**THE INSTITUTE OF NAVIGATION**

8551 Rixlew Lane, Ste. 360

Manassas, VA 20109-3701

PHN: (703) 366-2723/FAX: (703) 366-2724

[WWW.ION.ORG](http://WWW.ION.ORG)

**Global Navigation Satellite Systems**

**Software Defined Radio**

**Sampled Data**

**Metadata Standard**

Revision 0.1 (draft)

ION GNSS SDR Standard Working Group

**Abstract**

Abstract text here.

**Table of Contents**

[1 Introduction 3](#_Toc408782613)

[2 Scope 3](#_Toc408782614)

[3 Metadata Format 3](#_Toc408782615)

[4 SDR Data Collection Topologies 3](#_Toc408782616)

[4.1 Single Band, Single Stream, Single File 4](#_Toc408782617)

[4.2 Multi-Band, Single Stream, Single File 4](#_Toc408782618)

[4.3 Multi Stream, Single File 5](#_Toc408782619)

[4.4 Multi Stream, Single File (with Additional Data) 5](#_Toc408782620)

[4.5 Temporal Splitting of Files 5](#_Toc408782621)

[4.6 Spatial Splitting of Files 5](#_Toc408782622)

[4.7 Spatial-Temporal Splitting of Files 6](#_Toc408782623)

[5 Metadata File Naming and Association Mechanisms 6](#_Toc408782624)

[6 Metadata Parameters 6](#_Toc408782625)

[6.1 System Parameters 7](#_Toc408782626)

[6.2 Cluster Parameters 9](#_Toc408782627)

[6.3 Source Parameters 10](#_Toc408782628)

[6.4 Band Parameters 12](#_Toc408782629)

[6.5 Stream Parameters 13](#_Toc408782630)

[6.6 Lump Parameters 16](#_Toc408782631)

[6.7 Chunk Parameters 18](#_Toc408782632)

[6.8 Block Parameters 20](#_Toc408782633)

[6.9 Lane Parameters 21](#_Toc408782634)

[6.10 File Parameters 22](#_Toc408782635)

[6.11 Selection Parameters 22](#_Toc408782636)

[6.12 Metadata Order 23](#_Toc408782637)

[7 Working Group Membership 25](#_Toc408782638)

# Introduction

The past several years has seen a proliferation of software defined radio (SDR) data collection systems and processing platforms that are particularly designed for Global Navigation Satellite System (GNSS) receiver applications or those that support GNSS bands. For post-processing, correctly interpreting the GNSS SDR sampled datasets produced or consumed by these systems has historically been a cumbersome and error-prone process. This is because these systems necessarily produce datasets of various formats, the subtleties of which are often lost in translation when communicating between the producer and consumer of these datasets. This specification standardizes the metadata associated with GNSS SDR sampled data files.

# Scope

Datasets containing GNSS SDR samples may also contain other information such as data from other sensors and data from radio frequency (RF) bands other than GNSS bands. For non-RF data, this specification includes information needed to bypass this data during reading. For non-GNSS RF bands, only parameters common to GNSS bands are supported by this standard.

# Metadata Format

Extensible Markup Language (XML) is used in this standard. The XML schema are specified according to the XML Schema Definition (XSD) standard.

# SDR Data Collection Topologies

This standard is designed to support most (if not all) current and future GNSS SDR sampled data file formats. These formats stem from the fundamental data collection topologies illustrated in Figure 1. This section describes these topologies.

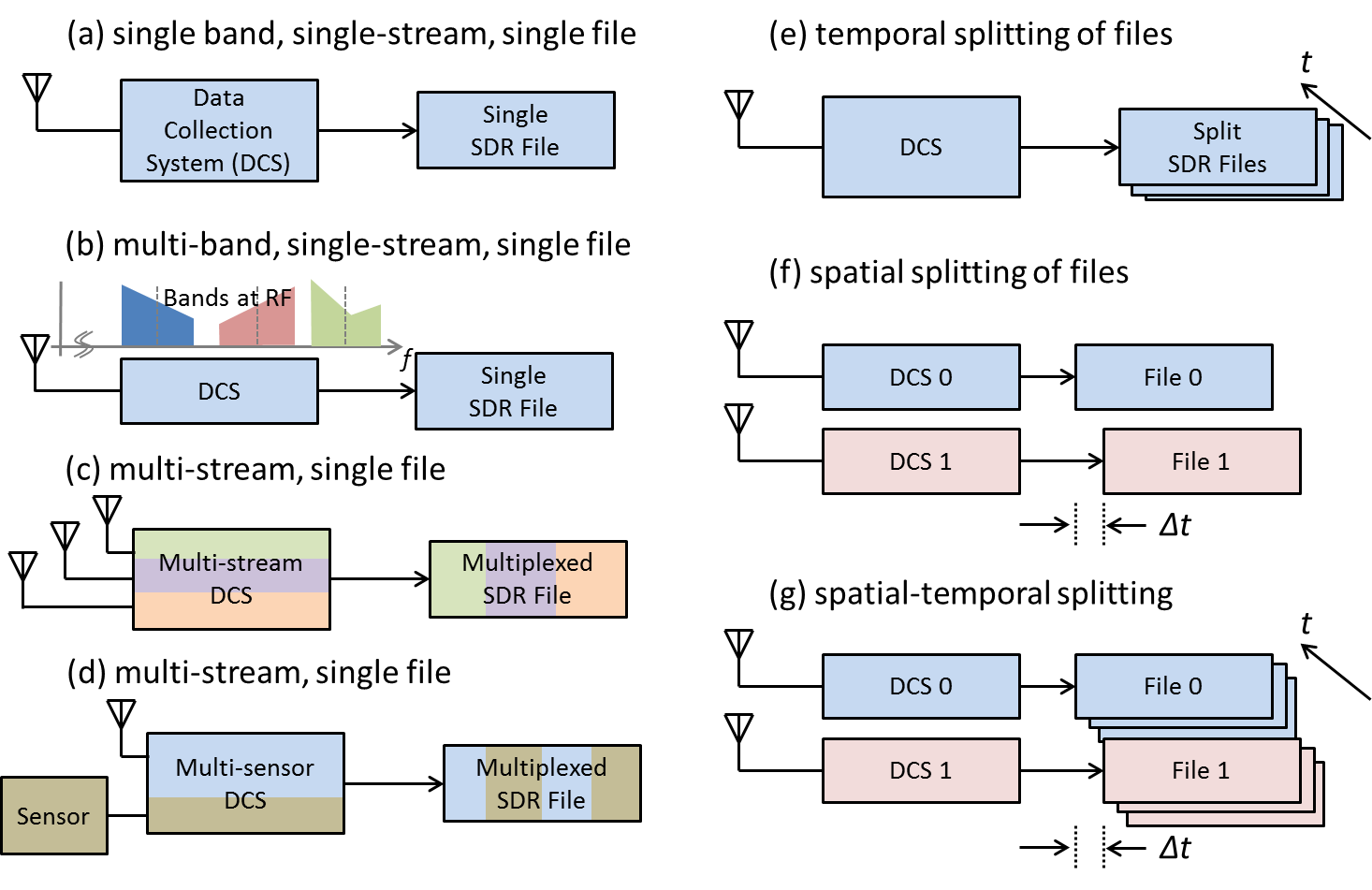


Figure 1 – Fundamental GNSS SDR Data Collection Topologies

## Single Band, Single Stream, Single File

Figure 1.a illustrates the simplest data collection topology that can exist. This is when a single contiguous region of RF spectrum (referenced henceforth as a ‘band’) is down-converted and sampled to produce a single data stream that is then written to a single data file.

