remote sensing technologies (e.g. LIDAR) have been used for years to detect and capture information about objects distant from the observation sensor. These technologies measure properties of scattered light to find range and/or other information of a distant target. The light may be emitted in the form of short pulses (ie, "pulsed"), or continuously (ie, "continuous-wave"). Like the similar radar technology, which uses radio waves instead of light, the range to an object is determined by measuring the time delay between transmission of a pulse and detection of the reflected signal.

This document is a part of a larger suite of document being developed to describe point cloud data. LASer (LAS) 2.0 defines a binary data exchange format for pulsed laser point cloud geospatial data using variable length records. The purpose of LAS 2.0 is make the exchange, manipulation, analysis, and storage of point cloud point data faster and easier.

LAS 2.0, although originally designed for the exchange of Light Detection and Ranging (LIDAR) data, can be used to exchange any point cloud geospatial data where the points that compose the point cloud can be encoded as a 3-tuple of x, y, and z values in a point cloud record. Additional Information and supporting documentation, such as point classification, about each point can also be encoded in the point cloud record.

The point cloud geospatial data are encoded in and decoded from this exchange format by software provided by sensor hardware vendors or point cloud processing software. In the case of hardware, laser pulse range data are combined with sensor orientation information to produce geocoded in 3-tuple sets of X, Y, and Z point data.

This document is the first major revision of the LAS file format specification since the Version 1.0 release in 2003. LAS 1.1 (the current version) was approved by the American Society for Photogrammetry and Remote Sensing in 2005. Since the change from LAS 1.1 to LAS 2.0 is so extensive, a modification document will not be issued. Thus this document presents the 2.0 specification without a change reference to the 1.1 specification.

The LAS 2.0 includes major new features such as infinitely variable point data records, extending the format to include terrestrial scanners, and the inclusion of general point cloud three dimensional data. The LAS 2.0 specification extends the LAS 1.X specifications by using the file format to encode point cloud data to the general case of N dimensional data.

In LAS 2.0, the point cloud data records are key word encoded and contain a set of mandatory fields. To these mandatory fields, users can add standard-based defined optional fields as well as their own user defined fields. This solves the significant problem in the 1.x specification of requiring a new record type (that then became proprietary) each time a new field needed to be defined. The intention of LAS 2.0 is to provide an open format that allows different hardware and software tools vendors to use a standardized point cloud data exchange format.

File Input/Output (I/O) software libraries that support LAS 1.0 or 1.1 will require significant modification in order to be compliant with LAS 2.0. A detailed change document that provides both an overview of the changes in the specification as well as the motivation behind each change is available from the ASPRS website (http://www.asprs.org)

1.2 Maintenance Authority:

The American Society for Photogrammetry and Remote Sensing (ASPRS) is the responsible organization for maintaining and coordinating any revisions of this standard.

1.3 Contact Information:

Address questions concerning this standard to:

American Society for Photogrammetry and Remote Sensing
5410 Grosvenor Lane, Suite 210
Bethesda, Maryland 20814-2160
Tel (301) 493-0290
Attention: Standards Committee/LAS Standard

1.4 Conformance:

LAS 2.0 files shall meet the following conformance.

- must have a ".LAS" file extension.
- must meet the record field conditionality and cardinality requirements as specified in Tables 4-1,5-1, and 6-1

1.5 Normative References:

ANSI/IEEE Std 754-1985 - Institute of Electrical and Electronics Engineers (IEEE) Standard for Binary Floating-Point Arithmetic

ISO/IEC 8859-1:1998 - International Electrotechnical Commission, Information technology - 8-bit single-byte coded graphic character sets

ITU-T X.667 (09/2004) – International Telecommunication Union Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 object identifier components ITU-T; Recommendation X.667, 09/2004
http://www.itu.int/ITU-T/studygroups/com17/oid/X.667-E.pdf

2 LAS 2.0 Point Cloud File Format Definition

The LAS 2.0 Point Cloud File Format is designed to permit the efficient exchange of a 3-tuple of X, Y, Z point cloud geospatial data records. LAS 2.0 formatted files end with a ".LAS" extension.

LAS 2.0 files contains 4-record types (sequenced in order): a FILE HEADER BLOCK record, a variable length POINT RECORD METADATA BLOCK record, VARIABLE LENGTH RECORDS records, and the POINT DATA records. See Figure 2-1.

FILE HEADER BLOCK
POINT RECORD METADATA BLOCK
VARIABLE LENGTH RECORDS
POINT DATA

Table 2-1 LAS 2.0 File Record Order

The FILE HEADER BLOCK record contains generic information and data such as point source identification and bounding geospatial coordinates. The POINT RECORD METADATA BLOCK record describes the structure of the POINT DATA records. The VARIABLE LENGTH RECORDS contain variable types of data including coordinate system information, metadata and user defined auxiliary data. The POINT DATA records contain the actual X, Y, Z point cloud geospatial data points.

The content and format of a LAS POINT DATA record is user definable as described by this standard. LAS 2.0 defines a set of mandatory and optional LAS point field records such as the X, Y, Z coordinates of the point, the reflectance classification and so forth. Users can extend the standard in any manner they choose given constraints described by this standard. Regardless of the user extension schema, any LAS 2.0 compliant software must be able to read and operate on the required data fields.

3 DATA TYPES:

LAS 2.0 uses and supports a variety of standardized computer language datatypes. Table 3-1 defines the moniker, byte sizes, and a description for each supported datatype.

Moniker	Data Type	Size (in Bytes)	Description
BOOL	Logical	1	Value range 1 = TRUE, 0 = FALSE
B1	Byte	1	Integer Value range from -128 to 127stored in 1 byte
UI1	Unsigned Byte	1	Integer value range between 0 and 255 stored in 1 byte. Also used to represent ASCII character data using character code convention (ISO 8859-1:1998).
I2	Signed Short Integer	2	Integer value range from -32768 to 32767 stored in 2 contiguous bytes
UI2	Unsigned Short Integer	2	Integer value from 0 to 65535 stored in 2 contiguous bytes
I4	Signed Long Integer	4	Integer values range between -2,147,483,648 to 2,147,483,647 stored in 4 contiguous bytes
UI4	Unsigned Long Integer	4	Integer value range from 0 to 4,294,967,295 stored in 2 contiguous bytes
I8	Signed 8 byte integer	8	Integer value range from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 stored in 8 contiguous bytes
UI8	Unsigned 8 byte integer	8	Integer value range from 0 to 18,446,744,073,709,551,615, stored in 8 contiguous bytes
R4	Real Float	4	Single-precision real floating-point values in IEEE S (ANSI/IEEE Std 754-1985)
R8	Real Double	8	Double-precision real floating-point values in IEEE T (ANSI/IEEE Std 754-1985)
R10	Real Extended precision Double	10	(ANSI/IEEE Std 754-1985)
STR	zero (null – binary 0000 0000) terminated variable length string	VAR	Variable length string of UI1 (characters) terminated by a UI1 whose value is zero (the null terminator). Note that the length of a String is always the number of characters in the string plus 1.
BFx[1]	bit field	x, where x = 1,2,4,8	An unsigned integer (of value 1, 2, 4 or 8 bytes) encoded as a bit field.
[n]	Array	VAR	Brackets indicate a fixed size array of n elements of the specified type (e.g. UI1[4] is an array of 4 characters)

Table 3-1 SUPPORTED DATA TYPES

LAS 2.0 uses Little Endian byte order for bit fields. See Annex A for further information and a description of Little Endian byte order notation..

