Initial Report of Activities of the GNSS SDR Metadata Standard Working Group

Session C4: Software Receivers

September 11, 2014

Tampa, FL

Background

- Proliferation of GNSS SDR technology in the past 5-10 years
  - Low-cost front-end hardware and data collection systems
  - Maturing GNSS SDR processors, receivers and software frameworks

Background

- Today: no established standard to convey GNSS SDR metadata
  - Existing metadata standards not well suited for needs of GNSS SDR and PNT community
    - VITA-49 (packet based, sample encoding already specified)
    - GNU Radio metadata standard (binary header based)

- ION SDR Metadata Standard
  - Objective: Interoperability between GNSS SDR data collection systems and processors
  - ‘Do no harm’

Actions to Date

- **Jan, 2014**
  - ION Council approved formation of working group
- **Feb, 2014** – Selection of Co-Chairs:
  - Dr. Sanjeev Gunawardena
  - Dr. Thomas Pany
- **Apr, 2014**
  - ION Executive Council reviewed and approved invite list and documented process to follow
  - Approx. 95 Invitations sent to representative international cross section of GNSS SDR subject matter experts from academia, industry and government
  - Invite material included ION legal reviewed forms to be returned with acceptance
    - Conflict of Interest Disclosure Statement
    - Request for Patents and Participant's Position Regarding Licensing of Essential Patent Claims
  - Returned forms reviewed by ION Executive Director

Actions to Date (contd.)

- **May, 2014**
  - Established Working Group (49 members) and ION-administrated online discussion group
  - Posted initial draft (developed by co-chairs) for WG feedback
  - Online WG discussions
- **Aug, 2014**
  - WG feedback incorporated into second draft and posted for discussion
- **Sep, 2014**
  - Discuss and finalize Initial Report and standard First Draft at first in-person WG meeting on Tuesday Sept 9 from 15:10 to 17:10
  - Initial Report Presentation at Software Radio Session (C4) on **Thursday, Sep. 11 at 17:08**

Working Group Members

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

Name	Company/Affiliation
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

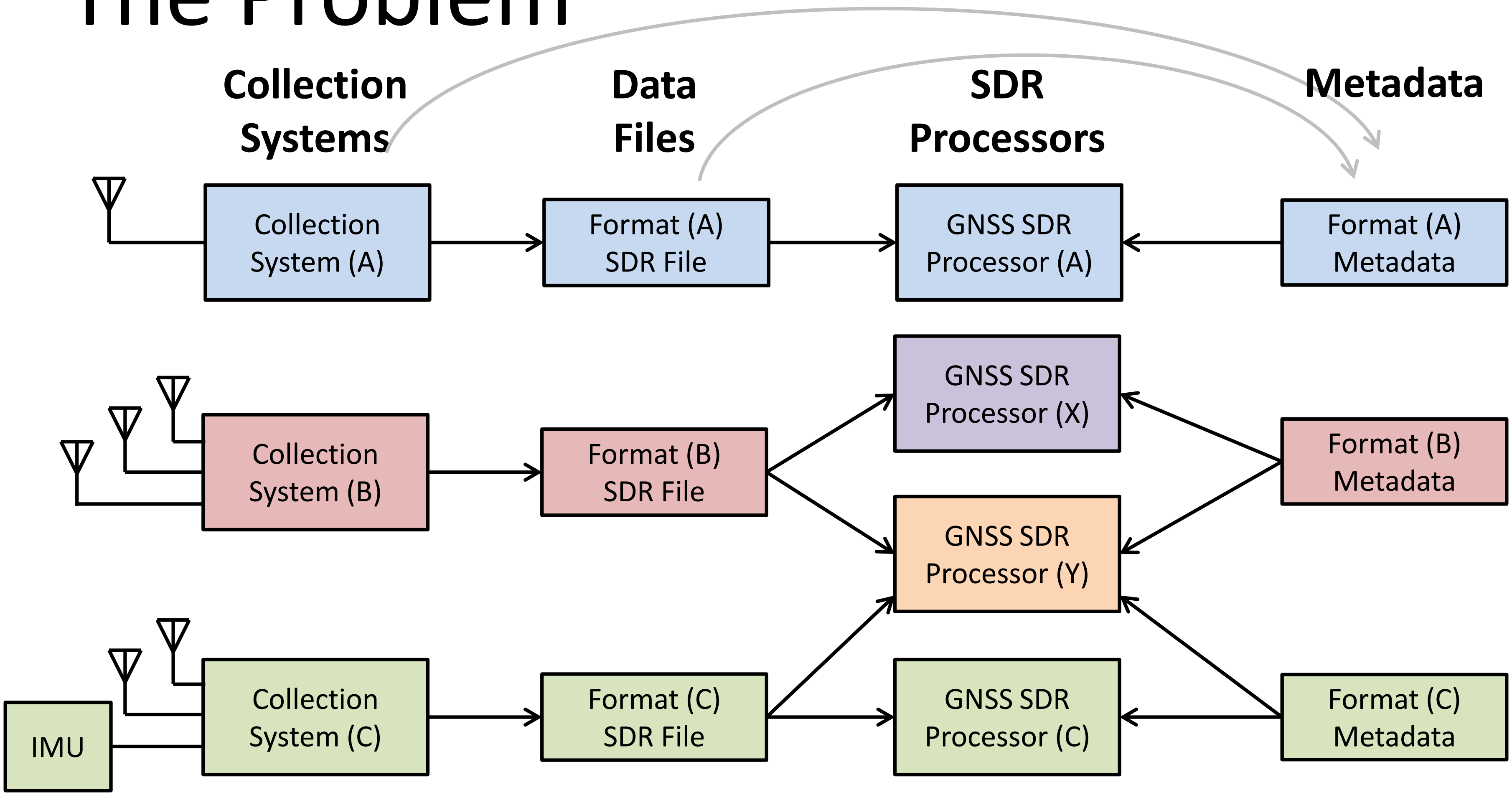
Summary of 9/9/14 WG Meeting

- Attendance: 30+, mostly WG members
- Motions passed, items agreed upon:
  - XML as metadata description format
  - ‘SDRX’ as metadata file extension
  - Documentation subcommittee
  - ‘Metadata codec’ reference implementation
  - Alpha testing
- 2nd draft standard accepted in principal with proposed revisions
- Several action items

