Sensor Independent Derived Data File Format Description Document

Version 0.1

10/30/2009

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Scope

The Sensor Independent Derived Data (SIDD) File Format Description Document specifies the placement of SIDD products into an approved SIDD container file format. The two file container formats that support SIDD products are the NITF 2.1 format and the GeoTIFF 1.0 format. SIDD products in NITF 2.1 and GeoTIFF 1.0 file containers use a limited set of features that those file containers support. The NITF 2.1 file container format is described in MIL-STD-2500C, National Imagery Transmission Format, Version 2.1, dated 01 May 2006. The GeoTIFF 1.0 file container format is based on TIFF 6.0, currently controlled by Adobe.

For each container file format, the following topics are covered.

- Capabilities & Limitations
- File Container Organization
- File Container Metadata
- SIDD Product Metadata
- SIDD Product Image Pixel Data

1.2 Applicable Documents

The documents listed in Table 1.2-1 and Table 1.2-2 are referenced throughout this document. All reference documents are subject to revision and users of this document should investigate recent editions and change notices.

Table 1.2-1 Government Documents and Publications

Number	Title & Website	Date
MIL-STD-2500C	National Imagery Transmission Format, Version 2.1 http://assist.daps.dla.mil/quicksearch/basic_profile.cfm?ident_number=112606	01 May 2006 (Revision C)
	Sensor Independent Complex Data (SICD) Design & Exploitation Description Document, version 0. 3 DRAFT	17 March 2009 version 0.3 DRAFT
	Sensor Independent Derived Data (SIDD) Design & Exploitation Description Document, version 0.1	30 October 2009 version 0.1

Table 1.2-2 Other Applicable Documents

Title & Website	Date
GeoTIFF Format Specification, GeoTIFF Revision 1.0 http://www.remotesensing.org/geotiff/spec/geotiffhome.html	28 December 2000
TIFF, Revision 6.0 Final http://partners.adobe.com/public/developer/en/tiff/TIFF6.pdf	3 June 1992

2 SIDD Products in NITF 2.1 Format

The purpose of this section is to define the items below for SIDD NITF 2.1 products.

- Capabilities and limitations of NITF 2.1 file container for SIDD products
- SIDD NITF 2.1 file container metadata
- Placement of SIDD product metadata in NITF 2.1 container
- Placement of SIDD product image data in NITF 2.1 container
- SIDD NITF 2.1 file container organization

2.1 Capabilities and Limitations

The NITF 2.1 container file format is capable of handling a wide array of possible SIDD products ranging from a single image product generated from a single SICD input to multiple product images generated from multiple SICD inputs. Additionally, the NITF 2.1 file container supports very large multi-image products. NITF 2.1 products use big-endian format.

2.1.1 Future Considerations

A number of potential updates may be incorporated into this specification for SIDD NITF 2.1 products as the need arises to support additional functionality. The potential updates include but are not limited to JPEG2000 compression, annotations, and blocking of image data.

2.2 SIDD NITF 2.1 File Container Metadata

The NITF 2.1 file container format requires a single NITF 2.1 File Header that describes the file layout and high level information for the file. A NITF 2.1 Image Sub-header must be present for each product image that describes the image segment the sub-header is attached to. Additionally, a SIDD NITF 2.1 product requires that a SIDD XML DES (Data Extension Segment) is present for each product image. Furthermore, if a SICD was used as an input, then its XML DES is required in the SIDD NITF 2.1. The following sections provide population instructions for each of the container metadata blocks.

2.2.1 SIDD NITF 2.1 File Header Description

The purpose of this section is to define the population instructions for each field of the NITF 2.1 file header for SIDD NITF 2.1 products. A SIDD NITF 2.1 product contains one NITF 2.1 file header. The information is in Table 2.2-1.

Table 2.2-1 SIDD NITF 2.1 File Header Definition

Field	Name/Description	Size (bytes)	Value Range	Type	Comment
FHDR	File Profile Name	4	BCS-A “NITF"	R	Per MIL-STD-2500c
FVER	File Version	5	BCS-A “02.10”	R	NITF Version 2.1
CLEVEL	Complexity Level	2	BCS-A 03, 05, 06, 07 or 09	R	Per MIL-STD-2500c Table A-10
STYPE	Standard Type	4	BCS-N “BF01”	R	Fixed value
OSTAID	Originating Station ID	10	BCS-A This field shall contain a meaningful value; it shall not be filled with BCS spaces (0x20)	R	Per the Product Specific Implementation Document

Table 2.2-1 SIDD NITF 2.1 File Header Definition

Field	Name/Description	Size (bytes)	Value Range	Type	Comment
FDT	File Date and Time	14	BCS-N CCYYMMDDhhmmss	R	File creation UTC date and time
FTITLE	File Title	80	ECS-A "SIDD: <ProductName>"	<R>	"SIDD: <ProductName>" where ProductName comes from *SIDD.ProductCreation.ProductName.* After SIDD colon space fill in with the 1st 74 characters of ProductName
Security Tags	Use "FS" prefix for Tag	167	As defined in Table 2.2-2	R	
FSCOP	File Copy Number	5	BCS-N 00000 = No tracking of numbered file copies	R	Not tracked Per MIL-STD-2500c
FSCPYS	File Number of Copies	5	BCS-N 00000 = No tracking of numbered file copies	R	Not tracked Per MIL-STD-2500c
ENCRYP	Encryption	1	BCS-N 0	R	No Encryption
FBKGC	File Background Color	3	Unsigned binary integer Default: 000 (0x00, 0x00, 0x00)	R	Default background color is black
ONAME	Originator's Name	24	ECS-A 24 ECS characters	<R>	Per the Product Specific Implementation Document
OPHONE	Originator's Phone	18	ECS-A 24 ECS characters	<R>	Per the Product Specific Implementation Document
FL	File Length	12	BCS-N Generate	R	Number of bytes
HL	NITF File Header Length	6	BCS-N Generate	R	Number of bytes in the header
NUMI	Number of Image Segments	3	BCS-N 001 to 999	R	See Sections 2.4.2 and 2.4.3
LISHn and LIn repeat as pairs as many times as specified in the NUMI field					
LISHn	Length of Image Sub-Header. This field shall occur as many times as specified in the NUMI field. Note: The largest image sub-header is limited to 999998 (10**6 -2) bytes.	6	BCS-N 000439 to 9999998 Generate	C	Per MIL-STD-2500c
LIn	Length of nth Image Segment. This field shall occur as many times as specified in the NUMI field. Note: The largest image is limited to	10	BCS-N 0000000001 to 9999999998 Generate	C	See Sections 2.4.2 and 2.4.3

