SEG-Y_r2.1: SEG-Y revision 2.1 Data Exchange format[1]

SEG Technical Standards Committee[2]

October 2023

[2] Editors Stewart A. Levin and Shawn New

Contents

FIGURES

TABLES

1. Introduction

Since the original SEG-Y Data Exchange Format (revision 0, - see Appendix G. References) was published in 1975 it has achieved widespread usage within the geophysical industry. The 2002 revision 1 extended the standard to 3-D acquisition and high-capacity media, reducing, though far from eliminating, the use of proprietary variations. Since the publication of SEG-Y rev 1, the nature of seismic data acquisition, processing and seismic hardware has continued to evolve and the SEG Technical Standards Committee undertook further revision. In line with the 2011 SEG-D rev 3.0 standard, this revision both addresses current industry data exchange needs and provides an explicit mechanism to support future expansion with both proprietary and officially adopted extensions. The SEG Technical Standards Committee strongly encourages producers and users of SEG-Y data sets to move to the revised standard in an expeditious fashion.

Users of this standard are cautioned that SEG-Y was not explicitly designed for use as a field recording format. The SEG-D or SEG-2 formats are recommended for this purpose.

2. Summary

2.1. Unchanged Items

- EBCDIC encoding allowed for most text
- The size of the original 3200-byte Textual File Header, 400-byte Binary File Header and initial 240-byte Trace Header

2.2. Changes from rev 2 to rev 2.1

- Binary header bytes 3507-10 changed to insert new survey type field
- Deprecated appendices D-3 through D-5 removed and subsequent appendices renumbered. They are still allowed by the format.
- A new stanza for XML-formatted Trace Header Layout was added. The existing Trace Header Mapping stanza is deprecated.
- Several corrections and clarifications in format description text

2.3. Changes from rev 1 to rev 2

- Provide for up to 65535 additional 240-byte trace headers with bytes 233-240 of each trace header reserved for trace header names
- Allow up to 2^{32} –1 samples per trace
- Allow arbitrarily large and small sample intervals
- Permit up to 2^{64}–1 traces per line and 2^{32}–1 traces per ensemble
- Support additional data sample formats, including IEEE double precision (64-bit)
- Support little-endian and pair-wise byte swapping to improve I/O performance.
- Support microsecond accuracy in time and date stamps
- Support additional precision on coordinates, depths, and elevations (especially useful for Lat/Long and UTM coordinates), and more options for coordinate reference system specification
- Require Extended Textual File Header stanzas to begin at 3200-byte boundaries and removed 40 80-byte line restriction
- Allow stanzas to appear after the last data trace
- Provide flexible trace header mapping options via Extended Textual File Headers. Because of this ability, we removed almost all “mandatory” and “highly recommended” header entry designations
- Allow XML-based Extended Textual File Header and Trailer stanzas for ease of machine encoding and decoding
- Include depth, velocity, EM, gravity and rotational sensor data

Optional 128-byte SEG-Y Tape Label	3200-byte Textual File Header	400-byte Binary File Header	1st 3200-byte Extended Textual File Header (Optional)	Nth 3200-byte Extended Textual File Header (Optional)	1 or more 240-byte Trace 1 Headers	1st Data Trace	1 or more 240-byte Trace M Headers	Mth Data Trace	Data Trailer 1 or more 3200-byte records (Optional)

Figure 1 Byte stream structure of a SEG-Y file with N Extended Textual File Header records and M traces records

2.4. Changes from rev 0 to rev 1

- A SEG-Y file may be written to any medium that is resolvable to a stream of variable length records
- The data word formats are expanded to include four-byte, IEEE floating-point and one-byte integer data words
- A small number of additional fields in the 400-byte Binary File Header and the 240-byte Trace Header are defined and the use of some existing entries is clarified
- An Extended Textual File Header consisting of additional 3200-byte Textual File Header blocks is introduced
- The data in the Extended Textual File Header uses a stanza layout and standard stanzas are defined
- Trace identification is expanded
- Engineering conversions are introduced
- The Textual File Header and the Extended Textual File Header can be encoded as EBCDIC or ASCII characters

2.5. Notation

The term CDP (common depth point) as used in this document is used as a synonym for the term CMP (common midpoint).

2.6. Controlling Organization

SEG-Y is administered by the SEG Technical Standards Committee. Any questions, corrections or problems encountered in the format should be addressed to:

Society of Exploration Geophysicists
10300 Town Park Dr. Ste SE 1000
Houston, TX 77072

Attention: SEG Technical Standards Committee

Phone: (918) 497-5500
Fax: (918) 497-5557
Internet site: www.seg.org

2.7. Acknowledgments

The SEG Technical Standards Committee would like to acknowledge the time and effort put forth by a great many individuals and organizations.

3. SEG-Y File Structure

The SEG-Y format is intended to be independent of the actual medium on which it is recorded. For this standard, the terms file and data set are synonymous. Both terms are a collection of logically related data traces or ensembles of traces and the associated ancillary data.

3.1. Recording Medium

A SEG-Y file may be written to any sequential medium. Whatever medium is used, the data must be resolvable to a stream of variable length logical records. This includes high-capacity tape devices, although with these it is desirable to use some kind of blocking and/or logical encapsulation such as SEG RODE (Booth et al., 1997), to use the tape more efficiently and possibly to allow the recording of associated metadata. Obviously, when seismic data are being exchanged in SEG-Y format, the medium and any blocking and/or encapsulation scheme used must be acceptable to both the provider and recipient of the data.

One important class of media on which SEG-Y data are exchanged is the byte stream without any record structure. It is common practice to write SEG-Y data to disk, including USB sticks, CD and DVD-ROM, or streamed through a network for data distribution. Certain rules must be followed for this to work correctly. Appendix A defines how SEG-Y data should be written as a byte stream.

In order to make SEG-Y consistent with the SEG-D Rev 3.0 standard, Appendix B defines a tape label for SEG-Y tapes, using a format based on the RP66 Storage Unit Label. Labels are not mandatory for SEG-Y, but their use is highly desirable in environments such as robotic tape libraries and large-scale processing centers.

Appendix C defines a simple blocking scheme for SEG-Y data to allow more efficient use of high-capacity tape media. This is based on the scheme defined in the SEG-D Rev 3.0 standard.

3.2. File Structure

Figure 1 illustrates the structure of a SEG-Y file. Following the optional SEG-Y Tape Label, the next 3600 bytes of the file are the Textual File Header and the Binary File Header written as a concatenation of a 3200-byte record and a 400-byte record. This is optionally followed by Extended Textual File Header(s), which consists of zero or more 3200-byte Extended Textual File Header records. The remainder of the SEG-Y file contains a variable number of Data Trace records that are each preceded by a 240-byte Standard Trace Header and zero or more 240-byte Trace Header Extensions. The Trace Header Extension mechanism was introduced in 2.0 and while not strictly backward compatible with prior SEG-Y formats, it has been carefully designed to have minimal impact on existing SEG-Y reader software. It should be simple for existing software to be modified to detect the presence of the optional trace headers and either process or ignore any Proprietary Trace Header Extensions. The format of Trace Header Extensions is described fully in section 7.1.

3.3. Number Formats

In earlier SEG-Y standards, all binary values were defined as using “big-endian” byte ordering. This means that, within the bytes that make up a number, the most significant byte (containing the sign bit) is written closest to the beginning of the file and the least significant byte is written closest to the end of the file. With SEG-Y rev 2, “little-endian” and “pairwise byte-swapped” byte ordering were introduced, primarily for I/O performance. This is independent of the medium to which a particular SEG-Y file is written (i.e., the byte ordering is no different if the file is written to tape on a mainframe or to disk on a PC). These alternate byte orders are identified by examining bytes 3297-3300 in the Binary File Header and apply only to the Binary File Header, Trace Headers, and Trace Samples[3].

All values in the Binary File Header and the SEG defined Trace Headers are to be treated as two's complement integers, whether two, four or eight bytes long, except for the 8-character Trace Header Extension name, an optional IEEE double precision sample rate, and fields that cannot be negative such as the number of samples per trace. *To aid in data recognition and*

[3] Textual Headers and Data Trailer records are always assumed to be text and so byte ordering is left untouched.

recovery, a value of zero in any SEG or user assigned fields of these headers should indicate an unknown or unspecified value unless explicitly stated otherwise.

Trace Data sample values are either integers or floating-point numbers. Signed integers are in two's complement format. SEG-Y revision 2 adds unsigned integers, 24 and 64-bit integers and IEEE floating-point data sample types.

3.4. Varying Trace Lengths

The SEG-Y standard specifies fields for sample interval and number of samples at two separate locations in the file. The Binary File Header contains values that apply to the whole file and the Trace Headers contain values that apply to the associated trace. In SEG-Y, varying trace lengths in a file are explicitly allowed. The values for sample interval and number of samples in the Binary File Header should be for the primary set of seismic data traces in the file. This approach allows the Binary File Header to be read and say, for instance, "this is six seconds data sampled at a two-millisecond interval". The value for the number of samples in each individual Trace Header may vary from the value in the Binary File Header and reflect the actual number of samples in a trace. The number of bytes in each trace record must be consistent with the number of samples in the Trace Header. This is particularly important for SEG-Y data written to disk files (see Appendix A).

Allowing variable length traces complicates random access in a disk file, since the locations of traces after the first are not known without pre-scanning the file. To facilitate the option of random access, a field in the Binary File Header defines a fixed length trace flag. If this flag is set, all traces in the file must have the same length. This will typically be the case for poststack data.

3.5. Coordinates

Knowing the source and receiver locations is a primary requirement for processing seismic data, and knowing the location of the processed data with respect to other data is essential for interpretation. Traditionally seismic coordinates have been supplied as geographic coordinates and/or grid coordinates. SEG-Y accommodates either form. However, locations are ambiguous without clear coordinate reference system (CRS) definition. SEG-Y provides the ability to define the CRS used for the coordinates contained within the Binary Header, the Extended Textual Header, and the Trace Headers. To avoid confusion, this standard requires that a single CRS ***must*** be used for all coordinates within an individual SEG-Y data set. Additionally, the coordinate units must be the same for all coordinates. The SEG-Y CRS definitions conform to those in SEG-D Rev 3.0 or the OGP P1/11 standards referenced in Appendix G.

4. Textual File Header

The first 3200-byte, Textual File Header record contains 40 lines of textual information, providing a human-readable description of the seismic data in the SEG-Y file. This information is free form and is the least well-defined of the headers in the 1975 standard, although the standard did provide a suggested layout for the first 20 lines. While there would be distinct advantages in making the layout of this header more rigid, it was decided that it would not be practicable to produce a layout that would be universally acceptable in the light of how it is currently used. It originally was encoded in the EBCDIC character set (see Appendix F) but ASCII is now allowed for all Textual File Headers.

SEG-Y revision 2 defines a separate textual header with a more comprehensively defined structure, where textual information can be stored in a machine-readable way. This Extended Textual File Header is described in detail in section 6. Note that the “traditional” Textual File Header is separate from the Extended Textual File Header and will still be the primary location for human-readable information about the contents of the file. In particular, it should contain information about any unusual features in the file, such as if the delay recording time in trace header bytes 109-110 is non-zero. The revision level of the SEG-Y format (Binary File Header bytes 3501-3502) being used *must* be included for all files written in the SEG-Y rev 2 format. It is mandatory that the SEG-Y revision level now be included in the Textual File Header.

Table 1 is an example Textual File Header with the SEG-Y revision level included in the 39th record.

Table 1 Textual File Header

3200-byte Textual File Header							
Cols 1-10	**Cols 11-20**	**Cols 21-30**	**Cols 31-40**	**Cols 41-50**	**Cols 51-60**	**Cols 61-70**	**Cols 71-80**
1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890	1234567890
C 1 CLIENT			COMPAN	Y		CREW N	O
C 2 LINE		AREA		MA	P ID		
C 3 REEL N	O	DAY-STAR	T OF REEL	YEAR	OBSERV	ER	
C 4 INSTRU	MENT: MFG		MODEL	SE	RIAL NO		
C 5 DATA T	RACES/RECO	RD	AUXILIARY	TRACES/REC	ORD	CDP FOLD	
C 6 SAMPLE	INTERVAL	SA	MPLES/TRAC	E BI	TS/IN	BYTES/SAMP	LE
C 7 RECORD	ING FORMAT	FO	RMAT THIS	REEL	MEASUREM	ENT SYSTEM	
C 8 SAMPLE	CODE: FLO	ATING PT	FIXED P	T FIXE	D PT-GAIN	CORREL	ATED
C 9 GAIN	TYPE: FIXE	D BINA	RY FLO	ATING POIN	T OTHE	R	
C10 FILTER	S: ALIAS	HZ NOT	CH HZ	BAND	- HZ	SLOPE -	DB/OCT
C11 SOURCE	: TYPE	NU	MBER/POINT	PO	INT INTERV	AL	
C12 PA	TTERN:			LENGTH	WID	TH	
C13 SWEEP:	START	HZ END	HZ LEN	GTH M	S CHANNEL	NO TY	PE
C14 TAPER:	START LEN	GTH	MS END LE	NGTH	MS TYPE		
C15 SPREAD	: OFFSET	MAX	DISTANCE	GROU	P INTERVAL		
C16 GEOPHO	NES: PER G	ROUP S	PACING	FREQUENCY	MFG	MO	DEL
C17 PA	TTERN:			LENGTH	WID	TH	
C18 TRACES	SORTED BY	: RECORD	CDP	OTHER			
C19 AMPLIT	UDE RECOVE	RY: NONE	SPHERI	CAL DIV	AGC	OTHER	
C20 MAP PR	OJECTION			ZONE ID	COORDI	NATE UNITS	[4]
C21 PROCES	SING:						
C22 PROCES	SING:						
C23							
...							
C38							
C39 SEG-Y	REV2.1						
C40 END TE	XTUAL HEAD	ER [5]					

[4] C20 is overridden by the contents of location data stanza in an Extended Textual Header record

[5] C40 END EBCDIC is also acceptable but C40 END TEXTUAL HEADER is the preferred encoding.

5. Binary File Header

The 400-byte Binary File Header record contains binary values relevant to the whole SEG-Y file. The values in the Binary File Header are defined as two-byte or four-byte, two's complement, or unsigned integers, with the exception of IEEE double precision sample intervals. Certain values in this header are crucial for the processing of the data in the file, particularly the sample interval, trace length and format code. Revision 2.1 defines a few additional fields in the optional portion, as well as providing some clarification on the use of some existing entries.

Table 2 Binary File Header[6]

400-byte Binary File Header	
Byte	**Description**
3201–3204	Job identification number.
3205–3208	Line number. For 3-D poststack data, this will typically contain the in-line number.
3209–3212	Reel number.
3213–3214	Number of data traces per ensemble. Mandatory for prestack data.
3215–3216	Number of auxiliary traces per ensemble. Mandatory for prestack data.
3217–3218	Sample interval. Microseconds (μs) for time data, Hertz (Hz) for frequency data, meters (m) or feet (ft) for depth data.
3219–3220	Sample interval of original field recording. Microseconds (μs) for time data, Hertz (Hz) for frequency data, meters (m) or feet (ft) for depth data.
3221–3222	Number of samples per data trace. Note: The sample interval and number of samples in the Binary File Header should be for the primary set of seismic data traces in the file.
3223–3224	Number of samples per data trace for original field recording.

[6] Where not otherwise indicated, a value of zero indicates an unknown or unspecified value.

400-byte Binary File Header	
Byte	**Description**
3225–3226	Data sample format code. Mandatory for all data. These formats are described in Appendix E. 1 = 4-byte IBM floating-point 2 = 4-byte, two's complement integer 3 = 2-byte, two's complement integer 4 = 4-byte fixed-point with gain (obsolete) 5 = 4-byte IEEE floating-point 6 = 8-byte IEEE floating-point 7 = 3-byte two's complement integer 8 = 1-byte, two's complement integer 9 = 8-byte, two's complement integer 10 = 4-byte, unsigned integer 11 = 2-byte, unsigned integer 12 = 8-byte, unsigned integer 15 = 3-byte, unsigned integer 16 = 1-byte, unsigned integer
3227–3228	Ensemble fold — The expected number of data traces per trace ensemble (e.g. the CMP fold).
3229–3230	Trace sorting code (i.e. type of ensemble) : –1 = Other (should be explained in a user Extended Textual File Header stanza) 0 = Unknown 1 = As recorded (no sorting) 2 = CDP ensemble 3 = Single fold continuous profile 4 = Horizontally stacked 5 = Common source point 6 = Common receiver point 7 = Common offset point 8 = Common mid-point 9 = Common conversion point
3231–3232	Vertical sum code: 1 = no sum, 2 = two sum, …, N = M–1 sum (M = 2 to 32,767)
3233–3234	Sweep frequency at start (Hz).
3235–3236	Sweep frequency at end (Hz).
3237–3238	Sweep length (ms).

400-byte Binary File Header	
Byte	**Description**
3239–3240	Sweep type code: 1 = linear 2 = parabolic 3 = exponential 4 = other
3241–3242	Trace number of sweep channel.
3243–3244	Sweep trace taper length in milliseconds at start if tapered (the taper starts at zero time and is effective for this length).
3245–3246	Sweep trace taper length in milliseconds at end (the ending taper starts at sweep length minus the taper length at end).
3247–3248	Taper type: 1 = linear 2 = cosine squared 3 = other
3249–3250	Correlated data traces: 1 = no 2 = yes
3251–3252	Binary gain recovered: 1 = yes 2 = no
3253–3254	Amplitude recovery method: 1 = none 2 = spherical divergence 3 = AGC 4 = other
3255–3256	Measurement system: If Location Data stanzas are included in the file, this entry would normally agree with the Location Data stanza. If there is a disagreement, the last Location Data stanza is the controlling authority. If units are mixed, e.g., meters on surface, feet in depth, then a Location Data stanza is mandatory. 1 = Meters 2 = Feet
3257–3258	Impulse signal polarity 1 = Increase in pressure or upward geophone case movement gives negative number on trace. 2 = Increase in pressure or upward geophone case movement gives positive number on trace.

400-byte Binary File Header	
Byte	**Description**
3259–3260	Vibratory polarity code: Seismic signal lags pilot signal by: 1 = 337.5° to 22.5° 2 = 22.5° to 67.5° 3 = 67.5° to 112.5° 4 = 112.5° to 157.5° 5 = 157.5° to 202.5° 6 = 202.5° to 247.5° 7 = 247.5° to 292.5° 8 = 292.5° to 337.5°
3261–3264	Extended number of data traces per ensemble. If nonzero, this overrides the number of data traces per ensemble in bytes 3213–3214.
3265–3268	Extended number of auxiliary traces per ensemble. If nonzero, this overrides the number of auxiliary traces per ensemble in bytes 3215–3216.
3269–3272	Extended number of samples per data trace. If nonzero, this overrides the number of samples per data trace in bytes 3221–3222.
3273–3280	Extended sample interval, IEEE double precision (64-bit). If nonzero, this overrides the sample interval in bytes 3217–3218 with the same units.
3281–3288	Extended sample interval of original field recording, IEEE double precision (64-bit). If nonzero, this overrides the sample interval of original field recording in bytes 3219–3220 with the same units.
3289–3292	Extended number of samples per data trace in original recording. If nonzero, this overrides the number of samples per data trace in original recording in bytes 3223–3224.
3293–3296	Extended ensemble fold. If nonzero, this overrides ensemble fold in bytes 3227–3228.
3297–3300	The integer constant 16909060_{10} (01020304_{16}). This is used to allow unambiguous detection of the byte ordering to expect for this SEG-Y file. For example, if this field reads as 67305985_{10} (04030201_{16}) then the bytes in every Binary File Header, Trace Header and Trace Data field must be reversed as they are read, i.e., converting the endian-ness of the fields. If it reads 33620995_{10} (02010403_{16}) then consecutive pairs of bytes need to be swapped in every Binary File Header, Trace Header and Trace Data field. If this value is zero, pre-rev 2 big-endian byte ordering should be assumed. The byte ordering of all other portions (the Extended Textual Header and Data Trailer) of the SEG-Y file is not affected by this field.
3301–3500	Unassigned

400-byte Binary File Header	
Byte	**Description**
3501	Major SEG-Y Format Revision Number. This is an 8-bit unsigned value. Thus for SEG-Y revision 2.1, as defined in this document, this will be recorded as 02_{16}. This field is mandatory for all versions of SEG-Y, although a value of zero indicates "traditional" SEG-Y conforming to the 1975 standard.
3502	Minor SEG-Y Format Revision Number. This is an 8-bit unsigned value with a radix point between the first and second bytes. Thus for SEG-Y revision 2.1, as defined in this document, this will be recorded as 01_{16}. This field is mandatory for all versions of SEG-Y.
3503–3504	Fixed length trace flag. A value of one indicates that all traces in this SEG-Y file are guaranteed to have the same sample interval, number of trace header blocks and trace samples, as specified in Binary File Header bytes 3217–3218 or 3273–3280, 3507–3508, and 3221–3222 or 3269–3272. A value of zero indicates that the length of the traces in the file may vary and the number of samples in bytes 115–116 of the Standard SEG-Y Trace Header and, if present, bytes 137–140 of SEG-Y Trace Header Extension 1 must be examined to determine the actual length of each trace. This field is mandatory for all versions of SEG-Y, although a value of zero indicates "traditional" SEG-Y conforming to the 1975 standard. Irrespective of this flag, it is strongly recommended that corect values for the number of samples per trace and sample interval appear in the appropriate trace Trace Header locations.
3505–3506	Number of 3200-byte, Extended Textual File Header records following the Binary Header. If bytes 3521–3528 are nonzero, that field overrides this one. A value of zero indicates there are no Extended Textual File Header records (i.e. this file has no Extended Textual File Header(s)). A value of -1 indicates that there are a variable number of Extended Textual File Header records and the end of the Extended Textual File Header is denoted by an ((SEG: EndText)) stanza in the final record (Section 6.2). A positive value indicates that there are exactly that many Extended Textual File Header records. Note that, although the exact number of Extended Textual File Header records may be a useful piece of information, it will not always be known at the time the Binary Header is written and it is not mandatory that a positive value be recorded here or in bytes 3521–3528. It is however recommended to record the number of records if possible as this makes reading more effective and supports direct access to traces on disk files. In the event that this number exceeds 32767, set this field to –1 and bytes 3521–3528 to 3600+3200*(number of Extended Textual File Header records). Add a further 128 if a SEG-Y Tape Label is present.
3507–3508	Maximum number of additional 240-byte trace headers. A value of zero indicates there are no additional 240-byte trace headers. The actual number for a given trace may be supplied in bytes 157–158 of SEG-Y Trace Header Extension 1.