Next Steps

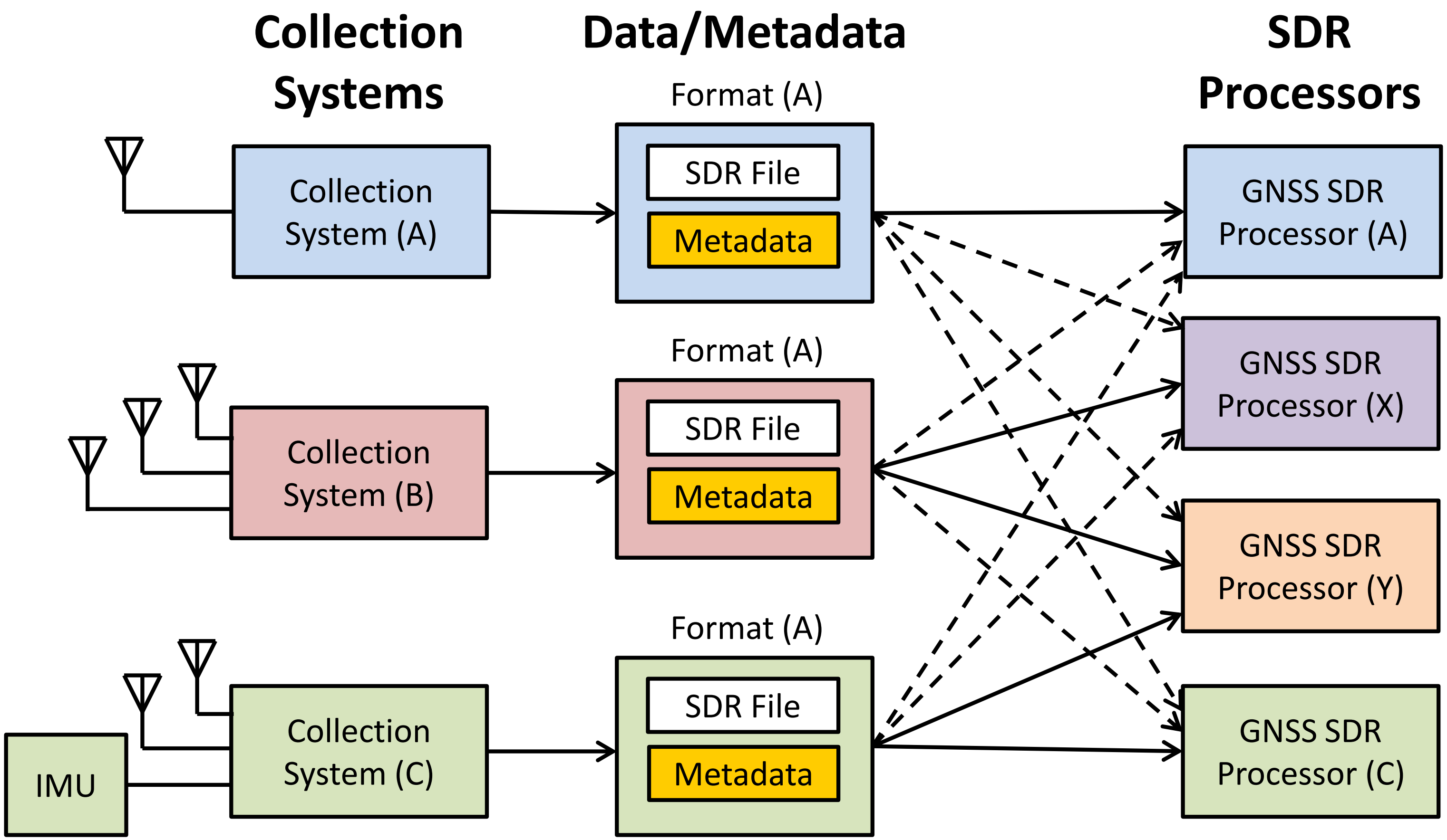
- **Oct 13** : Initial Draft Standard
- **Oct 13, 2014 to Jan 12, 2015** : 90-day Open Public Comment Period
  - ION.ORG home page notice during open comment period
  - Published on ION.ORG: Initial Report, Initial Draft, online comment form (similar to ION abstract portal)
  - WG members encouraged to initiate informal invitations to GNSS/PNT community for comment
  - WG members will review and discuss public comments as and when received
- Jan 26, 2015 – In-person WG meeting at ION ITM 2015
  - Proposed Draft Standard activities
  - Follow-on report at ION ITM
  - Some adoption of draft standard (reference implementation subcommittee)

The Problem



- Some front-end/DCS and SDR processors are 'married' to one another
- Ad hoc metadata exchange – prone to human error
- Does not promote interoperability
- Does not promote data/resource sharing and re-use

Proposed Solution: Metadata Standardization



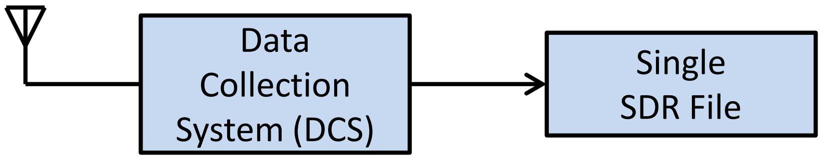
- Unambiguous transfer of all essential SDR metadata
- Standardization encourages vendors to support major formats
- Spurs community to develop open-source software handlers and plug-ins
- Promotes interoperability
- Promotes data portability, resource sharing and re-use

Challenges

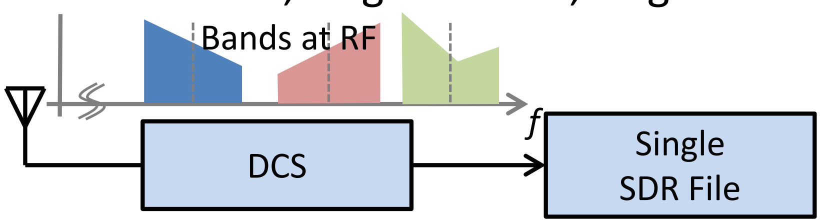
- Many SDR Data Collection Topologies
  - We're not defining SDR data standard
  - Need to get it 'mostly correct' on first try
    - Minor revisions okay
    - Maintain backwards compatibility between revisions
- Ensuring Widespread Adoption
  - Early adoption by academia & research labs is key
  - Open-source 'metadata codecs' and format handlers
- Vendor Support
  - Actively pursue vendor participation in WG
  - Report/demonstrate success of early-adoption efforts
  - Work through any 'proprietary format' issues