Table 2.2-1 SIDD NITF 2.1 File Header Definition					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
	9,999,999,998 (10**10 -2) bytes.				
End of LISH and LIn field repetition					
NUMS	Number of graphic Segments	3	BCS-N 000	R	Not used default to 000
NUMX	Reserved	3	BCS-N 000	R	Reserved
NUMT	Number of Text Segments	3	BCS-N 000 to 999	R	Text segments used for copyright info if needed
NUMDES	Number of data Extension Segments	3	BCS-N 001 to 999	R	# product images + # input SICDs
LDSHn and LDn repeat as pairs as many times as specified in the NUMDES field					
LDSHn	Length of nth Data Extension Segment Sub-Header	4	BCS-N 0200 to 9998	C	Length in bytes
LDn	Length of Data Extension Segment	9	BCS-N 000000001 to 999999998	C	Length in bytes
End of LDSH and LDn field repetition					
NUMRES	Number of Reserved	3	BCS-N 000	R	Not used
UDHDL	User-Defined Header Data Length	5	BCS-N 00000	R	No TREs allowed.
XHDL	Extended Header Data Length	5	BCS-N 00000	R	No TREs allowed.
Type – R = Required, C = Conditional, <> surrounding the type = BCS SPACES ALLOWED FOR ENTIRE FIELD					

Table 2.2-2 NITF 2.1 Security Fields				
Base Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**
xxCLAS	File Security Classification. This field shall contain a valid value representing the classification level of the entire file. Valid values are T (=Top Secret), S (=Secret), C (=Confidential), R (=Restricted), U (=Unclassified).	1	ECS-A "U" or Per the Program Specific Implementation Document	R
xxCLSY	File Security Classification System. This field shall contain valid values indicating the national or multinational security system used to classify the file. Country Codes per FIPS PUB 10-4 shall be used to indicate national security systems. The designator "XN" is for classified data generated by a component using NATO security system marking guidance. This code	2	ECS-A (Default is ECS spaces (0x20)) Per the Program Specific Implementation Document	<R>

Table 2.2-2 NITF 2.1 Security Fields				
Base Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**
	is outside the FIPS 10-4 document listing, and was selected to not duplicate that document's existing codes. If this field is all ECS spaces (0x20), it shall imply that no security classification system applies to the file.			
xxCODE	File Codewords. This field shall contain a valid indicator of the security compartments associated with the file. Values include one or more of the digraphs found in MIL-STD-2500c, table A-4. Multiple entries shall be separated by a single ECS space (0x20): The selection of a relevant set of codewords is application specific. If this field is all ECS spaces (0x20), it shall imply that no codewords apply to the file.	11	BCS-A (Default is BCS spaces (0x20)) Per the Program Specific Implementation Document	<R>
xxCTLH	File Control and Handling. This field shall contain valid additional security control and/or handling instructions (caveats) associated with the file. Values include digraphs found in MIL-STD-2500c, table A-4. The digraph may indicate single or multiple caveats. The selection of a relevant caveat(s) is application specific. If this field is all ECS spaces (0x20), it shall imply that no additional control and handling instructions apply to the file.	2	ECS-A (Default is ECS spaces (0x20)) Per the Program Specific Implementation Document	<R>
xxREL	File Releasing Instructions. This field shall contain a valid list of country and/or multilateral entity codes to which countries and/or multilateral entities the file is authorized for release. Valid items in the list are one or more country codes as found in FIPS PUB 10-4 separated by a single ECS space (0x20). If this field is all ECS spaces (0x20), it shall imply that no file release instructions apply.	20	ECS-A (Default is ECS spaces (0x20)) Per the Program Specific Implementation Document	<R>
xxDCTP	File Declassification Type. This field shall contain a valid indicator of the type of security declassification or downgrading instructions which apply to the file. Valid values are DD (=declassify on a specific date), DE (=declassify upon occurrence of an event), GD (=downgrade to a specified level on a specific date), GE (=downgrade to a specified level upon occurrence of an event), O (=OADR), and X (= exempt from automatic declassification). If this field is all ECS spaces (0x20), it shall imply that no file security declassification or downgrading instructions apply.	2	ECS-A (Default is ECS spaces (0x20)) Valid Values DD, DE, GD, GE, O, X Per the Program Specific Implementation Document	<R>
xxDCDT	File Declassification Date. This field shall indicate the date on which a file is to be declassified if the value in File Declassification Type is DD. If this field is all ECS spaces (0x20), it shall imply that no file	8	ECS-A (Default is ECS spaces (0x20)) Valid Format	<R>

Table 2.2-2 NITF 2.1 Security Fields				
Base Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**
	declassification date applies.		CCYYMMDD Per the Program Specific Implementation Document	
xxDCXM	File Declassification Exemption. This field shall indicate the reason the file is exempt from automatic declassification if the value in File Declassification Type is X. Valid values are X1 to X8 and X251 to X259. X1 to X8 correspond to the declassification exemptions found in DOD 5200.1-R, paragraphs 4- 202b(1) to (8) for material exempt from the 10-year rule. X251 to X259 correspond to the declassification exemptions found in DOD 5200.1-R, paragraphs 4- 301a(1) to (9) for permanently valuable material exempt from the 25-year declassification system. If this field is all ESC spaces (0x20), it shall imply that a file declassification exemption does not apply.	4	ECS-A (Default is ECS spaces (0x20)) Valid values X1 to X8, X251 to X259 Per the Program Specific Implementation Document	<R>
xxDG	File Downgrade. This field shall indicate the classification level to which a file is to be downgraded if the values in File Declassification Type are GD or GE. Valid values are S (=Secret), C (=Confidential), R (= Restricted). If this field contains an ECS space (0x20), it shall imply that file security downgrading does not apply.	1	ECS-A (Default is ECS space (0x20)) Valid Values S, C, R Per the Program Specific Implementation Document	<R>
xxDGDT	File Downgrade Date. This field shall indicate the date on which a file is to be downgraded if the value in File Declassification Type is GD. If this field is all ECS spaces (0x20), it shall imply that a file security downgrading date does not apply.	8	ECS-A (Default is ECS spaces (0x20)) Valid Format CCYYMMDD Per the Program Specific Implementation Document	<R>
xxCLTX	File Classification Text. This field shall be used to provide additional information about file classification to include identification of declassification or downgrading event if the values in File Declassification Type are DE or GE. It may also be used to identify multiple classification sources and/or any other special handling rules. Values are user defined free text. If this field is all ECS spaces (0x20), it shall imply that additional information about file classification does not apply.	43	ECS-A (Default is ECS spaces (0x20)) Per the Program Specific Implementation Document	<R>
xxCATP	File Classification Authority Type. This field shall indicate the type of authority used to classify the file. Valid values are O (= original classification authority), D (= derivative from a single source), and M (= derivative from multiple sources). If this field contains an ECS space (0x20), it shall imply that file classification authority type does not apply.	1	ECS-A (Default is ECS space (0x20)) Per the Program Specific Implementation Document	<R>