4 FILE HEADER BLOCK:

The first LAS 2.0 record is the “FILE HEADER BLOCK”. Table 4-1 describes the contents, format, byte size, and conditionality, cardinality of each field of the FILE HEADER BLOCK.

File Header Block Field Name	Format	Size	Conditionality, Cardinality (M=Mandatory, 1 occurrence)
FILE SIGNATURE (“LASF”)	UI1[4]	4 bytes	M, 1
FILE ID	UI4	4 bytes	M, 1
PROJECT ID - GUID DATA 1	UI4	4 bytes	M, 1
PROJECT ID - GUID DATA 2	UI2	2 bytes	M, 1
PROJECT ID - GUID DATA 3	UI2	2 bytes	M, 1
PROJECT ID - GUID DATA 4	UI1[8]	8 bytes	M, 1
VERSION MAJOR	UI1	1 byte	M, 1
VERSION MINOR	UI1	1 byte	M, 1
SYSTEM IDENTIFIER	UI1[32]	32 bytes	M, 1
GENERATING SOFTWARE	UI1[32]	32 bytes	M, 1
FILE CREATION DATE	UI4	4 bytes	M, 1
FILE CREATION TIME	UI4	4 bytes	M, 1
FILE HEADER SIZE	UI2	2 bytes	M, 1
SOURCE ID PACKET	BF2	2 bytes	M, 1
OFFSET TO POINT RECORD METADATA BLOCK	UI2	2 bytes	M, 1
SIZE OF POINT RECORD METADATA BLOCK	UI2	2 bytes	M, 1
POINT DATA RECORD LENGTH	UI2	2 bytes	M, 1
OFFSET TO POINT DATA	UI8	8 bytes	M, 1
NUMBER OF POINT RECORDS	UI8	8 bytes	M, 1
OFFSET TO FIRST VARIABLE LENGTH RECORD	UI4	4 bytes	M, 1
NUMBER OF VARIABLE LENGTH RECORDS	UI4	4 bytes	M, 1
NUMBER OF POINTS BY RETURN	UI4[16]	64 bytes	M, 1
POINT RECORD COMPATIBILITY	UI1	1 byte	M, 1
COORDINATE SYSTEM TYPE	UI1	1 byte	M, 1
HORIZONTAL METRIC UNITS	UI1	1 byte	M, 1
VERTICAL METRIC UNITS	UI1	1 byte	M, 1
X ORIGIN	R8	8 bytes	M, 1
Y ORIGIN	R8	8 bytes	M, 1
Z ORIGIN	R8	8 bytes	M, 1
APPLY SCALING	BOOL	1 byte	M, 1
X SCALE FACTOR	R8	8 bytes	M, 1
Y SCALE FACTOR	R8	8 bytes	M, 1
Z SCALE FACTOR	R8	8 bytes	M, 1
APPLY OFFSETS	BOOL	1 byte	M, 1
X OFFSET	R8	8 bytes	M, 1
Y OFFSET	R8	8 bytes	M, 1
Z OFFSET	R8	8 bytes	M, 1
MAX X	R8	8 bytes	M, 1
MIN X	R8	8 bytes	M, 1
MAX Y	R8	8 bytes	M, 1

File Header Block Field Name	Format	Size	Conditionality, Cardinality (M=Mandatory, 1 occurrence)
MIN Y	R8	8 bytes	M, 1
MAX Z	R8	8 bytes	M, 1
MIN Z	R8	8 bytes	M, 1

Table 4-1 FILE HEADER BLOCK Field Names

4.1 FILE HEADER BLOCK Field Descriptions

The following subsections describe the content of each field of the FILE HEADER BLOCK record.

4.1.1 FILE SIGNATURE

The FILE SIGNATURE field is used to determine that the file is a LAS point cloud data file. The FILE SIGNATURE value must be equal to the four ASCII characters “LASF”.

4.1.2 FILE ID

The FILE ID field is assigned to a particular file of POINT DATA records. A value of zero (null – binary 0000 0000) means that a FILE ID has not been assigned (a non-fatal error condition). Processing software may assign any valid number as the FILE ID and will assume that POINT_SOURCE_IDs have not been set (see POINT RECORD METADATA BLOCK information). A POINT_SOURCE_ID can be identified as an original data capture identification ID or it can be the result of merge and/or extract operations.

For an airborne system (ALS), a common practice is to assign the FILE ID to the Flight Line Number if the file was derived from an original airborne LIDAR flight line.

For a terrestrial scanner (TLS) system, a common practice is to assign the FILE ID to a unique value that identifies a single scanning operation. Note that two distinct FILE IDs do not necessarily imply two physically separate terrestrial scanning station operations.

The FILE ID is assigned when a file becomes part of a Project (see PROJECT ID). For example, if an existing digital elevation terrain model were converted to LAS format and introduced to a project, it would be assigned a unique ID prior to the inclusion process.

FILE IDs must be unique across a project. Any two files with the same PROJECT ID must have different FILE IDs.

4.1.3 PROJECT ID - GUID DATA

The Project Identifier (PROJECT ID) field is used to specify a Globally Unique Identifier (GUID), also commonly known as a Universally Unique Identifier (UUID) - see ITU-T X.667. GUID is used to uniquely identify a collection of LAS files as being members of the project identified by the PROJECT ID. The GUID is generated by a software driven algorithm that uses time, multiplexed with other parameters. GUIDs are most commonly written in text as a sequence of hexadecimal digits such as 3F2504E0-4F89-11D3-9A0C-0305E82C3301. The PROJECT ID should be the same for all files that are associated with a unique project. By assigning unique PROJECT ID and FILE IDs, every file within a project and every point within a file can be uniquely identified in all time and space.

A PROJECT ID equal to zero (null) implies that the Project ID scheme is not in use.

4.1.4 VERSION number (MAJOR, MINOR)

The Version number consists of a VERSION MAJOR and VERSION MINOR field and is used to specify the LAS version number for the LAS exchange file. For example, for a LAS 2.0 compliant file, the VERSION MAJOR field would equal a value of 2 and the VERSION MINOR field would equal a value of zero (null).

4.1.5 SYSTEM IDENTIFIER

The SYSTEM IDENTIFIER field is used to specify file unique source information and a generating agent such as a hardware system or software processing step (extraction, merging or modifying) of existing data files. Recommended System Identifiers are listed in Table 4-2.

Generating Agent	System Identifier
Hardware system	String identifying hardware (e.g. "ALTM 1210" or "ALS-50"
Merge of one or more files	"MERGE"
Modification of a single file	"MODIFICATION"
Extraction from one or more files	"EXTRACTION"
Reprojection, rescaling, warping, etc.	"TRANSFORMATION"
Some other operation	"OTHER" or a string up to 32 characters identifying the operation

Table 4-2 SYSTEM IDENTIFER

If the character data is less than 32 characters, the remaining data must be set to zero (null).

