LAS
Specification
Version 1.2
April 29, 2008

LAS FORMAT VERSION 1.2:

This document reflects the second revision of the LAS format specification since its initial version 1.0 release. Version 1.2 retains the same structure as version 1.1 including identical field alignment.   LAS 1.1 file Input/Output (I/O) libraries will require slight modifications in order to be compliant with this revision.   A LAS 1.1 Reader will read LAS 1.2 (without the new enhancements) with no modifications.

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 in the LIDAR committee section.

The additions of LAS 1.2 include:
- GPS Absolute Time (as well as GPS Week Time) – LAS 1.0 and LAS 1.1 specified GPS "Week Time" only. This meant that GPS time stamps "rolled over" at midnight on Saturday. This makes processing of LIDAR flight lines that span the time reset difficult. LAS 1.2 allows both GPS Week Time and Absolute GPS Time (POSIX) stamps to be used.
- Support for ancillary image data on a per point basis. You can now specify Red, Green, Blue image data on a point by point basis. This is encapsulated in two new point record types (type 2 and type 3).

LAS FORMAT DEFINITION:

The LAS file is intended to contain LIDAR point data records. The data will generally be put into this format from software (e.g. provided by LIDAR hardware vendors) which combines GPS, IMU, and laser pulse range data to produce X, Y, and Z point data. The intention of the data format is to provide an open format that allows different LIDAR hardware and software tools to output data in a common format.

The format contains binary data consisting of a header block, Variable Length Records, and point data.

PUBLIC HEADER BLOCK
VARIABLE LENGTH RECORDS
POINT DATA RECORDS
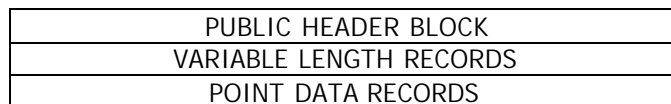
All data is in little-endian format. The header block consists of a public block followed by Variable Length Records. The public block contains generic data such as point numbers and coordinate bounds. The Variable Length Records contain variable types of data including projection information, metadata, and user application data.

DATA TYPES:

The following data types are used in the LAS format definition.

- char (1 byte)
- unsigned char (1 byte)
- short (2 bytes)
- unsigned short (2 bytes)

- long (4 bytes)
- unsigned long (4 bytes)
- double (8 byte IEEE floating point format)

PUBLIC HEADER BLOCK:

Item	Format	Size	Required
File Signature ("LASF")	char[4]	4 bytes	*
File Source ID	unsigned short	2 bytes	*
Global Encoding	unsigned short	2 bytes	*
Project ID - GUID data 1	unsigned long	4 bytes	
Project ID - GUID data 2	unsigned short	2 byte	
Project ID - GUID data 3	unsigned short	2 byte	
Project ID - GUID data 4	unsigned char[8]	8 bytes	
Version Major	unsigned char	1 byte	*
Version Minor	unsigned char	1 byte	*
System Identifier	char[32]	32 bytes	*
Generating Software	char[32]	32 bytes	*
File Creation Day of Year	unsigned short	2 bytes	
File Creation Year	unsigned short	2 bytes	
Header Size	unsigned short	2 bytes	*
Offset to point data	unsigned long	4 bytes	*
Number of Variable Length Records	unsigned long	4 bytes	*
Point Data Format ID (0-99 for spec)	unsigned char	1 byte	*
Point Data Record Length	unsigned short	2 bytes	*
Number of point records	unsigned long	4 bytes	*
Number of points by return	unsigned long[5]	20 bytes	*
X scale factor	double	8 bytes	*
Y scale factor	double	8 bytes	*
Z scale factor	double	8 bytes	*
X offset	double	8 bytes	*
Y offset	double	8 bytes	*
Z offset	double	8 bytes	*
Max X	double	8 bytes	*
Min X	double	8 bytes	*
Max Y	double	8 bytes	*
Min Y	double	8 bytes	*
Max Z	double	8 bytes	*
Min Z	double	8 bytes	*

Any field in the Public Header Block that is not required and is not used must be zero filled.

File Signature: The file signature must contain the four characters "LASF", and it is required by the LAS specification. These four characters can be checked by user software as a quick look initial determination of file type.

File Source ID (Flight Line Number if this file was derived from an original flight line): This field should be set to a value between 1 and 65,535, inclusive. A value of zero (0) is interpreted to mean that an ID has not been assigned. In this case, processing software is free to assign any

LAS 1.2   3

valid number. Note that this scheme allows a LIDAR project to contain up to 65,535 unique sources. A source can be considered an original flight line or it can be the result of merge and/or extract operations.