Table 2.2-2 NITF 2.1 Security Fields				
Base Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**
xxCAUT	File Classification Authority. This field shall identify the classification authority for the file dependent upon the value in File Classification Authority Type. Values are user defined free text which should contain the following information: original classification authority name and position or personal identifier if the value in File Classification Authority Type is O; title of the document or security classification guide used to classify the file if the value in File Classification Authority Type is D; and Derive-Multiple if the file classification was derived from multiple sources and the value of the FSCATP field is M. In the latter case, the file originator will maintain a record of the sources used in accordance with existing security directives. One of the multiple sources may also be identified in File Classification Text if desired. If this field is all ECS spaces (0x20), it shall imply that no file classification authority applies.	40	ECS-A (Default is ECS spaces (0x20)) Per the Program Specific Implementation Document	<R>
xxCRSN	File Classification Reason. This field shall contain values indicating the reason for classifying the file. Valid values are A to G. These correspond to the reasons for original classification per E.O. 12958, Section 1.5.(a) to (g). If this field contains an ECS space (0x20), it shall imply that no file classification reason applies.	1	ECS-A (Default is ECS space (0x20)) Valid Values A to G Per the Program Specific Implementation Document	<R>
xxSRDT	File Security Source Date. This field shall indicate the date of the source used to derive the classification of the file. In the case of multiple sources, the date of the most recent source shall be used. If this field is all ECS spaces (0x20), it shall imply that a file security source date does not apply.	8	ECS-A (Default is ECS spaces (0x20)) Valid Format CCYYMMDD Per the Program Specific Implementation Document	<R>
xxCTLN	File Security Control Number. This field shall contain a valid security control number associated with the file. The format of the security control number shall be in accordance with the regulations governing the appropriate security channel(s). If this field is all ECS spaces (0x20), it shall imply that no file security control number applies.	15	ECS-A (Default is ECS spaces (0x20)) Per the Program Specific Implementation Document	<R>
For the File Header replace the xx prefix with FS (e.g., FSCLAS) For the File Sub-Header replace the xx prefix with IS (e.g., ISCLAS) For the DES replace the xx prefix with DES (e.g., DESCLAS)				
Refer to MIL-STD-2500C Table A-1 for File Header and Table A-3 for Image Sub-Header for general description of all fields. Consult current security guidelines at time of production for proper entries.				
Type – R = REQUIRED, C = CONDITIONAL, <> surrounding the type = BCS SPACES ALLOWED FOR ENTIRE FIELD				

2.2.2 SIDD NITF 2.1 Image Sub-header Description

The purpose of this section is to define the population instructions for each field of the NITF 2.1 image sub-header for SIDD NITF 2.1 products. A NITF 2.1 image sub-header is present for each image segment in the file. The information is in Table 2.2-3.

Table 2.2-3 SIDD NITF 2.1 Image Sub-Header Definition

Field	Name/Description	Size (bytes)	Value Range	Type	Comment
IM	File Part Type	2	BCS-A IM	R	
IID1	Image Identifier 1	10	BCS-A SIDDmmmnnn	R	mmm correspond to product image number, nnn correspond to segment of image, starting at 001 for both mmm and nnn. See Section 2.4.2.
IDATIM	Image Date and Time	14	BCS-N CCYYMMDDhhmmss	R	Populated equivalent to SIDD XML – 1st instance of *SIDD.AdvancedExploitation.Collection.Information.CollectionDateTime*
TGTID	Target Identifier	17	BCS-A	<R>	Populated equivalent to SIDD XML – 1st instance of *SIDD.GeographicAndTarget.TargetInformation.Identifier* (if available)
IID2	Image Identifier 2	80	"SIDD: <ProductName>"	<R>	"SIDD: <ProductName>" where ProductName comes from *SIDD.ProductCreation.ProductName.* Fill with spaces at end as needed.
Security Tags	Use "IS" prefix for Tag	167	As defined in Table 2.2-2	R	
ENCRYP	Encryption	1	BCS-N positive integer 0	R	0 = not encrypted
ISORCE	Image Source	42	ECS-A Collector Name	<R>	Populate equivalent to SIDD XML – 1st instance of *SIDD.AdvancedExploitation.Collection.Information.SensorName*
NROWS	Number of Significant Rows in Image	8	BCS-N positive integer 00000001 to 9999999	R	See Section 2.4.2.
NCOLS	Number of Significant Columns in Image	8	BCS-N positive integer 00000001 to 9999999	R	See Section 2.4.2.
PVTYPE	Pixel Value Type	3	BCS-A INT	R	
IREP	Image Representation	8	BCS-A MONO, RGB/LUT, RGB	R	See Section 2.4.1.
ICAT	Image Category	8	BCS-A SAR, LEG	R	SAR for Synthetic Aperture Radar, LEG for Legend
ABPP	Actual Bits-Per-Pixel	2	BCS-N positive integer	R	8 unless using

Table 2.2-3 SIDD NITF 2.1 Image Sub-Header Definition					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
	Per Band		08 or 16		MONO16I
PJUST	Pixel Justification	1	BCS-A R	R	
ICORDS	Image Coordinate Representation	1	BCS-A G or ‘ ‘	<R>	G for geographic, Blank for legend segments
IGEOLO	Image Geographic Location	60	BCS-A ddmmssXdddmmssY (four times)	C	If ICORDS = G, dd(d) = degree mm = minute ss = seconds X = North or South Y = East or West. If ICORDS is blank omit this field
NICOM	Number of Image Comments	1	BCS-N positive integer 0 to 9	R	
NICOMn	Image Comment n	80	ECS-A User defined	C	
IC	Image Compression	2	BCS-A NC	R	NC for No Compression
NBANDS	Number of Bands	1	BCS-N positive integer 1,3	R	See Section 2.4.1
IREPBANDn, ISUBCATn, IFCn, IMFLTn, NLUTSn, and NELUTn repeat as a set as many times as specified in the NBANDS field					
IREPBANDn	nth Band Representation	2	BCS-A ‘LU’,’M ‘,’R’,’G’, or ‘B’	<R>	See Section 2.4.1.
ISUBCATn	nth Band Subcategory	6	BCS-A “ “ (spaces)	<R>	
IFCn	nth Band Image Filter Condition	1	BCS-A N	R	
IMFLTn	nth Band Standard Image Filter Code	3	BCS-A “ “	<R>	3 spaces
NLUTSn	Number of LUTS for the nth Image Band	1	BCS-N positive integer 0,1,2,3	R	See Table 2.4-1.
NELUTn	Number of LUT Entries for the n^{th} Image Band	5	BCS-N positive integer 00001 to 65536	C	Number of entries in each of the LUTs for n^{th} image band. Omitted if NLUTSn is 0
LUTDnm repeats as many times as specified in the NLUTSn field					
LUTDnm	n^{th} Image Band, m^{th} LUT	NELUTn	Unsigned binary integer. LUT Values	C	This field shall be omitted if the Number of LUTs (NLUTSn) is BCS zero (0x30). Otherwise, this field shall contain the data defining the mth LUT for the nth image band. Each entry in the LUT is composed of one byte, ordered from MSB to LSB, representing a binary value from zero (0x00) to 255 (0xFF). To use the LUT, for each integer k, 0 ≤ k ≤ (value of the NELUTn field) -1, the pixel value k in the nth image band shall be mapped to the value of