4.1.6 GENERATING SOFTWARE

The GENERATING SOFTWARE field is used to identify the generating software. This field provides a mechanism for specifying which generating software package and version was used during LAS file creation (e.g. "TerraScan V-10.8", "REALM V-4.2", "GeoCue 4.0" and etc.). If the character data are less than 32 characters, the remaining data must be set to zero (null).

4.1.7 FILE CREATION DATE

The FILE CREATION DATE field is used to specify the date the file was created and is expressed in Modified Julian Days (MJD). The MJD is the integer number of days that have elapsed since midnight, Universal Time, at the beginning of Wednesday November 17, 1858.

4.1.8 FILE CREATION TIME

The FILE CREATION TIME field is used to specify the time the file was created and is expressed as the number of seconds into the Modified Julian day, where midnight is equal to zero (null).

4.1.9 FILE HEADER SIZE

The FILE HEADER SIZE field is used to specify the number of bytes of the FILE HEADER BLOCK record. Unlike previous versions of the LAS specification, extension of the FILE HEADER BLOCK record by users is not permitted. This field has a value of 322 for Version 2.0.

4.1.10 SOURCE ID PACKET

The SOURCE ID PACKET field is used to identify the data source type(s) contained in the file. If all bits are zero (null), the Source has not been specified. If a particular source is used that corresponds to a SOURCE ID, the SOURCE ID bit for that source is set to 1. Table 4-3 identifies the SOURCE ID PACKET information.

Bit(s)	Source ID	Notes
0	Airborne Sensor only	If set, data was solely collected from an airborne sensor
1	Terrestrial Sensor only	If set, data originally was solely collected from a terrestrial sensor
2	Original Flight Line	Data in this file came from a single flight line (implies that bit 0 should also be set)
3	Original Terrestrial Scan	Data in this file came from a single terrestrial scan (implies that bit 1 should also be set)
4	Mixed Flight Line file	This file contains data from multiple flight lines (implies bit 0 should also be set)
5	Mixed Terrestrial Scans	This file contains data from multiple terrestrial scans (implies bit 1 should also be set)
6	Hybrid Airborne, Terrestrial data	Data in this file resulted from merging both airborne and terrestrial scan data – implies both bits 0 and 1 should be reset (zero)
7	Stereo Image automatic correlation	Data in this file were derived from stereo image extraction using an automatic correlation technique (Automatic Terrain Extraction, ATE)
8	Manual Stereo Collection	Data in this file were collected using interactive (manual) stereo collection from visible imagery
9	SAR	Data in this file were derived from synthetic aperture radar
10	Elevation Import	Data in this file were created by importing an elevation file product (such as a USGS DEM or an NGA DTED)
11:15	RESERVED	Reserved for future use

Table 4-3 SOURCE ID PACKET

4.1.11 OFFSET TO POINT RECORD METADATA BLOCK

The OFFSET TO POINT RECORD METADATA BLOCK field is used to specify the number of bytes from the beginning of the LAS file to the first byte of the POINT RECORD METADATA BLOCK. This value should be the same as the FILE HEADER SIZE value (i.e. there can be no additional fields inserted between the FILE HEADER BLOCK record and the start of the POINT RECORD METADATA BLOCK record). The value is equal to 322 for LAS Version 2.0.

4.1.12 SIZE OF POINT RECORD METADATA BLOCK

The SIZE OF POINT RECORD METADATA BLOCK field is used to specify the number of bytes in the POINT RECORD METADATA BLOCK. Note that the POINT RECORD METADATA BLOCK record is a variable size record and that value can vary from file to file.

4.1.13 POINT DATA RECORD LENGTH

The POINT DATA RECORD LENGTH field is used to specify the number of bytes of a POINT DATA record. Note that content and size of a POINT DATA record is not fixed in the LAS 2.0 specification. Any application that manipulates the length of the POINT DATA records must update the POINT DATA RECORD LENGTH value. All POINT DATA records in a single file must be the equal to the POINT DATA RECORD LENGTH value.

4.1.14 OFFSET TO POINT DATA

The OFFSET TO POINT DATA field is used to specify the number of bytes from the beginning of the file to the first byte of the first POINT DATA record data field. The OFFSET TO POINT DATA field must be updated if any software adds/removes information or data from the VARIABLE LENGTH RECORDS.

4.1.15 NUMBER OF POINT RECORDS

This NUMBER OF POINT RECORDS field is used to specify the total number of POINT DATA records within the file.

4.1.16 OFFSET TO FIRST VARIABLE LENGTH RECORD

The OFFSET TO FIRST VARIABLE LENGTH RECORD field is used to specify the number of bytes from the beginning of the file to the first byte of the first variable length record.

4.1.17 NUMBER OF VARIABLE LENGTH RECORDS

The NUMBER OF VARIABLE LENGTH RECORDS field is used to specify the number of variable length records. This field must be updated if the number of variable length records changes.

4.1.18 NUMBER OF POINTS BY RETURN

The NUMBER OF POINTS BY RETURN field is used to specify an array of the total POINT DATA records per return. The first UI4 value (array index 0) will be the total number of records for null returns (see POINT DATA Record description for the definition of a null return), the second UI4 value will contain the total count for the first returns, the third contains the total number for return two, and so forth up to 15 returns. If the collection system does not support multiple returns or the source was from a non-sensor then the second entry in this array should equal the total number of POINT DATA records and the remaining entries should be set to zero (null).

4.1.19 POINT RECORD COMPATIBILITY MODE

The POINT RECORD COMPATABILITY MODE field is used to indicate if the POINT DATA records are compatible with LAS 1.1 record formats. If set to 0 or 1, the POINT DATA records in the file are identical to the corresponding POINTS DATA record types in the prior specification formats (see Table 4-4). For POINT DATA records that are compatible, software, upon reading the compatibility flag, can skip the POINT DATA record descriptors.

Compatibility Mode	Definition
0	Point records are compatible with LAS 1.1 point record type 0
1	Point records are compatible with LAS 1.1 point record type 1
2-254	Reserved
255	No compatible mode implied

Table 4-4 POINT DATA RECORD COMPATIBILITY

4.1.20 COORDINATE SYSTEM TYPE

The COORDINATE SYSTEM TYPE field is used to define the supported coordinate system types (see Table 4-5)

Type	Definition
1	Geographic or projected coordinate system as defined by Well Known Text (WKT) definitions in variable length records
2	Cartesian, in which case X, Y and Z are a 3-dimensional standard Cartesian coordinate system
3	Positive Spherical (defined in Table 4-6)
4	Negative Spherical (defined in Table 4-7)

Table 4-5 COORDINATE SYSTEM TYPE

Table 4-6 POSITIVE SPHERICAL and Table 4-7 NEGATIVE SPHERICAL specify the spherical coordinate system values if a spherical coordinate system is used:

Value	Meaning
X = *r*	the radial distance of the point from the origin
Y = θ	the azimuthal angle in the X, Y plane from the *x*-axis with $0 \le \theta \le 2\pi$, positive direction counter-clockwise
Z = φ	the polar angle from the positive Z axis, $0 \le \varphi \le \pi$

Table 4-6 POSITIVE SPHERICAL

Value	Meaning
X = *r*	the radial distance of the point from the origin
Y = θ	the azimuthal angle in the X, Y plane from the *x*-axis with $0 \le \theta \le \pi$, positive direction counter-clockwise
Z = φ	the polar angle from the negative Z axis, $0 \le \varphi \le 2\pi$

Table 4-7 NEGATIVE SPHERICAL

Angles in spherical coordinates are always expressed in radians.