Global Encoding: This is a bit field used to indicate certain global properties about the file. In LAS 1.2 (the version in which this field was introduced), only the low bit is defined (this is the bit, that if set, would have the unsigned integer yield a value of 1). This bit field is defined as:

Global Encoding - Bit Field Encoding

Bits	Field Name	Description
0	GPS Time Type	The meaning of GPS Time in the Point Records 0 (not set) -> GPS time in the point record fields is GPS Week Time (the same as previous versions of LAS) 1 (set) -> GPS Time is POSIX Time. POSIX Time is defined to be the time, in seconds, measured since January 1, 1970 in Universal Coordinated Time (UTC). The resolution is typically microseconds.
1:15	Reserved	Must be set to zero

Note that in the previous version of LAS (LAS 1.1), this was a reserved field that had to be set to zero. Thus LAS 1.2 files that use GPS Week Time and point record types 0 and 1 (the only types supported in previous versions of LAS) will be identical to LAS 1.1 files with the exception of the minor version number.

Project ID (GUID data): The four fields that comprise a complete Globally Unique Identifier (GUID) are now reserved for use as a Project Identifier (Project ID). The field remains optional. The time of assignment of the Project ID is at the discretion of processing software. The Project ID should be the same for all files that are associated with a unique project. By assigning a Project ID and using a File Source ID (defined above) every file within a project and every point within a file can be uniquely identified, globally.

Version Number: The version number consists of a major and minor field. The major and minor fields combine to form the number that indicates the format number of the current specification itself. For example, specification number 1.2 (this version) would contain 1 in the major field and 2 in the minor field.

System Identifier: The version 1.0 specification assumes that LAS files are exclusively generated as a result of collection by a hardware sensor. Version 1.1 recognizes that files often result from extraction, merging or modifying existing data files. Thus System ID becomes:

Generating Agent	System ID
Hardware system	String identifying hardware (e.g. "ALTM 1210" or "ALS50"
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

Generating Software: This information is ASCII data describing the generating software itself. 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" and etc.). If the character data is less than 16 characters, the remaining data must be null.

File Creation Day of Year: Day, expressed as an unsigned short, on which this file was created. Day is computed as the Greenwich Mean Time (GMT) day. January 1 is considered day 1.

File Creation Year: The year, expressed as a four digit number, in which the file was created.

Header Size: The size, in bytes, of the Public Header Block itself. In the event that the header is extended by a software application through the addition of data at the end of the header, the Header Size field must be updated with the new header size. Extension of the Public Header Block is discouraged; the Variable Length Records should be used whenever possible to add custom header data. In the event a generating software package adds data to the Public Header Block, this data must be placed at the end of the structure and the Header Size must be updated to reflect the new size.

Offset to point data: The actual number of bytes from the beginning of the file to the first field of the first point record data field. This data offset must be updated if any software adds data from the Public Header Block or adds/removes data to/from the Variable Length Records.

Number of Variable Length Records: This field contains the current number of Variable Length Records. This number must be updated if the number of Variable Length Records changes at any time.

Point Data Format ID: The point data format ID corresponds to the point data record format type. LAS 1.2 defines types 0, 1, 2 and 3.

Point Data Record Length: The size, in bytes, of the Point Data Record.

Number of point records: This field contains the total number of point records within the file.

Number of points by return: This field contains an array of the total point records per return. The first unsigned long value will be the total number of records from the first return, and the second contains the total number for return two, and so forth up to five returns.

X, Y, and Z scale factors: The scale factor fields contain a double floating point value that is used to scale the corresponding X, Y, and Z long values within the point records. The corresponding X, Y, and Z scale factor must be multiplied by the X, Y, or Z point record value to get the actual X, Y, or Z coordinate. For example, if the X, Y, and Z coordinates are intended to have two decimal point values, then each scale factor will contain the number 0.01.

X, Y, and Z offset: The offset fields should be used to set the overall offset for the point records. In general these numbers will be zero, but for certain cases the resolution of the point data may not be large enough for a given projection system. However, it should always be assumed that these numbers are used. So to scale a given X from the point record, take the point record X multiplied by the X scale factor, and then add the X offset.

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

Max and Min X, Y, Z: The max and min data fields are the actual unscaled extents of the LAS point file data, specified in the coordinate system of the LAS data.

The projection information for the point data is required for all data. The projection information will be placed in the Variable Length Records. Placing the projection information within the Variable Length Records allows for any projection to be defined including custom projections. The GeoTIff specification http://www.remotesensing.org/geotiff/geotiff.html is the model for representing the projection information, and the format is explicitly defined by this specification.