Table 2.2-3 SIDD NITF 2.1 Image Sub-Header Definition					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
					the kth byte of this field (the LUT).
End of LUTDnm field repetition					
End of IREPBANDn, ISUBCATn, IFCn, IMFLTn, NLUTSn, and NELUTn field repetition					
ISYNC	Image Sync code	1	BCS-N positive integer 0	R	
IMODE	Image Mode	1	BCS-A 'B' = Band Interleaved by Block, 'P' = Band Interleaved by Pixel	R	If IREP=RGB, 'P'. 'B' otherwise. See Table 2.4-1.
NBPR	Number of Blocks Per Row	4	BCS-N positive integer 0001	R	
NBPC	Number of Blocks Per Column	4	BCS-N positive integer 0001	R	
NPPBH	Number of Pixels Per Block Horizontal	4	BCS-N positive integer 0001-8192 or 0000	R	If number of pixels in horizontal directional is more than 8192 populate with 0000 otherwise populate with zero padded version of number of columns
NPPBV	Number of Pixels Per Block Vertical	4	BCS-N positive integer 0001-8192 or 0000	R	If number of pixels in vertical directional is more than 8192 populate with 0000 otherwise populate with zero padded version of number of rows
NBPP	Number of Bits Per Pixel Per Band	2	BCS-N positive integer 08 or 16	R	08 unless using MONO16I
IDLVL	Image Display Level. This field shall contain a valid value that indicates the display level of the image relative to other displayed file components in a composite display.	3	BCS-N positive integer. 001 to 999	R	See Segmentation Rules – Section 2.4.1
IALVL	Attachment Level	3	BCS-N positive integer	R	See Segmentation Rules – Section 2.4.1
ILOC	Image Location. The image location is the location of the first pixel of the first line of the image. This field shall contain the image location offset from the ILOC or SLOC value of the segment to which the image is attached or from the origin of the CCS when the image	10	BCS-N RRRRRCCCCC For positive row and column values RRRRR and CCCCC are both in the range 00000 to 99999. For negative row and column values RRRRR and CCCCC are both in the range	R	See Segmentation Rules – Section 2.4.1

Table 2.2-3 SIDD NITF 2.1 Image Sub-Header Definition					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
	is unattached (IALVL contains 000).		-0001 to -9999		
IMAG	Image Magnification	4	BCS-A	R	Default is 1.0
UDIDL	User Defined Image Data Length	5	BCS-N positive integer 00000	R	No TREs allowed
IXSHDL	Image Extended Sub-Header Data Length	5	BCS-N positive integer 00000	R	No TREs allowed
Type – R = Required, C = Conditional, <> surrounding the type = BCS SPACES ALLOWED FOR ENTIRE FIELD					

2.2.3 SIDD XML DES Description

The purpose of this section is to define the population instructions for each field of the SIDD XML DES for SIDD NITF 2.1 products. The information is in Table 2.2-4. Additional information related to the number of SIDD XML DESs per file and the DESDATA field of the SIDD XML DES is provided in section 2.3.1.

Table 2.2-4 SIDD XML DES Description					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
DE	File Part Type	2	"DE"	R	
DESID	Unique DES Type Identifier. This field shall contain a valid alphanumeric identifier properly registered with the ISMC	25	"SIDD_XML"	R	
DESVER	Version of the Data Definition	2	01	R	
Security Tags	Use "DES" prefix for Tag	167	As defined in Table 2.2-2	R	
DESSHL	DES User-defined Subheader Length	4	0000	R	
DESDATA	DES User-Defined Data	Determined by User	<SIDD XML Content Body>	R	See Section 2.3.1
Type – R = Required, C = Conditional, <> surrounding the type = BCS SPACES ALLOWED FOR ENTIRE FIELD					

2.2.4 SICD XML DES Description

The purpose of this section is to define the population instructions for each field of the SICD XML DES for SIDD NITF 2.1 products. The information is in Table 2.2-5. Additional information related to the number of SICD XML DESs per file and the DESDATA field of the SICD XML DES is provided in section 2.3.2

Table 2.2-5 SICD XML DES Description					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
DE	File Part Type	2	"DE"	R	
DESID	Unique DES Type Identifier. This field shall contain a valid alphanumeric identifier properly registered with the ISMC	25	"SICD_XML"	R	
DESVER	Version of the Data Definition	2	01	R	

Table 2.2-5 SICD XML DES Description					
Field	**Name/Description**	**Size (bytes)**	**Value Range**	**Type**	**Comment**
Security Tags	Use “DES” prefix for Tag	167	As defined in Table 2.2-2	R	
DESSHL	DES User-defined Subheader Length	4	0000	R	
DESDATA	DES User-Defined Data	Determined by User	<SICD XML Content Body>	R	See Section 2.3.2
Type – R = Required, C = Conditional, <> surrounding the type = BCS SPACES ALLOWED FOR ENTIRE FIELD					

2.3 SIDD NITF 2.1 SIDD Product Metadata

The SIDD product metadata consists of metadata related to the SIDD image products and the input image products used to generate the SIDD image products that is independent of file container format. The following two sections provide information about this metadata.

2.3.1 Image Product Metadata

Each image product contains a unique SIDD XML DES; section 2.2.3 provides the format of the SIDD XML DES. The DESDATA portion of the SIDD XML DES Description is specified in the *Sensor Independent Derived Data (SIDD) Design & Exploitation Description Document* (See Table 1.2-1).

A SIDD XML DES is associated with a specific SIDD image product by the order in which they are placed in the container. The first DES in the NITF product corresponds to the first image product. A SIDD XML DES is linked to an image in the container by the *IID1* field in the NITF Image Sub-header. The mth DES is associated with the image product that has the *IID1* field in the NITF Image Sub-header having the format SIDD[mmm]001, where [mmm] is the zero padded version of the nth DES. For example the 7th DES in the NITF container refers to the image with NITF Image Sub-Header populated with *IID1* = SIDD007001. The last three digits in the *IID1* field are utilized for segmentation as described in section 2.4.2.

The convention described requires that the *IDLVL* fields be populated starting at one and incremented by one. A SIDD XML DES is required for each product image in the NITF 2.1 container. If a product image is segmented each segment points to the same XML DES; this procedure is described in section 2.4.2.

2.3.2 Input Image Metadata

When possible, the SICD XML DES instances are provided as reference for downstream processing. Therefore, for each SICD input image, a separate DES is required that contains the original, unaltered SICD XML instance in the input file. If the input to the SIDD product is not SICD, then a SICD XML DES will not be present. The ordering of the SICD DESs is unspecified but could be specified by an implementation specification. Section 2.2.4 details the format of the SICD DES. The DESDATA portion of the SICD XML DES Description is provided in the *Sensor Independent Complex Data (SICD) Design & Exploitation Description Document* (See Table 1.2-1).