4.1.21 HORIZONTAL METRIC UNITS

The HORIZONTAL METRIC UNITS field specifies the horizontal measurement units of the point cloud data (applies only to the *r* parameter for spherical coordinate systems; angles are always in radians) when the coordinate system is type 2 or 3 (units for type 1 are specified in the WKT record in which case this field should be set to 0). The valid values and meanings are listed in Table 4-8.

Value	Meaning
0	Units are defined in the WKT file
1	Meters
2	International Feet (1 international foot = 0.3048 meter)
3	U. S. Survey Feet (1 survey foot = 12/39.37 meter)
4	Arc Seconds (Cartesian coordinate systems only)

Table 4-8 HORIZONTAL METRIC UNITS

4.1.22 VERTICAL METRIC UNITS

(Only valid for Cartesian Coordinate Systems)

The VERTICAL METRIC UNITS field is used to specify the vertical measurement units of the point cloud data when the coordinate system is Cartesian and the vertical metric units are different from the horizontal units (see Table 4-9). This field is set to zero (null) when not applicable.

Value	Meaning
0	Use units defined in the Metric Units field
1	Meters
2	International Feet (1 international foot = 0.3048 meter)
3	U. S. Survey Feet (1 survey foot = 12/39.37 meter)

Table 4-9 VERTICAL METRIC UNITS

4.1.23 X, Y, Z ORIGIN

The X, Y, Z ORIGIN fields are used to define the sensor origin (within the specified coordinate system) if spherical coordinates are used. These fields must be set to 0.0 for Cartesian coordinates (they should be set to 0.0 when not applicable).

4.1.24 APPLY SCALING

The APPLY SCALING field is used to indicate if scale factors should be applied to the coordinates (see 4.1.25). If TRUE, the scale factors must be applied to <u>all</u> POINT DATA record coordinate values and MAX AND MIN X, Y, Z coordinate values.

4.1.25 X, Y, Z SCALE FACTORS

The X,Y, Z SCALE FACTORS fields are used to specify the scale factors to scale the corresponding X, Y, and Z POINT DATA record values when the “APPLY SCALING” flag is set to TRUE. The corresponding X, Y, and Z scale factor must be multiplied by the X, Y, or Z POINTS DATA record values to get the actual X, Y, or Z coordinate values.

4.1.26 APPLY OFFSETS

The APPLY OFFSETS field is used to indicate if offset factors should be applied to the coordinates (see 4.1.27). If TRUE, the offset factors must be applied to <u>all</u> POINTS DATA record coordinates and MAX AND MIN X, Y, Z coordinate values.

4.1.27 X, Y, Z OFFSET

The X, Y, Z OFFSET fields are used to specify the X, Y, Z offsets to offset the corresponding X,Y, and Z coordinate POINT DATA record values when the “APPLY OFFSETS” flag is set to TRUE. To compute the actual X coordinate value, a POINT DATA record X is multiplied by the X SCALE FACTOR, and then the X OFFSET is added. The same process is repeated for the Y and Z coordinate values.

If both APPLY SCALING and APPLY OFFSETS flag are TRUE, then the final coordinates are determined using the following equation:

$$X_{coordinate} = (X_{record} * X_{scale}) + X_{offset}$$
$$Y_{coordinate} = (Y_{record} * Y_{scale}) + Y_{offset}$$
$$Z_{coordinate} = (Z_{record} * Z_{scale}) + Z_{offset}$$

4.1.28 MAX and MIN X, Y, Z

The MAX and MIN X, Y, Z data fields are used to define the actual file X, Y, Z coordinate extents of the POINT DATA records. Note that ORIGIN, SCALE and OFFSET must be applied, where appropriate, to these values.

5 POINT RECORD METADATA BLOCK

5.1 POINT RECORD METADATA BLOCK Description

The POINT RECORD METADATA BLOCK record is used to specify the content, structure, and format of the POINT DATA records. Different POINT DATA record structures can be described by specifying the LAS 2.0 field names and DATATYPE Monikers.

The structure of the POINT RECORD METADATA BLOCK record is a consecutive set of zero (null) terminated string pairs that comprise the field names and the associated DATATYPE. All fields defined are prefixed with “LASF_”. Field Name and Format strings are not case sensitive. Thus the POINT RECORD METADATA BLOCK field value in the Figure 5-1 examples

Example 1		Example 2		Example 3	
Byte	Value	Byte	Value	Byte	Value
1	l	1	L	1	l
2	a	2	a	2	a
3	s	3	s	3	S
4	f	4	f	4	f
5	_	5	_	5	_
6	x	6	x	6	X
7	null	7	null	7	null
8	i	8	I	8	i
9	4	9	4	9	4
10	null	10	null	10	null

Table 5-1 POINT RECORD METADATA BLOCK Format Examples

all have the same meaning.

Users can extend and customize the point record specification by defining their own data types in an analogous manner. For example, if atmospheric pressure were desired by the XYZCorp with DATATYPE R4 on a point by point basis, the following field XYZCORP_Atm_Press could be added to the POINT RECORD METADATA BLOCK.

User Custom POINT RECORD METADATA BLOCK field example	
Byte	Value
1	X
2	Y
3	Z
4	C
5	o
6	r
7	p
8	_
9	A
10	t
11	m
12	_
13	P
14	r
15	e
16	s

17	s
18	null
19	R
20	4
21	null

Table 5-2 User Custom POINT RECORD METADATA BLOCK Field Example

Custom POINT DATA record definitions should be registered using the Key Registration process to ensure that their identifying string is unique (e.g. "XYZCorp" in the above example). Registration Keys can be obtained from ASPRS.

5.2 POINT DATA Record Defined Content

Table 5-1 lists all of the LAS 2.0 POINT DATA record standard fields and their conditionality and cardinality. The order of fields within the POINT DATA records is defined by the order of the field descriptors in the POINT RECORD METADATA BLOCK. If the file is to be compatible with LAS 1.1, the required fields must be in the same order as in the LAS 1.1 specification as listed Table 5-3 and use the same DATATYPES.

The field "LASF_END" is required to be the last field within the POINT RECORD METADATA BLOCK. The field "LASF_END" is only included in the POINT RECORD METADATA BLOCK. The POINT DATA records do not contain an end of record indicator.