For this and all subsequent topologies, the data stream may contain samples that are either real or complex valued depending on whether intermediate frequency (IF) or baseband sampling is used, respectively. These samples are packed according to a repetitive pattern. The repetitive pattern may also comprise of other information at the beginning and/or end of a block of samples This may include non-sample data such as headers and footers which may be used for data integrity check purposes. In this topology, this formatted data stream is written to one and only one file.

## Multi-Band, Single Stream, Single File

Figure 1.b is identical to Figure 1.a in terms of how the data stream may be formed and written to disk, except the data stream contains information from more than one RF band. An example of this topology is a direct RF sampling front-end architecture that intentionally aliases multiple bands to fall next to each other at baseband. In this case, some bands may be spectrally inverted as a result of the digital down-conversion process.

## Multi Stream, Single File

Figure 1.c illustrates a topology where multiple *sample streams* are combined into a single formatted *data stream* and written to a single file. The formatted data stream may contain additional information as described in 4.1. Each sample stream represents a distinct time series that is independent from any and all others (independent in a mathematical time series sense, not in a statistical sense).

NOTE:

The distinction of *sample stream* (i.e. mathematical time series) versus *data stream* (i.e. formatted data bytes that are ultimately written to disk) is made above. In this standard, the term *stream* shall always imply the former. The term *data stream* shall be used to refer to the latter.

In the example shown, each sample stream represents data collected from a different antenna whose signal passes through a different RF front-end channel. This is for illustration purposes only. The standard does not assume any dependence between streams (including common sample rates or quantization).

## Multi Stream, Single File (with Additional Data)

Figure 1.d illustrates a data stream containing GNSS samples as well as data from an additional sensor. For the purpose of this standard, any data that cannot be represented as GNSS sample streams are considered unknown data. The standard defines parameters necessary to skip over unknown data bytes when decoding the data stream.

The remaining topologies address how a data stream may be written to disk.

## Temporal Splitting of Files

The data rates of GNSS SDR streams are typically high (on the order of one to several hundred MB/sec). Hence, long-duration data collections can generate very large files that become cumbersome to manage. For this reason, the data may be written to smaller sets of files where the data stream continues from the end of one file to the beginning of another (possibly with some overlap to ensure data integrity). This is defined as *temporal file splitting* in this standard. The standard includes parameters that specify the order of temporally split files.

NOTE:

A metadata file typically exists for each data file. Optionally, all information for a multi-file set may be contained within one metadata file. For the former case, the first metadata file of a set must contain the complete set of metadata parameters and subsequent files may contain only those that change from file to file.

## Spatial Splitting of Files

A collection system or setup may write individual data streams to multiple files. These files may be written within the same host system (such as a personal computer (PC)) or multiple systems. This is defined as *spatial file splitting* in this standard.

NOTE:

This standard associates two or more spatially split files in a metadata file defined as the *Set Selection File*. Optionally, these set selection parameters may appear in the single metadata file architecture described in 4.5. In the event where duplicate parameters exist in a metadata file and the set selection file, the latter shall override.

## Spatial-Temporal Splitting of Files

Figure 1.g illustrates the combination of spatial and temporal splitting. In this case, the set selection parameters refer to the first of each temporally split file set.

# Metadata File Naming and Association Mechanisms

The official filename extension for a metadata file is ‘.SDRX’. Use of this extension is highly recommended.

For the case where a metadata file is paired with each SDR file, it is recommended that the metadata file

An SDR processing run is typically invoked with the SDR data file name as an input parameter. For Optionally,

# Metadata Parameters

As illustrated in Figure 2, metadata are defined in terms of 11 sets of parameters. These sets are defined hierarchically such that referenced parameters occur first.

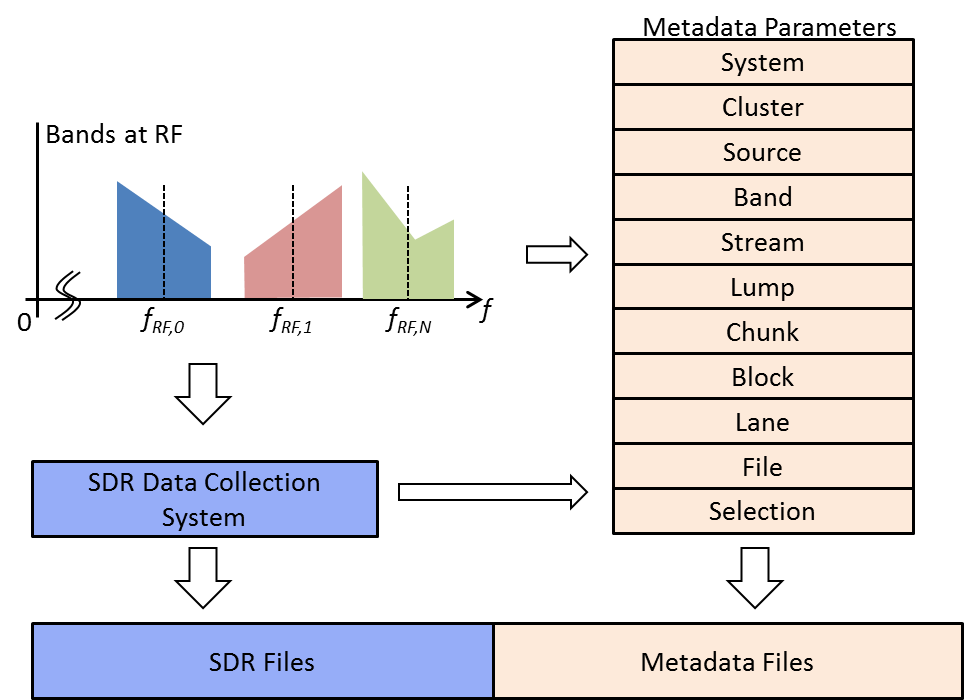


Figure 2 – Overview of Metadata Parameters, Generation and Association

## System Parameters

A System is defined as a complete data collection apparatus. The system comprises all antennas, sensors, and other information outputting equipment down to the disk arrays that store SDR files. The system may also include GNSS signal simulators. The standard includes geometrical parameters (location and attitude) to the extent that this information is necessary for post-processing SDR data stream. For example, initial position and platform orientation may be needed for a dynamic scenario. The relative position and orientation of antennas and their elements with respect to the platform coordinate frame are needed for adaptive antenna signal processing.