The SICD XML DESs directly follow the SIDD XML DESs with no empty DESs allowed.

2.4 SIDD Product Image Pixel Data

The purpose of this section is to specify the supported pixel types for SIDD NITF 2.1 products and the segmentation procedure for large image products.

2.4.1 Supported Pixel Types

The SIDD image pixel data may be stored in the NITF 2.1 container in one of the formats below as specified by the field *SIDD.Display.PixelType* in the SIDD XML for that image.

- MONO8I: Each pixel is stored as an 8-bit unsigned integer. The data is stored in a single channel.
- MONO8LU: Each pixel is stored as an 8-bit unsigned integer. A look-up table is used to decode the 256 values for monochromatic display. The 8-bit pixel data can be transformed to a space utilizing more than 8 bits. The data is stored in a single channel.
- MONO16I: Each pixel is stored as two 8-bit unsigned integers. The data is stored in a single channel.
- RGB8LU: Each pixel is stored as an 8-bit unsigned integer. A look-up table is used to decode the 256 values for RGB display. The data is stored in a single channel.
- RGB24I: Each pixel is stored as set of three 8-bit unsigned integers representing the Red-Green-Blue color components, respectively. The data is stored in three channels.

The SIDD NITF 2.1 container stores the pixels in a raster format, where the first pixel is associated with the visual upper left corner of the product image. For 24-bit RGB pixels the pixel is organized into three 8-bit blocks ordered red, green, and then blue. For 16-bit monochrome pixel follows big-endian convention.

The table below provides population instructions for NITF 2.1 image sub-header fields related to supporting the five pixel types.

Table 2.4-1 NITF 2.1 Image Sub-Header Population for Supported Pixel Type

NITF 2.1 Image Sub-Header Field Name	MONO8I	MONO8LU	MONO16I	RGB8LU	RGB24I
IREP	Mono	Mono	Mono	RGB/LUT	RGB
NBANDS	1	1	2	1	3
IREPBANDn	n =1 ➔M	n = 1 ➔ LU	n = 1 ➔ M n = 2 ➔ M	n = 1 ➔ LU	n = 1 ➔ R n = 2 ➔ G n = 3 ➔ B
NLUTSn	n = 1 ➔ 0	n = 1 ➔ 1 or 2	n = 1 ➔ 0 n = 2 ➔ 0	n = 1 ➔3	n = 1 ➔ 1 n = 2 ➔ 1 n = 3 ➔ 1
IMODE	B	B	B	B	P

2.4.2 Segmentation

The NITF 2.1 container can support large images; however, certain limitations built into the NITF specification require that these large images be segmented. A large image must be segmented once the raw image pixel data exceeds ~9.3 GB. Once the 9.3 GB limit has been exceeded each segment cannot contain more than 99,999 rows because of NITF limitations.

The NITF 2.1 file container supports large images using the common coordinate system (CCS). CCS is used to create a virtual window that is not limited in size. Each segment of a large image is placed into the CCS to create a seamless image when properly ingested by a downstream application.

When an image exceeds the NITF image segment size limitation (greater than 9.3 GB), the image must be split into multiple segments linked together by attachment levels, described below, and a defined relationship with another segment of the same image. Below are the two constraints on NITF 2.1 image segment size.

- $\boldsymbol{LI_{MAX}}$: The LI field in the NITF 2.1 file header can only accommodate an image segment with 9,999,999,998 bytes of data (~9.3 GB) because the field is limited to 10 digits
- $\boldsymbol{ILOC_{MAX}}$: The ILOC field provides the row offset and column offset, each represented with 5 digits, from the image segment (IDLVL) that a segment is attached to. Therefore, the maximum number of rows contained in a segment is 99,999. The ILOC restriction is only enforced once the LI_{MAX} threshold has been surpassed.

Proper CCS usage requires that the following information in the NITF 2.1 file container metadata be populated correctly:

- NITF 2.1 Image Sub-Header
  - IDLVL
  - IALVL
  - ILOC
  - IID1
- NITF 2.1 File Header
  - NUMI
  - LI*(z)*

The segmentation algorithm is provided in the next section.

2.4.2.1 Segmentation Algorithm

The segmentation algorithm divides the image on row boundaries such that the column offset is always zero. Pseudo-code shows how a SIDD product with M product images is segmented and how the associated NITF container fields are populated. Variables that contain ***FHDR_*** are related to the NITF 2.1 file header and fields with ***IMHDR(z)_*** are related to the z^{th} NITF 2.1

Image Sub-Header. The rows and columns of the M^th SIDD product images are defined as `NumCols(m)` and `NumRows(m)`.

Constants:

```
LI_MAX = 9,999,999,998
ILOC_MAX = 99,999
```

Number of Image Segments & Image Segment Size Parameters

```
z = 0; (Used to increment the image segments in the NITF 2.1 file
container.)
FHDR_NUMI = 0

For(k = 1,…,M) [Loop over product images]

      BytesPerPixel = IMHDR(z)_NBANDS
      BytesPerRow = BytesPerPixel*NumRows(k)
      NumRowsLimit(k) = min(floor(LI_max/BytesPerRow), ILOC_MAX)
      ProductSize = BytesPerPixel*NumRows(k)*NumCols(k)
      If(ProductSize ≤ LI_MAX) then,
            z =z+1
            FHDR_NUMI = FHDR_NUMI+1
            FHDR_LI(z) = ProductSize
            IMHDR(z)_IDLVL = z
            IMHDR(z)_IALVL = 0
            IMHDR(z)_ILOC = 0000000000
            IMHDR(z)_IID1 = "SIDD[mmm]001" where mmm is zero padded m
            IMHDR(z)_NROWS = NumRowsLimit(k)
            IMHDR(Z)_NCOLS = NumCols(k)
      Else
            NumSegPerImage(k) = NumRows(k)/NumRowsLimit(k)
            z = z+1
            FHDR_NUMI = FHDR_NUMI  + NumSegPerImage(k)
            FHDR_LI(z)= BytesPerPixel*NumRowsLimit(k)*NumCols(k)
            IMHDR(z)_IDLVL = z
            IMHDR(z)_IALVL = 0
            IMHDR(z)_ILOC = 0000000000
            IMHDR(z)_IID1 = "SIDD[mmm]001" where mmm is zero padded m
            IMHDR(Z)_NCOLS = NumCols(k)

            For(n=2,…,NumSegPerImage(k)-1)
                  z=z+1
                  FHDR_LI(z)= BytesPerPixel*NumRowsLimit(k)*NumCols(k)
                  IMHDR(z)_IDLVL=z
                  IMHDR(z)_IALVL=z-1
                  IMHDR(z)_ILOC = [RRRRR00000] →RRRRR= zero padded
         NumRowsLimit(k)
                  IMHDR(z)_IID1 = "SIDD[mmm][nnn]" (zero padded versions
            of numbers)
                  IMHDR(z)_NROWS = NumRowsLimit(k)
```

```
                    IMHDR(Z)_NCOLS = NumCols(k)
              End
              z=z+1
              lastSegRows=NumRows(k)-(NumSegPerImage(k)-
            1)*NumRowsLimit(k)
              FHDR_LI(z)=BytesPerPixel*lastSegRows*NumCols(k)
              IMHDR(z)_IDLVL = z
              IMHDR(z)_IALVL = z-1
              IMHDR(z)_ILOC = [RRRRR00000] →RRRRR=zero padded lastSegRows
              IMHDR(z)_IID1 = "SIDD[kkk][nnn]" (zero padded versions of
            numbers)
              IMHDR(z)_NROWS = lastSegRows
              IMHDR(z)_NCOLS = NumCols(k)
      End
End
```

The first 3 digits in the *IID1* field refer to the DES corresponding with the image product. Two graphical examples are provided below to show how a very large multiple product image requiring segmentation would look from a graphic and metadata stance.