3509–3510	Survey type. The sum of options from each group. <u>Group I (Environment)</u> 1 = Land 2 = Marine 3 = Transition 4 = Downhole 5–7 Reserved <u>Group II (Dimensionality)</u> 8 = 1D 16 = 2D 24 = 3D 32 = Time-lapse (added to one of the above three) 64–127 Reserved <u>Group III (Layout)</u> 128=parallel lines 256=cross-spread 684=patches 1024=towed streamer 1152=ocean bottom sensors/cables 1280=pseudo-random sensor and/or source arrangements 1281-4095 Reserved <u>Reserved</u> 4096-65535 user-defined
3511–3512	Time basis code: 1 = Local 2 = GMT (Greenwich Mean Time) 3 = Other, should be explained in a user defined stanza in the Extended Textual File Header 4 = UTC (Coordinated Universal Time) 5 = GPS (Global Positioning System Time)
3513–3520	Number of traces in this file or stream. (64-bit unsigned integer value). If zero, monitor the end of data flag in optional SEG-Y Trace Header Extension 1, bytes 159–160, if present, otherwise all bytes in the file or stream are part of this SEG-Y data set.
3521–3528	Byte offset of first trace relative to start of file or stream if known, otherwise zero. (64-bit unsigned integer value) This byte count will include the initial 3600 bytes of the Textual and this Binary File Header plus the Extended Textual Header if present. When nonzero, this field overrides the byte offset implied by any nonnegative number of Extended Textual Header records present in bytes 3505–3506.
3529–3532	Number of 3200-byte data trailer stanza records following the last trace (4-byte signed integer). A value of 0 indicates there are no trailer records. A value of -1 indicates an undefined number of trailer records (0 or more) following the data. It is, however, recommended to record the number of trailer records if possible as this makes reading more efficient.
3533–3600	Unassigned

6. Extended Textual File Header

If bytes 3505–3506 of the Binary File Header are non-zero, then an Extended Textual File Header[7] is present in the SEG-Y file. The Extended Textual File Header follows the Binary File Header record and precedes the first Data Trace record. Similarly, if bytes 3529–3532 are nonzero, a trailer with the same formatting appears after the last Data Trace. An Extended Textual File Header consists of one or more 3200-byte records and provides additional space for recording information about the SEG-Y file in a flexible but well-defined way. The kind of information recorded here will include trace header mappings, coordinate reference system details, 3-D bin grids, processing history and acquisition parameters. It is recommended that stanza information be included only once per SEG-Y rev 2 file. In the event multiple or conflicting data entries are included in the SEG-Y rev 2 file, the last data entry is assumed to be correct.

The data in the Extended Textual File Header are organized in the form of stanzas, a format also used for trailer records. Appendix D defines a set of predefined stanzas. It is intended that additional stanzas will be defined in the future revisions to this and other SEG-defined standards. However, the stanza mechanism is intended to be flexible and extensible and it is perfectly acceptable to define private stanzas. For the sake of usability, data exchange and maximum benefit, a standard SEG defined stanza should be used if it exists for the type of information required.

To avoid clashes of stanza names, a stanza name will be prefixed with the name of the company or organization that has defined the stanza. The company or organization name and the stanza name are separated by the character ":" (EBCDIC $7A_{16}$ or ASCII $3A_{16}$). Examples are: ((SEG: Location Data)) and ((JJ Example Seismic: Microseismic Geometry Definition)). The company or organization name can be an abbreviation or acronym; but the name must be sufficiently unique to unambiguously identify the originator of the stanza definition. If there is any question that the name may become non-unique, the first stanza keyword/value pair should be "Stanza Definer = Full Company Name".

All stanza names should be uniquely associated with a single parameter set, typically keyword/value pairs.[8] To ensure that there is always a unique association between the stanza names and the stanza content, revision numbering and/or stanza name modification should be employed for all user defined stanzas.

For stanza naming, the Society of Exploration Geophysicists reserves the acronym SEG and all variants of SEG for use by the SEG Technical Standards Committee.

A SEG-Y reader must be capable of ignoring stanzas that the reader does not comprehend (which may be the whole Extended Textual File Header). The data within stanzas will typically use keywords and values, which can be produced and read by machines, as well as remaining human-readable.

Possible user supplied stanzas which have been suggested are:

- General Data Parameters (e.g., License Block, Date, Operator, Line etc.)
- General Acquisition Parameters
- SP to CDP relationship

[7] While binary data may possibly appear in User Header stanzas, all SEG-defined Extended Textual Header stanzas have their numeric values encoded in text.

[8] The User Data stanza allows XML 1.0 format in Extended Textual Headers, permitting complex names and values. OGP stanzas adhere to OGP formatting.

- Usage of Optional parts of Trace Headers
- Decoded Binary Header

It is strongly recommended that the SEG-Y format be used principally to exchange seismic data. As part of that exchange, the SEG-Y file should contain sufficient information to identify the seismic data contained within the file and allow that seismic data to be processed. *The SEG-Y file is not intended as a comprehensive ancillary data exchange format.* The Extended Textual Header provides a means to include almost unlimited ancillary data in the SEG-Y file, but restraint should be exercised when selecting ancillary data to be included in the Extended Textual File Header. If significant amounts of ancillary data need to be exchanged, it is recommended that SEG Ancillary Data Standard data set(s) be used.

6.1. Structure of Extended Textual Header

The Extended Textual File Header consists of one or more 3200-byte records. Typically, each record contains lines[9] of textual card-image text. Note that, unlike the Textual File Header, lines in the Extended Textual File Header do not start with the character "C" (EBCDIC $C3_{16}$ or ASCII 43_{16}). For processing purposes, all of the Extended Textual File Header records shall be considered as being concatenated into a single logical file (i.e., the inter-record gaps between the 3200-byte records are not significant).

Text within the Extended Textual File Header is organized into stanzas. A stanza begins on a 3200-byte boundary with a stanza header, which is a line containing the name of the defining organization or company and the name of the stanza, separated by a colon ":", EBCDIC $7A_{16}$ or ASCII $3A_{16}$. A stanza ends with the start of a new stanza, or the end of the Extended Textual File Header and consists of one or more 3200-byte records. The stanza header begins with double left parentheses "((", EBCDIC $4D_{16}$ or ASCII 28_{16} characters, and ends with double right parentheses "))", EBCDIC $5D_{16}$ or ASCII 29_{16} characters. The first left parenthesis at the beginning of a stanza must be in column one. The case of stanza names shall not be significant. To aid readability, spaces (" ", EBCDIC 40_{16} or ASCII 20_{16}) within stanza names shall be allowed but ignored. Thus the stanza name ((SEG: Recording Parameters)) shall refer to the same stanza as ((seg:RECORDINGPARAMETERS)).

The format of the information within a stanza depends on the type of the data contained in the stanza, which is implicitly and uniquely defined by the name of the stanza. However, many stanzas will contain data organized as keyword/value pairs. The ground rules for stanzas that use this schema are as follows:

- Each line consists of a keyword/value pair in the form "keyword = value".
- The keywords and values can contain any printable character except double right or double left parentheses or the equal sign. However, the use of punctuation characters in keywords is not recommended.
- The case of a keyword is not significant.
- For readability, spaces within a keyword are allowed but ignored. Thus, the keyword "Line Name" refers to the same keyword as "LINENAME".
- The value associated with a keyword begins with the first non-blank character following the equal sign and extends to the last non-blank character on the line.

[9] A "line" is terminated by carriage return and linefeed characters. It need not be 80 characters long.

- The value field for a keyword may consist of multiple subfields, separated by commas (",", EBCDIC $6B_{16}$ or ASCII $2C_{16}$).
- Blank lines are ignored.
- If the first non-blank character in a line is the hash sign ("#", EBCDIC $7B_{16}$ or ASCII 23_{16}), the line is treated as a comment and ignored.
- For lines that are not comments, if the last non-blank character on the line is an ampersand ("&", EBCDIC 50_{16} or ASCII 16_{16}), the next line is considered to be a continuation of the current line (i.e., the next line is concatenated with the current line, with the ampersand removed). Note that blank lines and comments are bypassed when continuing the line.
- Each line in an Extended Textual File Header ends in carriage return and linefeed (EBCDIC $0D25_{16}$ or ASCII $0D0A_{16}$)

6.2. EndText stanza

The EndText stanza is required[10] if the Binary File Header value in bytes 3505–3506 is –1. If that value is greater than zero, the EndText stanza is optional but must in that case be included in the count of Extended Textual Header records if present. The stanza ((SEG: EndText)) is treated specially with regard to stanza concatenation. This stanza must appear as the final 3200-byte record in the Extended Textual File Header. The stanza header shall be on the first line in the record and must be the only non-blank text in the record (i.e., the stanza must be empty). This allows the end of the Extended Textual File Header to be located easily by SEG-Y readers and simplifies decoding for SEG-Y readers that do not wish to process the Extended Textual File Header.

[10] In principle, the EndText stanza could be omitted when bytes 3521–3528 of the Binary File Header are nonzero, but we feel it best to continue to require it for SEG-Y rev 1 software compatibility.

6.3. Stanza Examples

((JJ ESeis: Microseismic Geometry Definition ver 1.0))
Definer name = J and J Example Seismic Ltd.
Line Name Convention = CDA
Line Name = Sample MicroSeismic 1
First Trace In Data Set = 101
Last Trace In Data Set = 1021
First SP In Data Set = 2001
Last SP In Data Set = 6032
… additional blank lines to end of 3200-byte Extended Textual Header record

((OGP: P1/11 Data Geographic Extent))

```
H1,5,0,0,Survey Perimeter Definition… ,1,Full Fold Boundary,2,1,3,Full Fold Coverage,0,
M1,0,1,1,1,1,391194.94,4092809.86,,54.2344345434,-9.2344345434,,
M1,0,1,1,2,1,392747.34,4093232.60,,54.2655123423,-9.2435354534,,
M1,0,1,1,3,1,393576.45,4094267.73,,54.2834225677,-9.2578834354,,
M1,0,1,1,4,1,391243.56,4095786.14,,54.2535353553,-9.2367002431,,
M1,0,1,1,1, ,391194.94,4092809.86,,54.2344345434,-9.2344345434,,
```

… additional blank lines to end of 3200-byte Extended Textual Header record

((SEG: Measurement Units ver 1.0))
Data Sample Measurement Unit =Millivolts
Volt conversion =0.001
… additional blank lines to end of 3200-byte Extended Textual Header

7. Data Traces

7.1. Trace Header

The SEG-Y trace header contains trace attributes, most of which are defined with two-byte or four-byte, two's complement integers. The values in bytes 1–180 were defined in the 1975 standard and these entries remain unchanged, although clarification and extensions may be supplied where appropriate. Bytes 181–240 were for optional information in the 1975 standard and this has been the main area of conflict between different flavors of SEG-Y.

SEG-Y rev 1 defined standard locations in bytes 181–232 of the Standard Trace Header for certain values that are needed in modern data processing. In particular, standard locations for a shotpoint number and ensemble (CDP) coordinates are defined. Bytes 203 to 210 allow the measurement units for the Data Trace samples to be defined and transduction constants to be specified. These entries allow the Data Trace values to be converted to engineering units.

The present revision includes an optional SEG-Y Trace Header Extension to provide extra precision and range for certain fields (coordinates, depths and elevations, time stamps, sample rate and number of samples per trace) in the Standard Trace Header. When used, any SEG-Y Trace Header Extensions must immediately follow the SEG-Y Standard Trace Header.

This revision further provides for zero or more User-defined Proprietary Trace Header Extensions to appear after SEG-defined trace headers. If more than zero User-defined Proprietary Trace Header Extensions are provided, SEG-Y Trace

Header Extension 1 is mandatory. Also, unless the fixed length flag in Binary File Header bytes 3503–3504 is set, User-defined Proprietary Trace Header Extensions may appear in any order and number from trace to trace after all present SEG-defined trace headers. In the event the fixed length flag is set, the same sequence of trace headers and the same number and type of data samples must be present in every trace.

The values included in the SEG-defined Trace Headers are limited and intended to provide information that may change on a trace-by-trace basis and the basic information needed to process and identify the trace. The trace headers are not intended to be a repository for exceptional amounts of ancillary data. If great amounts of ancillary data need to be exchanged, it is recommended that one or more SEG Ancillary Data Standard data sets be used.

Table 3 Standard Trace Header[11]

240-byte Standard Trace Header	
Byte	**Description**
1–4	Trace sequence number within line — Numbers continue to increase if the same line continues across multiple SEG-Y files.
5–8	Trace sequence number within SEG-Y file — Each file starts with trace sequence one.
9–12	Original field record number.
13–16	Trace number within the original field record. If supplying multi-cable data with identical channel numbers on each cable, either supply the cable ID number in bytes 153–156 of SEG-Y Trace Header Extension 1 or enter (cable–1)*nchan_per_cable+channel_no here.
17–20	Energy source point number — Used when more than one record occurs at the same effective surface location. It is recommended that the new entry defined in Trace Header bytes 197–202 be used for shotpoint number.
21–24	Ensemble number (i.e. CDP, CMP, CRP, etc.)
25–28	Trace number within the ensemble — Each ensemble starts with trace number one.
29–30	Trace identification code: –1 = Other 0 = Unknown 1 = Time domain seismic data 2 = Dead 3 = Dummy 4 = Time break

[11] Where not otherwise indicated, a value of zero indicates an unknown or unspecified value.

240-byte Standard Trace Header	
Byte	**Description**
	5 = Uphole 6 = Sweep 7 = Timing 8 = Waterbreak 9 = Near-field gun signature 10 = Far-field gun signature 11 = Seismic pressure sensor 12 = Multicomponent seismic sensor – Vertical component 13 = Multicomponent seismic sensor – Cross-line component 14 = Multicomponent seismic sensor – In-line component 15 = Rotated multicomponent seismic sensor – Vertical component 16 = Rotated multicomponent seismic sensor – Transverse component 17 = Rotated multicomponent seismic sensor – Radial component 18 = Vibrator reaction mass 19 = Vibrator baseplate 20 = Vibrator estimated ground force 21 = Vibrator reference 22 = Time-velocity pairs 23 = Time-depth pairs 24 = Depth-velocity pairs 25 = Depth domain seismic data 26 = Gravity potential 27 = Electric field – Vertical component 28 = Electric field – Cross-line component 29 = Electric field – In-line component 30 = Rotated electric field – Vertical component 31 = Rotated electric field – Transverse component 32 = Rotated electric field – Radial component 33 = Magnetic field – Vertical component 34 = Magnetic field – Cross-line component 35 = Magnetic field – In-line component 36 = Rotated magnetic field – Vertical component 37 = Rotated magnetic field – Transverse component 38 = Rotated magnetic field – Radial component 39 = Rotational sensor – Pitch 40 = Rotational sensor – Roll 41 = Rotational sensor – Yaw 42 … 255 = Reserved 256 … N = optional use, (maximum N = 16,383) N+16,384 = Interpolated, i.e. not original, seismic trace.
31–32	Number of vertically summed traces yielding this trace. (1 is one trace, 2 is two summed traces, etc.)
33–34	Number of horizontally stacked traces yielding this trace. (1 is one trace, 2 is two stacked traces, etc.)

<table>
<tr><th colspan="3">240-byte Standard Trace Header</th></tr>
<tr><th>Byte</th><th colspan="2">Description</th></tr>
<tr><td>35–36</td><td colspan="2">Data use:
1 = Production
2 = Test</td></tr>
<tr><td>37–40</td><td colspan="2">Distance from center of the source point to the center of the receiver group (negative if opposite to direction in which line is shot).</td></tr>
<tr><td>41–44</td><td>Elevation of receiver group. This is, of course, normally equal to or lower than the surface elevation at the group location.</td><td rowspan="8">The scalar in Trace Header bytes 69–70 applies to these values.

The units are feet or meters as specified in Binary File Header bytes 3255–3256.
Elevations and depths and their signs (+ve or –ve) are tied to a vertical CRS defined through an Extended Textual Header (see Appendix D-1). Historical usage had been that all elevations above the vertical datum were positive and below were negative. Elevations should now be defined with respect to the CRS.</td></tr>
<tr><td>45–48</td><td>Surface elevation at source location.</td></tr>
<tr><td>49–52</td><td>Source depth below surface[12].</td></tr>
<tr><td>53–56</td><td>Seismic Datum[13] elevation at receiver group. (If different from the survey vertical datum, Seismic Datum should be defined through a vertical CRS in an extended textual stanza.)</td></tr>
<tr><td>57–60</td><td>Seismic Datum elevation at source. (As above)</td></tr>
<tr><td>61–64</td><td>Water column height at source location (at time of source event).</td></tr>
<tr><td>65– 68</td><td>Water column height at receiver group location (at time of recording of first source event into that receiver).</td></tr>
<tr><td>69–70</td><td colspan="2">Scalar to be applied to all elevations and depths specified in Standard Trace Header bytes 41–68 to give the real value. Scalar = 1, ±10, ±100, ±1000, or ±10,000. If positive, scalar is used as a multiplier; if negative, scalar is used as a divisor. A value of zero is assumed to be a scalar value of 1.</td></tr>
<tr><td>71–72</td><td colspan="2">Scalar to be applied to all coordinates specified in Standard Trace Header bytes 73–88 and to bytes Trace Header 181–188 to give the real value. Scalar = 1, ±10, ±100, ±1000, or ±10,000. If positive, scalar is used as a multiplier; if negative, scalar is used as divisor. A value of zero is assumed to be a scalar value of 1.</td></tr>
</table>

[12] Historically, taken as positive below the surface. We also note that bytes 41–48 of SEG-Y Trace Header Extension 1 provide a method for calculating receiver depth below the surface.

[13] Typically a floating or flat seismic processing datum that has been used in preprocessing the data in this SEG-Y dataset.