Table 1 – Definition of System Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique system identifier | String |  | Yes |  |
| TIMEZONE | Offset from UTC of time zone used to specify all absolute time parameters. | UINT8:UINT8 | Hours:Minutes | Yes | 0:00 |
| DATUM | Geode reference datum used for all absolute positions | String | WGS-84  (other datums TBD) | Yes | WGS-84 |
| DISTANCES1 | Units used for distance | String | m, cm, mm | Yes | m |
| ANGLES1 | Units used for angles | String | rad, deg, dms | Yes | deg |
| DELAYS | Units used for delays | String | ns, ps | Yes | ns |
| FREQUENCIES:  :FORMAT | Numerical format used to represent frequency values | String | INTFRAC, double | Yes | INTFRAC |
| FREQUENCIES:  :UNIT | Units for frequency values | String | Hz, kHz, MHz | Yes | Hz |
| TOA | Time of applicability for all position and attitude parameters | String | XSD Date and Time format2 | Yes3 |  |
| POSITION | Platform position at TOA expressed in Geoid frame |  |  | Yes3 |  |
| POSITION:FORMAT | Format of platform position. Attribute values that follow are named according to format | String | LLH, XYZ |  |  |
| POSITION: L,L,H OR X,Y,Z | Position values | double  INT8:UINT8:double |  |  |  |
| ATTITUDE | Attitude of the platform at TOA with respect to the local-level frame |  |  | Yes3 |  |
| ATTITUDE: FORMAT | Convention used for local-level frame | string | ENU, NED |  |  |
| ATTITUDE: TYPE | Type of rotation matrix used. | string | DCM |  |  |
| ATTITUDE: R11,…,R33 | Rotation matrix values | double | 3 x 3 matrix |  |  |
| FSBASE | Base frequency.  See UNITS\_FREQUENCY for format |  |  | Required |  |
| POC4 | Point of contact. Name of person or entity. | string |  | Yes |  |
| CONTACT4 | POC contact information (email) | string |  | Yes |  |
| EQUIPMENT4 | Equipment used for this data collection | string |  | Yes |  |
| CAMPAIGN4 | Data collection campaign | string |  | Yes |  |
| SCENARIO4 | Specific scenario for this collection | string |  | Yes |  |
| URL4 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT4 | Additional information | string |  | Yes |  |

NOTES:

1 Distance and angle units apply to all *relative* position and attitude parameters only

2 <http://www.w3schools.com/schema/schema_dtypes_date.asp>

3 TOA, Position and Attitude may be back-annotated into metadata file following post processing.

4 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. POC1, POC2, etc.).

XML Listing 1 – System Parameters

<!-- System Parameters -->

<SYSTEM ID=“SYS0” >

<TIMEZONE>+0.00</TIMEZONE>

<DATUM>WGS-84</DATUM>

<DISTANCES>M</DISTANCES>

<ANGLES>DEG</ANGLES>

<DELAYS>ns</DELAYS>

<FREQUENCIES> FORMAT=“INTFRAC” UNIT=“Hz”<FREQUENCIES>

<TOA>2012-12-13T12:12:12</TOA>

<POSITION> FORMAT=“LLH” L=“39.9” L=“82.0” H=“200.0”</POSITION>

<ATTITUDE> FORMAT=“ENU” TYPE=“DCM” R11=“0.0” R12=“0.0” R13=“0.0” R21=“0.0” R22=“0.0” R23=“0.0” R31=“0.0” R32=“0.0” R33=“0.0” </ATTITUDE>

<FSBASE>56320000.1</FSBASE>

<POC>Entity or person that collected data</POC>

<CONTACT>Contact information (email)</CONTACT>

<EQUIPMENT>Equipment used: Make/Mode/Serial</EQUIPMENT>

<CAMPAIGN>Campaign Text</CAMPAIGN>

<SCENARIO>Scenario Text</SCENARIO>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</SYSTEM>

## Cluster Parameters

Data collection setups may contain one or more antenna units where each antenna unit may comprise one or more elements. The position and orientation of each element’s phase center and the relative delay for each element must be known in order to perform multi-element signal processing. The antenna element geometry with respect to its reference frame may be supplied by the manufacturer. Hence, it is convenient to include these parameters directly as metadata. The standard defines the generic terms ‘cluster’ and ‘source’ to refer to an antenna unit and its elements respectively.

A cluster is defined as a grouping of sources. A coordinate frame is associated with a cluster. The origin and rotation of this frame is specified with respect to the platform coordinate frame.

Table 2 – Definition of Cluster Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Yes |  |
| ORIGIN | Origin with respect to platform |  |  | Yes |  |
| ORIGIN:FORMAT | Format used to specify origin | string | XYZ |  |  |
| ORIGIN:X,Y,Z | Origin coordinates. Units are: SYSTEM:  UNITS\_DISTANCE | double |  |  |  |
| ROTATION | Rotation of cluster frame w.r.t. platform |  |  | Yes3 |  |
| ROTATION: FORMAT | Format used to specify rotation | string | DCM |  |  |
| ROTATION: R11,…,R33 | Rotation matrix values | double | 3 x 3 matrix |  |  |
| VENDOR1 | Antenna vendor name | string |  | Yes |  |
| MODEL1 | Antenna model number | string |  | Yes |  |
| SERIAL1 | Antenna serial number | string |  | Yes |  |
| URL1 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT1 | Additional information | string |  | Yes |  |

NOTES:

1 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 2 – Cluster Parameters

<!-- Cluster Parameters -->

<CLUSTER ID=“ANT0” >

<ORIGIN> FORMAT=“XYZ” X=“0.1” Y=“0.2” Z=“0.3”</ORIGIN>

<ROTATION> FORMAT=“DCM” R11=“0.0” R12=“0.0” R13=“0.0” R21=“0.0” R22=“0.0” R23=“0.0” R31=“0.0” R32=“0.0” R33=“0.0” </ROTATION>

<VENDOR>Antenna vendor name</VENDOR>

<MODEL>Antenna model number</MODEL>

<SERIAL>Antenna serial number (text)</SERIAL>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</CLUSTER>

## Source Parameters

A source is defined as the originator of an electrical signal. A coordinate frame is associated with a cluster. The origin and rotation of this frame is specified with respect to the platform coordinate frame.