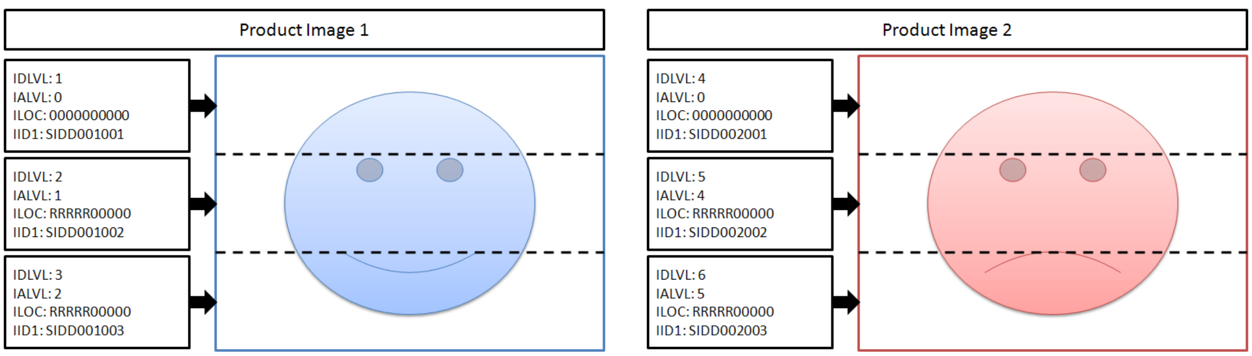

Figure 2.4-1 Example NITF 2.1 Image Segmentation for Multiple Image Product [RRRRR is defined as the zero padded version of NumRowsLimit(k)]

Image Segment Corner Coordinate Parameters

The four geodetic corners of the product image are located in the *SIDD.GeographicAndTarget.GeographicCoverage.Footprint* parameter of the SIDD XML. The four corners of the product image are defined below in the matrix `PCC(n,m)` where the n index refers to corner number and the m index refers to latitude if 1 and longitude if 2. The values 1, 2, 3, and 4 of the first index refer to the upper left, upper right, lower right, and lower left corners of the product image respectively.

```
PCC(1,1) = …Footprint(1).Lat      PCC(1,2) = …Footprint(1).Lon
PCC(2,1) = …Footprint(2).Lat      PCC(2,2) = …Footprint(2).Lon
PCC(3,1) = …Footprint(3).Lat      PCC(3,2) = …Footprint(3).Lon
PCC(4,1) = …Footprint(4).Lat      PCC(4,2) = …Footprint(4).Lon
```

For a product image that fits into a single Image Segment the corner coordinates of the image segment, `ISCC(z,n,m)` are defined below. The three indices for `ISCC` correspond to image segment number, corner number, and latitude/longitude. The indices n and m are defined the same for `ISCC` as they are for `PCC`.

```
ISCC(1,1,1) = PCC(1,1)          ISCC(1,1,2) = PCC(1,2)
ISCC(1,2,1) = PCC(2,1)          ISCC(1,2,2) = PCC(2,2)
ISCC(1,3,1) = PCC(3,1)          ISCC(1,3,2) = PCC(3,2)
ISCC(1,4,1) = PCC(4,1)          ISCC(1,4,2) = PCC(4,2)
```

If the product image requires segmentation to fit into the NITF file container the corners of each segment are determined by converting the corner coordinates to ECF and then interpolating to find the ECF position of each segment and then converting the new interpolated image segment corner coordinates back to LLA.

```
For(k = 1,…,M) [Loop over product images]

    For(z = 1,…,4)[ Convert each LLA to ECF for that product image ]

        Convert PCC (in LLA) to PCC_ECF(in ECF)

    End

    For(z = 1,…,NumSegPerImage(k) (defined above))

        wgt1 = ((z-1)*NumRowsLimit(k))/NumRows(k)
        wgt2 = 1-wgt1
        wgt3 = ((z-1)*NumRowsLimit(k)+IMHDR(z)_NROWS)/NumRows(k)
        wgt4 = 1-wgt3

        ISCC_ECF(z,1) = wgt2*PCC_ECF(1)+wgt1*PCC_ECF(4)
        ISCC_ECF(z,2) = wgt2*PCC_ECF(2)+wgt1*PCC_ECF(3)
        ISCC_ECF(z,3) = wgt4*PCC_ECF(2)+wgt3*PCC_ECF(3)
        ISCC_ECF(z,4) = wgt4*PCC_ECF(1)+wgt3*PCC_ECF(4)

        For(n = 1,…,4)

            Convert PCC_ECF(z,n) (in ECF) to ISCC(z,n) (in LLA)

        End
        IMHDR(z)_IGEOLO = [ISCC(z,1,1) ISCC(z,1,2) ISCC(z,2,1)
        ISCC(z,2,2) ISCC(z,3,1) ISCC(z,3,2) ISCC(z,4,1)
        ISCC(z,4,2)]

    End
End
```

2.4.3 Legends

The SIDD NITF 2.1 file container supports the usage of legends. A legend image in the NITF 2.1 file container requires that the *ICAT* field be populated with *LEG* instead of the standard *SAR*. The SIDD NITF 2.1 requires that any legend image segment be after the final *SAR* segment for that SIDD product image. Additionally, a legend image, which is not considered a product image, cannot exceed the 9.3 GB image segment constraint. Thus, if a SIDD product image required segmentation into N parts the legend(s) would start at N+1. The following are population instructions for legend images. Legend segments are not associated with any SIDD XML DES.