POINT DATA Record Field Name	Valid Formats	Conditionality, Cardinality[1]
LASF_X	I4, I8, R4, R8, R10	M, 1
LASF_Y	I4, I8, R4, R8, R10	M, 1
LASF_Z	I4, I8, R4, R8, R10	M, 1
LASF_INTENSITY	UI1, UI2, UI4, R4, R8	O, 1
LASF_RETURN_PACKET	BF1	C[2], 1
LASF_CLASSIFICATION	UI1	M, 1
LASF_AIRBORNE_SCANNER_PACKET	UI2	O, 1
LASF_USER_DATA	UI1	O, 1
LASF_POINT_SOURCE_ID	UI2, UI4, UI8	M, 1
LASF_GPS_WEEK_TIME	R8	O, 1
LASF_GPS_WEEK	UI2	O, 1
LASF_SIGMA_X	R4, R8	O, 1
LASF_SIGMA_Y	R4, R8	O, 1
LASF_SIGMA_Z	R4, R8	O, 1
LASF_PAN	UI1, UI2, UI4, R4, R8	O, 1
LASF_RED	UI1, UI2, UI4, R4, R8	O, 1
LASF_GREEN	UI1, UI2, UI4, R4, R8	O, 1
LASF_BLUE	UI1, UI2, UI4, R4, R8	O, 1
LASF_NIR	UI1, UI2, UI4, R4, R8	O, 1
LASF_ATTRIBUTES	BF1	M, 1
LASF_CLASS_CONFIDENCE	UI1	O, 1
LASF_GROUP_ID	UI1, UI2, UI4, UI8	O, 1
LASF_POSIX_TIME	R8	O, 1
LASF_PAD	UI1, UI2, UI4, UI8	O, 1
USER DEFINED FIELDS	User Defined	O, 1
LASF_END	N/A	M, 1

Table 5-3 POINT DATA Record Field Names

[1]M = Mandatory, C = Conditional (as determined by SYSTEM IDENTIFICATION in the FILE HEADER BLOCK), O = Optional, , 1 = Mandatory if compatible with LAS 1.1 Record Type 1 and must be of the same data type as in LAS 1.1.

[2]Mandatory if these data were produced by an airborne sensor

5.3 POINT DATA Record Field Definitions

The definition of the POINT DATA record fields are contained in the following subsections.

5.3.1 LASF_X, LASF_Y, LASF_Z

The LASF_X, LASF_Y, and LASF_Z fields are used to specify the POINT DATA coordinate values expressed in the units defined by COORDINATE SYSTEM TYPE in the FILE HEADER BLOCK. The corresponding X SCALE, Y SCALE, and Z SCALE and X OFFET, Y OFFSET, and Z OFFSET values from the FILE HEADER BLOCK must be applied if the corresponding “APPLY SCALING” and/or “APPLY OFFSETS” flag values are set to “TRUE”.

5.3.2 LASF_INTENSITY

The LASF_INTENSITY field is used to specify the return pulse integrated energy magnitude value. This value is system specific. This field should never be used for encoding an auxiliary value such as intensity provided by an associated photographic camera.

5.3.3 LASF_RETURN_PACKET

The LASF_RETURN_PACKET field is used to specify the number of this return pulse (return number) and the total number of returns. The lower 4 bits represent the number of this return pulse (return number) and the upper four bits represent the total number of returns that were detected for this pulse.

The return number is the pulse return number for a given output pulse. A given output laser pulse can have many returns and they must be marked in sequence of return. The first return will have a return number of one, the second a return number of two, and so on up to fifteen possible returns. Multi-return pulses <u>must</u> be sequentially placed in the LAS file. That is, for a single fired pulse with 3 detected returns, the data records must be in the sequence: return 1 followed by return 2 followed by return 3 with no other intervening data. Software applications must be extremely careful regarding this rule since it is easy to mix data pulses when merging or extracting files.

The number of returns for an emitted pulse is the total number of detected returns for that given pulse. For example, a laser data point may be return two (return number) with a total number of five detected returns.

The value zero (null) is reserved for the special case of a pulse being emitted by the sensor but no return pulse was detected (a null pulse). For this case, the number of returns field will also be set to zero (null). The X, Y, Z coordinate values are invalid but the GPS Time, if included in the data format, represents the time the pulse was fired. Not all sensors are capable of detecting the occurrence of these null pulses.

5.3.4 LASF_CLASSIFICATION

The LASF_CLASSIFICATION field is used to specify the classification value of the object reflected by the laser pulse. If a POINT DATA record has never been classified, the LASF_CLASSIFICATION value is set to zero. If custom-defined classification values are required, the user should define a new custom-defined field for this purpose (e.g. XYZCORP_User_Classification) by setting the b:7 bit. If a custom CLASSIFICATION field is added, the LASF_CLASSIFICATION standard field must also be populated. For example, if a user elects to define a number of different building classifications, then the LASF_CLASSIFICATION Building (Generic) field value should be set in the LASF_CLASSIFICATION standard classification and the specific type of building in the user defined classification. Setting the high bit of the LASF_CLASSIFICATION byte indicates that the user has supplied a custom classification field that contains additional classification information. In the above example, the user could set a value of 6 for Building (Generic), setting the high bit of the LASF_CLASSIFICATION byte and indicate specific building types in a custom classification field.

Organizations wishing to submit for consideration additional standard LASF_CLASSIFICATION values for inclusion in this standard should contact ASPRS. Table 5-2 defines the LASF_CLASSIFICATION Point Cloud Classifications.

Classification Value (b:0 –b:6)	Classification Name	Description
0	Created, never classified	Points whose parent dataset have never been assigned classification values
1	Unclassified[3]	Points that have been subjected to a classification algorithm or manual classification but emerged in an undefined state
2	Ground (bare earth surface)	Points that have been classified as representing the bare earth surface.
3	Low Vegetation	Points reflected from vegetative features that are within the vertical range of .03 meter to 3 meters above the bare earth surface. (From FAO Land Cover Classification System, 2000)
4	Medium Vegetation	Points reflected from vegetative features that are within the vertical range of .3 meter to 5 meters above the bare earth surface. (From FAO Land Cover Classification System, 2000)
5	High Vegetation	Points reflected from vegetative features that are within the vertical range of 3 meter to over 30 meters above the bare earth surface. (From FAO Land Cover Classification System, 2000)
6	Building (Generic)	Points that have reflected off a man-made planar surfaces and are above the bare earth surface.

Classification Value (b:0 –b:6)	Classification Name	Description
7	Low Point	Erroneous or variable points which are uncharacteristically lower than other points in the vicinity. Variable and flexible comparison
8	High Point	Erroneous or variable points which are uncharacteristically higher than other points in the vicinity. Variable and flexible comparison
9	Water	Points that have reflected off of the water surface
10	Bridge	Points that could be classified as bare earth, but are man-made structures and must be removed from the bare earth surface to remove impediments to stream or road flow
11	Road	Points that are classified as having been reflected off of man-made automotive transportation networks
12	Miscellaneous Structure	Any man-made structure that does not meet the criteria of a building.
13	Railroad	Points that are classified as having been reflected off of railroad transportation networks
14	Stream	A point that has not been classified as water, and was reflected off a natural directional moving water source, such as a stream, river or creek.
15	Power Line	Points that are reflections from overhead power transmission lines
16	Power Line Tower	Points that are reflections from structures or pylons that are used to support overhead electricity conductors
17	Power Line Tower to wire connection	Points that represent the intersection between Power Line Towers and Power Lines.
18	Seafloor Bottom	Points that are reflections of the seabed.