Table 3 – Definition of Source Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Yes |  |
| CLUSTER | Cluster identifier that this source belongs to | string |  | Yes |  |
| TYPE | Electrical type of this source | string | PATCH,  DIPOLE,  HELICAL, QUADRIFILAR,  SIMULATOR | Yes |  |
| POLARIZATION | Element polarization | string | RHCP, LHCP, LINEAR, HORI, VERT | Yes |  |
| ORIGIN | Origin with respect to platform |  |  | Yes |  |
| ORIGIN:FORMAT | Format used to specify origin | string | XYZ |  |  |
| ORIGIN:X,Y,Z | Origin coordinates. Units are: SYSTEM:  UNITS\_DISTANCE | double |  |  |  |
| ROTATION | Rotation of cluster frame w.r.t. platform |  |  | Yes3 |  |
| ROTATION: FORMAT | Format used to specify rotation | string | DCM |  |  |
| ROTATION: R11,…,R33 | Rotation matrix values | double | 3 x 3 matrix |  |  |
| URL1 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT1 | Additional information | string |  | Yes |  |

NOTES:

1 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 3 - Source Parameters

<!-- Source Parameters -->

<SOURCE> ID=“J1” CLUSTER=“ANT0” TYPE=“PATCH” POLARIZATION=“RHCP” DELAY=“0.0”

<ORIGIN> FORMAT=“XYZ” X=“1.0” Y=“0.0” Z=“0.1”</ORIGIN>

<ROTATION> FORMAT=“DCM” R11=“0.0” R12=“0.0” R13=“0.0” R21=“0.0” R22=“0.0” R23=“0.0” R31=“0.0” R32=“0.0” R33=“0.0” </ROTATION>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</SOURCE>

## Band Parameters

A Band is defined as a finite span of RF spectrum. Each Band is received from a single Source and converted to a sample stream by a signal processor that is typically referred to as an RF front-end. This analog signal represented by the Band experiences the following changes as it passes through this mixed-signal processing chain:

* The RF center frequency, FRF, is translated to FIF
* The spectrum may become inverted such that the frequency FRF+dF is translated to FIF-dF
* The sampled representation of the band is delayed with respect to the signal incident at the phase center of the source (i.e. antenna element). This delay may vary with time, and is hence defined at the System time of applicability, TOA.
* An approximate double-sided half power bandwidth can be specified for the Stream representation of the Band.

The above are specified in terms of Band Parameters.

Table 4 – Definition of Band Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Required |  |
| SOURCE | Source identifier of received signal | string |  | Required |  |
| FRF | Center frequency of band incident at source | SYSTEM:  FREQUENCIES |  | Required |  |
| FIF | Translated center frequency of band | SYSTEM:  FREQUENCIES |  | Required |  |
| INV | Binary flag indicating spectral inversion | Boolean | 0, 1 | Yes | 0 |
| DELAY | Delay of band measured from source to sampled stream, specified at TOA. | double |  | Yes | 0 |
| BW31 | Approximate double-sided half power bandwidth |  |  | Yes |  |
| URL2 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT2 | Additional information | string |  | Yes |  |

NOTES:

1 This bandwidth is measured by processing the sample stream. For streams containing multiple bands, it is recommended that other bands be muted to measure a given BW3 of a Band.

2 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 4- Band Parameters

<!-- Band Parameters -->

<BAND ID=“L1\_0” SOURCE=“J1” FRF=“1575420000.1” FIF=“13680000.1” INV=“0” DELAY=“0.0” BW3=“20000000.1”>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</BAND>

## Stream Parameters

A frequency-translated signal may contain more than one band. For example, in a direct RF sampling front-end, the sample rate is chosen such that multiple passbands are intentionally aliased to fall adjacent to each other in the sampled signal spectrum. This is illustrated in Figure 3.

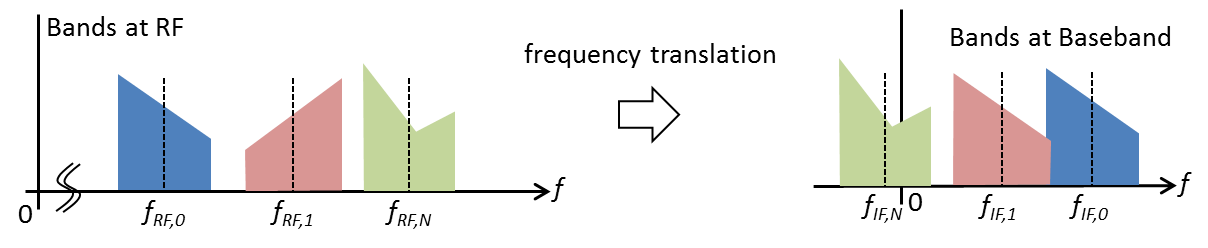


Figure 3 - Intentional Aliasing of a Multiband signal to Baseband

Figure 4 illustrates the conceptual representation of the digitization of a signal containing multiple bands. The output of this process is a sampled representation of the multi-band signal referred to as a Sample Stream.

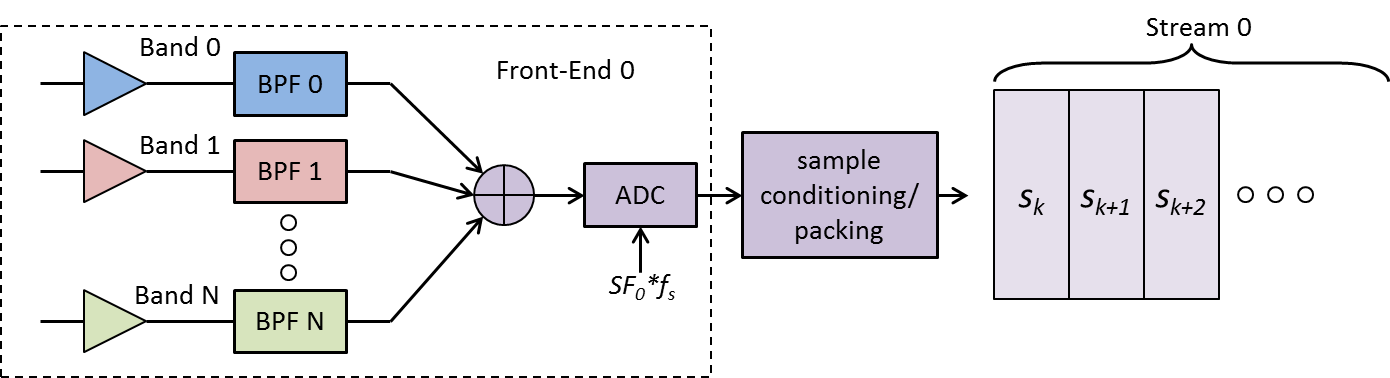


Figure 4 - Illustration of Multiple Bands Present in a Stream

A (Sample) Stream is defined as a discrete-time discrete-amplitude series that is the sampled representation of a combination of one or more bands.