Table 2.4-2 SIDD NITF 2.1 File Container Metadata Population Instructions for Legends		
File Header/Image Sub-Header	**Field**	**Instruction**
Image Sub-Header	ICAT	Populate with “LEG”
Image Sub-Header	IID1	“SIDDmmmnnn” where mmm is the SIDD product image number associated with the legend and nnn is the number of image segments that the SIDD product image required+ 1 (or more for multiple legends). The first legend segment (ICAT = LEG) must have a larger nnn value than the highest valued image segment (ICAT = SAR) for that SIDD product image.
Image Sub-Header	IDLVL	The IDLVL of a legend segment corresponding to a SIDD product image must be higher than the highest SAR segment. In a multiple product image SIDD, the IDLVL value for a legend image related to the m^{th} product image must be higher than the highest SAR segment for the m^{th} product image but less than the lowest SAR segment of the m+1 product image.
Image Sub-Header	IALVL	The legend must be attached to the image segment of the SIDD product image where it’s display is desired. For example, if the legend is meant to be in the top right of the SIDD product image the attachment level should reference the IDLVL level of the first segment of the SIDD product image. If the desire is to place the legend in the lower right then the attachment level should be equal to the IDLVL level of the last segment of the SIDD product image.
Image Sub-Header	ILOC	The row and column offsets are specified in this field are relative to the upper left corner (0,0) of the image segment that the legend is being attached to. For example, if the desire is to place legend in the upper left this field would be (0,0)→0000000000. Likewise if the desire is to place it near the lower right corner of the segment that it is being attached to then this field would be (NumRows-x,NumCols-y).
File Header	NUMI	The procedure outlined in section 2.4.2 should be followed but for each legend added NUMI should be incremented by 1.
File Header	LIn	The procedure outlined in section 2.4.2 for calculating an image segment size should be used for legend segments

2.5 SIDD NITF 2.1 File Container Organization

The purpose of this section is to describe the NITF 2.1 file container organization for SIDD NITF 2.1 products. The organizational description of a SIDD NITF 2.1 product specifies the location and relationship between five components:

- NITF 2.1 file header
- NITF 2.1 image sub-header
- SIDD product images
- SIDD XML DESs
- SICD XML DESs

SIDD products in the NITF 2.1 file container format use the same basic organization as that of a standard NITF 2.1 with the additional requirement that a SIDD NITF 2.1 product must follow a specific organizational layout. The organizational structure outlined below starts with the NITF 2.1 File Header and ends with the SICD XML DES (Data Extension Segment). The DESs must start at 1 and no empty DESs are allowed. The basic outline is based on a SIDD NITF product containing N product images generated from M SICD inputs.

- NITF 2.1 File Header
- Repeat for N SIDD product images
  - NITF 2.1 Image Sub-Header
  - SIDD Product Image (Raster format, starting at visual upper left corner of image)
- Repeat for N SIDD product images
  - SIDD XML DES
- Repeat for M SICD input images
  - SICD XML DES (if available)

The next five sections provide examples of the organization of SIDD NITF 2.1 products that are intended for clarifying the details presented heretofore.

2.5.1 Single Input Image - Single Product Image

The simplest SIDD product is a single product image generated from a single input image, which is shown in Figure 2.5-1. In this example, the SIDD NITF 2.1 file is organized in the following order: NITF 2.1 File Header, Product Image 1, DES 1 – SIDD XML, and DES 2 – SICD XML. DES 1 SIDD XML is linked with the first product image by setting the *IID1* field in the Image Sub-Header to *SIDD001001*. The single SICD XML DES from the input SICD image is contained in DES 2. Additional details on how these components are organized are in sections 2.2, 2.3, and 2.4.

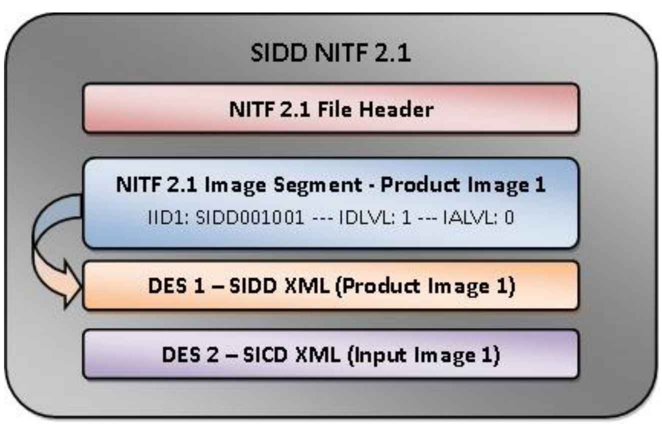

Figure 2.5-1 - SIDD NITF 2.1 File Container Organization - Single Input Image/Single Product Image

2.5.2 Multiple Input Images – Single Product Image

A SIDD NITF 2.1 product containing a single SIDD image product generated from multiple SICD inputs is shown in Figure 2.5-2. This product is organized the same as the product in section

2.5.1 except there are two SICD DESs present. The example product shown in the figure is for a SIDD product generated from 2 SICD inputs. The ordering of the SICD DESs is unspecified but could be specified by an implementation specification.

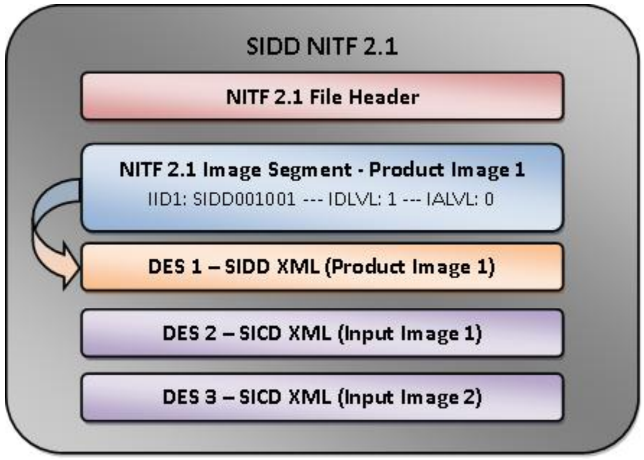

Figure 2.5-2 - SIDD NITF 2.1 File Container Organization - Multiple Input Images/Single Product Image

2.5.3 Single Input Image – Multiple Product Images

A SIDD NITF 2.1 Product containing multiple SIDD image products generated from a single input SICD is shown in Figure 2.5-3. This product is organized the same as the product in section 2.5.1 except that a second product image and SIDD XML DES instance are present. The product images are linked to their associated SIDD XML via the *IID1* field in the NITF 2.1 image sub-header. The SICD XML DES is now the 3rd DES because two product images exist and their associated SIDD XML DESs must precede the SICD XML DES.