VARIABLE LENGTH RECORDS:

The Public Header Block is followed by one or more Variable Length Records (There is one mandatory Variable Length Record, **GeoKeyDirectoryTag**). The number of Variable Length Records is specified in the "Number of Variable Length Records" field in the Public 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 54 bytes in length.

VARIABLE LENGTH RECORD HEADER

Item	Format	Size	Required
Reserved	unsigned short	2 bytes	
User ID	char[16]	16 bytes	*
Record ID	unsigned short	2 bytes	*
Record Length After Header	unsigned short	2 bytes	*
Description	char[32]	32 bytes	

User ID: The User ID field is 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 must be null. The User ID must be registered with the LAS specification managing body. The management of these User IDs ensures that no two individuals accidentally use the same User ID. The specification will initially use two IDs: one for globally specified records (LASF_Spec), and another for projection types (LASF_Projection). Keys may be requested at http://www.asprs.org/lasform/keyform.html.

Record ID: The Record ID is dependent upon the User ID. There can be 0 to 65535 Record IDs for every User ID. The LAS specification manages its own Record IDs (User IDs owned by the specification), 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 left to the owner of the given User ID. Unknown User ID/Record ID combinations should be ignored.

Record Length after Header: The record length is the number of bytes for the record after the end of the standard part of the header. Thus the entire record length is 54 bytes (the header size in version 1.2) plus the number of bytes in the variable length portion of the record.

Description: Optional, null terminated text description of the data. Any remaining characters not used must be null.

POINT DATA RECORD

 NOTE: Point Data Start Signature was removed in LAS Version 1.1. LAS file I/O software must use the **Offset to Point Data** field in the Public Header Block to locate the starting position of the first Point Data Record. Note that all Point Data Records must be the same type (0, 1, 2 or 3).

POINT DATA RECORD FORMAT 0:

Item	Format	Size	Required
X	long	4 bytes	*
Y	long	4 bytes	*
Z	long	4 bytes	*
Intensity	unsigned short	2 bytes	
Return Number	3 bits (bits 0, 1, 2)	3 bits	*
Number of Returns (given pulse)	3 bits (bits 3, 4, 5)	3 bits	*
Scan Direction Flag	1 bit (bit 6)	1 bit	*
Edge of Flight Line	1 bit (bit 7)	1 bit	*
Classification	unsigned char	1 byte	*
Scan Angle Rank (-90 to +90) – Left side	char	1 byte	*
User Data	unsigned char	1 byte	
Point Source ID	unsigned short	2 bytes	*

X, Y, and Z: The X, Y, and Z values are stored as long integers. The X, Y, and Z values are used in conjunction with the scale values and the offset values to determine the coordinate for each point as described in the Public Header Block section.
Intensity: The intensity value is the integer representation of the pulse return magnitude. This value is optional and system specific. However, it should always be included if available.

NOTE: The following four fields (Return Number, Number of Returns, Scan Direction Flag and Edge of Flight Line) are bit fields within a single byte.

Return Number: 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 five returns.

Number of Returns (for this emitted pulse): The Number of Returns is the total number of returns for a given pulse. For example, a laser data point may be return two (Return Number) within a total number of five returns.

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 (where positive scan direction is a scan moving from the left side of the in-track direction to the right side and negative the opposite).

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 it changes direction.

Classification: Classification in LAS 1.0 was essentially user defined and optional. LAS 1.1 defines a standard set of ASPRS classifications. In addition, the field is now mandatory. If a

point has never been classified, this byte must be set to zero. There are no user defined classes since both point format 0 and point format 1 supply 8 bits per point for user defined operations.

Note that the format for classification is a bit encoded field with the lower five bits used for class and the three high bits used for flags. The bit definitions are:

Classification Bit Field Encoding

Bits	Field Name	Description
0:4	Classification	Standard ASPRS classification as defined in the following classification table.
5	Synthetic	If set then this point was created by a technique other than LIDAR collection such as digitized from a photogrammetric stereo model.
6	Key-point	If set, this point is considered to be a model key-point 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).

Note that bits 5, 6 and 7 are treated as flags and can be set or clear in any combination. For example, a point with bits 5 and 6 both set to one and the lower five bits set to 2 (see table below) would be a *Ground* point that had been *Synthetically* collected and marked as a *model key-point*.