A Stream has the following properties:

* The Stream contains the sampled representation of one or more bands.
* A Stream is sampled at a given sample rate. This sample rate may be different to other streams in the system. The sample rate of a stream is specified as an integer multiple of the System base sample rate.
* The sample values may be real or complex values depending on whether IF sampling or baseband sampling is used, respectively. Some or all of the numerical values expressed in the stream may be inverted.
* Each sample value is represented by one or more bits, known as quantization. These values may be encoded using various established schemes.
* A sample of QTZ quantized bits may be expressed by a value of BTP packed bits in the Stream where BTP ≥ QTZ.
* When BTP>QTZ, the alignment of the quantized sample with respect to the packed sample must be known in order to interpret the sample value correctly. This interpretation also depends on the encoding used.

The above are specified in terms of Stream Parameters.

Table 5 – Definition of Band Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Required |  |
| BAND1 | Identifiers of all Bands present in this stream | Comma-separated string |  | Required |  |
| SF | Sample rate factor | UINT16 |  | Required |  |
| QTZ | Sample quantization (bits) | UINT8 |  | Required |  |
| BTP | Packed representation (bits) | UINT8 |  | Required |  |
| ALN | Sample alignment | string | ‘L’ – left aligned  ‘R’ – right aligned  ‘N’ – N/A | Required |  |
| FORMAT | Sample representation | string | IF, IF’, IQ, IQ’, I’Q, I’Q’, QI, QI’, Q’I, Q’I’  (where ‘ signifies inversion) | Required |  |
| ENC | Encoding | string | SIGN – sign bit  SM – sign-magnitude  INT – twos complement  BIN – offset binary  FP – floating point | Required |  |
| URL2 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT2 | Additional information | string |  | Yes |  |

NOTES:

1 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 5- Stream Parameters

<!-- Stream Parameters -->

<STREAM ID=“S0” BAND=“L1\_0, L2\_0” SF=“1” QTZ=“1” BTP=“1” ALN=“N” FORMAT=“IQ” ENC=“SIGN”>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</STREAM>

Table 6 - Sample Encoding Schemes

| QTZ | | Encoding | Set | Range Min | Range Max |
| --- | --- | --- | --- | --- | --- |
| 1 | sign | | {-1, +1} | -1 | +1 |
| 2 | sign-magnitude | | {-3, -1, +1, +3} | -3 | +3 |
| signed integer | | {-2, -1, 0, 1} | -2 | +1 |
| offset binary | | {-2, -1, 0, 1} | -2 | +1 |
| 4 | sign-magnitude | | {-8, -7,…,-1,+1,…, +8} | -8 | +8 |
| signed integer | | {-8, …, 0…, +7} | -8 | +7 |
| offset binary | | {-8, …, 0…, +7} | -8 | +7 |
| 8 | sign-magnitude | | {-128, -127,…, +127, +128} | -127 | +128 |
| signed integer | | {-128, …, 0…, 127} | -128 | +127 |
| offset binary | | {-128, …, 0…, 127} | -128 | +127 |
| 16 | sign-magnitude | | {-215,… ,-1,+1,…, +215} | -215 | +215 |
| signed integer | | {-215, …, 0…, 215-1} | -215 | +215-1 |
| offset binary | | {-215, …, 0…, 215-1} | -215 | +215-1 |
| floating point | | IEEE 754-2008, FP16 | | |
| 32 | sign-magnitude | | {-231,… ,-1,+1,…, +231} | -231 | +231 |
| signed integer | | {-231, …, 0…, 231-1} | -231 | +231-1 |
| offset binary | | {-231, …, 0…, 231-1} | -231 | +231-1 |
| floating point | | IEEE 754-2008, FP32 | | |
| 64 | sign-magnitude | | {-263,… ,-1,+1,…, +263} | -263 | +263 |
| signed integer | | {-263, …, 0…, 263-1} | -263 | +263-1 |
| offset binary | | {-263, …, 0…, 263-1} | -263 | +263-1 |
| floating point | | IEEE 754-2008, FP64 | | |

## Lump Parameters

Samples from two or more Sample Streams may be time multiplexed to form a single Data Stream that is ultimately written to disk (after additional formatting is applied, as described later in this document). This standard assumes that all samples belonging to a finite interval of time are packed into a contiguous grouping of bits, known as a Lump.

A Lump is defined as the ordered containment of all samples occurring within an interval ts=1/fs.

Figure 5 illustrates a *Lump* containing all samples from *N* *Sample Streams*.

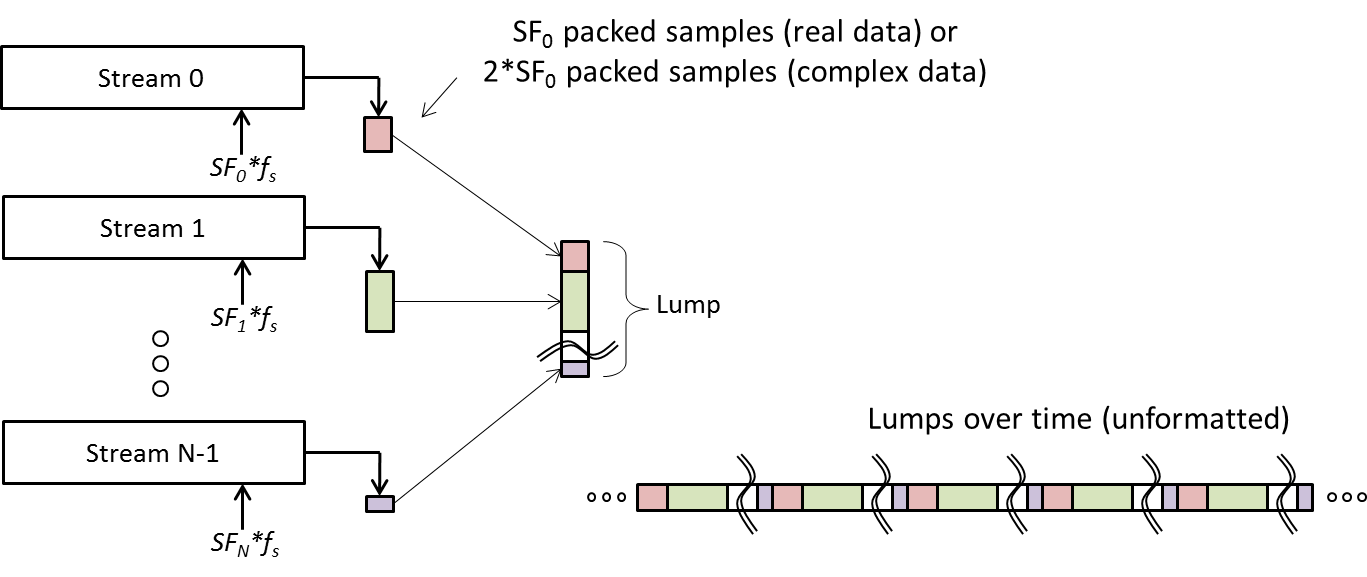


Figure 5 - Illustration of a lump Containing Samples from N Streams

Table 7 – Definition of Lump Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Required |  |
| STREAM1 | Identifiers of all Streams present in this lump | Comma-separated string |  | Required |  |
| URL2 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT2 | Additional information | string |  | Yes |  |

NOTES:

1 The Stream identifier order must match the order of Streams multiplexed in Lump

2 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 6- Lump Parameters

<LUMP ID=“LUMP\_0” STREAM=“S0, S1, S2, S3”>

<URL>link containing additional info (browser format) </URL>

<COMMENT>this is a comment</COMMENT>

</LUMP>

## Chunk Parameters

The packing scheme of samples in a data stream must be known to correctly decode them. For example, consider 32 1-bit real samples packed into two UINT16 words represented in little-endian format. Due to the little-endian representation, these samples will be decoded incorrectly if read back as a single UINT32 word and shifted out. Further, some systems pack samples from left to right within a word whereas others perform the opposite.