<table>
<tr><th colspan="4">240-byte Standard Trace Header</th></tr>
<tr><th>Byte</th><th colspan="3">Description</th></tr>
<tr><td>73–76</td><td>Source coordinate – X.</td><td colspan="2" rowspan="4">The coordinate reference system should be identified through an Extended Textual Header (see Appendix D-1).
If the coordinate units are in seconds of arc, decimal degrees or DMS, the X values represent longitude and the Y values latitude. A positive value designates east of Greenwich Meridian or north of the equator and a negative value designates south or west.</td></tr>
<tr><td>77–80</td><td>Source coordinate – Y.</td></tr>
<tr><td>81–84</td><td>Group coordinate – X.</td></tr>
<tr><td>85–88</td><td>Group coordinate – Y.</td></tr>
<tr><td>89–90</td><td colspan="3">Coordinate units:
1 = Length (meters or feet as specified in Binary File Header bytes 3255-3256 and in Extended Textual Header if Location Data are included in the file)
2 = Seconds of arc (deprecated)
3 = Decimal degrees (preferred degree representation)
4 = Degrees, minutes, seconds (DMS)[14]
Note: To encode ±DDDMMSS set bytes 73–88 = ±DDD*10^4 + MM*10^2 + SS with bytes 71–72 set to 1; To encode ±DDDMMSS.ss set bytes 73–88 = ±DDD*10^6 + MM*10^4 + SS*10^2 + ss with bytes 71–72 set to –100.</td></tr>
<tr><td>91–92</td><td colspan="3">Weathering velocity. (ft/s or m/s as specified in Binary File Header bytes 3255–3256)</td></tr>
<tr><td>93–94</td><td colspan="3">Subweathering velocity. (ft/s or m/s as specified in Binary File Header bytes 3255–3256)</td></tr>
<tr><td>95–96</td><td colspan="2">Uphole time at source in milliseconds.</td><td rowspan="7">Time in milliseconds as scaled by the scalar specified in Standard Trace Header bytes 215-216.</td></tr>
<tr><td>97–98</td><td colspan="2">Uphole time at group in milliseconds.</td></tr>
<tr><td>99–100</td><td colspan="2">Source static correction in milliseconds.</td></tr>
<tr><td>101–102</td><td colspan="2">Group static correction in milliseconds.</td></tr>
<tr><td>103–104</td><td colspan="2">Total static applied in milliseconds. (Zero if no static has been applied,)</td></tr>
<tr><td>105–106</td><td colspan="2">Lag time A — Time in milliseconds between end of 240-byte trace identification header and time break. The value is positive if time break occurs after the end of header; negative if time break occurs before the end of header. Time break is defined as the initiation pulse that may be recorded on an auxiliary trace or as otherwise specified by the recording system.</td></tr>
<tr><td>107–108</td><td colspan="2">Lag Time B — Time in milliseconds between time break and the initiation time of the energy source. May be positive or negative.</td></tr>
</table>

[14] With two decimal places (±DDMMSS.ss) resolution is approximately ±0.3 meters. If longitudes are in the range ±180 degrees, a third decimal place is available.

240-byte Standard Trace Header	
Byte	**Description**
109–110	Delay recording time — Time in milliseconds between initiation time of energy source and the time when recording of data samples begins. In SEG-Y rev 0 this entry was intended for deep-water work if data recording did not start at zero time. The entry can be negative to accommodate negative start times (i.e. data recorded before time zero, presumably as a result of static application to the data trace). If a non-zero value (negative or positive) is recorded in this entry, a comment to that effect should appear in the Textual File Header.
111–112	Mute time — Start time in milliseconds.
113–114	Mute time — End time in milliseconds.
115–116	Number of samples in this trace. The number of bytes in a trace record must be consistent with the number of samples written in the Binary File Header and/or the SEG-defined Trace Header(s). This is important for all recording media; but it is particularly crucial for the correct processing of SEG-Y data in disk files (see Appendix A). If the fixed length trace flag in bytes 3503–3504 of the Binary File Header is set, the number of samples in every trace in the SEG-Y file is assumed to be the same as the value recorded in the Binary File Header and this field is ignored. If the fixed length trace flag is not set, the number of samples may vary from trace to trace.
117–118	Sample interval for this trace. Microseconds (µs) for time data, Hertz (Hz) for frequency data, meters (m) or feet (ft) for depth data. If the fixed length trace flag in bytes 3503–3504 of the Binary File Header is set, the sample interval in every trace in the SEG-Y file is assumed to be the same as the value recorded in the Binary File Header and this field is ignored. If the fixed length trace flag is not set, the sample interval may vary from trace to trace.
119–120	Gain type of field instruments: 1 = fixed 2 = binary 3 = floating point 4 … N = optional use
121–122	Instrument gain constant (dB).
123–124	Instrument early or initial gain (dB).
125–126	Correlated: 1 = no 2 = yes
127–128	Sweep frequency at start (Hz).
129–130	Sweep frequency at end (Hz).

240-byte Standard Trace Header	
Byte	**Description**
131–132	Sweep length in milliseconds.
133–134	Sweep type: 1 = linear 2 = parabolic 3 = exponential 4 = other
135–136	Sweep trace taper length at start in milliseconds.
137–138	Sweep trace taper length at end in milliseconds.
139–140	Taper type: 1 = linear 2 = $\cos^2$ 3 = other
141–142	Alias filter frequency (Hz), if used.
143–144	Alias filter slope (dB/octave).
145–146	Notch filter frequency (Hz), if used.
147–148	Notch filter slope (dB/octave).
149–150	Low-cut frequency (Hz), if used.
151–152	High-cut frequency (Hz), if used.
153–154	Low-cut slope (dB/octave)
155–156	High-cut slope (dB/octave)
157–158	Year data recorded — The 1975 standard was unclear as to whether this should be recorded as a 2-digit or a 4-digit year and both have been used. For SEG-Y revisions beyond rev 0, the year should be recorded as the complete 4-digit Gregorian calendar year, e.g., the year 2001 should be recorded as 2001_{10} ($07D1_{16}$).
159–160	Day of year[15] (Range 1–366 for GMT, UTC, and GPS time basis).
161–162	Hour of day (24-hour clock).
163–164	Minute of hour.
165–166	Second of minute.

[15] Also known as Julian Day, albeit distinguished from the Julian Day in the Julian Date system that astronomers use.

240-byte Standard Trace Header	
Byte	**Description**
167–168	Time basis code. If nonzero, overrides Binary File Header bytes 3511–3512. 1 = Local 2 = GMT (Greenwich Mean Time) 3 = Other, should be explained in a user defined stanza in the Extended Textual File Header 4 = UTC (Coordinated Universal Time) 5 = GPS (Global Positioning System Time)
169–170	Trace weighting factor — Defined as 2^{-N} units (volts unless bytes 203–204 specify a different unit) for the least significant bit. (N = 0, 1, …, 32767)
171–172	Geophone group number of roll switch position one.
173–174	Geophone group number of trace number one within original field record.
175–176	Geophone group number of last trace within original field record.
177–178	Gap size (total number of groups dropped).
179–180	Over travel associated with taper at beginning or end of line: 1 = down (or behind) 2 = up (or ahead)
181–184	X coordinate of ensemble (CDP) position of this trace (scalar in Standard Trace Header bytes 71–72 applies). The coordinate reference system should be identified through an Extended Textual Header (see Appendices D-1 or D-3).
185–188	Y coordinate of ensemble (CDP) position of this trace (scalar in Standard Trace Header bytes 71–72 applies). The coordinate reference system should be identified through an Extended Textual Header (see Appendices D-1 or D-3).
189–192	For 3-D poststack data, this field should be used for the in-line number. If one in-line per SEG-Y file is being recorded, this value should be the same for all traces in the file and the same value will be recorded in bytes 3205–3208 of the Binary File Header.
193–196	For 3-D poststack data, this field should be used for the cross-line number. This will typically be the same value as the ensemble (CDP) number in Standard Trace Header bytes 21–24, but this does not have to be the case.
197–200	Shotpoint number — This is probably only applicable to 2-D poststack data. Note that it is assumed that the shotpoint number refers to the source location nearest to the ensemble (CDP) location for a particular trace. If this is not the case, there should be a comment in the Textual File Header explaining what the shotpoint number actually refers to.
201–202	Scalar to be applied to the shotpoint number in Standard Trace Header bytes 197–200 to give the real value. If positive, scalar is used as a multiplier; if negative as a divisor; if zero the shotpoint number is not scaled (i.e. it is an integer. A typical value will be –10, allowing shotpoint numbers with one decimal digit to the right of the decimal point).

240-byte Standard Trace Header	
Byte	**Description**
203–204	Trace value measurement unit: –1 = Other (should be described in Data Sample Measurement Units Stanza) 0 = Unknown 1 = Pascal (Pa) 2 = Volts (v) 3 = Millivolts (mV) 4 = Amperes (A) 5 = Meters (m) 6 = Meters per second (m/s) 7 = Meters per second squared (m/s^2) 8 = Newton (N) 9 = Watt (W) 10-255 = reserved for future use 256 … N = optional use. (maximum N = 32,767)
205–210	Transduction Constant — The multiplicative constant used to convert the Data Trace samples to the Transduction Units (specified in Standard Trace Header bytes 211–212). The constant is encoded as a four-byte, two's complement integer (bytes 205–208) which is the mantissa and a two-byte, two's complement integer (bytes 209–210) which is the power of ten exponent (i.e. Bytes 205–208 * 10**Bytes 209–210).
211–212	Transduction Units — The unit of measurement of the Data Trace samples after they have been multiplied by the Transduction Constant specified in Standard Trace Header bytes 205–210. –1 = Other (should be described in Data Sample Measurement Unit stanza, page 36) 0 = Unknown 1 = Pascal (Pa) 2 = Volts (v) 3 = Millivolts (mV) 4 = Amperes (A) 5 = Meters (m) 6 = Meters per second (m/s) 7 = Meters per second squared (m/s^2) 8 = Newton (N) 9 = Watt (W)
213–214	Device/Trace Identifier — The unit number or id number of the device associated with the Data Trace (i.e. 4368 for vibrator serial number 4368 or 20316 for gun 16 on string 3 on vessel 2). This field allows traces to be associated across trace ensembles independently of the trace number (Standard Trace Header bytes 25–28).
215–216	Scalar to be applied to times specified in Trace Header bytes 95–114 to give the true time value in milliseconds. Scalar = 1, +10, +100, +1000, or +10,000. If positive, scalar is used as a multiplier; if negative, scalar is used as divisor. A value of zero is assumed to be a scalar value of 1.

240-byte Standard Trace Header	
Byte	**Description**
217–218	Source Type/Orientation — Defines the type and the orientation of the energy source. The terms vertical, cross-line and in-line refer to the three axes of an orthogonal coordinate system. The absolute azimuthal orientation of the coordinate system axes should be defined in the CRS (see Appendix D-1). –1 to –n = Other (should be described in Source Type/Orientation stanza, page 38) 0 = Unknown 1 = Vibratory - Vertical orientation 2 = Vibratory - Cross-line orientation 3 = Vibratory - In-line orientation 4 = Impulsive - Vertical orientation 5 = Impulsive - Cross-line orientation 6 = Impulsive - In-line orientation 7 = Distributed Impulsive - Vertical orientation 8 = Distributed Impulsive - Cross-line orientation 9 = Distributed Impulsive - In-line orientation
219–224	Source Energy Direction with respect to the source orientation — Three two-byte two's complement binary integers for vertical, cross-line and in-line inclinations respectively. The positive orientation direction is defined in Bytes 217–218 of the Standard Trace Header. The energy direction is encoded in tenths of degrees (i.e. 347.8º is encoded as 3478_{10} ($0D96_{16}$)).
225–230	Source Measurement — Describes the source effort used to generate the trace. The measurement can be simple, qualitative measurements such as the total weight of explosive used or the peak air gun pressure or the number of vibrators times the sweep duration. Although these simple measurements are acceptable, it is preferable to use true measurement units of energy or work. The constant is encoded as a four-byte, two's complement integer (bytes 225–228) which is the mantissa and a two-byte, two's complement integer (bytes 209–230) which is the power of ten exponent (i.e. Bytes 225–228 * 10**Bytes 229–230).
231–232	Source Measurement Unit — The unit used for the Source Measurement, Standard Trace header bytes 225–230. –1 = Other (should be described in Source Measurement Unit stanza, page 39) 0 = Unknown 1 = Joule (J) 2 = Kilowatt (kW) 3 = Pascal (Pa) 4 = Bar (Bar) 4 = Bar-meter (Bar-m) 5 = Newton (N) 6 = Kilograms (kg)
233–240	Either binary zeros or the eight-character trace header name "SEG00000". May be ASCII or EBCDIC text.

Table 4 details the contents of SEG-Y Trace Header Extension 1. This extension allows one to override or supplement entries in the SEG-Y Standard Trace Header, providing more numeric precision or additional information. If used, it should appear immediately after the SEG-Y Standard Trace Header. In the event that any Proprietary User-defined Trace Headers are present, then SEG-Y Trace Header Extension 1 bytes 157-158 are used to account for their number.

Table 4 Trace Header Extension 1[16]

240-byte SEG-Y Trace Header Extension 1	
Byte	**Description**
1–8	Extended trace sequence number within line — Numbers continue to increase if the same line continues across multiple SEG-Y files. If nonzero, overrides trace sequence number within line (SEG-Y Standard Trace Header bytes 1–4). 64-bit unsigned integer.
9–16	Extended trace sequence number within SEG-Y file — Each file starts with trace sequence one. If nonzero, overrides trace sequence number within SEG-Y file (SEG-Y Standard Trace Header bytes 5–8). 64-bit unsigned integer.
17–24	Extended original field record number. If nonzero, overrides original field record number (SEG-Y Standard Trace Header bytes 9–12). 64-bit two's-complement integer.
25–32	Extended ensemble number (i.e. CDP, CMP, CRP, etc.) If nonzero, overrides ensemble number (SEG-Y Standard Trace Header bytes 21–24). 64-bit two's-complement integer.
33–40	Extended elevation of receiver group. If nonzero, overrides receiver group elevation (SEG-Y Standard Trace Header bytes 41–44). IEEE double precision (64-bit) value.
41–48	Receiver group depth below the surface location of receiver group. Positive for receiver group below surface[17]. IEEE double precision (64-bit) value.
49–56	Extended surface elevation at source location. If nonzero, overrides surface elevation at source location (SEG-Y Standard Trace Header bytes 45–48). IEEE double precision (64-bit) value.
57-64	Extended source depth below surface. If nonzero, overrides source depth below surface (SEG-Y Standard Trace Header bytes 49-52). IEEE double precision (64-bit) value.
65-72	Extended Seismic Datum elevation at receiver group. If nonzero, overrides Seismic Datum elevation at receiver group (SEG-Y Standard Trace Header bytes 53–56). IEEE double precision (64-bit) value.

[16] Where not otherwise indicated, a value of zero indicates an unknown or unspecified value.

[17] Note that there is no corresponding entry in the SEG-Y Standard Trace Header.

240-byte SEG-Y Trace Header Extension 1	
Byte	**Description**
73-80	Extended Seismic Datum elevation at source. If nonzero, overrides Seismic Datum elevation at source (SEG-Y Standard Trace Header bytes 57–60). IEEE double precision (64-bit) value.
81-88	Extended water column height at source location (at time of source event). If nonzero, overrides water column height at source location (SEG-Y Standard Trace Header bytes 61–64). IEEE double precision (64-bit) value.
89-96	Extended water column height at receiver group location (at time of recording of first source event into that receiver). If nonzero, overrides water column height at receiver group location (SEG-Y Standard Trace Header bytes 65–68). IEEE double precision (64-bit) value.
	For values in bytes 33–96, signs of elevation or depth values (+ve or –ve) must be in accordance with the respective vertical CRS defined through a Location Data Stanza (see Appendices D-1 or D-3).
97-104	Extended source coordinate - X. If nonzero, overrides Source coordinate - X (SEG-Y Standard Trace Header bytes 73–76). Will be negative for negative coordinates. IEEE double precision (64-bit) value.
105-112	Extended Source coordinate - Y. If nonzero, overrides Source coordinate - Y (SEG-Y Standard Trace Header bytes 77–80). Will be negative for negative coordinates. IEEE double precision (64-bit) value.
113-120	Extended group coordinate - X. If nonzero, overrides group coordinate - X (SEG-Y Standard Trace Header bytes 81–84). Will be negative for negative coordinates. IEEE double precision (64-bit) value.
121-128	Extended group coordinate - Y. If nonzero, overrides group coordinate - Y (SEG-Y Standard Trace Header bytes 85–88). Will be negative for negative coordinates. IEEE double precision (64-bit) value.
129-136	Extended Distance from center of the source point to the center of the receiver group (negative if opposite to direction in which line is shot). If nonzero, overrides Distance from center of the source point to the center of the receiver group (SEG-Y Standard Trace Header bytes 37-40). IEEE double precision (64-bit) value.
137-140	Extended number of samples in this trace. If nonzero, overrides number of samples in this trace (SEG-Y Standard Trace Header bytes 115–116). 4 byte unsigned integer value.
141-144	Nanoseconds to add to Second of minute (SEG-Y Standard Trace Header bytes 165–166). May be negative. 4 byte signed integer value.
145-152	If nonzero, IEEE double precision (64-bit) microsecond sample interval, overriding bytes 117–118 of the Standard Trace Header.
153-156	Cable number for multi-cable acquisition or Recording Device/Sensor ID number. 4 byte signed integer value.

240-byte SEG-Y Trace Header Extension 1	
Byte	**Description**
157-158	Number of additional trace header extension blocks including this one. If zero, the value in the Binary File Header bytes 3507–3510 is assumed. 2-byte unsigned integer value.
159-160	Last trace flag — a sum of the appropriate integers, leave zero otherwise. 1 = Last trace in ensemble (CDP, Shot Record, …) 2 = Last trace in line 4 = Last trace in this data file or data stream 8 = Last trace in current survey
161–168	Extended X coordinate of ensemble (CDP) position of this trace. If nonzero, overrides X coordinate of ensemble (CDP) (SEG-Y Standard Trace Header bytes 181–184). Will be negative for negative coordinates. IEEE double precision (64-bit) value.
169–176	Extended Y coordinate of ensemble (CDP) position of this trace. If nonzero, overrides Y coordinate of ensemble (CDP) (SEG-Y Standard Trace Header bytes 185–188). Will be negative for negative coordinates. IEEE double precision (64-bit) value.
177–232	Reserved
233–240	Eight-character trace header name “SEG00001”. May be ASCII or EBCDIC text.

All nonstandard 240-byte trace header extensions must take the following form. A corresponding record or records defining the proprietary layout should always be included in a SEG-Y Extended Textual Header ((SEG: Trace Header Mapping)) and include the trace header extension name in its stanza.

Table 5 Proprietary Trace Header Extension

240-byte Proprietary Trace Header Extensions	
Byte	**Description**
1–232	User defined
233–240	Eight-character trace header extension name, left justified and blank padded. May be ASCII or EBCDIC text and must consist of printable characters in the selected character set. (Refer to Appendix F, p. 76.) Names “SEG00000” through “SEG99999” are reserved.

7.2. Trace Data

Trace Data immediately follow their attached Trace Header(s), with the trace data arranged in samples of fixed size (1, 2, 3, 4, or 8 bytes) described in Appendix E.

The format of the data sample is specified in the Binary File Header (bytes 3225–3226). With SEG-Y revision 2, provision has been made via bytes 3297–3300 of the Binary File Header to consistently support little-endian byte ordering or pairwise byte swapping of the Binary File Header and both Trace Headers and Trace Data.

The seismic data in a SEG-Y file is organized into ensembles of traces or as a series of stacked traces. When the trace data is organized into ensembles of traces, the ensemble type may be identified (Binary File Header bytes 3229–3230).

8. User Header stanza and Data Trailer

The User Header and Data Trailer allow for arbitrary binary or textual data within a SEG-Y file. The User Header is provided as an Extended Textual Header stanza option and the Data Trailer follows the last trace.

The User Header and Data Trailer use the same basic format, and consist of a set of user-defined blocks separated by a description block identifying the producer, contents, size and data format of the data block following it.

Desc block 1	Data block 1	Desc block 2	Data block 2	…	Desc block n	Data block n

Figure 2 Structure of a User Header or Data Trailer containing *n* data blocks.

Even though the format is the same except for the preceding ((SEG:UserHeader)) specific to the User Header Extended Textual Header stanza line, we recommended the User Header be used for information that is required for interpreting the data traces. Large data blocks such as backups of positioning data, databases etc. should be stored in the Data Trailer. This will simplify decoding and speed up access to the data.

The description block is a well-formed XML document with a strictly defined format providing information about the following data block. It must be written in ASCII text (1-byte characters), and contain English text. This is done to simplify decoding of the description block on all platforms and parsing in any language by all readers of the SEG-Y format. Please refer to D-8 for a detailed description of the description block.

The data block following the description block can be of any format and have any contents in any language (binary, text, Unicode, big-endian, little-endian etc.)

The format is designed to make it easy to append and insert new blocks and delete existing blocks. This allows systems to add information to the record as processing stages refine and update the data.

Examples of data that can be stored in the trailer are information about survey, contract, processing system, edits, trace data description, deliverables, processing notes - decisions and result evaluation, etc.