Classification Value (b:0 –b:6)	Classification Name	Description
19	Underwater Navigational Hazard	Points that are reflections of any obstacle encountered by a vessel in route posing risk or danger to the vessel, its contents or the environment
20	Water Surface object	Points that are reflections from bathymetric structures or objects on the water surface (e.g. a buoy)
21	Aquatic vegetation	Points that are reflections of aquatic vegetation in the water column between the water surface and seafloor
22-127	Reserved for Future Classifications	
b7 (high bit set)	Classification provided in a custom classification field	

Table 5-4 LASF_CLASSIFICATION Point Cloud Classifications

[3]Both a classification value of 0 and 1 may used to indicate a classification of ***Unclassified*** to maintain compatibility with current popular classification software such as TerraScan. A classification value 1 may include cases in which data have been subjected to a classification algorithm but emerged in an undefined state. For example, data with class 0 is sent through an algorithm to detect man-made structures – points that emerge without having been assigned as belonging to structures could be remapped from class 0 to class 1.

5.3.5 LASF_AIRBORNE_SCANNER_PACKET

The LASF_AIRBORNE_SCANNER_PACKET field is used to specify information about the scanner mirror (or equivalent fiber-optic data) for airborne laser post-processing software packages. Table 5-3 defines the encoded fields for the AIRBORNE_SCANNER_PACKET.

Bit(s)	**Name**	**Description**
0	Scan Direction Flag	The scan direction flag denotes the direction at which the scanner mirror was traveling at the time of the output pulse. A bit value of 1 is a positive scan direction, and a bit value of 0 is a negative scan direction. The positive direction is taken to be from left side of flight line to right side of flight line when viewed in the direction of travel of the platform.
1	Edge of Flight Line	The edge of flight line data bit has a value of 1 only when the point is at the end of a scan. It is the last point on a given scan line before the scanner mirror changes direction.
2	Mirror Centered	This point was acquired with the scan mirror at its most central position (This would be nadir if the aircraft roll were zero)
3:7	RESERVED	

8:15	Scan Angle Rank	The scan angle rank is a signed 8 bit number with a valid range from -90.00 to +90.00. The scan angle rank is the angle (rounded to the nearest integer in the absolute value sense) at which the laser point was output from the laser system excluding[4] the roll of the aircraft. The scan angle is within 1 degree of accuracy from +90 to –90 degrees. The scan angle is an angle based on 0 degrees being NADIR, and –90 degrees to the left side of the aircraft in the direction of flight.

Table 5-5 LASF_AIRBORNE_SCANNER_PACKET

The LASF_AIRBORNE_SCANNER_PACKET is used when it is a requirement to indicate backward compatibility with LAS 1.0 and LAS 1.1. It is strongly recommended that a custom field be added for rich data content, when the fields of LASF_AIRBORNE_SCANNER_PACKET cannot adequately represent the system parameters

[4]Note: This is a change from version 1.1 where aircraft roll was included in scan angle rank.

5.3.6 LASF_USER_DATA

The LASF_USER_DATA field is used to specify a file exchange compatibility value with LAS 1.1. It is required only when LAS 2.0 POINT DATA record types impersonate LAS 1.1 POINT DATA record types.

5.3.7 LASF_POINT_SOURCE_ID

The LASF_POINT_SOURCE_ID field is used to specify the initial source data file value from which a POINT DATA record originated. This field value is used as a trace back to original point sources. For example, if the original files were from airborne flight lines (one file per line) and a composite file were created by merging a number of these flight line files, the LASF_POINT_SOURCE_ID would be equivalent to the flight line number.

Once assigned, a LASF_POINT_SOURCE_ID for a POINT DATA record never changes. Therefore the LASF_POINT_SOURCE_ID combined with the PROJECT ID (see FILE HEADER BLOCK) can be used to trace the original source of an LAS POINT DATA record regardless of the number of files through which the point has propagated.

Valid values for this field are 1 to the maximum count that can be contained in the data type selected. The numerical value corresponds to the FILE ID from which this point originated.

A LASF_POINT_SOURCE_ID value of zero is reserved to identify that a specific point originated in a specific file. Therefore processing software must set the LASF_POINT_SOURCE_ID equal to the FILE ID of the file containing this point during a merge or extract processing operation.

5.3.8 LASF_GPS WEEK TIME

The GPS Week Time is measured in seconds and is associated with a point. The time must match the time used in an associated trajectory file. The time starts at zero on midnight of Saturday and repeats each week. The typical resolution is 200 Hz. GPS Week Time has been carried over to LAS 2.0 to allow point record compatibility with LAS 1.1 Point Record type 1. Users are strongly discouraged from using GPS Week Time since merging files from different GPS Weeks results in ambiguous times.

5.3.9 LASF_GPS_WEEK

Users are strongly encouraged to use GPS Week number if they are using GPS Week Time (and not trying to clone LAS 1.1 record structures). The GPS Week Number count began at approximately midnight on the evening of 05 January 1980 / morning of 06 January 1980. Since that time, the count has been incremented by 1 each week, and broadcast as part of the GPS message. The GPS Week Number field is modulo 1024. This means that at the completion of week 1023, the GPS week number rolled over to 0 on midnight GPS Time of the evening of 21 August 1999 / morning of 22 August 1999. Note that this corresponded to 23:59:47 UTC on 21 August 1999.

5.3.10 LASF_SIGMA_X, LASF_SIGMA_Y, LASF_SIGMA_Z

The LASF_SIGMA_X, LASF_SIGMA_Y, LASF_SIGMA_Z fields are used to specify the standard deviation value of each POINT DATA record X, Y and Z coordinate value. The LASF_SIGMA_X, LASF_SIGMA_Y, LASF_SIGMA_Z values should be set to 0 for absolute known values and less than zero for completely unknown values. The LASF_SIGMA_X, LASF_SIGMA_Y, LASF_SIGMA_Z metric units are specified by the HORIZONTAL METRIC UNITS and VERTICAL METRIC UNITS in the FILE HEADER BLOCK.

5.3.11 LASF_PAN, LASF_RED, LASF_GREEN, LASF_BLUE, LASF_NIR

The LASF_PAN, LASF_RED, LASF_GREEN, LASF_BLUE, LASF_NR fields are used to specify the potential Panchromatic – LASF_PAN (sensitive to all visible wavelengths; 0.4 – 0.7 microns), Red – LASF_RED (approximately 0.6-0.7 microns), Green – LASF_GREEN (approximately 0.5-0.6 microns), Blue – LASF_BLUE (approximately 0.4-0.5 microns) and/or Near-infrared – LASF_NIR (approximately 0.75-1.4 microns) values that have been associated with the POINT DATA records (usually from an auxiliary camera in the case of a laser sensor or from image draping in the case of generalized elevation data).