This standard defines a metadata parameter known as a *Chunk* that together with *Stream* and *Lump* parameters completely and unambiguously describes how samples shall be decoded from a data stream.

A *Chunk* is defined as a segment of data consisting of one or more lumps that have been packed using one of four standard unsigned integer data types.

Table 8 – Definition of Chunk Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Required |  |
| LUMP1 | Identifiers of all Lumps present in this chunk | Comma-separated string |  | Required |  |
| TYPE | Word data type for reading and decoding samples | string | UINT8, UINT16, UINT32, UINT64 | Required |  |
| WORDS | Total number of words to be read in order to read/decode this chunk | UINT8 |  | Required |  |
| ENDIAN | Endianness of words stored in chunk | string | ‘L’ – Little  ‘B’ – Big  ‘N’ – not applicable | yes | ‘N’ |
| PAD | Padding applied during encoding | string | ‘H’ – head padding  ‘T’ – Tail padding  ‘N’ – No padding | yes | ‘N’ |
| SHIFT | Word shift direction | string | ‘L’ – Left shift  ‘R’ – Right shift | Required |  |
| URL2 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT2 | Additional information | string |  | Yes |  |

NOTES:

1 The number of lumps stored in a chunk is derived from the number of comma separated identifiers present. If multiple lumps of the same type are present, repeat the same lump ID that many times

2 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 7- Chunk Parameters

<!-- Chunk Parameters -->

<CHUNK ID=“1” LUMP=“LUMP0” TYPE=“UINT32” WORDS=“1” ENDIAN=“L” PAD=“N” SHIFT=“R”>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</CHUNK>

Figure 6 illustrates four different schemes where a single 7-bit Lump may be encoded within a Chunk. The number of bits of information contained within a lump (and hence the number of bits to discard while decoding a chunk – shown as whitespace) is determined implicitly by parsing the referenced Lump and Stream parameters.

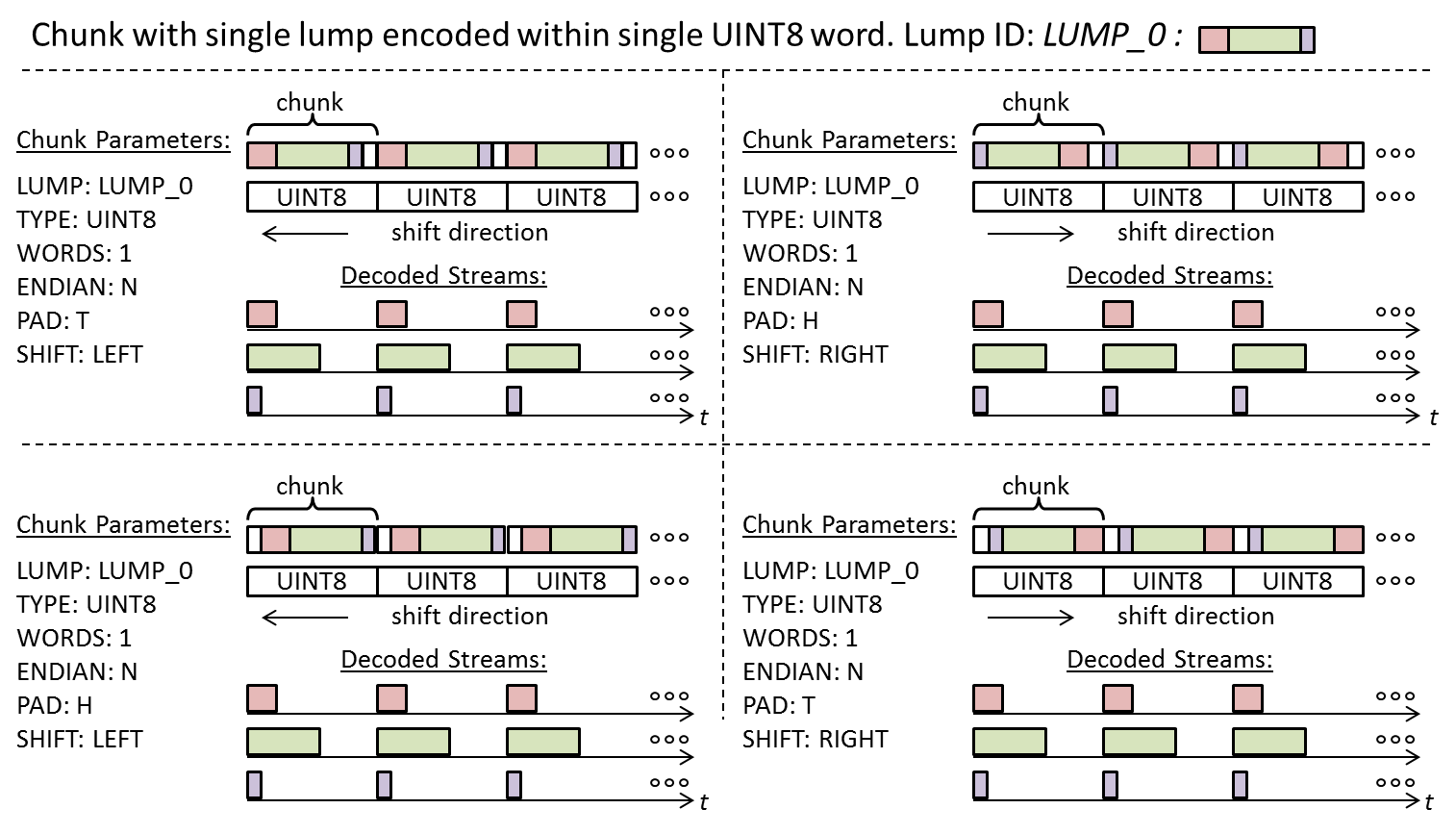


Figure 6 - Encoding Schemes for a single Lump within a single Chunk

Figure 7 illustrates two schemes where ten lumps are encoded within a chunk comprised of 9 UINT16 words. In the first case, the UINT16 words are written to disk in big-endian format. The bytes are swapped in the second case since the words are written in little-endian format.