The User Header and Data Trailer can also be used as a backup of information like positioning files (P1, P6, SPS etc.) or equipment/sensor information (from e.g. SEG-D).

They also allow bundling of any meta data with the seismic traces/cubes (e.g. database files, velocity models, processing scripts, observer logs). This can be used to automate and simplify the workflows, and improve consistency, correctness, robustness and repeatability of the seismic processing and interpretation.

Please refer to D-8 for more detailed description of the User header and Data trailer.

Appendix A. Writing SEG-Y Data to a Disk File

On modern UNIX and PC systems, a disk file is defined at the operating system level as a byte stream without any structure. It has become common practice for SEG-Y data to be streamed into a disk file, without any kind of encapsulation to recover record boundaries. Such a disk file can only be read by software that comprehends the SEG-Y format, since it must use certain values in the SEG-Y headers to reconstruct the original record stream. This appendix describes the rules that must be followed when un-encapsulated SEG-Y data is written to a disk file or network byte stream.

The first 3600 bytes of the file[18] are the "traditional" SEG-Y File Header (i.e. the 3200-byte Textual File Header followed by the 400-byte Binary Header). The Binary Header may be followed by zero or more 3200-byte Extended Textual File Header records, as indicated in bytes 3505–3506 of the Binary Header.

The first Data Trace record, beginning with the 240-byte SEG-Y Standard Trace Header, immediately follows the Binary File Header or if supplied, the last Extended Textual File Header. The number of bytes of Data Trace sample values that follow the Trace Header is determined from the value for number of samples in bytes 115–116 in the Trace Header and, if present, bytes 129–132 of SEG-Y Trace Header Extension 1, together with the sample format code in bytes 3225–3226 of the Binary Header. For format codes 1, 2, 4, 5 and 10, the number of bytes of sample data is four times the number of samples. For format codes 6, 9 and 12, the number of bytes of sample data is eight times the number of samples. For format codes 3 and 11, the number of bytes of sample data is twice the number of samples. For format codes 7 and 15, the number of bytes of sample data is three times the number of samples. For format codes 8 and 16, the number of bytes of sample data is the same as the number of samples.

The Trace Header for the second Data Trace in the file follows immediately after the sample data for the first trace and so on for subsequent traces in the file.

As with tape, all values may be written to the disk file using either the traditional "big-endian" byte ordering or the "little-endian" byte ordering now overwhelmingly used in modern computer architectures. For exchange purposes text in the Textual File Header and Extended Textual File Headers may be written in EBCDIC or ASCII (UTF-8) character code.

[18] Beware that a tape label may have been copied in front if the Textual File Header.

Appendix B. SEG-Y Tape Labels

In order to bring SEG-Y into line with SEG-D Rev 3.0, a label may, and should, be written at the front of a SEG-Y file on record-oriented removable media such as magnetic tape. This is a single record consisting of 128 bytes of ASCII characters, the same length as the SEG-D label and sharing a similar format description. A SEG-Y tape label is optional and is only valid on SEG-Y files written to unformatted, removable media. However, a label must be present if the blocking scheme described in Appendix C is being used. In this case the label must appear as a separate 128-byte record at the beginning of the file. There must be no file mark between the label record and the first data record.

If the recording medium supports multiple partitions, each partition is treated in isolation as if it were a separate unit. Thus, if labels are being used, each partition must begin with a label. Data from one partition cannot "run-over" into a subsequent partition. Each partition must be capable of being decoded in isolation. On one recording medium, it is permissible to mix partitions containing SEG-Y data with partitions containing non-SEG-Y formatted information.

The format of a SEG-Y Tape Label is summarized in Table 6.

Table 6 SEG-Y Tape Label

Field	Description	Bytes	Start - end byte
1	Storage Unit Sequence Number	4	1 – 4
2	SEG-Y Revision	5	5 – 9
3	Storage Unit Structure (fixed or variable)	6	10 – 15
4	Binding Edition	4	16 – 19
5	Maximum Block Size	10	20 – 29
6	Producer Organization Code	10	30 – 39
7	Creation Date	11	40 – 50
8	Serial Number	12	51 – 62
9	Reserved	6	63 – 68
10	Storage Set Identifier	60	
11	External Label Name	12	69 – 80
12	Recording Entity Name	24	81 – 104
13	User defined	14	105 – 118
14	Reserved	10	119 – 128

Field 1
The Storage Unit Sequence Number is an integer in the range 1 to 9999 that indicates the order in which the current storage unit occurs in the storage set. The first storage unit of a storage set has sequence number 1, the second 2 and so on. This number is represented using the characters 0 to 9,

right justified with leading blanks if needed to fill out the field (no leading zeros). The right-most character is in byte 4 of the label. *This field is optional.* If not used, it must be blank (filled with blank characters). This implies that this is the only storage unit within the storage set. Separate storage sets should be used for different data types.

Field 2
The SEG-Y Revision field indicates which revision of SEG-Y was used to record the data on this tape. SY2.1 indicates that the data was formatted using SEG-Y revision 2.1. *This field is required.*

Field 3
Storage Unit Structure is a name indicating the record structure of the storage unit. This name is left justified with trailing blanks if needed to fill out the field. The leftmost character is in byte 10 of the label. For SEG-Y revision 2 tapes, this field must contain "RECORD". *This field is required.*

"RECORD" - Records may be of variable length, ranging up to the block size length specified in the maximum block size field of the storage unit label (if not zero). If the maximum block size specified is zero, records may be of any length.

Field 4
Binding Edition is the character B in byte 16 of the label followed by a positive integer in the range 1 to 999 (no leading zeros), left justified with trailing blanks if needed to fill out the field. The integer value corresponds to the edition of the Part 3 of the API RP66 standard[19] used to describe the physical binding of the logical format to the storage unit. 'B2' is an appropriate value. ('B1' requires a file mark after the label.) *This field is required.*

Field 5
Maximum Block Size is an integer in the range of 0 to 4,294,967,295 (2^{32}-1), indicating the maximum block length for the storage unit, or 0 (zero) if undeclared. This number is represented using the characters 0 to 9, right justified, with leading blanks if necessary to fill out the field (no leading zeros). The rightmost character is byte 29 of the label. *A valid value or 0 (zero) must be recorded.*

Field 6
Producer Organization Code is an integer in the range of 0 to 4,294,967,295 (2^{32}-1) indicating the organization code of the storage unit producer. This number is represented using the characters 0 to 9, right justified, with leading blanks if necessary to fill out the field (no leading zeros). The rightmost character is byte 39 of the label. *This field is required.*

RP 66 Organization codes are assigned and maintained by Energistics, an affiliate of the Open Group. Please visit the Energistics website[20] for a list of the currently assigned codes. To request a new organization code, email Energistics at energisticsinfo@opengroup.org.

Field 7
Creation date is the earliest date that any current information was recorded on the storage unit. The date is represented in the form dd-MMM-yyyy, where yyyy is the year (e.g. 1996), MMM is one of (JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC) and dd is the day of the month in the range 1 to 31. Days 1 to 9 may have one leading blank. The separator is a hyphen (code 45_{10}). *This field is required.*

Field 8
Serial Number is an identifier used to distinguish the storage unit from other storage units in an archive of an enterprise. The specification and management of serial numbers is delegated to organizations using this standard. This field may be empty (i.e.

[19] w3.energistics.org/RP66/V2/rp66v2.html

[20] www.energistics.org/rp66-organization-codes

may contain all blanks, in which case no serial number is specified).

Field 9

This field is reserved and should be recorded as all blanks (code 32_{10}).

Field 10

The Storage set identifier is a descriptive name for the storage set. Every storage unit in the same storage set shall have the same value for the user defined portion of the storage set identifier in its storage unit label. It is subdivided into four individual fields.

Field 11

The External Label Name is nonblank when the tape has an external label. The characters in this field are right justified with leading blank characters as required. A physical label is optional, but if it exists, then this field is required only if the external label is different from the lower 6 characters of the Serial Number in Field 8.

Field 12

The Recording Entity Name should provide the originator of these SEG-Y data, whether from field recording or subsequent processing. Leave blank if unknown.

Field 13

The User defined field may contain any other user input information. The only restriction is that the data must be in ASCII.

Field 14

Reserved and should be recorded as all blanks (code 32_{10}).

Appendix C. Blocking of SEG-Y Files on Tape

This appendix describes a simple blocking scheme for writing SEG-Y files to tape. This is effectively a special encapsulation layer for SEG-Y and may be advantageous with certain tape devices that require a large block size, e.g., megabytes, to use the tape efficiently. Note however that this is not the only way to achieve SEG-Y file blocking and it may be preferable to use another encapsulation scheme such as SEG RODE.

In the following explanation, a SEG-Y record means a record defined in the SEG-Y standard (i.e., a 3200-byte Textual File Header record, a 400-byte Binary Header record, a 3200-byte Extended Textual File Header record or the Trace Header with its associated Data Trace). A tape record means a variable length physical record written to the tape device.

A tape containing SEG-Y data written using this blocking scheme must begin with a SEG-Y tape label, as described in Appendix B. The label must be written as a separate tape record 128 bytes long. If the tape medium supports partitioning, each partition is treated in isolation and must have its own label.

Each subsequent tape record may contain one or more SEG-Y records concatenated together. Each tape record must contain an integral number of SEG-Y records (i.e., the start of a tape record must coincide with the start of a SEG-Y record). The first tape record following the SEG-Y Tape Label must begin with the 3200-byte Textual File Header record. For all tape records in a file, the record length must be less than or equal to the maximum record length for the tape medium being used.

A SEG-Y reader program that comprehends this blocking scheme must unblock the data to reproduce the original SEG-Y record stream. In particular, it must use the number of samples in each trace and the sample format to determine the actual length of the trace record.

When this blocking scheme is being used, it is permitted to end one SEG-Y file and start a new one either with or without an intervening file mark. If a file mark is present, it signifies the end of a SEG-Y file and a file mark must be followed by either a tape record beginning with a 3200-byte Textual File Header record or another file mark. Alternatively, a new SEG-Y file can be identified by the start of a 3200-byte Textual File Header record. The Textual File Header would begin with a ‘C’ character ($C3_{16}$ in EBCDIC or ASCII 43_{16}), which is taken as the beginning of the new SEG-Y data set and have the SEG-Y revision level encoded in record C39 as described in section 4. It follows that the start of a SEG-Y file must start on a tape record boundary (i.e., any tape record contains data from only one SEG-Y file). In either case, a double file mark signifies end of data.

Appendix D. Extended Textual Stanzas

The structure for Extended Textual stanzas is described in section 6.1. The following stanzas are SEG-defined, standard stanzas. User-defined stanzas are permitted and provide a means to logically extend and customize the SEG-Y format to a user's particular needs. *It is highly advisable to use standard SEG-defined or SEG-approved stanza definitions.* When additional information is required beyond the standard definitions, a user defined stanza can be used to extend the standard stanza without repeating the information contained in the standard stanza.

All parameters defined in each stanza are required unless the stanza definition specifically notes that the inclusion of a parameter is optional. Since it is always assumned to be text, byte reordering does not apply to the Extended Textual Header.

Location Data Stanzas

Revision 2 of the SEG-Y format supports the International Association of Oil and Gas Producers (IOGP) P1/11 and P6/11 formats for defining location data. ***Legacy formats for location data have been removed from this document in favor of the IOGP formats.*** Summary information describing location data stanza structures and links to the IOGP information are detailed in Appendices D-1 and D-2.

D-1. Coordinate Reference System Definition: International Association of Oil and Gas Producers P1/11

The OGP P1/11 Geophysical position data exchange format is preferred for defining location data and data geographic extent. It can also be used interchangeably with OGP P6/11 for defining a survey coverage perimeter, although for a 3D survey this should more logically appear in OGP P6/11.

The stanza for OGP P1/11 Coordinate Reference System (CRS) definition takes the simple form of an initial

((OGP:P1/11 CRS))

followed by the text of the OGP P1/11 CRS definition, with sufficient trailing space characters to make the stanza a multiple of 3200 bytes. The stanza identifies the CRS to which source, receiver group and CDP coordinates given in the Standard Trace Header and Header Extension are referenced. The SEG recommends using the OGP P1/11 version 1.1 standard when backward compatibility is not an issue. Note that the “keyword = value” structure is not used in the OGP formats.

The OGP P1/11 standard is maintained by the International Association of Oil and Gas Producers (IOGP). The recommended method for defining the coordinate reference system (CRS) of location data is found in the Common Header section of the P1/11 standard. For further details and more information on usage please refer to the Geomatics Committee page of the IOGP website. In the Geophysical Operations Sub-Committee section there are links to the P1/11 Format (483-1) and User Guide (483-1u) with more examples and usage details.

The old CRS stanza format from SEG-Y rev 1 is still supported but deprecated (see Appendix D-3 of the rev 2.0 document). All new systems should use the format described in Appendix D-1 as it is computer readable and ensures compatibility between modern formats.

D-1.1 Format overview

All SEG formats, and SEG-Y and SEG-D in particular, aim to use the same CRS definition format, and will also share the definition text with IOGP P-formats to ensure compatibility throughout the seismic acquisition and processing.

The CRS identification blocks contain text like this:

```
((OGP:P1/11 CRS))
P-format CRS definition records (lines of text)
    :
```

The P-format CRS records are the same records as can be found in a P1/11 file. Table numbering in this appendix may not be sequential with the rest of this document but replicates table numbering in the P1/11 format description document.

After removing the ((OGP:P1/11 CRS)) string, the SEG-Y CRS definition string can be passed directly into any P1/11 compatible position or processing package. To that end the CRS section should begin with the 'OGP File Identification Record' to enable format recognition by the reading device.

The P1/11 CRS format supports lines longer than 80 characters. Also note the use of # (comment fields) and & (line breaks) used in the other SEG-Y stanzas are not allowed in the P-format. In P1/11 lines starting with 'C' are comments.

The string is broken up into multiples of 3200 bytes. If the string does not fit into multiples of 3200-byte blocks, the last block is padded with space (code 20_{16}).

As with any stanza header the text in (( )) is case and white space insensitive, so ((OGP:P1/11 CRS)) may be written ((ogp:p1/11crs)).

D-1.2 Logical File Structure

The data is stored in a series of variable length ASCII comma-separated data records, each terminated by a carriage return (Hex 0x0D) and/or a line feed (Hex 0x0A) character. Line termination shall be consistent throughout each file.

As the format is designed primarily for access by a computer program, there is no fixed limit on the length of each individual data record, and many record definitions allow multiple data items to be written into a single record. However, while it is recommended that systems make use of this facility to reduce file size where it is possible to do so, it is also recommended that records should not be written to excessive length but should instead be split across multiple records.

Although the format is primarily intended for computer access, it is also common for the file to be visually inspected, particularly the Common Header records. Thus, it is recommended that, particularly for the Common Header block, systems writing the files make use of spaces to pad any repeated records to ensure the data is aligned in columns to facilitate readability.

Thus, if possible, Common Header records should be written as:

```
HC,1,5,2,Latitude of natural origin                     ,1,8801,     0,3,degree
HC,1,5,2,Longitude of natural origin                    ,1,8802,   -15,3,degree
HC,1,5,2,Scale factor at natural origin                 ,1,8805,0.9996,4, unity
HC,1,5,2,False easting                                  ,1,8806,500000,1, metre
HC,1,5,2,False northing                                 ,1,8807,     0,1, metre
```

However, it should be noted, unless the field width is specifically stated in the record field definition, this padding of records for readability is a recommendation and not an absolute requirement. Note that for recording in SEG-Y format, if the record (or record set) does not fit a multiple of 3200-byte blocks the last block should be padded with space (code 20_{16}).

D-2. Seismic Bin Grid Definition: International Association of Oil and Gas Producers P6/11

The OGP P6/11 Seismic bin grid data exchange format is preferred for bin grid definition and survey coverage perimeter. It must start with an OGP File Identification record, followed by the CRS definition section in order to define the bin grid geometry, survey perimeter or bin node coordinates, depending on the purpose of the data.

The bin grid definition blocks contain text like this:

```
((OGP:P1/11 CRS))
P1-format CRS definition records (lines of text)
.
((OGP:P6/11 Coverage Perimeter))
P6-format perimeter position records (lines of text)
.
((OGP:P6/11 Bin Node Coordinates))
P6-format bin node position records (lines of text)
     :
```

There are header records specific to P6/11 necessary to give further definition to bin grid position records.

The P6/11 standard is maintained by the International Association of Oil and Gas Producers (IOGP). For further details and more information on usage please refer to the Geomatics Committee page of the IOGP website (http://www.iogp.org/Geomatics). In the Geophysical Operations Sub-Committee section there are links to the P6/11 Format (486-1) and User Guide (486-1u) with more examples and usage details.

The old Bin Grid Definition stanza format from SEG-Y rev 1 is still supported but deprecated (see Appendix D-4 of the rev 2.0 document).

The format and layout of this stanza is consistent with that described above in Appendix D-1.

D-3. Data Sample Measurement Unit

D-3.1 Stanza for Data Sample Measurement Unit

The Data Sample Measurement Unit stanza provides a means of defining a measurement unit other than the measurement units defined in SEG-Y Standard Trace Header bytes 203–204.

Table 7 Stanza for Data Sample Measurement Unit

Stanza and Keyword	Format	Comment
((SEG: Data Sample Measurement Unit ver 1.0))	Text	Stanza name
Data Sample Measurement Unit =	Text	A textual description of the measurement unit used for the data samples (i.e. millivolts, meters)
Volt conversion =	Real Number	The multiplicative constant that converts the Data Sample Measurement Unit to Volts.

D-3.2 Example stanza for Data Sample Measurement Unit

```
((SEG: Data Sample Measurement Unit ver 1.0))
Data Sample Measurement Unit =          Millivolts
Volt conversion =                       0.001
```

D-4. Processing History

The Processing History stanza provides a means to track the processing history of the seismic data traces.

D-4.1 Stanza for Processing History

Table 8 Stanza for Processing History

Stanza and Keyword	Format	Comment
((SEG: Processing History ver 1.0))	Text	Stanza name
The following six entries are repeated as needed to define all processing steps applied to the data traces.		
Processing Company =	Text	
Processing Software =	Text	
Input Data Set =	Text	Data set name or data set id of the data traces being processed
Processing Date =	Text	Date in YYYYMMDD-HHMMSS format
Process Applied =	Text	Name of the algorithm or program being applied to the data traces
Process Parameters =	Text	

D-4.2 Example stanza for Processing History

((SEG: Processing History ver 1.0))	
Processing Company =	**Expert Processing Inc.**
Processing Software =	**Omega**
Input Data Set =	**$ADBigDo_FieldSeq463**
Processing Date =	**20010519-231643**
Processing Applied =	**SEG-D edit**
Process Parameters =	**MP factor applied**
Processing Company =	**Expert Processing Inc.**
Processing Software =	**Omega**
Input Data Set =	**$ADBigDo_FieldSeq463**
Processing Date =	**20010519-231643**
Processing Applied =	**Trace select/sort**
Process Parameters =	**Data traces, Common Rcv Sort**
Processing Company =	**Expert Processing Inc.**
Processing Software =	**Omega**
Input Data Set =	**$ADBigDo_FieldSeq463_Edit**
Processing Date =	**20010520-115959**
Processing Applied =	**Predictive deconvolution**
Process Parameters =	**Surface consistent, 130 ms, 3 windows**

In this example, a field data set was edited and sorted into common receiver order as an initial process. In a second step, the edited data was deconvolved using a surface consistent deconvolution operator.

D-5. Source Type/Orientation

D-5.1 Stanza for Source Type/Orientation

The Source Type stanza allows the source types used during the data acquisition to be uniquely identified. The source type identifier is used in Trace Header bytes 217–218. This stanza is used when the predefined source types in the Trace Header bytes 217–218 do not adequately identify the sources used for acquisition or an expanded description is desired. When a source type is capable of generating energy in multiple orientations, a Source Type/Orientation stanza should be defined for each orientation.

Table 9 Stanza for Source Type/Orientation

Stanza and Keyword	Format	Comment
((SEG: Source Type/Orientation ver 1.0))	Text	Stanza name
Source description =	Text	A textual description of the source.
Source description (continued 1) =	Text	A textual description of the source.
Source description (continued 2) =	Text	A textual description of the source.
Source type identifier =	Negative Integer	The negative integer that will be used in SEG-Y Standard Trace Header bytes 217–218 to identify this source.