Classification must adhere to the following standard (we expect to assign the ASPRS Reserved values as LAS Version 1.1a, 1.1b, etc. augmentations):

ASPRS Standard LIDAR Point Classes

Classification Value (bits 0:4)	Meaning
0	Created, never classified
1	Unclassified[1]
2	Ground
3	Low Vegetation
4	Medium Vegetation
5	High Vegetation
6	Building
7	Low Point (noise)
8	Model Key-point (mass point)
9	Water
10	*Reserved for ASPRS Definition*
11	*Reserved for ASPRS Definition*
12	Overlap Points[2]
13-31	*Reserved for ASPRS Definition*

---

[1] We are using both 0 and 1 as *Unclassified* to maintain compatibility with current popular classification software such as TerraScan. We extend the idea of classification value 1 to 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.
[2] Overlap Points are those points that were immediately culled during the merging of overlapping flight lines. In general, the *Withheld* bit should be set since these points are not subsequently classified.

[A note on Bit Fields – The LAS storage format is "Little Endian." This means that multi-byte data fields are stored in memory from least significant byte at the low address to most significant byte at the high address. Bit fields are always interpreted as bit 0 set to 1 equals 1, bit 1 set to 1 equals 2, bit 2 set to 1 equals 4 and so forth.]

Scan Angle Rank: The Scan Angle Rank is a signed one-byte number with a valid range from -90 to +90. 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 including 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.

User Data: This field may be used at the user's discretion.

Point Source ID: This value indicates the file from which this point originated. Valid values for this field are 1 to 65,535 inclusive with zero being used for a special case discussed below. The numerical value corresponds to the File Source ID from which this point originated. Zero is reserved as a convenience to system implementers. A Point Source ID of zero implies that this point originated in this file. This implies that processing software should set the Point Source ID equal to the File Source ID of the file containing this point at some time during processing.

NOTE: The File Marker field in the LAS 1.0 structure was generally miscoded and/or not implemented by users. The entire concept was removed from LAS 1.1 and this single byte field has been renamed User Data and is available for any use. The extended records associated with this field in the original LAS 1.0 specification are removed. Please note that the field named User Bit Field has been renamed Point Source ID and is no longer available for general use.

POINT DATA RECORD FORMAT 1:

Item	Format	Size	Required
X	long	4 bytes	*
Y	long	4 bytes	*
Z	long	4 bytes	*
intensity	unsigned short	2 bytes	
Return Number	3 bits (bits 0, 1, 2)	3 bits	*
Number of Returns (given pulse)	3 bits (bits 3, 4, 5)	3 bits	*
Scan Direction Flag	1 bit (bit 6)	1 bit	*
Edge of Flight Line	1 bit (bit 7)	1 bit	*
Classification	unsigned char	1 byte	*
Scan Angle Rank (-90 to +90) – Left side	unsigned char	1 byte	*
User Data	unsigned char	1 byte	
Point Source ID	unsigned short	2 bytes	*
GPS Time	double	8 bytes	*

Point Data Record Format 1 is the same as Point Data Record Format 0 with the addition of GPS Time.

GPS Time: The GPS Time is the double floating point time tag value at which the point was acquired. It is GPS Week Time if the Global Encoding low bit is clear and POSIX Time if the Global Encoding low bit is set (*see Global Encoding in the Public Header Block description).*

POINT DATA RECORD FORMAT 2:

Item	Format	Size	Required
X	long	4 bytes	*
Y	long	4 bytes	*
Z	long	4 bytes	*
Intensity	unsigned short	2 bytes	
Return Number	3 bits (bits 0, 1, 2)	3 bits	*
Number of Returns (given pulse)	3 bits (bits 3, 4, 5)	3 bits	*
Scan Direction Flag	1 bit (bit 6)	1 bit	*
Edge of Flight Line	1 bit (bit 7)	1 bit	*
Classification	unsigned char	1 byte	*
Scan Angle Rank (-90 to +90) – Left side	unsigned char	1 byte	*
User Data	unsigned char	1 byte	
Point Source ID	unsigned short	2 bytes	*
Red	unsigned short	2 bytes	*
Green	unsigned short	2 bytes	*
Blue	unsigned short	2 bytes	*

Point Data Record Format 2 is the same as Point Data Record Format 0 with the addition of three color channels. These fields are used when "colorizing" a LIDAR point using ancillary data, typically from a camera.