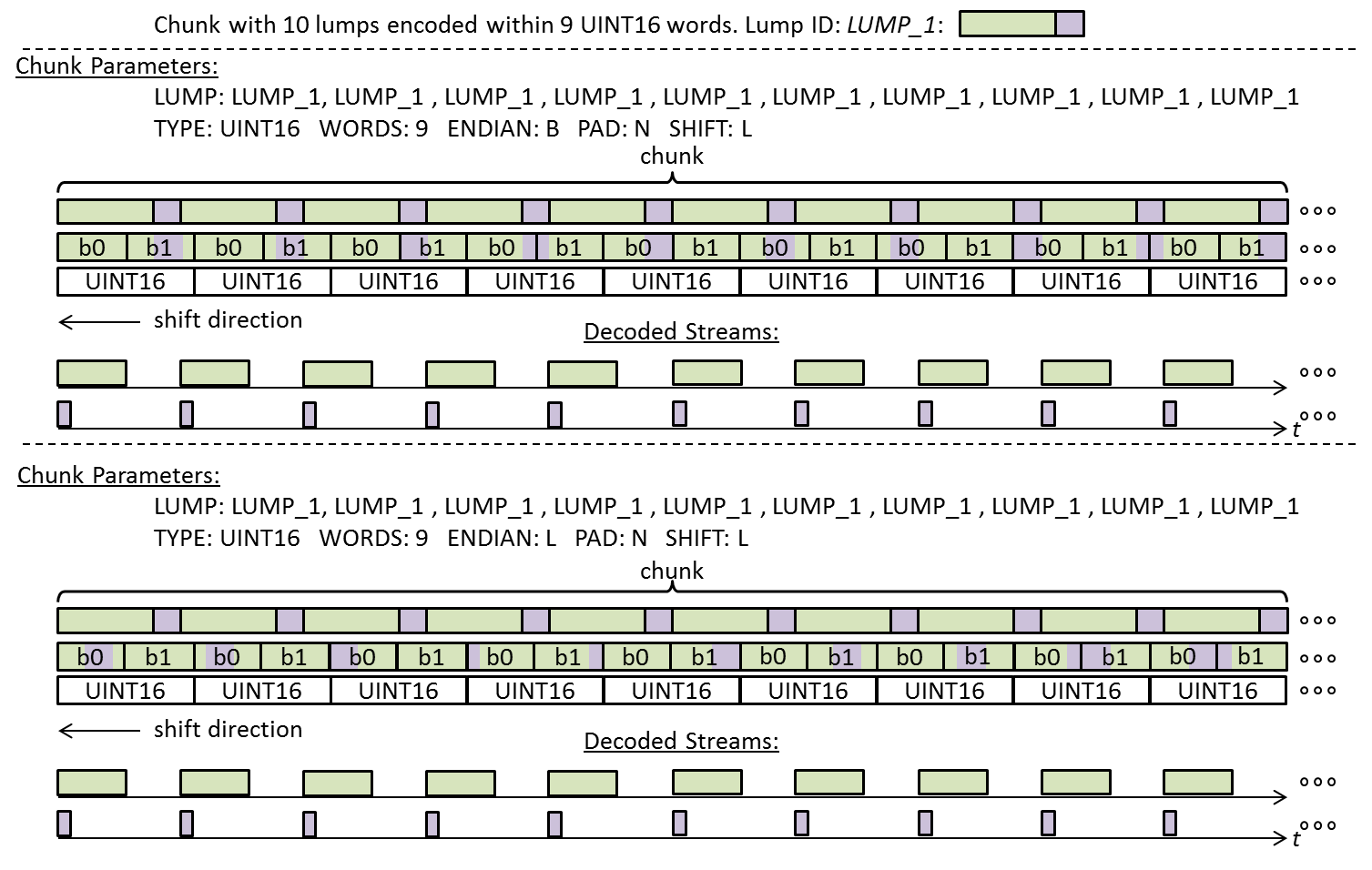


Figure 7 - Encoding Schemes for a Chunk containing 10 Lumps

## Block Parameters

A data stream may contain other undefined bytes of information. This standard includes parameters necessary to skip over these bytes while decoding sample streams. This information is contained within a metadata object referred to as a Block.

A Block has the following properties:

* A Block is comprised of a finite integer number of chunks greater than zero
* Chunks within a Block are sequential and contiguous
* A Block may begin with a data segment of arbitrary size (integer number of bytes) known as a *Header*.
* A Block may end with a data segment of arbitrary size (integer number of bytes) known as a *Footer*.
* A Block may contain data integrity features that are implemented within the Header and/or Footer segments.
* The Block data structure shall remain constant for the entire data collection session (i.e. Block format shall not change dynamically).

A Block is defined as a data segment comprised of one or more Chunks, where the Chunk data appears contiguously anywhere within said segment.

Table 9 – Definition of Block Parameters

| Parameter | Description | Type | Enumeration | Optional | Default (if not specified) |
| --- | --- | --- | --- | --- | --- |
| ID | Unique identifier | string |  | Required |  |
| CHUNK | Identifiers of all Chunks present in this Block | Comma-separated string |  | Required |  |
| CYCLES | For the Chunk ID pattern described in CHUNK, the integer number of cycles that this pattern repeats within a Block | UINT32 |  | Required |  |
| HEADER | Integer number bytes to skip in order to access first byte of chunk data | UINT32 |  | yes | 0 |
| FOOTER | Integer number of bytes to skip in order to access first byte of next block | UINT32 |  | yes | 0 |
| URL1 | Uniform resource locator of a network location or file containing additional information | string |  | Yes |  |
| COMMENT1 | Additional information | string |  | Yes |  |

NOTES:

1 Multiple instances of these parameters may exist. The parser shall enumerate accordingly (e.g. URL1, URL2, etc.).

XML Listing 8- Block Parameters

<!-- Block Parameters -->

<BLOCK ID=“BLOCK\_0” CHUNK=“CHUNK\_0” CYCLES=“56320” HEADER=“0” FOOTER=“4”>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</BLOCK>

## Lane Parameters

A Lane is defined as a conduit that transports data comprised of Blocks

XML Listing 9- Lane Parameters

<!-- Lane Parameters -->

<LANE ID=“LANE\_0” BLOCK=“BLOCK\_0” DELAY=“0.0” >

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</LANE>

## File Parameters

A File is defined as the ordered collection of bytes retrieved from a single Lane over a finite interval of time and stored in a digital media device.

When a lane is written to a file, it may or may not be synchronized to the start of a block. For this reason there may be a byte offset from the beginning of the file to the first byte of the first block. This offset is different for each file.

The creation time of the file may be tagged as metadata. This time is typically obtained from the system RTC.