SDR Data Collection Topologies

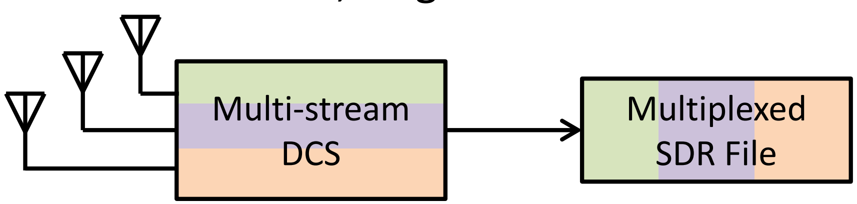
Single band, single-stream, single file



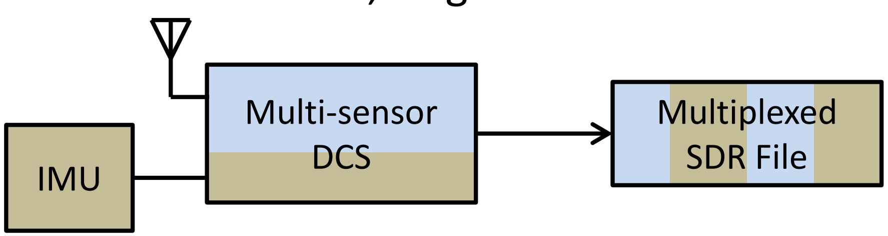
Multi-band, single-stream, single file



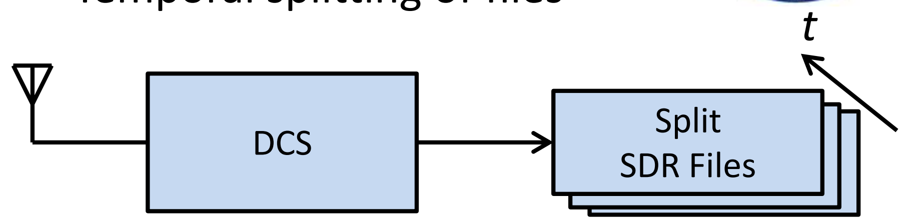
Multi-stream, single file



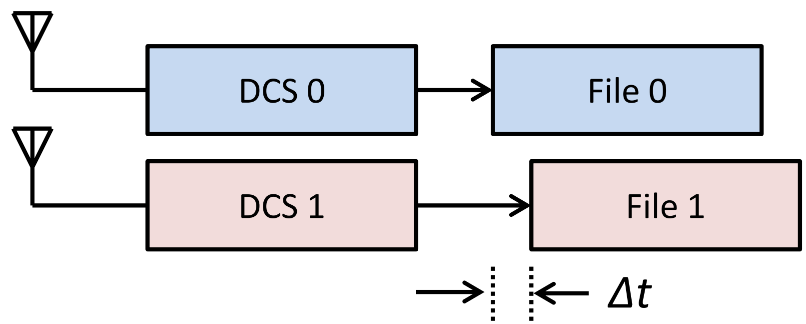
Multi-stream, single file



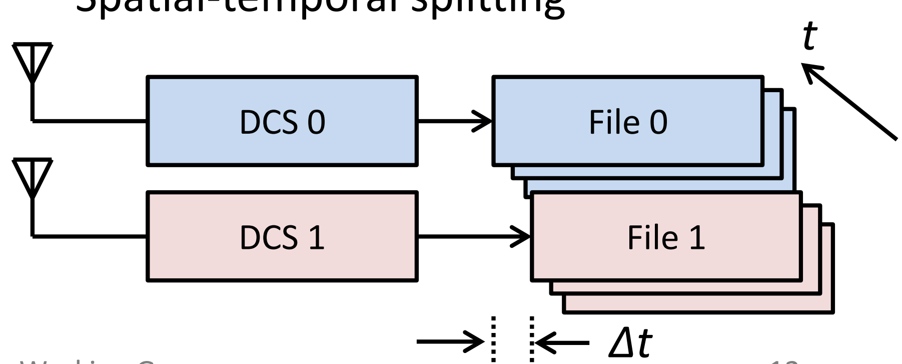
Temporal splitting of files



Spatial splitting of files

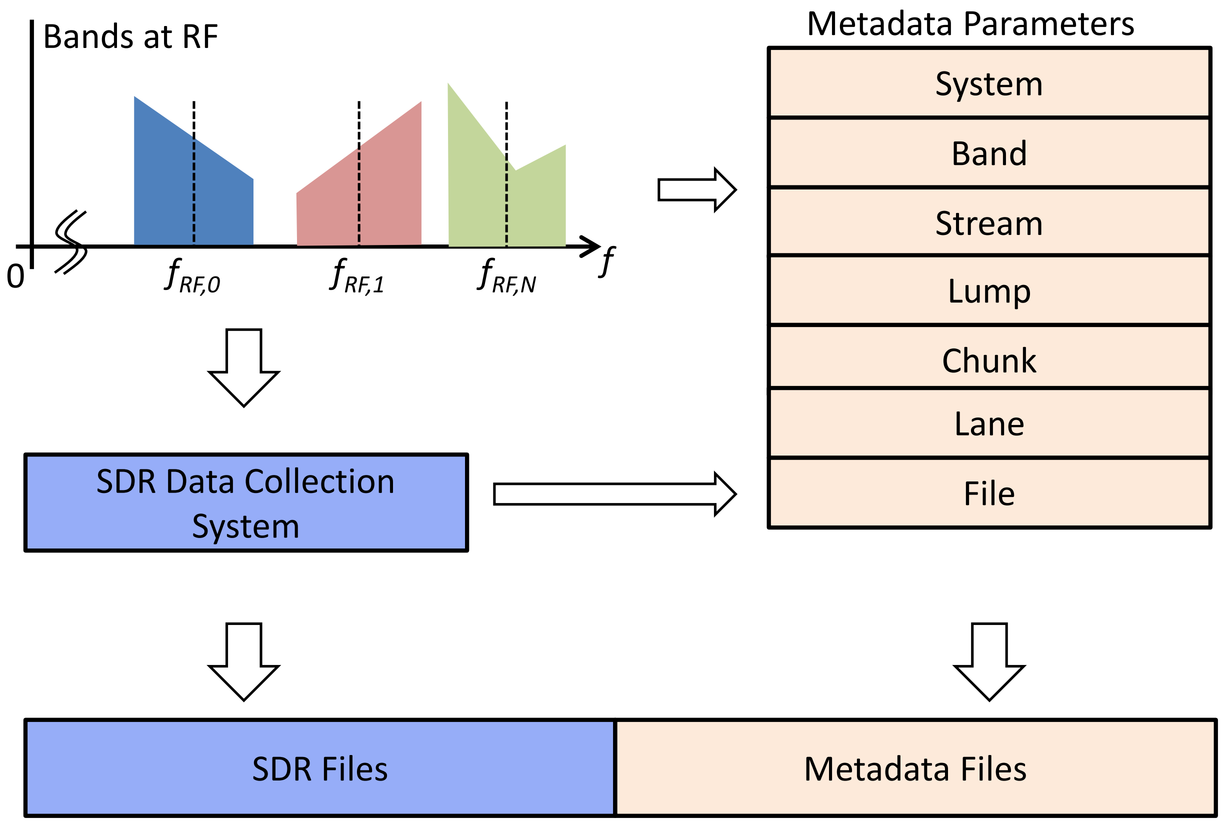


Spatial-temporal splitting



Metadata Parameters

To support various topologies, data formats and their combinations, the SDR metadata standard defines a set of parameters to fully describe SDR file(s). We refer to these as metadata parameters. The figure below depicts the hierarchy of metadata parameters.



Todo: Finalize these terms

The following slides define each of these Metadata Parameters

eXtensible Markup Language (XML) as Metadata Format

- Free and open standard
- Wide industry adoption
- API support in all OS and app development frameworks (C/C++, python, MATLAB, …)
- Lightweight, text-based format
- Human readable/editable
- Extensibility (supports proprietary metadata)
- Supports back-annotation

System Parameters

- Specifies top-level parameters related to data collection campaign and equipment

Table 1. System Parameters

Parameter	Symbol	Enumeration	Type	Units
Time zone	**TIMEZONE**	TBD	string (3 letters?)	
Lowest common factor sample rate, f_s	FSBASE		**UINT32.UINT32 (N/D)**	samples/second
Information relating to data collection campaign	CAMPAIGN		string	
Scenario contained in data	SCENARIO		string	
Information on data collection equipment and/or configuration	EQUIPMENT		string	
Antenna: make, model, type, etc.	ANTENNA		string	
approximate (antenna) location of data collection. Lat, Lon, Height coordinates	LOCATION		string	
Point-of-contact for data collection (person's name)	POC		string	
Contact info for POC. E.g. email	CONTACT		string	

System Parameters: XML

```xml
<!-- System Parameters -->
 <TIMEZONE>UTC</TIMEZONE>
 <FSBASE>56320000.1</FSBASE>
 <CAMPAIGN>Collection Campaign</CAMPAIGN>
 <SCENARIO>Specific Scenario</SCENARIO>
 <EQUIPMENT>Equipment used: Make/Mode/Serial</EQUIPMENT>
 <ANTENNA>Antenna Make/Model</ANTENNA>
 <LOCATION>Location Coordinates</LOCATION>
 <POC>Entity or person that collected data</POC>
 <CONTACT>Contact information</CONTACT>
```

Band Definition and Parameters

- **A band is defined as a finite span of radio frequency (RF) spectrum.**
- Band parameters include information on frequency translation to baseband.