Only LASF_PAN, LASF_RED, LASF_GREEN, LASF_BLUE, LASF_NR values that actually originated from the specified channel type in these parameters should be stored. For example, storing panchromatic data values in the green channel will cause data from different files to be improperly merged.

5.3.12 LASF_ATTRIBUTES

The LASF_ATTRIBUTES field is used to define flag values that signify other general information conditions of the POINT DATA records. Table 5-4 defines the LASF_ATTRIBUTE bit field values.

Bits	LASF_ATTRIBUTE type	Description
0	Class Verified	This point's class is considered known (e.g. if the class is Ground and this bit is set, the point is known to be ground to a "very high" degree of certainty). Note that if this bit is not set, the meaning is that no statement is being made regarding confidence of classification. This field is useful when the optional "LASF_CLASS_CONFIDENCE" byte is not in use.
1	Control Point	This point has been inserted or marked for control purposes.
2	Breakline[5]	This point forms a vertex of a breakline – a line of defined elevation that is added to the surface model to represent and model surface discontinuities

Bits	LASF_ATTRIBUTE type	Description
3	Noise	This point has been identified as noise, i.e., a point that is identified as not consistent and beyond the normal bounds of the landscape the project has captured.
4	Overlap	If set, this point has been marked as being in the overlap region of two or more scans (NOTE – It is not mandatory that this bit be set and thus one should not conclude that a Zero in this position implies that the point is not in an overlap region)
5	Synthetic	If set then this point was created by a technique other than direct sensor collection such as digitized from a photogrammetric stereo model.
6	Key	If set, this point is considered to be a model key-point (a subset of points that retains all or most the elevation changes in the dataset) and thus generally should not be withheld in a thinning algorithm.
7	Withheld	If set, this point should not be included in processing (synonymous with Deleted).

Table 5-6 LASF_ATTRIBUTES bit field values

[5]Breakline topology can be determined by using Group number (identifies membership in a specific breakline) and Time_Stamp (identifies the order of vertices within the group).

Note that bits are treated as flags and can be set or clear in any combination. For example, a POINT DATA record with LASF_ATTRIBUTE bits 1, 5 and 6 all set to one and other bits set to 0 would be a *control* point that had been *Synthetically* collected and marked as a *model key-point*.

5.3.13 LASF_CLASS_CONFIDENCE

The LAS_CLASS_CONFIDENCE field is used to specify the confidence of the classification of the POINT DATA record. A value of 0 means no confidence and a value of 255 means absolute confidence.
Note that the LASF_ATTRIBUTES bit field 0 contains a bit value that can be used to signify absolute confidence in a classification. This is useful when the confidence only needs to be known for the absolute condition.

5.3.14 LASF_GROUP_ID

The LASF_GROUP_ID field is used to specify a value that indicates an association between POINT DATA records. For example, LASF_GROUP_ID could be used as the index (key value) in an external database used for feature coding (e.g. GROUP_ID 1124 = First State Bank implying all points encoded with a LASF_GROUP_ID of 1124 are reflecting from this particular structure).

LASF_GROUP_ID used in conjunction with LASF_ATTRIBUTES and LASF_CLASSIFICATION to allow the construction of context. For example, points with the breakline attribute set, class set to Stream and GLASF_GROUP_ID equal to 117 would be known to be forming a linear breakline for hydrography. The POINT DATA record LASF_POSIX TIME value would be used to determine vertex order.

5.3.15 LASF_POSIX_TIME

The LASF_POSIX_TIME field is used to specify the time, in seconds, measured since January 1, 1970 in Universal Coordinated Time (UTC), in a POINT DATA record. This time must match the time used in an associated trajectory file. The typical resolution is microseconds. Users are strongly encouraged to use POSIX time rather than GPS Week Time since POSIX time will be unique across all data files.

5.3.16 LASF_PAD

The LASF_PAD field is used to define a value that is used to “pad” a POINT DATA record to a particular length or number of bytes. It is typically used to address machine specific address alignment issues. The content of the LASF_PAD field is undefined. LASF_PAD field values should never be used to carry information because they are not required to be included or used during file merge and extract operations.

5.3.17 USER DEFINED FIELDS

Users can extend POINT DATA records to meet unique user requirements. POINT DATA records are extended by creating a unique field name and supporting data type declaration and providing that extension information in the POINT RECORD METADATA BLOCK. The unique field name must be prefixed by the user’s Registration Key. Users can obtain a Registration Key from ASPRS.

For example if the XXZ Corporation registered the registration key of “XYZCORP”, they could add a custom POINT DATA record field by adding the custom registration key field name and data type to the POINT RECORD METADATA BLOCK record description as shown in Table 5-7.:

BYTE										
Value	X	Y	Z	C	O	R	P	_	E	n
BYTE										
Value	e	r	g	y	null	R	4	null		

Table 5-7 USER DEFINED FIELDS Example

Note that the only field that must be placed in a specific location in the POINT RECORD METADATA BLOCK record is the terminal field, “LASF_END” – see section 5.3.18

5.3.18 LASF_END

The LASF_END field is used to define the end of the POINT RECORD METADATA BLOCK record by the string “LASF_END”. This string must be the last field in the POINT RECORD METADATA BLOCK record description. It is immediately followed by the string null terminator which is then immediately followed by the first byte of the first variable length record. All LAS 2.0 compliant files must end the POINT RECORD METADATA BLOCK with a value “LASF”.

5.4 POINT RECORD METADATA BLOCK Example

An example of a typical POINT RECORD METADATA BLOCK record for an airborne sensor is:

BYTE	1	2	3	4	5	6	7	8	9	10
Value	L	A	S	F	_	P	O	S	I	X
BYTE	11	12	13	14	15	16	17	18	19	20
Value	_	T	I	M	E	n	R	8	n	L
BYTE	21	22	23	24	25	26	27	28	29	30
Value	A	S	F	_	X	n	R	4	n	L

BYTE	31	32	33	34	35	36	37	38	39	40
Value	A	S	F	_	Y	n	R	4	n	L
BYTE	41	42	43	44	45	46	47	48	49	60
Value	A	S	F	_	Z	n	I	4	n	L
BYTE	61	62	63	64	65	66	67	68	69	70
Value	A	S	F	_	I	N	T	E	N	S
BYTE	71	72	73	74	75	76	77	78	79	80
Value	I	T	Y	n	U	I	1	n	L	A
BYTE	81	82	83	84	85	86	87	88	89	90
Value	S	F	_	C	L	A	S	S	I	F
BYTE	91	92	93	94	95	96	97	98	99	100
Value	I	C	A	T	I	O	N	n	U	I
BYTE	101	102	103	104	105	106	107	108	109	110
Value	1	n	L	A	S	F	_	A	T	T
BYTE	111	112	113	114	115	116	117	118	119	120
Value	R	I	B	U	T	E	S	n	U	I
BYTE	121	122	123	124	125	126	127	128	129	130
Value	1	L	A	S	F	_	R	e	t	u
BYTE	131	132	133	134	135	136	137	138	139	140
Value	r	n	_	p	a	c	k	e	t	n
BYTE	141	142	143	144	145	146	147	148	149	150
Value	U	I	1	L	A	S	F	_	P	O
BYTE	151	152	153	154	155	156	157	158	159	160
Value	I	N	T	_	S	O	U	R	C	E
BYTE	161	162	163	164	165	166	167	168	169	170
Value	_	I	D	n	U	I	4	n	L	A
BYTE	171	172	173	174	175	176				
Value	S	F	_	E	N	D				

Table 5-8 POINT RECORD METADATA BLOCK Example

Note n=null

Note that in this example, the data types for LASF_X, LASF_Y and LASF_Z coordinates are different and the POINT DATA record fields are in a different order then that listed in Table 5-1 POINT DATA Record Field Names. Also notice that there is a mix of upper and lower case characters. For this example, the size of each POINT DATA record is 28 bytes.