Red: The Red image channel value associated with this point

Green: The Green image channel value associated with this point

Blue: The Blue image channel value associated with this point

POINT DATA RECORD FORMAT 3:

Item	Format	Size	Required
X	long	4 bytes	*
Y	long	4 bytes	*
Z	long	4 bytes	*
Intensity	unsigned short	2 bytes	
Return Number	3 bits (bits 0, 1, 2)	3 bits	*
Number of Returns (given pulse)	3 bits (bits 3, 4, 5)	3 bits	*
Scan Direction Flag	1 bit (bit 6)	1 bit	*
Edge of Flight Line	1 bit (bit 7)	1 bit	*
Classification	unsigned char	1 byte	*
Scan Angle Rank (-90 to +90) – Left side	unsigned char	1 byte	*
User Data	unsigned char	1 byte	
Point Source ID	unsigned short	2 bytes	*
GPS Time	double	8 bytes	*
Red	unsigned short	2 bytes	*
Green	unsigned short	2 bytes	*
Blue	unsigned short	2 bytes	*

Point Data Record Format 3 is the same as Point Data Record Format 2 with the addition of GPS Time.

DEFINED VARIABLE LENGTH RECORDS:

Georeferencing Information

Georeferencing for the LAS format will use the same robust mechanism that was developed for the GeoTIFF standard. The variable length header records section will contain the same data that would be contained in the GeoTIFF key tags of a TIFF file. With this approach, any vendor that has existing code to interpret the coordinate system information from GeoTIFF tags can simply feed the software with the information taken from the LAS file header. Since LAS is not a raster format and each point contains its own absolute location information, only 3 of the 6 GeoTIFF tags are necessary. The ModelTiePointTag (33922), ModelPixelScaleTag (33550), and ModelTransformationTag (34264) records can be excluded. The GeoKeyDirectoryTag (34735), GeoDoubleParamsTag (34736), and GeoASCIIParamsTag (34737) records are used.

Only the GeoKeyDirectoryTag record is required. The GeoDoubleParamsTag and GeoASCIIParamsTag records may or may not be present, depending on the content of the GeoKeyDirectoryTag record.

GeoKeyDirectoryTag Record: (mandatory)

User ID: LASF_Projection
Record ID: 34735

This record contains the key values that define the coordinate system. A complete description can be found in the GeoTIFF format specification. Here is a summary from a programmatic point of view for someone interested in implementation.

The GeoKeyDirectoryTag is defined as just an array of unsigned short values. But, programmatically, the data can be seen as something like this:

```
struct sGeoKeys
{
   unsigned short wKeyDirectoryVersion;
   unsigned short wKeyRevision;
   unsigned short wMinorRevision;
   unsigned short wNumberOfKeys;
   struct sKeyEntry
   {
     unsigned short wKeyID;
     unsigned short wTIFFTagLocation;
     unsigned short wCount;
     unsigned short wValue_Offset;
   } pKey[1];
};
```

Where:
wKeyDirectoryVersion = 1;   // Always
wKeyRevision = 1;   // Always
wMinorRevision = 0;   // Always

wNumberOfKeys   // Number of sets of 4 unsigned shorts to follow

For each set of 4 unsigned shorts:

Name	Definition
wKeyID	Defined key ID for each piece of GeoTIFF data. IDs contained in the GeoTIFF specification.
wTIFFTagLocation	Indicates where the data for this key is located: 0 means data is in the wValue_Offset field as an unsigned short. 34736 means the data is located at index wValue_Offset of the GeoDoubleParamsTag record. 34767 means the data is located at index wValue_Offset of the GeoAsciiParamsTag record.
wCount	Number of characters in string for values of GeoAsciiParamsTag , otherwise is 1
wValue_Offset	Contents vary depending on value for wTIFFTagLocation above

GeoDoubleParamsTag Record: (optional)

User ID: LASF_Projection
Record ID: 34736
This record is simply an array of doubles that contain values referenced by tag sets in the GeoKeyDirectoryTag record.

GeoAsciiParamsTag Record: (Optional)

User ID: LASF_Projection
Record ID: 34737
This record is simply an array of ASCII data. It contains many strings separated by null terminator characters which are referenced by position from data in the GeoKeyDirectoryTag record.

Classification lookup: (optional)

User ID: LASF_Spec
Record ID: 0
Record Length after Header: 255 recs X 16 byte struct len
struct CLASSIFICATION
{
        unsigned char ClassNumber;
        char Description[15];
};

Header lookup for flight-lines:
(Removed with Version 1.1 - Point Source ID in combination with Source ID provides the new scheme for directly encoding flight line number. Thus variable Record ID 1 now becomes reserved for future use.)

User ID: LASF_Spec
Record ID: 1

Histogram: (optional)

User ID: LASF_Spec
Record ID: 2

Text area description: (optional)

User ID: LASF_Spec
Record ID: 3