D-5.2 Example stanza for Source Type/Orientation

((SEG: Source Type/Orientation ver 1.0))	
Source Description =	**Inclined Impactor**
Source description (continued) =	**80-45 -45 incident angle**
Source description (continued) =	
Source type identifier =	**-6**
((SEG: Source Type/Orientation ver 1.0))	
Source Description =	**Inclined Impactor**
Source description (continued) =	**100-135-135 incident angle**
Source description (continued) =	
Source type identifier =	**-7**
((SEG: Source Type/Orientation ver 1.0))	
Source Description =	**Mini-shallow water air gun**
Source description (continued) =	**182 ci at 10,000 psi**
Source description (continued) =	
Source type identifier =	**-8**

D-6. Source Measurement Unit

D-6.1 Stanza for Source Measurement Unit

The Source Measurement Unit stanza provides a means of defining a measurement unit other than the measurement units defined in the Trace Header bytes 231-232.

Table 10 Stanza for Source Measurement Unit

Stanza and Keyword	Format	Comment
((SEG: Source Measurement Unit ver 1.0))	Text	Stanza name
Source Measurement Unit =	Text	A textual description of the measurement unit used for the source measurement (i.e. joules, millivolts, meters, vibrators, kilograms of dynamite, etc.)
Joule conversion =	Real Number	The multiplicative constant that converts the Source Measurement Unit to Joules. Specify the value of zero if the Source Measurement Unit cannot be converted to joules

D-6.2 Example stanza for Source Measurement Unit

((SEG: Source Measurement Unit ver 1.0))	
Source Measurement Unit =	**Vibrators * sweep length in seconds**
Joule conversion =	**0.0**

D-7. Stanza for Trace Header Mapping (Deprecated)

(Replaced by D-8. Stanza for Trace Header Layout)

The Trace Header Mapping stanza provides definitions for user-defined trace 240-byte header extensions and overrides of SEG-defined trace header standard mappings. The enumeration 1 … N is optional but is an aid to readability by grouping related entries.

Table 11 Stanza for Trace Header Mapping

Stanza and Keyword	Format	Comment
((SEG: Trace Header Mapping ver 1.0))	Text	Stanza name
Header name =	Text 8	The name that appears in the last 8 bytes of the trace header being mapped. SEG00000 represents the initial Standard SEG-Y Trace Header. It is suggested that these stanzas occur in the same sequence as the trace headers being mapped.
Delete field name =	Text	The name of an existing trace header field definition to be deleted. This optional keyword is provided to allow removal or replacement of a predefined trace header field. It is suggested that all “Delete field name” keywords appear immediately after the “Header name” entry.
Field name 1 =	Text	The name assigned to a new trace header field. If omitted, a name is autogenerated from the “Header name” and the field’s byte location, e.g. USER0001.21 for the field starting in byte 21 of header USER0001.
Byte position 1 =	Integer	The starting byte position of new trace header field 1. This must be in the range 1 to 240 and is required.
Value format code 1 =	Integer	The format code as listed for bytes 3225-3226 of the 400-byte Binary File Header. Use code 0 for ASCII or UTF-8 text. This keyword is required.
Number of values 1 =	Integer	The number of values provided in new trace header 1. If omitted, 1 is assumed. Byte position 1 plus the Number of values times the length of the Value format must not exceed 241.

Stanza and Keyword	Format	Comment
Field description 1 =	Text	Description of the trace header field.
…	…	…
Field name N =	Text	The name assigned to a new trace header field. If omitted, a name is autogenerated from the “Header name” and the field’s byte location, e.g. USER0001.211 for the field in byte 211 of header USER0001.
Byte position N =	Integer	The starting byte position of new trace header field N. This must be in the range 1 to 240 and is required.
Value format code N =	Integer	The format code as listed for bytes 3225-3226 of the 400-byte Binary File Header. Use code 0 for ASCII or UTF-8 text. This keyword is required.
Number of values N =	Integer	The number of values provided in new trace header N. If omitted, 1 is assumed. Byte position 1 plus the Number of values times the length of the Value format must not exceed 241.
Field description N =	Text	Description of the trace header field.

D-7.1 Example Stanzas for Trace Header Mapping

Stanzas for the two currently predefined SEG-Y trace headers are defined below. The field names have been chosen, where feasible, to match those used in the widely used open-source SU seismic processing package from the Colorado School of Mines Center for Wave Phenomena. Fields may also be referred to as SEG00000.x and SEG00001.x respectively, irrespective of these predefined names.

((SEG: Trace Header Mapping ver 1.0))

HEADER NAME = ...
SEG00000

FIELD NAME 1 = ...
TRACL

Byte position 1 =	**1**
Value format code 1 =	**2**
Field description 1 =	**Trace sequence number within line**
Field name 2 =	**tracr**
Byte position 2 =	**5**
Value format code 2 =	**2**
Field description 2 =	**Trace sequence number within SEG-Y file**
Field name 3 =	**fldr**
Byte position 3 =	**9**
Value format code 3 =	**2**
Field description 3 =	**Original field record number**
Field name 4 =	**tracf**
Byte position 4 =	**13**
Value format code 4 =	**2**
Field description 4 =	**Trace number within the original field record**
Field name 5 =	**ep**
Byte position 5 =	**17**
Value format code 5 =	**2**
Field description 5 =	**Energy source point number**
Field name 6 =	**cdp**
Byte position 6 =	**21**
Value format code 6 =	**2**
Field description 6 =	**Ensemble number**
Field name 7 =	**cdpt**
Byte position 7 =	**25**
Value format code 7 =	**2**
Field description 7 =	**Trace number within the ensemble**
Field name 8 =	**trid**
Byte position 8 =	**29**
Value format code 8 =	**3**
Field description 8 =	**Trace identification code**
Field name 9 =	**nvs**
Byte position 9 =	**31**
Value format code 9 =	**3**
Field description 9 =	**Number of vertically summed traces yielding this trace**
Field name 10 =	**nhs**
Byte position 10 =	**33**
Value format 10 =	**3**
Field description 10 =	**Number of horizontally stacked traces yielding this trace**
Field name 11 =	**duse**
Byte position 11 =	**35**

Value format 11 =	**3**
Field description 11 =	**Data use**
Field name 12 =	**offset**
Byte position 12 =	**37**
Value format 12 =	**2**
Field description 12 =	**Distance from center of the source point to the center of the receiver group**
Field name 13 =	**gelev**
Byte position 13 =	**41**
Value format 13 =	**2**
Field description 13 =	**Receiver group elevation**
Field name 14 =	**selev**
Byte position 14 =	**45**
Value format 14 =	**2**
Field description 14 =	**Surface elevation at source**
Field name 15 =	**sdepth**
Byte position 15 =	**49**
Value format 15 =	**2**
Field description 15 =	**Source depth below surface**
Field name 16 =	**gdel**
Byte position 16 =	**53**
Value format 16 =	**2**
Field description 16 =	**Seismic Datum elevation at receiver group**
Field name 17 =	**sdel**
Byte position 17 =	**57**
Value format 17 =	**2**
Field description 17 =	**Seismic Datum elevation at source**
Field name 18 =	**swdep**
Byte position 18 =	**61**
Value format 18 =	**2**
Field description 18 =	**Water column height at source location**
Field name 19 =	**gwdep**
Byte position 19 =	**65**
Value format 19 =	**2**
Field description 19 =	**Water column height at receiver group location**
Field name 20 =	**scalel**
Byte position 20 =	**69**
Value format 20 =	**3**
Field description 20 =	**Scalar to be applied to all elevations and depths in fields 13-19 and any extension in SEG-Y Trace Header SEG00001.**
Field name 21 =	**scalco**
Byte position 21 =	**71**
Value format 21 =	**3**
Field description 21 =	**Scalar to be applied to all coordinates specified in fields 22-25, 72-73, and any extension in SEG-Y Trace Header SEG00001.**
Field name 22 =	**sx**
Byte position 22 =	**73**
Value format 22 =	**2**
Field description 22 =	**Source coordinate - X**
Field name 23 =	**sy**
Byte position 23 =	**77**
Value format 23 =	**2**
Field description 23 =	**Source coordinate - Y**
Field name 24 =	**gx**
Byte position 24 =	**81**
Value format 24 =	**2**

Field description 24 =	**Group coordinate - X**
Field name 25 =	**gy**
Byte position 25 =	**85**
Value format 25 =	**2**
Field description 25 =	**Group coordinate - Y**
Field name 26 =	**counit**
Byte position 26 =	**89**
Value format 26 =	**3**
Field description 26 =	**Coordinate units**
Field name 27 =	**wevel**
Byte position 27 =	**91**
Value format 27 =	**3**
Field description 27 =	**Weathering velocity**
Field name 28 =	**swevel**
Byte position 28 =	**93**
Value format 28 =	**3**
Field description 28 =	**Subweathering velocity**
Field name 29 =	**sut**
Byte position 29 =	**95**
Value format 29 =	**3**
Field description 29 =	**Uphole time at source**
Field name 30 =	**gut**
Byte position 30 =	**97**
Value format 30 =	**3**
Field description 30 =	**Uphole time at group**
Field name 31 =	**sstat**
Byte position 31 =	**99**
Value format 31 =	**3**
Field description 31 =	**Source static correction**
Field name 32 =	**gstat**
Byte position 32 =	**101**
Value format 32 =	**3**
Field description 32 =	**Group static correction**
Field name 33 =	**tstat**
Byte position 33 =	**103**
Value format 33 =	**3**
Field description 33 =	**Total static applied**
Field name 34 =	**laga**
Byte position 34 =	**105**
Value format 34 =	**3**
Field description 34 =	**Lag time A**
Field name 35 =	**lagb**
Byte position 35 =	**107**
Value format 35 =	**3**
Field deswcription 35 =	**Lag time B**
Field name 36 =	**delrt**
Byte position 36 =	**109**
Value format 36 =	**3**
Field description 36 =	**Delay recording time**
Field name 37 =	**muts**
Byte position 37 =	**111**
Value format 37 =	**3**
Field description 37 =	**Mute time start**
Field name 38 =	**mute**
Byte position 38 =	**113**
Value format 38 =	**3**

Field description 38 =	**Mute time end**
Field name 39 =	**ns**
Byte position 39 =	**115**
Value format 39 =	**11**
Field description 39 =	**Number of samples in this trace**
Field name 40 =	**dt**
Byte position 40 =	**117**
Value format 40 =	**11**
Field description 40 =	**Sample interval for this trace**
Field name 41 =	**gain**
Byte position 41 =	**119**
Value format 41 =	**3**
Field description 41 =	**Gain type of field instruments**
Field name 42 =	**igc**
Byte position 42 =	**121**
Value format 42 =	**3**
Field description 42 =	**Instrument gain constant**
Field name 43 =	**igi**
Byte position 43 =	**123**
Value format 43 =	**3**
Field description 43 =	**Instrument early or initial gain**
Field name 44 =	**corr**
Byte position 44 =	**125**
Value format 44 =	**3**
Field description 44 =	**Correlated**
Field name 45 =	**sfs**
Byte position 45 =	**127**
Value format 45 =	**3**
Field description 45 =	**Sweep frequency at start**
Field name 46 =	**sfe**
Byte position 46 =	**129**
Value format 46 =	**3**
Field description 46 =	**Sweep frequency at end**
Field name 47 =	**slen**
Byte position 47 =	**131**
Value format 47 =	**3**
Field description 47 =	**Sweep length**
Field name 48 =	**styp**
Byte position 48 =	**133**
Value format 48 =	**3**
Field description 48 =	**Sweep type**
Field name 49 =	**stas**
Byte position 49 =	**135**
Value format 49 =	**3**
Field description 49 =	**Sweep trace taper length at start**
Field name 50 =	**stae**
Byte position 50 =	**137**
Value format 50 =	**3**
Field description 50 =	**Sweep trace taper length at end**
Field name 51 =	**tatyp**
Byte position 51 =	**139**
Value format 51 =	**3**
Field description 51 =	**Taper type**
Field name 52 =	**afilf**
Byte position 52 =	**141**
Value format 52 =	**3**

Field description 52 = **Alias filter frequency, if used**
Field name 53 = **afils**
Byte position 53 = **143**
Value format 53 = **3**
Field description 53 = **Alias filter slope**
Field name 54 = **nofilf**
Byte position 54 = **145**
Value format 54 = **3**
Field description 54 = **Notch filter frequency, if used**
Field name 55 = **nofils**
Byte position 55 = **147**
Value format 55 = **3**
Field description 55 = **Notch filter slope**
Field name 56 = **lcf**
Byte position 56 = **149**
Value format 56 = **3**
Field description 56 = **Low-cut frequency, if used**
Field name 57 = **hcf**
Byte position 57 = **151**
Value format 57 = **3**
Field description 57 = **High-cut frequency, if used**
Field name 58 = **lcs**
Byte position 58 = **153**
Value format 58 = **3**
Field description 58 = **Low-cut slope**
Field name 59 = **hcs**
Byte position 59 = **155**
Value format 59 = **3**
Field description 59 = **High-cut slope**
Field name 60 = **year**
Byte position 60 = **157**
Value format 60 = **3**
Field description 60 = **Year data recorded**
Field name 61 = **day**
Byte position 61 = **159**
Value format 61 = **3**
Field description 61 = **Day of year**
Field name 62 = **hour**
Byte position 62 = **161**
Value format 62 = **3**
Field description 62 = **Hour of day**
Field name 63 = **minute**
Byte position 63 = **163**
Value format 63 = **3**
Field description 63 = **Minute of hour**
Field name 64 = **sec**
Byte position 64 = **165**
Value format 64 = **3**
Field description 64 = **Second of minute**
Field name 65 = **timbas**
Byte position 65 = **167**
Value format 65 = **3**
Field description 65 = **Time basis code**
Field name 66 = **trwf**
Byte position 66 = **169**
Value format 66 = **3**

Field description 66 =	**Trace weighting factor**
Field name 67 =	**grnors**
Byte position 67 =	**171**
Value format 67 =	**3**
Field description 67 =	**Geophone group number of roll switch position one**
Field name 68 =	**grnofr**
Byte position 68 =	**173**
Value format 68 =	**3**
Field description 68 =	**Geophone group number of trace number one within original field record**
Field name 69 =	**grnlof**
Byte position 69 =	**175**
Value format 69 =	**3**
Field description 69 =	**Geophone group number of last trace within original field record**
Field name 70 =	**gaps**
Byte position 70 =	**177**
Value format 70 =	**3**
Field description 70 =	**Gap size**
Field name 71 =	**otrav**
Byte position 71 =	**179**
Value format 71 =	**3**
Field description 71 =	**Over travel associated with taper at beginning or end of line**
Field name 72 =	**cdpx**
Byte position 72 =	**181**
Value format 72 =	**2**
Field description 72 =	**X coordinate of ensemble (CDP) position of this trace**
Field name 73 =	**cdpy**
Byte position 73 =	**185**
Value format 73 =	**2**
Field desciption 73 =	**Y coordinate of ensemble (CDP) position of this trace**
Field name 74 =	**iline**
Byte position 74 =	**189**
Value format 74 =	**2**
Field description 74 =	**3-D poststack data in-line number**
Field name 75 =	**xline**
Byte position 75 =	**193**
Value format 75 =	**2**
Field description 75 =	**3-D poststack data cross-line number**
Field name 76 =	**sp**
Byte position 76 =	**197**
Value format 76 =	**2**
Field description 76 =	**Shotpoint number**
Field name 77 =	**spscal**
Byte position 77 =	**201**
Value format 77 =	**3**
Field description 77 =	**Scalar to be applied to shotpoint number in field 76**
Field name 78 =	**tvmu**
Byte position 78 =	**203**
Value format 78 =	**3**
Field description 78 =	**Time value measurement unit**
Field name 79 =	**trdman**
Byte position 79 =	**205**
Value format 79 =	**2**
Field description 79 =	**Transduction constant mantissa**
Field name 80 =	**trdexp**
Byte position 80 =	**209**

Value format 80 = **3**
Field description 80 = **Transduction constant exponent**
Field name 81 = **trdun**
Byte position 81 = **211**
Value format 81 = **3**
Field description 81 = **Transduction Units**
Field name 82 = **dti**
Byte position 82 = **213**
Value format 82 = **3**
Field description 82 = **Device/Trace Identifier**
Field name 83 = **timscal**
Byte position 83 = **215**
Value format 83 = **3**
Field description 83 = **Scalar to be applied to times in fields 29-**
Field name 84 = **stypor**
Byte position 84 = **217**
Value format 84 = **3**
Field description 84 = **Source Type/Orientation**
Field name 85 = **sedir**
Byte position 85 = **219**
Value format 85 = **3**
Number of values 85 = **3**
Field description 85 = **Source Energy Direction with respect to the source orientation**
Field name 86 = **smman**
Byte position 86 = **225**
Value format 86 = **2**
Field description 86 = **Source Measurement mantissa**
Field name 87 = **smexp**
Byte position 87 = **229**
Value format 87 = **3**
Field description 87 = **Source Measurement exponent**
Field name 88 = **smun**
Byte position 88 = **231**
Value format 88 = **3**
Field description 88 = **Source Measurement Unit**

((SEG: Trace Header Mapping ver 1.0))

HEADER NAME = ...
SEG00001

Field name 1 = **etracl**
Byte position 1 = **1**
Value format 1 = **12**
Field description 1 = **Extended trace sequence number within line**
Field name 2 = **etracr**
Byte position 2 = **9**
Value format 2 = **12**
Field description 2 = **Extended trace sequence number within SEG-Y file**
Field name 3 = **efldr**
Byte position 3 = **17**
Value format 3 = **9**
Field description 3 = **Extended original field record number**
Field name 4 = **ecdp**
Byte position 4 = **25**
Value format 4 = **9**

Field description 4 = **Extended ensemble number**
Field name 5 = **egelev**
Byte position 5 = **33**
Value format 5 = **6**
Field description 5 = **Extended elevation of receiver group**
Field name 6 = **gdepth**
Byte position 6 = **41**
Value format 6 = **6**
Field description 6 = **Receiver group depth below surface**
Field name 7 = **eselev**
Byte position 7 = **49**
Value format 7 = **6**
Field description 7 = **Extended surface elevation at source**
Field name 8 = **esdepth**
Byte position 8 = **57**
Value format 8 = **6**
Field description 8 = **Extended source depth below surface**
Field name 9 = **egdel**
Byte position 9 = **65**
Value format 9 = **6**
Field description 9 = **Extended Seismic Datum elevation at receiver group**
Field name 10 = **esdel**
Byte position 10 = **73**
Value format 10 = **6**
Field description 10 = **Extended Seismic Datum elevation at source**
Field name 11 = **eswdep**
Byte position 11 = **81**
Value format 11 = **6**
Field description 11 = **Extended water column height at source location**
Field name 12 = **egwdep**
Byte position 12 = **89**
Value format 12 = **6**
Field description 12 = **Extended water column height at receiver group location**
Field name 13 = **esx**
Byte position 13 = **97**
Value format 13 = **6**
Field description 13 = **Extended source coordinate — X**
Field name 14 = **esy**
Byte position 14 = **105**
Value format 14 = **6**
Field description 14 = **Extended source coordinate — Y**
Field name 15 = **egx**
Byte position 15 = **113**
Value format 15 = **6**
Field description 15 = **Extended group coordinate — X**
Field name 16 = **egy**
Byte position 16 = **121**
Value format 16 = **6**
Field description 16 = **Extended group coordinate — Y**
Field name 17 = **eoffset**
Byte position 17 = **129**
Value format 17 = **6**
Field description 17 = **Extended distance center of source to center of receiver group**
Field name 18 = **ens**
Byte position 18 = **137**
Value format 18 = **10**

Field description 18 =	**Extended number of samples in this trace**
Field name 19 =	**secfrac**
Byte position 19 =	**141**
Value format 19 =	**2**
Field description 19 =	**Nanoseconds to add to Second of minute**
Field name 20 =	**edt**
Byte position 20 =	**145**
Value format 20 =	**6**
Field description 20 =	**Extended sample interval in microseconds**
Field name 21 =	**cable**
Byte position 21 =	**153**
Value format 21 =	**2**
Field description 21 =	**Cable number for multicable acquisition**
Field name 22 =	**nthe**
Byte position 22 =	**157**
Value format 22 =	**11**
Field description 22 =	**Number of trace header extensions**
Field name 23 =	**lasttr**
Byte position 23 =	**159**
Value format 23 =	**11**
Field description 23 =	**Last trace flag**
Field name 24 =	**ecdpx**
Byte position 24 =	**161**
Value format 24 =	**6**
Field description 24 =	**Extended X coordinate of ensemble (CDP) position of this trace**
Field name 25 =	**ecdpy**
Byte position 25 =	**169**
Value format 25 =	**6**
Field description 25 =	**Extended Y coordinate of ensemble (CDP) position of this trace**

D-8. Stanza For Trace Header Layout

This appendix describes a schema for an XML file that defines the layout of the trace header entries in a SEG-Y file.