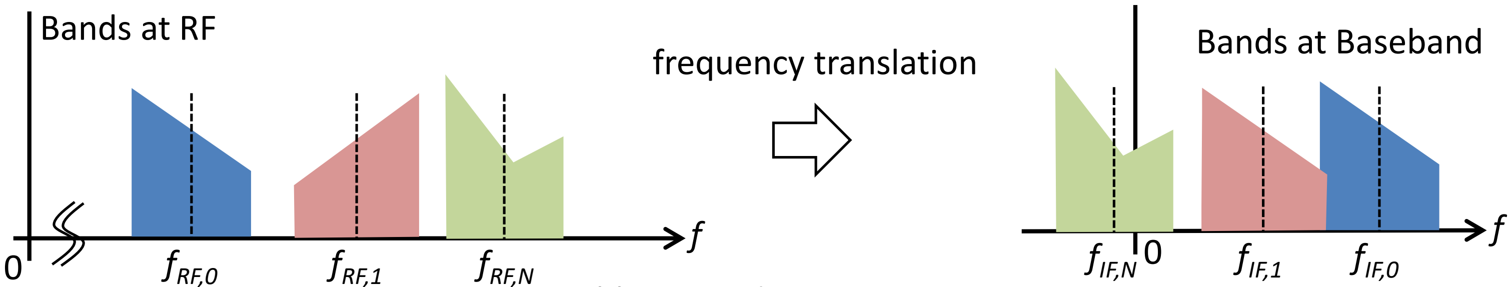


Table 1. Band Parameters

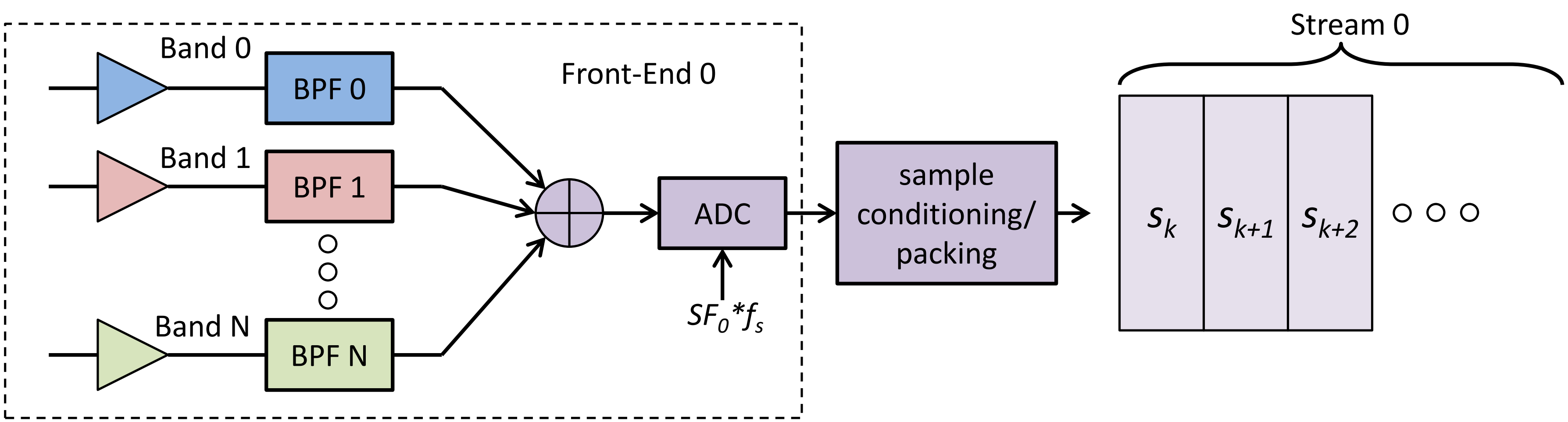
Parameter	Symbol	Enumeration	Type	Units
number of bands	NUMBANDS		UINT16	
containing-stream index	STRIDX		UINT8	zero-referenced index
unique identifier	NAME		string	
RF center frequency	FRF		**UINT32.UINT32 (N/D)**	cycles/second
translated center frequency	FIF		**UINT32.UINT32 (N/D)**	cycles/second
spectral inversion flag	INV	0: non-inverted 1: inverted	UINT8	
approximate double-sided half-power bandwidth	BW3		double	cycles/second

Band Parameters (XML)

```
<NUMBANDS>4</NUMBANDS>
  <BAND STRIDX="0" NAME=“L1 ANT1” FRF=“1575420000.1” FIF=“13680000.1”
INV=“0” BW3=“20000000.1”></BAND>
  <BAND STRIDX="1" NAME=“L2 ANT1” FRF=“1227600000.1” FIF=“13680000.1”
INV=“0” BW3=“24000000.1”></BAND>
  <BAND STRIDX="2" NAME=“G1 ANT1” FRF=“1602000000.1” FIF=“13680000.1”
INV=“0” BW3=“26000000.1”></BAND>
  <BAND STRIDX="3" NAME=“G2 ANT1” FRF=“1246000000.1” FIF=“13680000.1”
INV=“0” BW3=“26000000.1”></BAND>
```

Stream

- **A stream is defined as a series of binary values that are derived from samples produced by a digitizing radio frequency front-end or equivalent source.**
- Stream sample rate specified as integer multiple (*SF*) of the system base sample rate f_s
- Stream data contains information from one or more bands
- Front-end samples are conditioned and packed into a stream according to the following stream parameters:
  - Baseband or IF sampling (including inversion of numerical values)
  - Sample quantization (for baseband sampling, I & Q samples treated as two distinct values with same quantization. i.e. two values for each sample)
  - Packed bits per sample
  - Sample encoding scheme
  - Packed sample alignment



Stream Parameters

Parameter	Symbol	Enumeration	Type	Units
number of streams	NUMSTR		UINT16	
unique identifier	NAME		string	
sample rate factor	SF		UINT8	integer scale factor
sampling method and representation		IF, IF’, IQ, IQ’, I’Q, I’Q’, QI, QI’, Q’I, Q’I’ (where ‘ signifies inversion)	string	
sample quantization	QTZ		UINT8	bits/sample
Packed bits/sample	BTP		UINT8	bits/sample
Sample packing alignment	ALN	‘L’ – left aligned ‘R’ – right aligned ‘N’ – not applicable	string	
Sample encoding	ENC	SIGN, SM, INT, BIN, FP	string	