6 VARIABLE LENGTH RECORDS

The POINT RECORD METADATA BLOCK is followed by one or more VARIABLE LENGTH RECORDS. There is one mandatory VARIABLE LENGTH RECORD – LASF_WKT_Primary,. The number of variable length records is specified in the NUMBER OF VARIABLE LENGTH RECORDS field in the FILE HEADER BLOCK. The variable length records must be accessed sequentially since the size of each variable length record is contained in the VARIABLE LENGTH RECORD Header. Each VARIABLE LENGTH RECORD Header is 58 bytes in length.

6.1 VARIABLE LENGTH RECORD Header

Each VARIABLE LENGTH RECORD begins with a fixed size header: Table 6-1 defines the VARIABLE LENGTH RECORD Header

Item	Format	Size	Conditionality M = mandatory O = optional
USER ID	UI1[16]	16 bytes	M
RECORD ID	UI2	2 bytes	M
DESCRIPTION	UI1[32]	32 bytes	O
RECORD LENGTH AFTER HEADER	UI8	8 bytes	M

Table 6-1 VARIABLE LENGTH RECORD Header

6.1.1 USER ID

The USER ID field is used to define ASCII character data that identifies the user which created the variable length record. It is possible to have many variable length records from different sources with different user IDs. If the character data is less than 16 characters, the remaining data values must be zero (null). Users are encouraged to register their USER ID registration key with ASPRS. The registration of USER IDs ensures that they are unique. The user is not permitted to redefine the POINT RECORD METADATA BLOCK value “LASF”.

6.1.2 RECORD ID

The RECORD ID field is a value from 0 to 65,535 and is used to associate the RECORD ID with the USER ID. Each user manages their own RECORD IDs. USER IDs should be registered with ASPRS otherwise RECORD IDs will be managed by the owner of the given User ID. Thus each USER ID is allowed to assign 0 to 65535 RECORD IDs in any manner they desire. Publicizing the meaning of a given RECORD ID is the responsibility of the given USER ID. Unknown USER ID/RECORD ID combinations should be ignored.

6.1.3 DESCRIPTION

The DESCRIPTION field is used to provide additional user text descriptions of the data. Any characters not used must be set to a value zero (null).

6.1.4 RECORD LENGTH AFTER HEADER

The RECORD LENGTH AFTER HEADER field is used to specify the number of bytes for the RECORD LENGTH AFTER HEADER record after the end of the standard part of the header. Thus the entire record length is 58 bytes (the header size) plus the number of bytes in the variable length portion of the record.

6.2 Georeferencing Information

LAS 2.0 uses the Well Known Text (WKT) mechanism (promulgated by the Open Geospatial Consortium (OGC) and commonly supported by software vendors) for georeferencing. The WKT convention used is the ESRI[6] compatible version (that is, the LAS WKT can be directly used in ArcGIS products).

6.2.1 PRIMARY WKT RECORD (MANDATORY IF COORDINATE SYSTEM TYPE = 1)

USER ID: LASF
RECORD ID: 0
DESCRIPTION: Primary_WKT
RECORD LENGTH AFTER HEADER: Variable

The PRIMARY WKT RECORD uses the literal ESRI compatible Well Known Text (WKT) definition of the coordinate system when the coordinate system type equals a value of 1. It should encode both the horizontal and vertical systems. The WKT is a null terminated string. Note that this string, if literally written to disk as a text file with the extension ".prj" can be read in as the coordinate system definition to any ESRI coordinate system compatible software.

Record Content:
STR – ESRI compliant WKT coordinate system null terminated string

6.2.2 SECONDARY WKT RECORD (OPTIONAL)

USER ID: LASF
RECORD ID: 1
DESCRIPTION: Secondary_WKT
RECORD LENGTH AFTER HEADER: Variable

The Secondary WKT Record uses the literal OGC compatible Well Known Text (WKT) definition of the coordinate system. It should encode both the horizontal and vertical systems. The WKT is a null terminated string.

Record Content:
STR – OGC compliant WKT coordinate system null terminated string

Note: This secondary record is used for two purposes. The first is that standard coordinate systems passed around by processing software often do not encode the vertical coordinate system. In these cases, users can encode the vertical system in the secondary key without rendering the primary key incompatible with processing software that misinterprets the vertical tags. The second use of this record is to support cases in which processing software can interpret OCS WKS encodings but not ESRI WKT variations.

[6]ESRI and ArcGIS are registered trademarks of Environmental Sciences Research Institute, Inc.

Annex A

Endian Byte Notation

Data storage in different computers uses a byte notation convention known as either little endian or big endian storage. The storage convention generally applies to numeric values that span multiple bytes, as follows:

- Little endian storage occurs when:
  - The least significant bit (LSB) value is in the byte with the lowest address.
  - The most significant bit (MSB) value is in the byte with the highest address.
  - The address of the numeric value is the byte containing the LSB. Subsequent bytes with higher addresses contain more significant bits.
- Big endian storage occurs when:
  - The least significant bit (LSB) value is in the byte with the highest address.
  - The most significant bit (MSB) value is in the byte with the lowest address.
  - The address of the numeric value is the byte containing the MSB. Subsequent bytes with higher addresses contain less significant bits.

Figure C-1 shows the difference between the two byte-ordering schemes:

Little and Big Endian Storage of an INTEGER Value

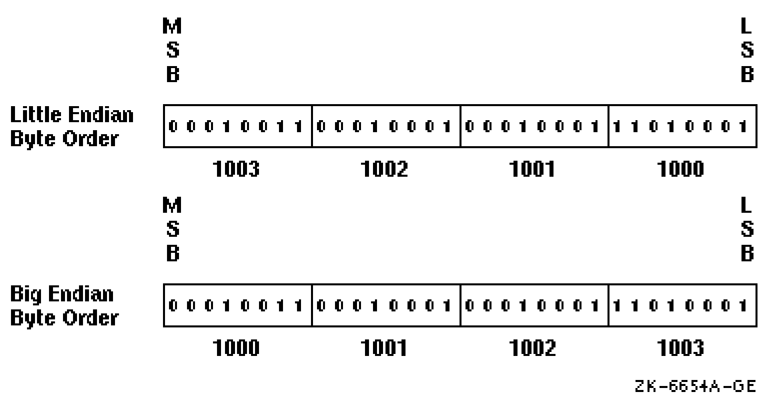

Figure A-1 Endian Byte Notation