With SEG-Y revision 2, this XML file can be embedded in a SEG-Y file as Extended Textual Header stanza ((SEG:Layout)). In this context, the XML file is an alternative to the method described in Appendix D-7: "Stanza for Trace Header Mapping", and the old method may be removed from future versions of the standard.

The XML file can also exist as a separate text file that sits alongside a SEG-Y file. In fact, this can be particularly useful for legacy SEG-Y data (typically SEG-Y revision 0), to define the locations of key entries which did not have a standard location in the original standard, e.g. inline and crossline. This information would typically have been described in the EBCDIC reel header, but the XML file provides a more structured method for capturing this information.

Each entry in a trace header layout is given a name. These names are not under dictionary control, so that any name can be used. However, for maximum benefit when the layout is being used for exchange purposes, whenever possible it is recommended that the names used should correspond to those used in the examples in this appendix. If a name appears in more than one entry definition in a layout, the last entry will supersede the earlier definitions. This is of particular use when the layout contains definitions of entries in Trace Header Extension 1, where an entry can supersede an entry with the same name in the first 240-byte trace header.

Note that the XML file does not define the byte ordering of the entries in the SEG-Y file. For SEG-Y revision 2, the byte ordering used in the file can be determined from the value in bytes 3297-3300 of the file header. SEG-Y revision 0 and revision 1 always use big-endian byte ordering.

Table 12 describes the XML elements that are used in a SEG-Y Layout file.

Figure 3 shows a sample XML file for the trace header in a standard SEG-Y revision 2 file.
Figure 4 shows a sample XML file for the trace header in a standard SEG-Y revision 1 file.
Figure 5 shows a sample XML file for the trace header in a SEG-Y revision 0 file.

Table 12 XML Elements

Element Name	Description
segy-layout	The top level element of the XML file. Contains the following attributes: name The name of the layout (optional)
desc	Text containing a description of the layout. This element is optional.
entry	Contains the definition of a trace header entry. If the <entry> element appears at the top level of the XML file, the entry is contained in the first 240

Element Name	Description
	bytes of the trace header, i.e. the traditional SEG-Y trace header. If the <entry> element appears inside an <extension> element, the entry is contained within that trace header extension. Contains the following attributes: name The name of the entry. This should be an alphanumeric string with no blanks. byte The byte number of the first byte of the entry in the trace header (or trace header extension). The first byte in the trace header (or extension) is byte number 1. type Keyword defining the type of the entry. Possible values for type are shown in Table 13. if-non-zero The value of this attribute is a boolean value indicating that the value of this entry should only be used if its value is non-zero. If the attribute is not present, it is assumed to be false. This is to allow the definition of entries that appear in Trace Header Extension 1 where a non-zero value overrides a value in the Standard Trace Header.
extension	Contains the definition of entries contained in a 240-byte trace header extension. The <extension> element contains one or more <entry> elements. The byte number for an entry is relative to the start of the extension, i.e. the first byte of the extension is byte number 1. Contains the following attributes: name The name of the trace header extension

Element Name	Description
	This corresponds to the name in bytes 233-240 of the extension, e.g. the name of Trace Header Extension 1 is SEG00001..

Table 13 Trace Header Entry Types

Element Name	Description
int2	2-byte signed integer
int4	4-byte signed integer
int8	8-byte signed integer
uint2	2-byte unsigned integer
uint4	4-byte unsigned integer
uint8	8-byte unsigned integer
ibmfp	4-byte IBM floating-point
ieee32	4-byte IEEE floating-point
ieee64	8-byte IEEE floating-point
linetrc	4-byte unsigned integer representing trace sequence within line. On input this is treated the same as uint4. On output a new trace sequence will be generated.
reeltrc	4-byte unsigned integer representing trace sequence within reel. On input this is treated the same as uint4. On output a new trace sequence will be generated.
linetrc8	8-byte unsigned integer representing trace sequence within line. On input this is treated the same as uint8. On output a new trace sequence will be generated.
reeltrc8	8-byte unsigned integer representing trace sequence within reel. On input this is treated the same as uint8. On output a new trace sequence will be generated.
coor4	4-byte signed integer representing a coordinate using the scale value in entry co_scal (normally bytes 71-72).
elev4	4-byte signed integer representing an elevation or depth using the scale value in entry ed_scal (normally bytes 69-70).
time2	2-byte signed integer representing a time using

Element Name	Description
	the scale value in entry tm_scal (normally bytes 215-216).
spnum4	4-byte signed integer representing a shotpoint number using the scale value in entry sp_scal (normally bytes 201-202).
scale6	6-byte value representing a scale value to be applied to sample values. The value is made up of a 4-byte signed value which is the mantissa and a 2-byte signed value which is the power of ten exponent.

Figure 3 Sample Layout file for a standard SEG-Y revision 2 file

```
<?xml version="1.0" encoding="utf-8"?>
<segy-layout name="rev2">
  <desc>SEGY Rev 2 standard entries</desc>
  <entry name="linetrc" byte="1" type="linetrc"/>
  <entry name="reeltrc" byte="5" type="reeltrc"/>
  <entry name="ffid" byte="9" type="int4"/>
  <entry name="chan" byte="13" type="int4"/>
  <entry name="espnum" byte="17" type="int4"/>
  <entry name="cdp" byte="21" type="int4"/>
  <entry name="cdptrc" byte="25" type="int4"/>
  <entry name="trctype" byte="29" type="int2"/>
  <entry name="vstack" byte="31" type="int2"/>
  <entry name="fold" byte="33" type="int2"/>
  <entry name="rectype" byte="35" type="int2"/>
  <entry name="offset" byte="37" type="int4"/>
  <entry name="relev" byte="41" type="elev4"/>
  <entry name="selev" byte="45" type="elev4"/>
  <entry name="sdepth" byte="49" type="elev4"/>
  <entry name="rdatum" byte="53" type="elev4"/>
  <entry name="sdatum" byte="57" type="elev4"/>
  <entry name="wdepthso" byte="61" type="elev4"/>
  <entry name="wdepthrc" byte="65" type="elev4"/>
  <entry name="ed_scal" byte="69" type="int2"/>
  <entry name="co_scal" byte="71" type="int2"/>
  <entry name="sht_x" byte="73" type="coor4"/>
  <entry name="sht_y" byte="77" type="coor4"/>
  <entry name="rec_x" byte="81" type="coor4"/>
  <entry name="rec_y" byte="85" type="coor4"/>
  <entry name="coorunit" byte="89" type="int2"/>
  <entry name="wvel" byte="91" type="int2"/>
```

```
<entry name="subwvel" byte="93" type="int2"/>
<entry name="shuphole" byte="95" type="time2"/>
<entry name="rcuphole" byte="97" type="time2"/>
<entry name="shstat" byte="99" type="time2"/>
<entry name="rcstat" byte="101" type="time2"/>
<entry name="stapply" byte="103" type="time2"/>
<entry name="lagtimea" byte="105" type="time2"/>
<entry name="lagtimeb" byte="107" type="time2"/>
<entry name="delay" byte="109" type="time2"/>
<entry name="mutestrt" byte="111" type="time2"/>
<entry name="muteend" byte="113" type="time2"/>
<entry name="nsamps" byte="115" type="int2"/>
<entry name="dt" byte="117" type="int2"/>
<entry name="gaintype" byte="119" type="int2"/>
<entry name="ingconst" byte="121" type="int2"/>
<entry name="initgain" byte="123" type="int2"/>
<entry name="corrflag" byte="125" type="int2"/>
<entry name="sweepsrt" byte="127" type="int2"/>
<entry name="sweepend" byte="129" type="int2"/>
<entry name="sweeplng" byte="131" type="int2"/>
<entry name="sweeptyp" byte="133" type="int2"/>
<entry name="sweepstp" byte="135" type="int2"/>
<entry name="sweepetp" byte="137" type="int2"/>
<entry name="tapertyp" byte="139" type="int2"/>
<entry name="aliasfil" byte="141" type="int2"/>
<entry name="aliaslop" byte="143" type="int2"/>
<entry name="notchfil" byte="145" type="int2"/>
<entry name="notchslp" byte="147" type="int2"/>
<entry name="lowcut" byte="149" type="int2"/>
<entry name="highcut" byte="151" type="int2"/>
<entry name="lowcslop" byte="153" type="int2"/>
<entry name="hicslop" byte="155" type="int2"/>
<entry name="year" byte="157" type="int2"/>
<entry name="day" byte="159" type="int2"/>
<entry name="hour" byte="161" type="int2"/>
<entry name="minute" byte="163" type="int2"/>
<entry name="second" byte="165" type="int2"/>
<entry name="timebase" byte="167" type="int2"/>
<entry name="trweight" byte="169" type="int2"/>
<entry name="rstaswp1" byte="171" type="int2"/>
<entry name="rstatrc1" byte="173" type="int2"/>
<entry name="rstatrcn" byte="175" type="int2"/>
<entry name="gapsize" byte="177" type="int2"/>
<entry name="overtrvl" byte="179" type="int2"/>
<entry name="cdp_x" byte="181" type="coor4"/>
```

```
  <entry name="cdp_y" byte="185" type="coor4"/>
  <entry name="iline" byte="189" type="int4"/>
  <entry name="xline" byte="193" type="int4"/>
  <entry name="sp" byte="197" type="spnum4"/>
  <entry name="sp_scal" byte="201" type="int2"/>
  <entry name="samp_unit" byte="203" type="int2"/>
  <entry name="trans_const" byte="205" type="scale6"/>
  <entry name="trans_unit" byte="211" type="int2"/>
  <entry name="dev_id" byte="213" type="int2"/>
  <entry name="tm_scal" byte="215" type="int2"/>
  <entry name="src_type" byte="217" type="int2"/>
  <entry name="src_dir1" byte="219" type="int2"/>
  <entry name="src_dir2" byte="221" type="int2"/>
  <entry name="src_dir3" byte="223" type="int2"/>
  <entry name="smeasure" byte="225" type="scale6"/>
  <entry name="sm_unit" byte="231" type="int2"/>
  <extension name="SEG00001">
    <entry name="linetrc" byte="1" type="linetrc8" if-non-zero="1"/>
    <entry name="reeltrc" byte="9" type="reeltrc8" if-non-zero="1"/>
    <entry name="ffid" byte="17" type="int8" if-non-zero="1"/>
    <entry name="cdp" byte="25" type="int8" if-non-zero="1"/>
    <entry name="relev" byte="33" type="ieee64" if-non-zero="1"/>
    <entry name="rdepth" byte="41" type="ieee64"/>
    <entry name="selev" byte="49" type="ieee64" if-non-zero="1"/>
    <entry name="sdepth" byte="57" type="ieee64" if-non-zero="1"/>
    <entry name="rdatum" byte="65" type="ieee64" if-non-zero="1"/>
    <entry name="sdatum" byte="73" type="ieee64" if-non-zero="1"/>
    <entry name="wdepthso" byte="81" type="ieee64" if-non-zero="1"/>
    <entry name="wdepthrc" byte="89" type="ieee64" if-non-zero="1"/>
    <entry name="sht_x" byte="97" type="ieee64" if-non-zero="1"/>
    <entry name="sht_y" byte="105" type="ieee64" if-non-zero="1"/>
    <entry name="rec_x" byte="113" type="ieee64" if-non-zero="1"/>
    <entry name="rec_y" byte="121" type="ieee64" if-non-zero="1"/>
    <entry name="offset" byte="129" type="ieee64" if-non-zero="1"/>
    <entry name="nsamps" byte="137" type="uint4" if-non-zero="1"/>
    <entry name="nanosecs" byte="141" type="int4"/>
    <entry name="dt" byte="145" type="ieee64" if-non-zero="1"/>
    <entry name="cable_num" byte="153" type="int4"/>
    <entry name="last_trc" byte="159" type="int2"/>
    <entry name="cdp_x" byte="161" type="ieee64" if-non-zero="1"/>
    <entry name="cdp_y" byte="169" type="ieee64" if-non-zero="1"/>
  </extension>
</segy-layout>
```

Figure 4 Sample Layout file for a standard SEG-Y revision 1 file

```
<?xml version="1.0" encoding="utf-8"?>
<segy-layout name="rev1">
  <desc>SEGY Rev 1 standard entries</desc>
  <entry name="linetrc" byte="1" type="linetrc"/>
  <entry name="reeltrc" byte="5" type="reeltrc"/>
  <entry name="ffid" byte="9" type="int4"/>
  <entry name="chan" byte="13" type="int4"/>
  <entry name="espnum" byte="17" type="int4"/>
  <entry name="cdp" byte="21" type="int4"/>
  <entry name="cdptrc" byte="25" type="int4"/>
  <entry name="trctype" byte="29" type="int2"/>
  <entry name="vstack" byte="31" type="int2"/>
  <entry name="fold" byte="33" type="int2"/>
  <entry name="rectype" byte="35" type="int2"/>
  <entry name="offset" byte="37" type="int4"/>
  <entry name="relev" byte="41" type="elev4"/>
  <entry name="selev" byte="45" type="elev4"/>
  <entry name="sdepth" byte="49" type="elev4"/>
  <entry name="rdatum" byte="53" type="elev4"/>
  <entry name="sdatum" byte="57" type="elev4"/>
  <entry name="wdepthso" byte="61" type="elev4"/>
  <entry name="wdepthrc" byte="65" type="elev4"/>
  <entry name="ed_scal" byte="69" type="int2"/>
  <entry name="co_scal" byte="71" type="int2"/>
  <entry name="sht_x" byte="73" type="coor4"/>
  <entry name="sht_y" byte="77" type="coor4"/>
  <entry name="rec_x" byte="81" type="coor4"/>
  <entry name="rec_y" byte="85" type="coor4"/>
  <entry name="coorunit" byte="89" type="int2"/>
  <entry name="wvel" byte="91" type="int2"/>
  <entry name="subwvel" byte="93" type="int2"/>
  <entry name="shuphole" byte="95" type="time2"/>
  <entry name="rcuphole" byte="97" type="time2"/>
  <entry name="shstat" byte="99" type="time2"/>
  <entry name="rcstat" byte="101" type="time2"/>
  <entry name="stapply" byte="103" type="time2"/>
  <entry name="lagtimea" byte="105" type="time2"/>
  <entry name="lagtimeb" byte="107" type="time2"/>
  <entry name="delay" byte="109" type="time2"/>
  <entry name="mutestrt" byte="111" type="time2"/>
  <entry name="muteend" byte="113" type="time2"/>
  <entry name="nsamps" byte="115" type="int2"/>
  <entry name="dt" byte="117" type="int2"/>
```

```
<entry name="gaintype" byte="119" type="int2"/>
<entry name="ingconst" byte="121" type="int2"/>
<entry name="initgain" byte="123" type="int2"/>
<entry name="corrflag" byte="125" type="int2"/>
<entry name="sweepsrt" byte="127" type="int2"/>
<entry name="sweepend" byte="129" type="int2"/>
<entry name="sweeplng" byte="131" type="int2"/>
<entry name="sweeptyp" byte="133" type="int2"/>
<entry name="sweepstp" byte="135" type="int2"/>
<entry name="sweepetp" byte="137" type="int2"/>
<entry name="tapertyp" byte="139" type="int2"/>
<entry name="aliasfil" byte="141" type="int2"/>
<entry name="aliaslop" byte="143" type="int2"/>
<entry name="notchfil" byte="145" type="int2"/>
<entry name="notchslp" byte="147" type="int2"/>
<entry name="lowcut" byte="149" type="int2"/>
<entry name="highcut" byte="151" type="int2"/>
<entry name="lowcslop" byte="153" type="int2"/>
<entry name="hicslop" byte="155" type="int2"/>
<entry name="year" byte="157" type="int2"/>
<entry name="day" byte="159" type="int2"/>
<entry name="hour" byte="161" type="int2"/>
<entry name="minute" byte="163" type="int2"/>
<entry name="second" byte="165" type="int2"/>
<entry name="timebase" byte="167" type="int2"/>
<entry name="trweight" byte="169" type="int2"/>
<entry name="rstaswp1" byte="171" type="int2"/>
<entry name="rstatrc1" byte="173" type="int2"/>
<entry name="rstatrcn" byte="175" type="int2"/>
<entry name="gapsize" byte="177" type="int2"/>
<entry name="overtrvl" byte="179" type="int2"/>
<entry name="cdp_x" byte="181" type="coor4"/>
<entry name="cdp_y" byte="185" type="coor4"/>
<entry name="iline" byte="189" type="int4"/>
<entry name="xline" byte="193" type="int4"/>
<entry name="sp" byte="197" type="spnum4"/>
<entry name="sp_scal" byte="201" type="int2"/>
<entry name="samp_unit" byte="203" type="int2"/>
<entry name="trans_const" byte="205" type="scale6"/>
<entry name="trans_unit" byte="211" type="int2"/>
<entry name="dev_id" byte="213" type="int2"/>
<entry name="tm_scal" byte="215" type="int2"/>
<entry name="src_type" byte="217" type="int2"/>
<entry name="src_dir1" byte="219" type="int2"/>
<entry name="src_dir2" byte="221" type="int2"/>
```

```
  <entry name="src_dir3" byte="223" type="int2"/>
  <entry name="smeasure" byte="225" type="scale6"/>
  <entry name="sm_unit" byte="231" type="int2"/>
</segy-layout>
```

Figure 5 Sample Layout file for a standard SEG-Y revision 0 file

```
<?xml version="1.0" encoding="utf-8"?>
<segy-layout name="rev0">
  <desc>SEGY Rev 0 standard entries</desc>
  <entry name="linetrc" byte="1" type="linetrc"/>
  <entry name="reeltrc" byte="5" type="reeltrc"/>
  <entry name="ffid" byte="9" type="int4"/>
  <entry name="chan" byte="13" type="int4"/>
  <entry name="espnum" byte="17" type="int4"/>
  <entry name="cdp" byte="21" type="int4"/>
  <entry name="cdptrc" byte="25" type="int4"/>
  <entry name="trctype" byte="29" type="int2"/>
  <entry name="vstack" byte="31" type="int2"/>
  <entry name="fold" byte="33" type="int2"/>
  <entry name="rectype" byte="35" type="int2"/>
  <entry name="offset" byte="37" type="int4"/>
  <entry name="relev" byte="41" type="elev4"/>
  <entry name="selev" byte="45" type="elev4"/>
  <entry name="sdepth" byte="49" type="elev4"/>
  <entry name="rdatum" byte="53" type="elev4"/>
  <entry name="sdatum" byte="57" type="elev4"/>
  <entry name="wdepthso" byte="61" type="elev4"/>
  <entry name="wdepthrc" byte="65" type="elev4"/>
  <entry name="ed_scal" byte="69" type="int2"/>
  <entry name="co_scal" byte="71" type="int2"/>
  <entry name="sht_x" byte="73" type="coor4"/>
  <entry name="sht_y" byte="77" type="coor4"/>
  <entry name="rec_x" byte="81" type="coor4"/>
  <entry name="rec_y" byte="85" type="coor4"/>
  <entry name="coorunit" byte="89" type="int2"/>
  <entry name="wvel" byte="91" type="int2"/>
  <entry name="subwvel" byte="93" type="int2"/>
  <entry name="shuphole" byte="95" type="int2"/>
  <entry name="rcuphole" byte="97" type="int2"/>
  <entry name="shstat" byte="99" type="int2"/>
  <entry name="rcstat" byte="101" type="int2"/>
  <entry name="stapply" byte="103" type="int2"/>
  <entry name="lagtimea" byte="105" type="int2"/>
  <entry name="lagtimeb" byte="107" type="int2"/>
  <entry name="delay" byte="109" type="int2"/>
```

```
  <entry name="mutestrt" byte="111" type="int2"/>
  <entry name="muteend" byte="113" type="int2"/>
  <entry name="nsamps" byte="115" type="int2"/>
  <entry name="dt" byte="117" type="int2"/>
  <entry name="gaintype" byte="119" type="int2"/>
  <entry name="ingconst" byte="121" type="int2"/>
  <entry name="initgain" byte="123" type="int2"/>
  <entry name="corrflag" byte="125" type="int2"/>
  <entry name="sweepsrt" byte="127" type="int2"/>
  <entry name="sweepend" byte="129" type="int2"/>
  <entry name="sweeplng" byte="131" type="int2"/>
  <entry name="sweeptyp" byte="133" type="int2"/>
  <entry name="sweepstp" byte="135" type="int2"/>
  <entry name="sweepetp" byte="137" type="int2"/>
  <entry name="tapertyp" byte="139" type="int2"/>
  <entry name="aliasfil" byte="141" type="int2"/>
  <entry name="aliaslop" byte="143" type="int2"/>
  <entry name="notchfil" byte="145" type="int2"/>
  <entry name="notchslp" byte="147" type="int2"/>
  <entry name="lowcut" byte="149" type="int2"/>
  <entry name="highcut" byte="151" type="int2"/>
  <entry name="lowcslop" byte="153" type="int2"/>
  <entry name="hicslop" byte="155" type="int2"/>
  <entry name="year" byte="157" type="int2"/>
  <entry name="day" byte="159" type="int2"/>
  <entry name="hour" byte="161" type="int2"/>
  <entry name="minute" byte="163" type="int2"/>
  <entry name="second" byte="165" type="int2"/>
  <entry name="timebase" byte="167" type="int2"/>
  <entry name="trweight" byte="169" type="int2"/>
  <entry name="rstaswp1" byte="171" type="int2"/>
  <entry name="rstatrc1" byte="173" type="int2"/>
  <entry name="rstatrcn" byte="175" type="int2"/>
  <entry name="gapsize" byte="177" type="int2"/>
  <entry name="overtrvl" byte="179" type="int2"/>
</segy-layout>
```

D-8.1 Minimum Requirements by Data Type

The following lists the minimum entry names required for Field, Pre-Stack and Post-Stack data. Using these in combination with the binary header flags for format version, data type and fixed trace length enables a computer to auto-read the dataset.