Sample Encoding Schemes

Bits (QTZ)	Encoding	Set	Range Min	Range Max	Code
1	sign	{-1, +1}	-1	+1	SIGN
2	sign-magnitude	{-3, -1, +1, +3}	-3	+3	SM2
	signed integer	{-2, -1, 0, 1}	-2	+1	INT2
	offset binary	{-2, -1, 0, 1}	-2	+1	BIN2
3	sign-magnitude	{-7, -5, -3, -1, +1, +3, +5, +7}	-7	+7	SM3
	signed integer	{-4, -3, -2, -1, 0, 1, 2, 3}	-4	+3	INT3
	offset binary	{-4, -3, -2, -1, 0, 1, 2, 3}	-4	+3	BIN3
4	sign-magnitude	{-8, -7,…,-1,+1,…, +8}	-8	+8	SM4
	signed integer	{-8, …, 0…, +7}	-8	+7	INT4
	offset binary	{-8, …, 0…, +7}	-8	+7	BIN4
8	sign-magnitude	{-128, -127,…, +127, +128}	-127	+128	SM8
	signed integer	{-128, …, 0…, 127}	-128	+127	INT8
	offset binary	{-128, …, 0…, 127}	-128	+127	BIN8
16	sign-magnitude	{-2^{15},… ,-1,+1,…, $+2^{15}$}	-2^{15}	$+2^{15}$	SM16
	signed integer	{-2^{15}, …, 0…, $2^{15}-1$}	-2^{15}	$+2^{15}-1$	INT16
	offset binary	{-2^{15}, …, 0…, $2^{15}-1$}	-2^{15}	$+2^{15}-1$	BIN16
	floating point	IEEE 754-2008, FP16			FP16
32	sign-magnitude	{-2^{31},… ,-1,+1,…, $+2^{31}$}	-2^{31}	$+2^{31}$	SM32
	signed integer	{-2^{31}, …, 0…, $2^{31}-1$}	-2^{31}	$+2^{31}-1$	INT32
	offset binary	{-2^{31}, …, 0…, $2^{31}-1$}	-2^{31}	$+2^{31}-1$	BIN32
	floating point	IEEE 754-2008, FP32			FP32
64	sign-magnitude	{-2^{63},… ,-1,+1,…, $+2^{63}$}	-2^{63}	$+2^{63}$	SM64
	signed integer	{-2^{63}, …, 0…, $2^{63}-1$}	-2^{63}	$+2^{63}-1$	INT64
	offset binary	{-2^{63}, …, 0…, $2^{63}-1$}	-2^{63}	$+2^{63}-1$	BIN64
	floating point	IEEE 754-2008, FP64			FP64

Other QTZs and encodings exist. WG should determine if we need to specify all of them

Stream Parameters (contd.)

- QTZ, BTP, and ALN specify how extra bits are discarded in the decoding process.
- ALN is ‘N’ when QTZ equals BTP (i.e. no extra bits to consider in packed format)
- When BTP is greater than QTZ, <u>discard extra bits when decoding</u> (i.e. do not assume sign extension or zero padding)
- QTZ, BTP and ALN parameters only specify how quantized samples of a specified encoding are packed. They do not specify which bits are MSB/LSB. This information is specified in chunk parameters.

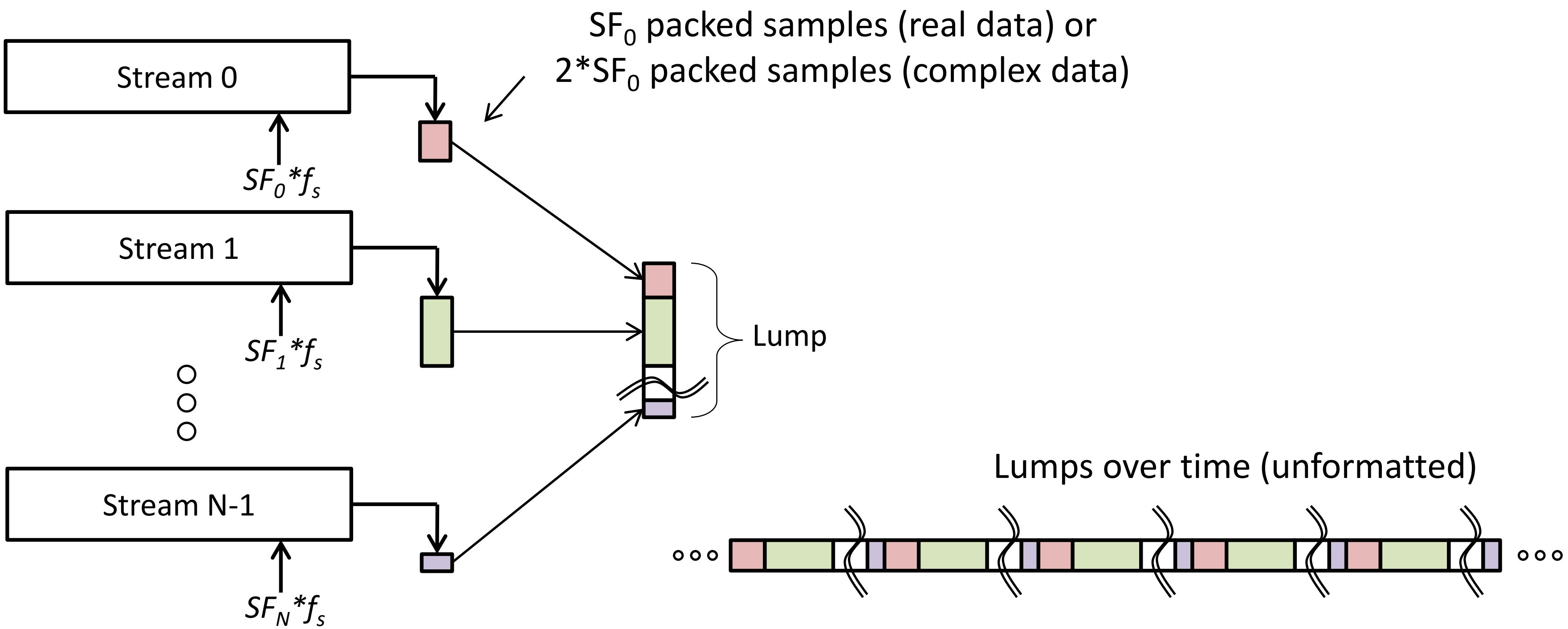
XML:

```
<!-- Stream Parameters -->
<NUMSTR>4</NUMSTR>
<STREAM NAME=“S0” SF=“1” QTZ=“1” BTP=“1” ALN=“N” ENC=“SIGN”></STREAM>
<STREAM NAME=“S1” SF=“1” QTZ=“1” BTP=“1” ALN=“N” ENC=“SIGN”></STREAM>
<STREAM NAME=“S2” SF=“1” QTZ=“1” BTP=“1” ALN=“N” ENC=“SIGN”></STREAM>
<STREAM NAME=“S3” SF=“1” QTZ=“1” BTP=“1” ALN=“N” ENC=“SIGN”></STREAM>
```

Lump

Samples from multiple coherently-sampled streams may be multiplexed in time to form a single ‘collection’ of sequential samples. This requires parameters to describe how streams are multiplexed. The term ‘chunk’ is used later to describe a segment of data for which an encoding/decoding scheme is specified. Hence, the term ‘lump’ is used to describe the collection of all binary data that gets generated during a single $1/f_s$ sampling epoch.