XML Listing 10- File Parameters

<!-- File Parameters -->

<FILE ID=“FILE\_0” LANE=“LANE\_0” >

<OFFSET>1234</OFFSET>

<TIMESTAMP>2012-12-13T12:12:12</TIMESTAMP>

<PREVIOUS>PreviousFileName.ext</PREVIOUS>

<NEXT>NextFileName.ext</NEXT>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</FILE>

## Selection Parameters

For spatially and spatial-temporally split files, the file set must be identified. This is done by the file selection parameters. The selection parameters identify the first set of files. All other information can be obtained by parsing the metadata of those files.

XML Listing 11- Selection Parameters

<!-- Selection Parameters -->

<SELECT ID=“SEL\_0” >

<FILE>FirstFileName.ext</FILE>

<FILE>SecondFileName.ext</FILE>

<FILE>ThirdFileName.ext</FILE>

<FILE>FourthFileName.ext</FILE>

<URL>link containing additional info (browser format)</URL>

<COMMENT>this is a comment</COMMENT>

</SELECT>

## Metadata Order

The following XML listing shows structure of a metadata file and the order in which the parameters shall appear.

XML Listing 12- Example Metadata File Structure (Type 1)

<!—ION SDR Metadata File Structure: Type 1 -->

<?XML VERSION=“1.0” ENCODING=“UTF8” ?>

<!-- IONSDR Tag, Type and Version numbers -->

<!-- Type1:metadata, Type2:selection parameters -->

<IONSDR TYPE=“1” VERSION=“1.0.0.0”>

<!-- System Parameters -->

<SYSTEM ID=“SYS0” >

…

</SYSTEM>

<!-- Cluster Parameters -->

<CLUSTER ID=“ANT\_0” >

…

</CLUSTER>

<CLUSTER ID=“ANT\_1” >

…

</CLUSTER>

<!-- Source Parameters -->

<SOURCE ID=“J1” CLUSTER=“ANT0” … >

…

</SOURCE>

<SOURCE ID=“J2” CLUSTER=“ANT0” … >

…

</SOURCE>

<SOURCE ID=“J3” CLUSTER=“ANT1” … >

…

</SOURCE>

<SOURCE ID=“J4” CLUSTER=“ANT1” … >

…

</SOURCE>

<!-- Band Parameters -->

<BAND ID=“L1\_J1” SOURCE=“J1” …>

…

</BAND>

<BAND ID=“L1\_J2” SOURCE=“J2” …>

…

</BAND>

<BAND ID=“L1\_J3” SOURCE=“J3” …>

…

</BAND>

<BAND ID=“L1\_J4” SOURCE=“J4” …>

…

</BAND>

<!-- Stream Parameters -->

<STREAM ID=“S0” BAND=“L1\_J1, L1\_J2, L1\_J3, L1\_J4” … >

…

</STREAM>

<!-- Lump Parameters -->

<LUMP ID=“LUMP\_0” STREAM=“S0”>

…

</LUMP>

<!-- Chunk Parameters -->

<CHUNK ID=“CHUNK\_0” LUMP=“LUMP\_0” … >

…

</CHUNK>

<!-- Block Parameters -->

<BLOCK ID=“BLOCK\_0” CHUNK=“CHUNK\_0” … >

…

</BLOCK>

<!-- Lane Parameters -->

<LANE ID=“LANE\_0” BLOCK=“BLOCK\_0” DELAY=“0.0” >

…

</LANE>

<!-- File Parameters -->

<FILE ID=“FILE\_1” LANE=“LANE\_0” >

<OFFSET>1234</OFFSET>

<TIMESTAMP>2012-12-13T12:12:12</TIMESTAMP>

<PREVIOUS>File0.dat</PREVIOUS>

<NEXT>File2.ext</NEXT>

</FILE>

</IONSDR>

XML Listing 13- Example Selection File Structure (Type 2)

<!—ION SDR Metadata File Structure: Type 2 -->

<?XML VERSION=“1.0” ENCODING=“UTF8” ?>

<!-- IONSDR Tag, Type and Version numbers -->

<!-- Type1:metadata, Type2:selection parameters -->

<IONSDR TYPE=“2” VERSION=“1.0.0.0”>

<SELECT ID=“SEL\_0” >

<FILE>File0.dat</FILE>

</SELECT>

</IONSDR>

# Working Group Membership

|  |  |
| --- | --- |
| **Name** | **Company/Affiliation** |
| Akos, Dennis M. | University of Colorado |
| Al-Masyabi, Walid | Raytheon |
| Bavaro, Michele | One Talent GNSS |
| Belabbas, Boubeker | DLR |
| Braasch, Michael | Ohio University |
| Chansarkar, Mangesh | CSR |
| Cosgrove, Mathew | Northrop Grumman NSD |
| Crampton, Paul G. | Spirent Federal Systems |
| Dovis, Fabio | Politecnico di Torino |
| Eissfeller, Bernd | University FAF |
| Fernández Hernández, Ignacio | Galileo Supervisory Agency |
| Fernández-Prades, Carles | Centre Tecnològic de Telecomunicacions de Catalunya (CTTC) |
| Gavrilov, Artyom | GNSS-SDR |
| Glennon, Eamonn | University of New South Wales |
| Goodrich, Brian | NavCom |
| Gunawardena, Sanjeev | Air Force Institute of Technology |
| Kou, Yanhong | Beihang University |
| Kubo, Nobuaki | Tokyo University of Marine Science and Technology |
| Langer, Markus | Karlsruhe Institute of Technology (KIT) |
| Ledvina, Brent | Coherent Navigation |
| Little, Jon C. | Applied Research Laboratories of the University of Texas at Austin |
| Lohan, Elena-Simona | Tampere University of Technology |
| López-Almansa, José María | GMV |
| Lopez-Risueño, Gustavo | European Space Agency |
| Mathews, Michael B. | Loctronix |
| Morton, Yu (Jade) | Colorado State University |
| O’Brien, Andrew J. | Ohio State University |
| Pany, Thomas | Ifen GmbH |
| Parsons, Bryan Masamitsu | Applied Research Laboratories of the University of Texas at Austin |
| Pelosi, Lou | Cast Navigation |
| Petovello, Mark | University of Calgary |
| Pinchin, James | University of Nottingham |
| Psiaki, Mark | Cornell University |
| Rudra, Angsuman | D-TA Systems |
| Rügamer, Alexander | Fraunhofer IIS |
| Schipper, Brian | Honeywell |
| Schleppe, John B. | NovAtel |
| Scott, Logan | LS Consulting |
| Shivaramaiah, Nagaraj | GNSS Labs |
| Soloviev, Andrey | Qunav |
| Suzuki, Taro | Tokyo University of Marine Science and Technology |
| Tkatch, Alex | Rohde & Schwarz USA Inc. |
| Vinande, Eric | AFRL Sensors Directorate |
| Wesson, Kyle | University of Texas at Austin/Zeta Associates |
| WON, Jong-Hoon | ISTA at University FAF Munich |
| Yang, Ning | Draper Laboratory |
| Yao, Zheng | Tsinghua University |
| Yu, Jim | Trimble |
| Yu-Hsuan Chen | Stanford University |