Field Data

```
<entry name="ffid" byte="9" type="int4"/>
<entry name="chan" byte="13" type="int4"/>
<entry name="espnum" byte="17" type="int4"/>
```

Pre-Stack Data

```
<entry name="ffid" byte="9" type="int4"/>
<entry name="chan" byte="13" type="int4"/>
<entry name="espnum" byte="17" type="int4"/>
<entry name="cdp" byte="21" type="int4"/>
<entry name="cdptrc" byte="25" type="int4"/>
<entry name=”trctype” byte=”29” type="int2"/>
<entry name="co_scal" byte="71" type="int2"/>
<entry name="sht_x" byte="73" type="coor4"/>
<entry name="sht_y" byte="77" type="coor4"/>
<entry name="rec_x" byte="81" type="coor4"/>
<entry name="rec_y" byte="85" type="coor4"/>
<entry name="coorunit" byte="89" type="int2"/>
```

Post-Stack Data

```
<entry name="cdp" byte="21" type="int4"/>
<entry name="cdptrc" byte="25" type="int4"/>
<entry name “offset” byte=”37” type=”int4”/>
<entry name="co_scal" byte="71" type="int2"/>
<entry name="sht_x" byte="73" type="coor4"/>
<entry name="sht_y" byte="77" type="coor4"/>
<entry name="rec_x" byte="81" type="coor4"/>
<entry name="rec_y" byte="85" type="coor4"/>
<entry name="coorunit" byte="89" type="int2"/>
<entry name=”inline” byte=”189” type=”int4”/>
<entry name=”xline” byte=”193” type=”int4”/>
```

D-9. User Data stanza

The User Data stanza **((SEG:UserData))** contains a set of user-defined blocks separated by a description block identifying the producer, contents, size and data format of the data block following it.

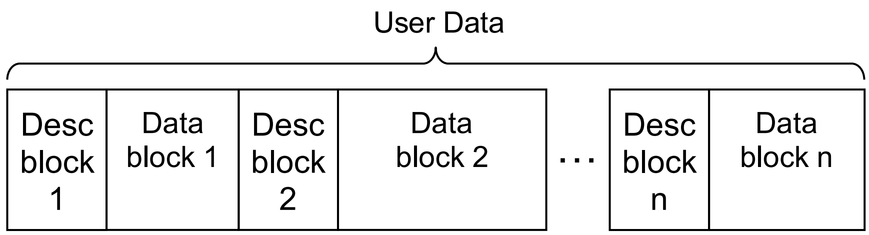

Figure 6 Structure of a User Data stanza containing n data blocks

For efficiency we recommended the User Data stanza in the Extended Textual Header be used for information that is required for interpreting the data traces. Large data blocks like backup of positioning data, databases etc. should be stored in the User Data trailer stanza. This will simplify decoding and interpretation of, and speed up access to, the data.

The description block is a well-formed XML document with a strictly defined format providing information about the following data block. It must be written in ASCII text (1 byte characters), and contain English text only. This simplifies decoding of the description block on all platforms and in any locale for all readers of the SEG-Y format.

The data blocks can be any size—there is no padding between blocks.

The description block follows directly after the last data byte of the previous block.

The last data block is padded with binary 0's to fill the last 3200-byte record if necessary.

The data block following the description block can be of any format and have any contents in any language (binary, text, Unicode, big-endian, little-endian etc.)

The format is designed to make it easy to append and insert new blocks, and delete existing blocks. This allows systems to add information to the SEG-Y record as processing stages refines and updates the data.

Examples of data that can be stored in the User Data stanza are information about survey, contract, processing system, edits, trace data description, parameters and deliverables, processing notes - decisions and result evaluation, etc.

The User Data stanza also allows bundling of any meta data with the seismic traces/cubes (e.g. database files, velocity models, processing scripts, observer logs). This can be used to automate and simplify the workflows, and improve consistency, correctness, robustness and repeatability of the seismic processing and interpretation.

The trailer can also be used as a backup of information like positioning files (P1, P6, SPS etc.) or equipment/sensor information (from e.g. SEG-D).

D-9.1 Description block

The description block is a well-formed XML 1.0 compatible[21] document that describes the following data block. It must be written in ASCII (1 byte characters), and contain only English text. This is done to simplify decoding of the description block on all platforms (no endian or encoding issues) and simplify parsing in any language by all readers of the SEG-Y format.

The XML structure is also simplified to allow decoding by software without implementing a full XML reader.

The XML structure is single level only, and contains only tags, attributes are not allowed. Also comments and other special tags are not allowed (<?> <!> etc.).

The XML structure may contain line change (ASCII CR($0D_{16}$) or ASCII LF($0A_{16}$) or CRLF($0D0A_{16}$)) and indentation (ASCII SPACE(20_{16}) and TAB(09_{16})) to improve human readability, though this is not required.

Tags are written in lower case. All values are case insensitive.

The following tags are supported in the description block, other tags are not allowed:

Tag	Description	Required
segydescblock	Top level tag. If the User header or Data trailer contains errors readers can search for this tag to find the start of next block.	X
version	Version number of description block format, this table describes format version **1**	X
blocksize	Size of data block in bytes.	X
dataformat	Short string describing the format of the data in the data block Starts with *text-* for textual data, and *bin-* for binary data. Text formats can either contain encoding information in the datablock, or specify it explicitly in the dataformat tag (ex: text-utf-8). Html, xml and xhtml are examples of text formats that contain encoding information in the data itself. Encoding follows the html standard for naming (us-ascii, utf-8, ISO-8859-1, iso-8859-5, x-euc-jp etc.) If the byte order/endian is required (the data format does not imply byte order, or specify it in the data format itself) *be-* and *le-* is used to indicate byte order. Some examples below may list multiple similar type dataformats in one table cell for efficiency reasons, but one block can only have one dataformat. Examples (this is just a set of examples, not a complete list):	X

[21] See http://www.w3.org/TR/2008/REC-xml-20081126/ for a suitable XML 1.0 reference.

dataformat

Description

text-us-ascii

ASCII text

text-xml

XML text containing definition of encoding (in e.g. <?xml> tag)

text-be-utf-8

UTF 8 big endian unicode text

text-ISO-8859-1

ISO-8859-1 (latin1) Western European text

text-p1/11

P1/11 file

text-html

HTML text containing definition of encoding (in e.g. <meta> tag)

text-other

Other unspecified formatted text. Only use if no other option is available. Readers need to use datalabel, creator and dataformat tags to determine how to decode.

Binary formats will typically use a file format as dataformat code.

Examples (this is just a set of examples, not a complete list):

dataformat

Description

bin-doc
bin-docx
bin-xlsx
bin-ppt
bin-ods
bin-odt

bin-odp
bin-pdf

Data block contains a Microsoft Word/Excel/Powerpoint, an Open Office word processor/spreadsheet/presentation or a pdf document.

The content of the document is specified in datalabel/description tags. This may be any document like survey description, contract information, job specifications, observer log, processing notes, result evaluations, information to downstream processing and interpretation, etc.

Note: These are human readable formats only, and the use of computer readable formats is recommended when possible.

bin-hdf5

HDF 5 formatted binary data.

bin-zip
bin-tar

Data block contains multiple files compressed into a single zip or tar.

Useful for backup, but the use of individual blocks for each file is recommended if the archive contains multiple data types.

Refer to datalabel and description tags for more information about actual contents.

bin-jar

Java archive containing executable Java byte code

bin-python

Python script code

bin-iso

CD or DVD ISO-9660 image

bin-mpeg4

MPEG4 encoded video file

bin-mp3

MP3 encoded audio file

	bin-sqlite SQLite database bin-segd bin-segy Data block contains a SEG-D or SEG-Y file. text-myformat bin-myformat Format of data block is a format called "myformat". The format is described in another block. The labelname tag for the format description block should be TextMyformatDesc or BinMyformatDesc respectively. The format description is typically a Word/xml/html/text document or similar. Description tag text should also state which data block (by labelname) contains the format description. bin-le-other Unspecified binary little endian format. Typically used for a proprietary binary formats. Readers need to use datalabel, creator and dataformat to determine how to decode. It is recommended to use the "text-myformat/bin-myformat" method described above if possible, and include the format description in another block.	
datalabel	The label describes the type of data in a short string. The field can consist of any alphanumeric characters (a-zA-Z0-9). The datalabel is defined by the recorder. Example: SURVEYINFO, VELMODEL. The datalabel must be unique for each creator, i.e. the combination creator and datalabel must identify one and only one type of data block. Data labels starting with the letters *SEG* is reserved for usage by the Society of Exploration Geophysicists.	X
description	A textual description of the contents of the data block. Designed for human consumption and can contain more details about creator and data (who, when, what, why), comments, URL references to more information, etc.	

creationtime	Timestamp for creation of the data block. May be set to the recording time for the SEG-Y record if unknown. Cannot be empty or dummy timestamp. Timestamp format is UTC time (zero GMT time offset) according to ISO 8601 (24 hour clock): Format: **YYYY-MM-DD hh:mm:ssZ** Example: **2015-09-01 22:04:00Z**	X
creator	Name of creator of the data block, usually the name of the system making the data block, for example “NavProc”.	X
dataversion	Version number of the contents of the data block, user defined, for example “1.1”, “2002.b2”. If omitted it implies dataversion “1”, or the data block itself contains a version number.	
compression	If the data block is compressed this specifies the compression method. Proprietary compression methods are allowed, but must be specified by name. Uncompressing by the specified method is necessary to get to the format specified in the dataformat tag. Omit the compression tag or set the value to **none** if data block is not compressed. Examples: zip, bzip2, gzip, compress, snappy	

The combination dataformat, datalabel, dataversion and creator is used to determine how the data should be decoded and used.

D-9.2 Data block

The data block consists of a number of bytes of data as described in the description block. The format of the data block is completely user defined. The data block starts at the first byte after `</segydescblock>`.

D-9.3 Examples

Description block example:

```
<segydescblock>
  <version>1</version>
  <blocksize>16340</blocksize>
  <dataformat>text-xml</dataformat>
  <datalabel>MSPJobParams</datalabel>
  <description>Full set of job parameters in XML format as described in
My Seismic Processing v3.14 User manual Appendix F. Available for
download at
http://www.myseismicproc.com/sw/3.14/downloads.php</description>
  <creationtime>2015-10-30 19:58:17Z</creationtime>
  <creator>My Seismic Processing</creator>
  <dataversion>14</dataversion>
  <compression>none</compression>
</segydescblock>
```

Full User header / Data trailer example consisting of two blocks

Text blocks only as binary data is difficult to write in a document. Padding to fill 3200-byte block is also not shown.

```
((SEG:User Data ver 1.0))
<segydescblock>
  <version>1</version>
  <blocksize>155</blocksize>
  <dataformat>text-xml</dataformat>
  <datalabel>SurvPars</datalabel>
  <description>Short example of survey parameters in xml.</description>
  <creationtime>2015-10-03 00:00:00Z</creationtime>
  <creator>DataProc</creator>
</segydescblock><?xml version=”1.0” encoding=”UTF-8”?>
<dataproc>
<surveyparams version=”3.0”>
  <area>南海</area>
  <contractid>C320012/2015</contractid>
</surveyparams><segydescblock>
  <version>1</version>
  <blocksize>240</blocksize>
  <dataformat>text-iso-8859-1</dataformat>
  <datalabel>ObsLog</datalabel>
  <description>Short textual observer log example</description>
  <creationtime>2015-07-04 09:00:00Z</creationtime>
  <creator>MyAcqSystem</creator>
  <dataversion>7</dataversion>
</segydescblock>Sequence 001:
11:38    Start of line SK15P1001
12:03    Shots 2001-3200 - Seismic interference from vessel “Støyende
Ørn” NW of survey area
13:18    Missed shots 2155-2157
15:12    End of line SK15P1001
Sequence 002:
17:45    Start of line SK15P1312
```

Example of a user defined block format with description

```
((SEG:User Data ver 1.0))
<segydescblock>
  <version>1</version>
  <blocksize>810221411</blocksize>
  <dataformat>bin-mspprj3</dataformat>
  <datalabel>MSPProjectFile</datalabel>
  <description>My Seismic Processing v3.14 project database for survey
SK1501NS. File format described in block labelled
BinMSPPRJ3Desc.</description>
  <creationtime>2015-10-30 19:58:17Z</creationtime>
  <creator>My Seismic Processing</creator>
  <dataversion>3.14</dataversion>
  <compression>none</compression>
</segydescblock>
…………
<segydescblock>
  <version>1</version>
  <blocksize>154340</blocksize>
```

```
  <dataformat>bin-docx</dataformat>
  <datalabel>BinMSPPRJ3Desc</datalabel>
  <description>My Seismic Processing v3.14 project database format
description.</description>
  <creationtime>2015-10-30 19:58:17Z</creationtime>
  <creator>My Seismic Processing</creator>
</segydescblock>
…………
```

P1 file backup using tar and gzip

```
((SEG:User Data ver 1.0))
<segydescblock>
  <version>1</version>
  <blocksize>27320014096</blocksize>
  <dataformat>bin-tar</dataformat>
  <datalabel>MSPPosBackup</datalabel>
  <description>Tar and gzipped file containing all final P1/90 or P1/11
files used for processing this survey</description>
  <creationtime>2015-10-30 19:58:17Z</creationtime>
  <creator>My Seismic Processing</creator>
  <dataversion>1</dataversion>
  <compression>gzip</compression>
</segydescblock>
```

Determine how to decode data block

This is done by examining dataformat, compression, datalabel, dataversion and creator tags in the description block.

The description tag should also provide help information on how to decode

1. Description contains information about how to decode

```
<segydescblock>
  <version>1</version>
  <blocksize>16340</blocksize>
  <dataformat>text-xml</dataformat>
  <datalabel>MSPJobParams</datalabel>
  <description>Full set of job parameters in XML format as described in
My Seismic Processing v3.14 User manual Appendix F. Available for
download at
http://www.myseismicproc.com/sw/3.14/downloads.php</description>
  <creationtime>2015-10-30 19:58:17Z</creationtime>
  <creator>My Seismic Processing</creator>
  <dataversion>14</dataversion>
  <compression>none</compression>
</segydescblock>
```

In this example the data block consists of uncompressed XML text. The contents are job parameters for the My Seismic Processing system. The format of the parameter structure is version 14 as defined by the manufacturer of the MSP system.

If this is already a known format to the reader, it can easily be automatically decoded.

If not the, the description contains information on how to download the format description. The reader may either implement decoding in software, or present the parameter text to the user and do manual interpretation of them.

2. No description text

```
<segydescblock>
  <version>1</version>
  <blocksize>16340</blocksize>
  <dataformat>text-xml</dataformat>
  <datalabel>MSPJobParams</datalabel>
  <creationtime>2015-10-30 19:58:17Z</creationtime>
  <creator>My Seismic Processing</creator>
  <dataversion>14</dataversion>
  <compression>none</compression>
</segydescblock>
```

In this example there are no description showing how to interpret the job parameter format. The reader will then have to try to find the format description somewhere (e.g. by contacting the manufacturers of the My Seismic Processing system), or display the text to the user who can then try to interpret the parameters manually.

For binary formats the result of lack of format description is basically that unknown formats must be ignored by the SEG-Y reader.

It is therefore highly recommended to include information about format decoding in any user defined block.

Appendix E. Data Word Format

This appendix details the coding of values stored in SEG-Y trace samples. To convert these values to the units give by trace header bytes 203–204, multiply by 2^{-N} where N is given by the Trace weighting factor in trace header bytes 169–170. Trace header bytes 205–212 are further used to convert to alternative physical units. If all of bytes 203–212 ore zero, the units after trace weighting should be assumed to be volts unless external information such as that contained in an "SEG:Data Sample Measurement Unit" Extended Textual Header says otherwise.

Note: All formats described in this appendix are laid out in big-endian, i.e. most significant byte first, format used in all SEG-Y versions prior to 2.0. When nonzero, Field 3297–3300 of the 400-byte Binary File Header is used to determine the actual byte ordering in the SEG-Y dataset.

Code 1 — 4-byte hexadecimal exponent data (i.e. IBM single precision floating point)

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	S	C_6	C_5	C_4	C_3	C_2	C_1	C_0
Byte 2	Q_{-1}	Q_{-2}	Q_{-3}	Q_{-4}	Q_{-5}	Q_{-6}	Q_{-7}	Q_{-8}
Byte 3	Q_{-9}	Q_{-10}	Q_{-11}	Q_{-12}	Q_{-13}	Q_{-14}	Q_{-15}	Q_{-16}
Byte 4	Q_{-17}	Q_{-18}	Q_{-19}	Q_{-20}	Q_{-21}	Q_{-22}	Q_{-23}	Q_{-24}

S = sign bit. — (One = negative number).

C = excess 64 hexadecimal exponent. — This is a binary exponent of 16. The exponent has been biased by 64 such that it represents $16^{(CCCCCCC-64)}$ where CCCCCCC can assume values from 0 to 127.

Q_{1-24} = magnitude fraction. — This is a 24-bit positive binary fraction (i.e., the number system is sign and magnitude). The radix point is to the left of the most significant bit (Q_{-1}) with the MSB being defined as 2^{-1}. The sign and fraction can assume values from ($1 - 2^{-24}$ to $-1 + 2^{-24}$). If this fraction is zero, the sign and exponent must also be zero (i.e., the entire word is zero).

Value = S.QQQQ,QQQQ,QQQQ,QQQQ,QQQQ,QQQQ x $16^{(CCCCCCC-64)}$

Code 2 — 4-byte, two's complement integer

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	I_{31}	I_{30}	I_{29}	I_{28}	I_{27}	I_{26}	I_{25}	I_{24}
Byte 2	I_{23}	I_{22}	I_{21}	I_{20}	I_{19}	I_{18}	I_{17}	I_{16}
Byte 3	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_9	I_8
Byte 4	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $(I_{31}*2^{31} + I_{30}*2^{30} + \ldots + I_1*2^1 + I_0*2^0 + 2^{31}) \bmod 2^{32} - 2^{31}$

Code 3 — 2-byte, two's complement integer

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_9	I_8
Byte 2	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $(I_{15}*2^{15} + I_{14}*2^{14} + \ldots + I_1*2^1 + I_0*2^0 + 2^{15}) \bmod 2^{16} - 2^{15}$

Code 4 — 32-bit fixed point with gain values (*Obsolete*)

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	0	0	0	0	0	0	0	0
Byte 2	G_7	G_6	G_5	G_4	G_3	G_2	G_1	G_0
Byte 3	S	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_9	I_8
Byte 4	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $S\,(I_{14}*2^{14} + I_{13}*2^{13} + \ldots + I_1*2^1 + I_0*2^0) * 2^{-GGGGGGGG}$

Code 5 — 4-byte, IEEE Floating Point

The IEEE format is fully documented in the IEEE standard, "ANSI/IEEE Std 754 - I985", available from the IEEE. The IEEE 32-bit floating point format is summarized as follows:

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	S	C_7	C_6	C_5	C_4	C_3	C_2	C_1
Byte 2	C_0	Q_{-1}	Q_{-2}	Q_{-3}	Q_{-4}	Q_{-5}	Q_{-6}	Q_{-7}
Byte 3	Q_{-8}	Q_{-9}	Q_{-10}	Q_{-11}	Q_{-12}	Q_{-13}	Q_{-14}	Q_{-15}
Byte 4	Q_{-16}	Q_{-17}	Q_{-18}	Q_{-19}	Q_{-20}	Q_{-21}	Q_{-22}	Q_{-23}

The value (v) of a floating-point number represented in this format is determined as follows:

if $e = 255$ & $f \neq 0$. .v = NaN — Not-a-Number (see Note 1)
if $e = 255$ & $f = 0$. .$v = (-1)^s \times \infty$ — Overflow
if $0 < e < 255$. . . . $v = (-1)^s \times 2^{e-127} \times (1.f)$ — Normalized
if $e = 0$ & $f \neq 0$. . . .$v = (-1)^s \times 2^{e-126} \times (0.f)$ — Denormalized
if $e = 0$ & $f = 0$. . . .$v = (-1)^s \times 0$ — ± zero
where e = binary value of all C's (exponent)
f = binary value of all Q's (fraction)

NOTE:
1. A Not-a-Number (NaN) is interpreted as an invalid number. All other numbers are valid and interpreted as described above.