A lump is defined as the ordered containment of all samples occurring within the interval ***$t_s=1/f_s$***



Lump Parameters

Parameter	Symbol	Enumeration	Type	Units
number of multiplexed streams	NMXSTR		UINT8	Streams
stream indices (in order of appearance)	STRIDX		CSV String of NSTR UINT16 values	Zero-referenced indices

XML:

```
<LUMP NMXSTR="4" STRIDX="0,1,2,3"></LUMP>
```

Chunk Definition

In general, the sample packing scheme must be known to correctly decode the samples. For example, consider 16 1-bit real samples packed into a UINT16 words written in little-endian format. These samples would be decoded incorrectly if shifted out from a UINT32 word (due to little-endianness).

Another consideration is whether bits are packed from left to right or vice versa.

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.

The Stream, Lump and Chunk parameters together, completely and unambiguously describe the sample decoding scheme.

Chunk Parameters

Parameter	Symbol	Enumeration	Type	Units
number of lumps	NLMP		UINT8	lumps
packed word type	TYPE	‘UINT8’, ‘UINT16’, ‘UINT32’, ‘UINT64’	String	
number of words	NWORDS		UINT8	words
word endianness	ENDIAN	‘L’ – little endian ‘B’ – big endian ‘N’ – not applicable	string	
Padding	PAD	‘H’ – head padding ‘T’ – tail padding ‘N’ – no padding	string	bits (value implied)
word shift direction	SHIFT	‘L’ – left shift ‘R’ – right shift	string	

XML:

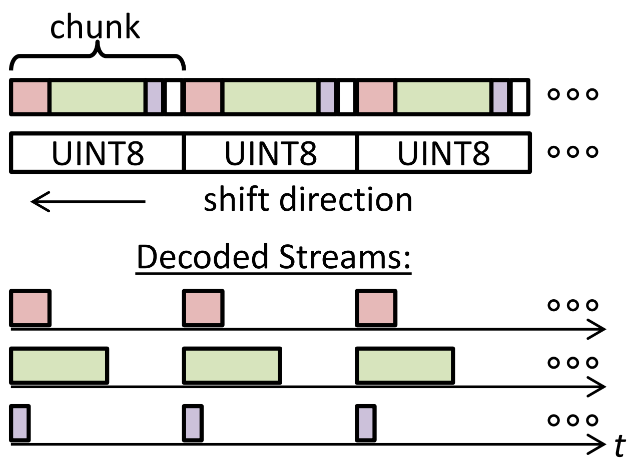
```xml
<!-- Chunk Parameters -->
<CHUNK NLMP=“1” TYPE=“UINT32” NWORDS=“1” ENDIAN=“L” PAD=“N” SHIFT=“R”></CHUNK>
```

Chunk Examples

Examples: Chunk with single lump encoded within single UINT8 word. Lump: 

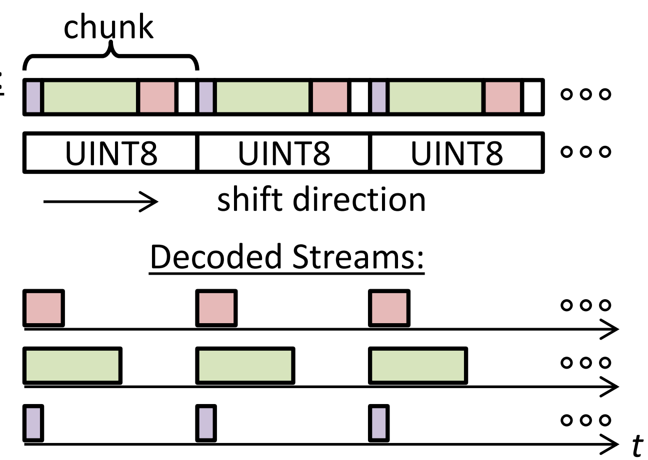
Chunk Parameters:

NLMP: 1
TYPE: UINT8
WORDS: 1
ENDIAN: N
PAD: T
SHIFT: LEFT



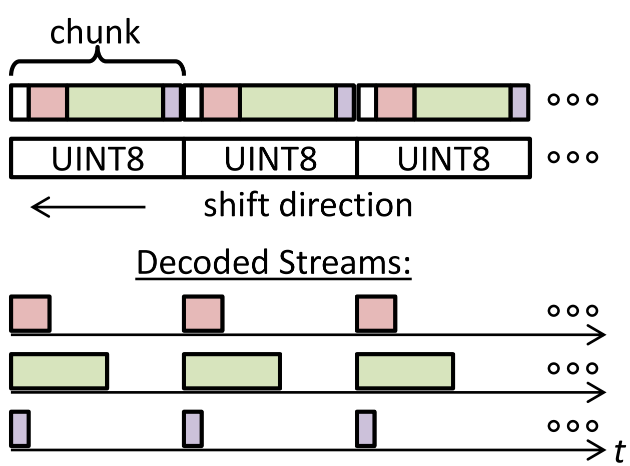
Chunk Parameters:

NLMP: 1
TYPE: UINT8
WORDS: 1
ENDIAN: N
PAD: H
SHIFT: RIGHT



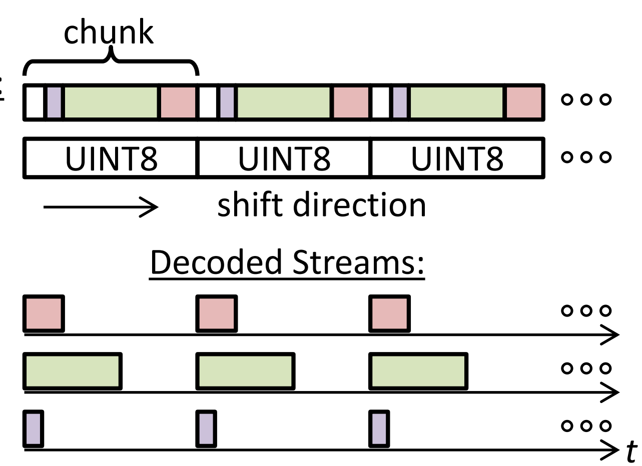
Chunk Parameters:

NLMP: 1
TYPE: UINT8
WORDS: 1
ENDIAN: N
PAD: H
SHIFT: LEFT



Chunk Parameters:

NLMP: 1
TYPE: UINT8
WORDS: 1
ENDIAN: N
PAD: T
SHIFT: RIGHT

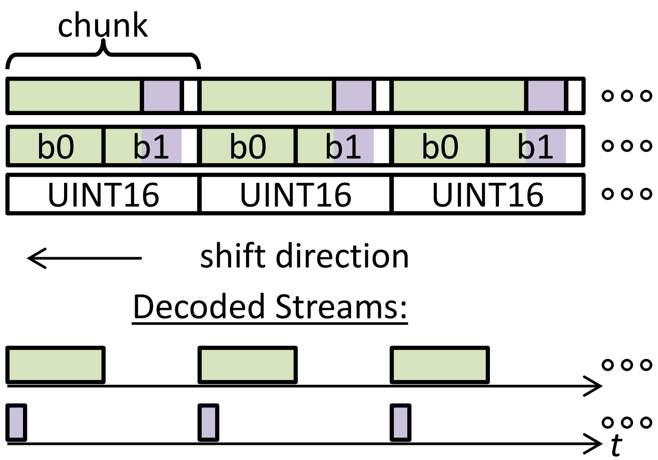


Chunk Examples (contd.)