Code 6 — 8-byte, IEEE Floating Point

The IEEE format is fully documented in the IEEE standard, "ANSI/IEEE Std 754 - I985", available from the IEEE. The IEEE format is summarized as follows:

Bit	0	1	2	3	4	5	6	7
Byte 1	S	C_{11}	C_{10}	C_9	C_8	C_7	C_6	C_5
Byte 2	C_4	C_3	C_2	C_1	Q_{-1}	Q_{-2}	Q_{-3}	Q_{-4}
Byte 3	Q_{-5}	Q_{-6}	Q_{-7}	Q_{-8}	Q_{-9}	Q_{-10}	Q_{-11}	Q_{-12}
Byte 4	Q_{-13}	Q_{-14}	Q_{-15}	Q_{-16}	Q_{-17}	Q_{-18}	Q_{-19}	Q_{-20}
Byte 5	Q_{-21}	Q_{-22}	Q_{-23}	Q_{-24}	Q_{-25}	Q_{-26}	Q_{-27}	Q_{-28}
Byte 6	Q_{-29}	Q_{-30}	Q_{-31}	Q_{-32}	Q_{-33}	Q_{-34}	Q_{-35}	Q_{-36}
Byte 7	Q_{-37}	Q_{-38}	Q_{-39}	Q_{-40}	Q_{-41}	Q_{-42}	Q_{-43}	Q_{-44}
Byte 8	Q_{-45}	Q_{-46}	Q_{-47}	Q_{-48}	Q_{-49}	Q_{-50}	Q_{-51}	Q_{-52}

The value (v) of a floating-point number represented in this format is determined as follows:

if $e = 2047$ & $f \neq 0$. . $v = NaN$ Not-a-Number (see Note 1)
if $e = 2047$ & $f = 0$. . $v = (-1)^s \times \infty$ Overflow
if $0 < e < 2047$. . . . . $v = (-1)^s \times 2^{e-1023} \times (1.f)$ Normalized
if $e = 0$ & $f \neq 0$. . . . . $v = (-1)^s \times 2^{e-1022} \times (0.f)$ Denormalized
if $e = 0$ & $f = 0$. . . . . $v = (-1)^s \times 0$ ± zero
where e = binary value of all C's (exponent)
f = binary value of all Q's (fraction)

NOTE:
1. A Not-a-Number (NaN) is interpreted as an invalid number. All other numbers are valid and interpreted as described above.

Code 7 — 3-byte, two's complement integer

Bit	0	1	2	3	4	5	6	7
Byte 1	I_{23}	I_{22}	I_{21}	I_{20}	I_{19}	I_{18}	I_{17}	I_{16}
Byte 2	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_9	I_8
Byte 3	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $(I_{23}*2^{23} + I_{22}*2^{22} + \ldots + I_1*2^1 + I_0*2^0 + 2^{23}) \bmod 2^{24} - 2^{23}$

Code 8 — 1-byte, two's complement integer

Bit	0	1	2	3	4	5	6	7
Byte 1	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $(I_7*2^7 + I_6*2^6 + \ldots + I_1*2^1 + I_0*2^0 + 2^7) \bmod 2^8 - 2^7$

Code 9 — 8-byte, two's complement integer

Bit	0	1	2	3	4	5	6	7
Byte 1	I_{63}	I_{62}	I_{61}	I_{60}	I_{59}	I_{58}	I_{57}	I_{56}
Byte 2	I_{55}	I_{54}	I_{53}	I_{52}	I_{51}	I_{50}	I_{49}	I_{48}
Byte 3	I_{47}	I_{46}	I_{45}	I_{44}	I_{43}	I_{42}	I_{41}	I_{40}
Byte 4	I_{39}	I_{38}	I_{37}	I_{36}	I_{35}	I_{34}	I_{33}	I_{32}
Byte 5	I_{31}	I_{30}	I_{29}	I_{28}	I_{27}	I_{26}	I_{25}	I_{24}
Byte 6	I_{23}	I_{22}	I_{21}	I_{20}	I_{19}	I_{18}	I_{17}	I_{16}
Byte 7	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_{9}	I_{8}
Byte 8	I_{7}	I_{6}	I_{5}	I_{4}	I_{3}	I_{2}	I_{1}	I_{0}

Value = $(I_{63}*2^{63} + I_{62}*2^{62} + \ldots + I_1*2^1 + I_0*2^0 + 2^{63}) \bmod 2^{64} - 2^{63}$

Code 10 — 4-byte, unsigned integer

Bit	0	1	2	3	4	5	6	7
Byte 1	I_{31}	I_{30}	I_{29}	I_{28}	I_{27}	I_{26}	I_{25}	I_{24}
Byte 2	I_{23}	I_{22}	I_{21}	I_{20}	I_{19}	I_{18}	I_{17}	I_{16}
Byte 3	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_{9}	I_{8}
Byte 4	I_{7}	I_{6}	I_{5}	I_{4}	I_{3}	I_{2}	I_{1}	I_{0}

Value = $I_{31}*2^{31} + I_{30}*2^{30} + \ldots + I_1*2^1 + I_0*2^0$

Code 11 — 2-byte, unsigned integer

Bit	0	1	2	3	4	5	6	7
Byte 1	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_{9}	I_{8}
Byte 2	I_{7}	I_{6}	I_{5}	I_{4}	I_{3}	I_{2}	I_{1}	I_{0}

Value = $I_{15}*2^{15} + I_{14}*2^{14} + \ldots + I_1*2^1 + I_0*2^0$

Code 12 — 8-byte, unsigned integer

Bit	0	1	2	3	4	5	6	7
Byte 1	I_{63}	I_{62}	I_{61}	I_{60}	I_{59}	I_{58}	I_{57}	I_{56}
Byte 2	I_{55}	I_{54}	I_{53}	I_{52}	I_{51}	I_{50}	I_{49}	I_{48}
Byte 3	I_{47}	I_{46}	I_{45}	I_{44}	I_{43}	I_{42}	I_{41}	I_{40}
Byte 4	I_{39}	I_{38}	I_{37}	I_{36}	I_{35}	I_{34}	I_{33}	I_{32}
Byte 5	I_{31}	I_{30}	I_{29}	I_{28}	I_{27}	I_{26}	I_{25}	I_{24}
Byte 6	I_{23}	I_{22}	I_{21}	I_{20}	I_{19}	I_{18}	I_{17}	I_{16}

Byte 7	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_9	I_8
Byte 8	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $I_{63}*2^{63} + I_{62}*2^{62} + \ldots + I_1*2^1 + I_0*2^0$

Code 15 — 3-byte, unsigned integer

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	I_{23}	I_{22}	I_{21}	I_{20}	I_{19}	I_{18}	I_{17}	I_{16}
Byte 2	I_{15}	I_{14}	I_{13}	I_{12}	I_{11}	I_{10}	I_9	I_8
Byte 3	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $I_{23}*2^{23} + I_{22}*2^{22} + \ldots + I_1*2^1 + I_0*2^0$

Code 16 — 1-byte, unsigned integer

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Byte 1	I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0

Value = $I_7*2^7 + I_6*2^6 + \ldots + I_1*2^1 + I_0*2^0$

Appendix F. EBCDIC and ASCII Codes

Table 14 IBM 3270 Char Set Ref Ch 10, GA27-2837-9, April 1987

Char	EBCDIC (hex)	ASCII (hex)	Description
NU	x00	x00	Null (nul)
SH	x01	x01	Start of heading (soh)
SX	x02	x02	Start of text (stx)
EX	x03	x03	End of text (etx)
ST	x04	x9C	String terminator (st)
HT	x05	x09	Character tabulation (ht)
SA	x06	x86	Start of selected area (ssa)
DT	x07	7F	Delete (del)
EG	x08	x97	End of guarded area (epa)
RI	x09	x8D	Reverse line feed (ri)
S2	x0A	x8E	Single-shift two (ss2)
VT	x0B	x0B	Line tabulation (vt)
FF	x0C	x0C	Form feed (ff)
CR	x0D	x0D	Carriage return (cr)
SO	x0E	x0E	Shift out (so)
SI	x0F	x0F	Shift in (si)
DL	x10	x10	Datalink escape (dle)
D1	x11	x11	Device control one (dc1)
D2	x12	x12	Device control two (dc2)
D3	x13	x13	Device control three (dc3)
OC	x14	x9D	Operating system command (osc)
NL	x15	x85	Next line (nel)
BS	x16	x08	Backspace (bs)
ES	x17	x87	End of selected area (esa)
CN	x18	x18	Cancel (can)
EM	x19	x19	End of medium (em)
P2	x1A	x92	Private use two (pu2)
S3	x1B	x8F	Single-shift three (ss3)
FS	x1C	x1C	File separator (is4)
GS	x1D	x1D	Group separator (is3)
RS	x1E	x1E	Record separator (is2)
US	x1F	x1F	Unit separator (is1)
PA	x20	x80	Padding character (pad)
HO	x21	x81	High octet preset (hop)
BH	x22	x82	Break permitted here (bph)
NH	x23	x83	No break here (nbh)
IN	x24	x84	Index (ind)
LF	x25	x0A	Line feed (lf)
EB	x26	x17	End of transmission block (etb)
EC	x27	x1B	Escape (esc)
HS	x28	x88	Character tabulation set (hts)
HJ	x29	x89	Character tabulation with justification (htj)
VS	x2A	x8A	Line tabulation set (vts)
PD	x2B	x8B	Partial line forward (pld)
PU	x2C	x8C	Partial line backward (plu)
EQ	x2D	x05	Enquiry (enq)
AK	x2E	x06	Acknowledge (ack)
BL	x2F	x07	Bell (bel)
DC	x30	x90	Device control string (dcs)
P1	x31	x91	Private use one (pu1)
SY	x32	x16	Synchronous idle (syn)
TS	x33	x93	Set transmit state (sts)
CC	x34	x94	Cancel character (cch)
MW	x35	x95	Message waiting (mw)
SG	x36	x96	Start of guarded area (spa)
ET	x37	x04	End of transmission (eot)
SS	x38	x98	Start of string (sos)
GC	x39	x99	Single graphic character introducer (sgci)

Char	EBCDIC (hex)	ASCII (hex)	Description
SC	x3A	x9A	Single character introducer (sci)
CI	x3B	x9B	Control sequence introducer (csi)
D4	x3C	x14	Device control four (dc4)
NK	x3D	x15	Negative acknowledge (nak)
PM	x3E	x9E	Privacy message (pm)
SB	x3F	x1A	Substitute (sub)
SP	x40	x20	Space, Blank
¢	x4A	xA2	Cent sign
.	x4B	x2E	Full stop, Period
<	x4C	x3C	Less-than sign
(	x4D	x28	Left parenthesis
+	x4E	x2B	Plus sign
|	x4F	x7C	Vertical line, Logical OR
&	x50	x26	Ampersand
!	x5A	x21	Exclamation mark
$	x5B	x24	Dollar sign
*	x5C	x2A	Asterisk
)	x5D	x29	Right parenthesis
;	x5E	x3B	Semicolon
¬	x5F	xAC	Not sign
-	x60	x2D	Hyphen, Minus
/	x61	x2F	Solidus, Forward slash
BB	x6A	xA6	Broken bar
,	x6B	x2C	Comma
%	x6C	x25	Percent sign
_	x6D	x5F	Low line, Underline, Underscore
>	x6E	x3E	Greater-than sign
?	x6F	x3F	Question mark
`	x79	x60	Grave accent
:	x7A	x3A	Colon
#	x7B	x23	Number sign, Pound sign, hash mark
@	x7C	x40	Commercial at
'	x7D	x27	Apostrophe
=	x7E	x3D	Equals sign

Char	EBCDIC (hex)	ASCII (hex)	Description
"	x7F	x22	Quotation mark
a	x81	x61	Latin small letter A
b	x82	x62	Latin small letter B
c	x83	x63	Latin small letter C
d	x84	x64	Latin small letter D
e	x85	x65	Latin small letter E
f	x86	x66	Latin small letter F
g	x87	x67	Latin small letter G
h	x88	x68	Latin small letter H
i	x89	x69	Latin small letter I
j	x91	x6A	Latin small letter J
k	x92	x6B	Latin small letter K
l	x93	x6C	Latin small letter L
m	x94	x6D	Latin small letter M
n	x95	x6E	Latin small letter N
o	x96	x6F	Latin small letter O
p	x97	x70	Latin small letter P
q	x98	x71	Latin small letter Q
r	x99	x72	Latin small letter R
~	xA1	x7E	Tilde
s	xA2	x73	Latin small letter S
t	xA3	x74	Latin small letter T
u	xA4	x75	Latin small letter U
v	xA5	x76	Latin small letter V
w	xA6	x77	Latin small letter W
x	xA7	x78	Latin small letter X
y	xA8	x79	Latin small letter Y
z	xA9	x7A	Latin small letter Z
{	xC0	x7B	Left curly bracket
A	xC1	x41	Latin capital letter A
B	xC2	x42	Latin capital letter B
C	xC3	x43	Latin capital letter C
D	xC4	x44	Latin capital letter D
E	xC5	x45	Latin capital letter E
F	xC6	x46	Latin capital letter F
G	xC7	x47	Latin capital letter G
H	xC8	x48	Latin capital letter H

Char	EBCDIC (hex)	ASCII (hex)	Description
I	xC9	x49	Latin capital letter I
}	xD0	x7D	Right curly bracket
J	xD1	x4A	Latin capital letter J
K	xD2	x4B	Latin capital letter K
L	xD3	x4C	Latin capital letter L
M	xD4	x4D	Latin capital letter M
N	xD5	x4E	Latin capital letter N
O	xD6	x4F	Latin capital letter O
P	xD7	x50	Latin capital letter P
Q	xD8	x51	Latin capital letter Q
R	xD9	x52	Latin capital letter R
\	xE0	x5C	Reverse solidus, Back slash
S	xE2	x53	Latin capital letter S
T	xE3	x54	Latin capital letter T
U	xE4	x55	Latin capital letter U
V	xE5	x56	Latin capital letter V
W	xE6	x57	Latin capital letter W
X	xE7	x58	Latin capital letter X
Y	xE8	x59	Latin capital letter Y
Z	xE9	x5A	Latin capital letter Z
0	xF0	x30	Digit zero
1	xF1	x31	Digit one
2	xF2	x32	Digit two
3	xF3	x33	Digit three
4	xF4	x34	Digit four
5	xF5	x35	Digit five
6	xF6	x36	Digit six
7	xF7	x37	Digit seven
8	xF8	x38	Digit eight
9	xF9	x39	Digit nine
AC	xFF	x9F	Application program command (apc)

Char	EBCDIC (hex)	ASCII (hex)	Description
NUL	x00	x00	Nul
SOH	x01	x01	Start of heading (soh)
STX	x02	x02	Start of text (stx)
ETX	x03	x03	End of text (etx)
EOT	x37	x04	End of transmission (eot)
ENQ	x2D	x05	Enquiry (enq)
ACK	x2E	x06	Acknowledge (ack)
alert	x2F	x07	Bell (bel)
BEL	x2F	x07	Bell (bel)
backspace	x16	x08	Backspace (bs)
tab	x05	x09	Character tabulation (ht)
newline	x25	x0A	Line feed (lf)
vertical-tab	x0B	x0B	Line tabulation (vt)
form-feed	x0C	x0C	Form feed (ff)
carriage-return	x0D	x0D	Carriage return (cr)
DLE	x10	x10	Datalink escape (dle)
DC1	x11	x11	Device control one (dc1)
DC2	x12	x12	Device control two (dc2)
DC3	x13	x13	Device control three (dc3)
DC4	x3C	x14	Device control four (dc4)
NAK	x3D	x15	Negative acknowledge (nak)
SYN	x32	x16	Synchronous idle (syn)
ETB	x26	x17	End of transmission block (etb)
CAN	x18	x18	Cancel (can)
SUB	x3F	x1A	Substitute (sub)
ESC	x27	x1B	Escape (esc)
IS4	x1C	x1C	File separator (is4)
IS3	x1D	x1D	Group separator (is3)
intro	x1D	x1D	Group separator (is3)
IS2	x1E	x1E	Record separator (is2)
IS1	x1F	x1F	Unit separator (is1)
DEL	x07	x7F	Delete (del)
space	x40	x20	Space
!	x5A	x21	Exclamation mark

Char	EBCDIC (hex)	ASCII (hex)	Description
"	x7F	x22	Quotation mark
#	x7B	x23	Number sign
$	x5B	x24	Dollar sign
%	x6C	x25	Percent sign
&	x50	x26	Ampersand
'	x7D	x27	Apostrophe
(	x4D	x28	Left parenthesis
)	x5D	x29	Right parenthesis
*	x5C	x2A	Asterisk
+	x4E	x2B	Plus sign
,	x6B	x2C	Comma
-	x60	x2D	Hyphen, Minus
.	x4B	x2E	Full stop, Period
/	x61	x2F	Solidus, Slash
0	xF0	x30	Digit Zero
1	xF1	x31	Digit one
2	xF2	x32	Digit two
3	xF3	x33	Digit three
4	xF4	x34	Digit four
5	xF5	x35	Digit five
6	xF6	x36	Digit six
7	xF7	x37	Digit seven
8	xF8	x38	Digit eight
9	xF9	x39	Digit nine
:	x7A	x3A	Colon
;	x5E	x3B	Semicolon
<	x4C	x3C	Less-than sign
=	x7E	x3D	Equals sign
>	x6E	x3E	Greater-than sign
?	x6F	x3F	Question mark
@	x7C	x40	Commercial at
[	X00	x5B	Left square bracket
\	xE0	x5C	Reverse solidus, Backslash
]	x00	x5D	Right square bracket
^	x00	x5E	Circumflex, Caret
_	x6D	x5F	Low line, Underscore
`	x79	x60	Grave accent
{	xC0	x07B	Left curly bracket
|	x4F	x7C	Vertical line
}	xD0	x7D	Right curly bracket
~	xA1	x7E	Tilde

Appendix G. References

Barry, K. M., Cavers, D. A. and Kneale, C. W., 1975, Report on recommended standards for digital tape formats: Geophysics, 40, no. 02, 344-352[22].

Booth, Algan, Duke, Guyton, Norris, Stainsby, Theriot, Wildgoose and Wilhelmsen, 1997, SEG Rode Format Record Oriented Data Encapsulation: Geophysics, 61, no. 05, 1545-1558.

Geomatics Committee, 2015, OGP P1/11 Geophysical position data exchange format, International Association of Oil & Gas Producers, Report 483-1, https://www.iogp.org.

Geomatics Committee, 2017, OGP P6/11 Seismic bin grid data exchange format, International Association of Oil & Gas Producers, Report 483-6, https://www.iogp.org.

Norris, M. W. and Faichney, A.K. (Eds.), 2002, SEG-Y rev1 Data Exchange format: SEG, Tulsa, OK.

Northwood, E. J., Wisinger, R. C., and Bradley, J. J., 1967, Recommended standards for digital tape formats: Geophysics, v32, p. 1073-1084.

SEG Comm. Field Tape Std., 1994, Digital field tape standards - SEG-D, revision 1 (special report): Geophysics, 59, no. 04, 668-684.

[22] SEG-Y rev 0 format

SEG Technical Standards Committee, 2011, SEG-D Rev. 3.0 update, SEG, Tulsa, OK, www.seg.org.

Wood, Behn, Borresen, Guyton, Miles, O'Neill, Oishi and Scales, 1998, SEG-D, Rev 2 Field Tape Standards, Geophysics, 62, no. 03, 1004-1031 .