Example: Chunk with single lump encoded within single UINT16 word. Lump:

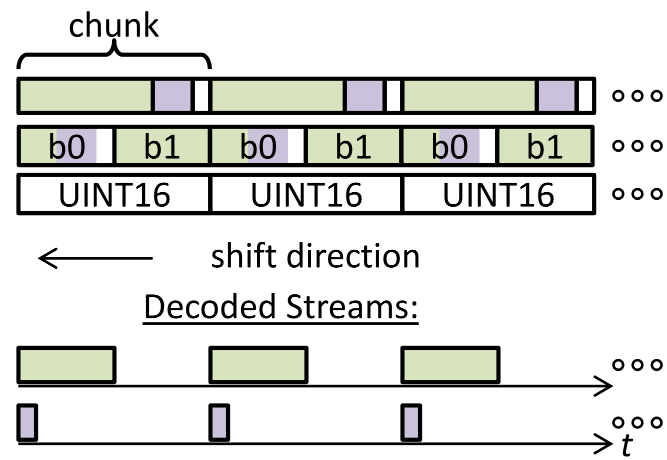
Chunk Params:

NLMP: 1
TYPE: UINT16
WORDS: 1
ENDIAN: B
PAD: T
SHIFT: LEFT



Chunk Params:

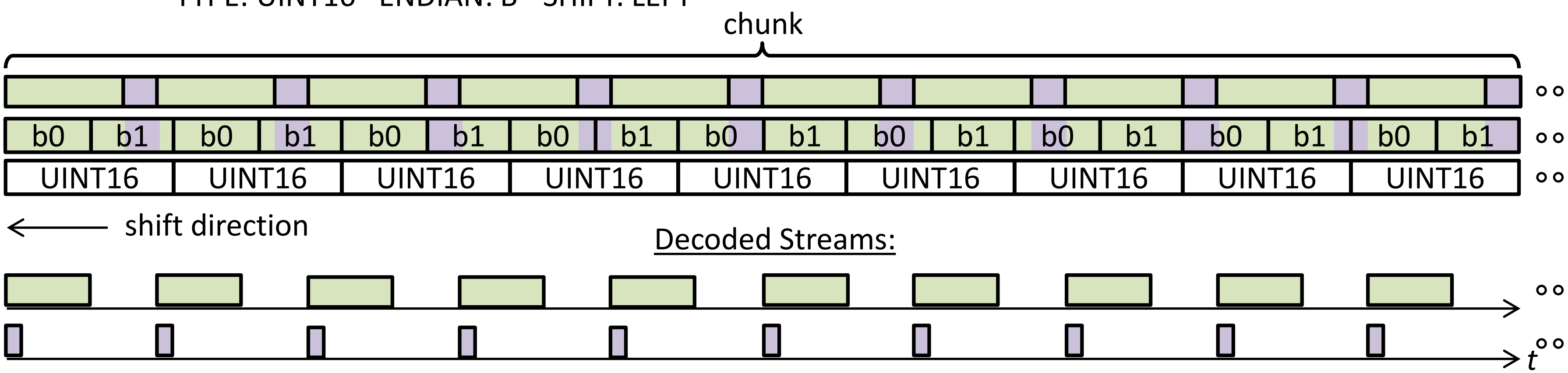
NLMP: 1
TYPE: UINT16
WORDS: 1
ENDIAN: L
PAD: T
SHIFT: LEFT



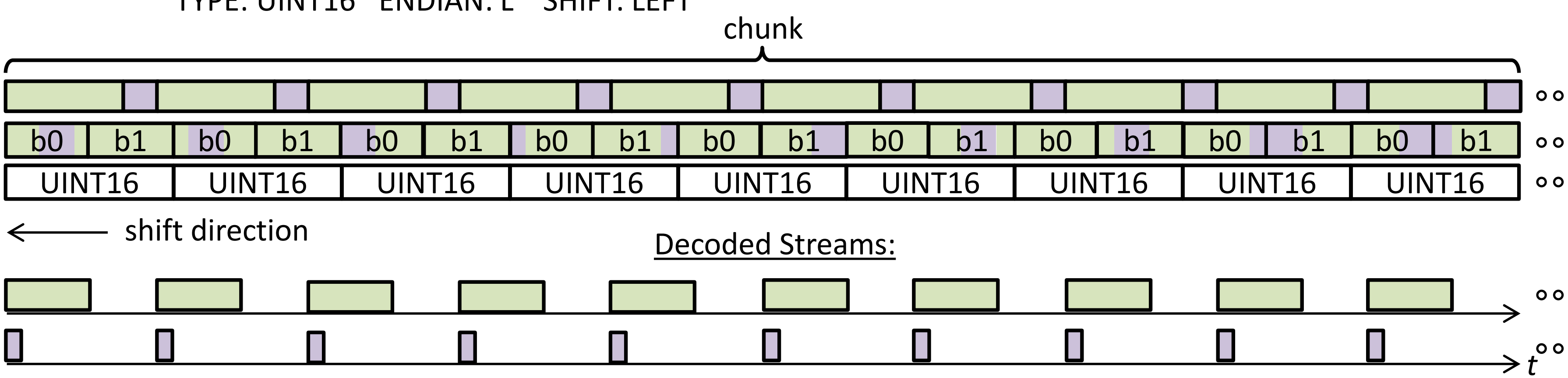
Chunk Examples (contd.)

Example: Chunk with 10 lumps encoded within 9 UINT16 words. Lump:

Chunk Params: NLMP: 10 WORDS: 9 PAD: N
TYPE: UINT16 ENDIAN: B SHIFT: LEFT



Chunk Params: NLMP: 10 WORDS: 9 PAD: N
TYPE: UINT16 ENDIAN: L SHIFT: LEFT

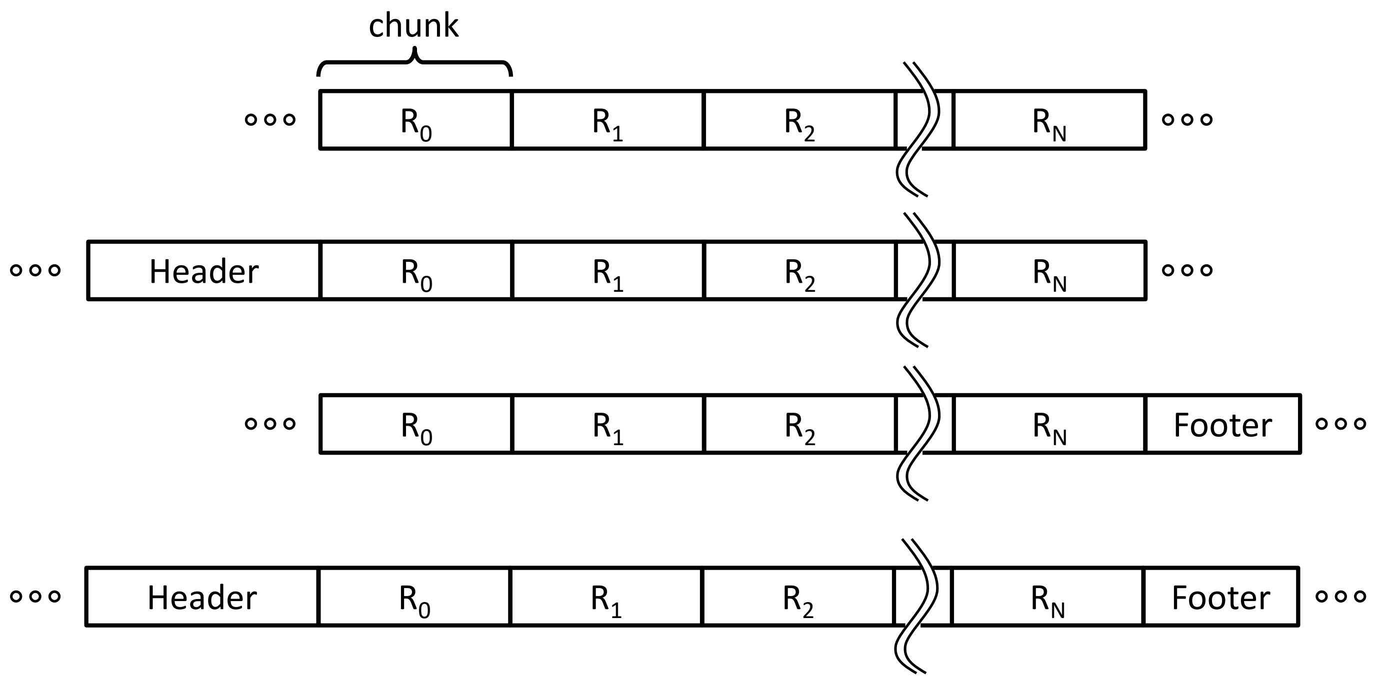


Frame Definition

A data file may contain other data in addition to SDR data that are multiplexed together. Or, the SDR data may be contained in a payload of a packet (example: when a VITA-59 stream is written to file). At a minimum, the standard must contain information to skip over bytes and point to the start of one or more chunks to be decoded. This is accomplished by defining a frame.

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

Valid Frames



Frame Parameters

Parameter	Symbol	Enumeration	Type	Units
total number of bytes in frame	NUMBYTES		UINT32	bytes
number of bytes in header	HEADERBYTES		UINT32	bytes
number of bytes in footer	FOOTERBYTES		UINT32	bytes
number of chunks in frame	NUMCHUNKS		UINT32	

XML:

```
<!-- Frame Parameters -->
 <FRAME NUMBYTES=“7044” HEADERBYTES=“0” FOOTERBYTES=“4” NUMCHUNKS=“56320”></FRAME>
```

Concept of a Lane

SDR data collection systems write formatted data into files in numerous ways. We refer to these as system topologies. The main topologies are:

- Single ‘stream of data’ written to a single file
- Single ‘stream of data’ containing two or more multiplexed streams written to a single file
- Two or more ‘streams of data’ written to individual files (each containing one stream, referred to as ‘spatial splitting’)
- Two or more ‘streams of data,’ each containing two or more multiplexed streams written to individual files
- The term ‘file’ above may be one large file or multiple files split across time intervals (temporal splitting)

The ‘stream of data’ referenced above is typically written to a high-speed disk array. Sector-aligned write modes may be used for performance reasons. For this and possibly other reasons, the start of a frame may not be aligned to the beginning of a file.

Concept of a Lane (contd.) and Definition

For spatially split files, each metadata file must identify its paired SDR file's association with respect to other SDR files in the data set.

Spatially split files may be written across multiple computer systems. Hence, the time offsets between these systems may be needed to synchronize streams.

These time offsets may not be known at collection time but may be resolved later. Hence the format must support back-annotation

The 'stream of data' terminology used above is henceforth termed 'Lane'

A Lane is defined as a conduit that transports formatted data.

Lane Parameters

Parameter	Symbol	Enumeration	Type	Units
total number of lanes in data collection system	NUMLANES		UINT16	Lanes
Lane index associated with this metadata file	LANEINDEX		UINT32	zero-referenced index
Time offset of this lane w.r.t. Lane Index 0	TOFFSETMS		INT32.UINT32 (N/D)	milliseconds

XML:

```
<!-- Lane Parameters -->
<NUMLANES>1</NUMLANES>
<LANEINDEX>0</LANEINDEX>
<TOFFSETMS>0.1</TOFFSETMS>
```

File Definition and Parameters

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

Parameter	Symbol	Type	Units
Integer byte offset to start of first frame of Lane	BYTEOFFSET	UINT32	Bytes
Time that this file was created	CREATIONTIME	TimeCSV string	
Previous file name of temporally split sequence	PREVIOUSFILE	String	
Next file name	NEXTFILE	String	

XML:

```
<!-- File Parameters -->
<BYTEOFFSET>1234</BYTEOFFSET>
<CREATETIME>2014,12,31,23,59,59,999></CREATETIME>
<PREVIOUSFILE></PREVIOUSFILE>
<NEXTFILE>WideBand_1bit_L1L2R1R2_001.dat</NEXTFILE>
```

Nice to Haves

- Ref. oscillator type and clock model (af0, af1, af2)
- SVs in view. Type, PRN, Doppler, CodePhase
- Orbit info for SVs in view. Embedded TLE data
- Events (such as anomalous events)
- Front-end gain settings/ change events
- Data/Metadata association check (e.g. MD5 hash)

Documentation Subcommittee

- Committed to all WG documentation activities
  - Especially Sept 15 to Oct 15 2014
- 2-3 persons (in addition to co-chairs)
- Responsibilities
  - Review related standards to adopt/maintain 'best practices' (ION Std. 101, VITA-49, IEEE standards, etc.)
  - Create/review presentations, reports and draft standard documents
  - Ensure quality of all ION SDR WG publications
- Qualifications
  - Excellent writing/editing skills
  - (preferred) Previous experience/involvement writing standards
  - (preferred) Individuals with recognized track record of publication excellence

Reference Implementation Subcommittee

- Open source implementation of metadata 'codec'
  - Goes a long way towards successful adoption
  - Languages: C/C++, Python, MATLAB (MEX)
  - Scope: Structure(s) <-> Metadata file(s)
  - Based on free OS XML libraries (e.g. TinyXML)
  - Public repository (on GitHub, SourceForge)
- Coding group: 2-3 persons (balanced representation)
- Responsibilities:
  - Develop metadata 'codec' implementation according to draft standard and software development best practices
  - Has authority to revise XML format for implementability as needed
  - Support industry-standard ports and tool-flows (gcc, Make, MSVS, etc.)
  - Create and maintain public repository (gatekeeper)

How you can get involved

- Join the working group / online discussion group
- Join Documentation Subcommittee
- Join Reference Implementation Subcommittee
- Send us your SDR files and format descriptions (upload)
- Sign up for alpha testing
- **Oct 13**: Initial Draft Standard on ION.org
- **Oct 13, 2014 to Jan 12, 2015** : 90-day Open Public Comment Period
- **ION ITM 2015** : Next Working Group Meeting

Contact WG Co-Chairs:
Dr. Sanjeev Gunawardena: sanjeev@ChameleonChips.com
Dr. Thomas Pany: t.pany@ifen.com