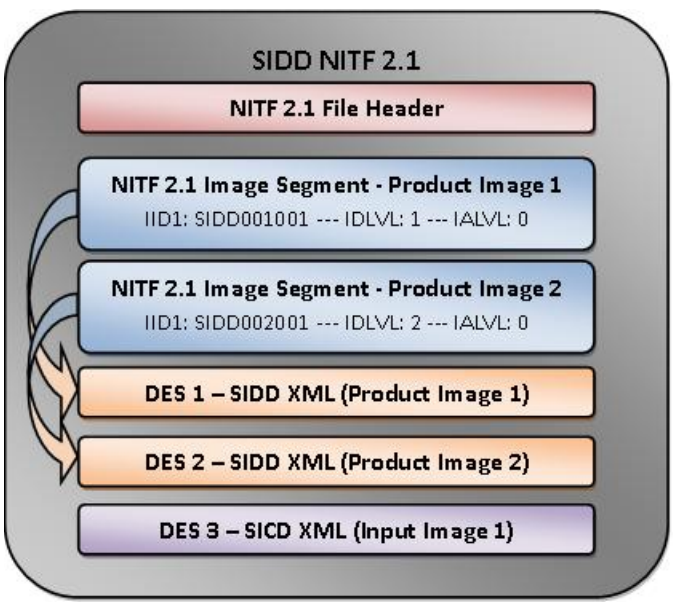

Figure 2.5-3 - SIDD NITF 2.1 File Container Organization - Single Input Image/Multiple Product Images

2.5.4 Single Input Image – Single Product Image Requiring Segmentation

A SIDD NITF 2.1 product containing a single SIDD image product requiring segmentation generated and that is from a single input SICD is shown in Figure 2.5-4. This product is organized the same as the product in section 2.5.1 except for that the product image requires segmentation due to NITF file container constraints.

The segmentation procedure is detailed in section 2.4.1. However, a couple high level details are also noted here. The first segment in each product image has the *IALVL* set to 0 to indicate that it is attached directly to the CCS. The second segment is linked to the first by setting its *IALVL* value to the *IDLVL* value of the first segment. Within the container, the second segment of the product image follows the first segment and appears before the DESs. For this example, the *IID1* field for the second segment of Product Image 1 is set to *SIDD001002* where the *001* indicate the product image and the *002* indicate the second segment. Thus, this product image is associated with DES 1 because of the *001* in the *IID1* field. Additional fields must be set to ensure that segmentation works properly; the section on segmentation, section 2.4.2, specifies these details.

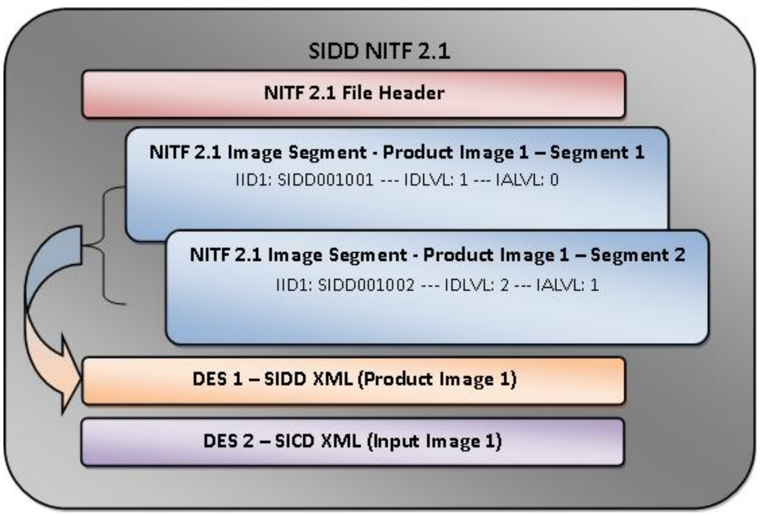

Figure 2.5-4 - SIDD NITF 2.1 File Container Organization - Single Input Image/Single Product Image Requiring Segmentation

2.5.5 Multiple Input Image – Multiple Product Images Requiring Segmentation

A SIDD NITF 2.1 product consisting of multiple product images requiring segmentation generated from multiple SICD inputs is shown in Figure 2.5-5.

The product images are linked to the SIDD XML DESs through the *IID1* field in the image sub-header as mentioned in previous sections. The two SICD XML DESs for the example product are located in the file container after the SIDD XML DESs in an unspecified order; however, this order can be specified in an implementation specification.

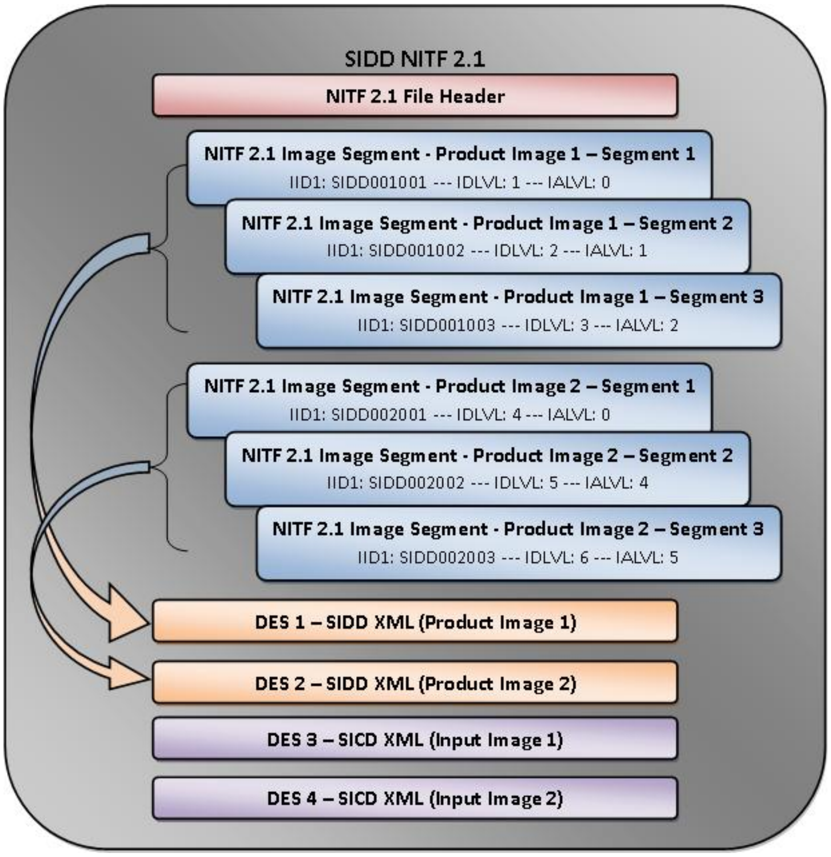

Figure 2.5-5 - SIDD NITF 2.1 File Container Organization - Multiple Input Image/Multiple Product Image Requiring Segmentation

3 SIDD Products in GeoTIFF 1.0 Format

GeoTIFF is a format for storing geo-referenced data and is an extension of the standard TIFF definition. SIDD products contained in a GeoTIFF 1.0 file container use GeoTIFF tags to tie the raster image to a known model space. GeoTIFF 1.0 is an extension of TIFF 6.0. The GeoTIFF 1.0 and TIFF 6.0 specifications are referenced in Table 1.2-2.

The purpose of this section is to define the items below for SIDD NITF 2.1 products.

- Capabilities and limitations of GeoTIFF 1.0 file container for SIDD products
- SIDD GeoTIFF 1.0 file container metadata
- Placement of SIDD product metadata in GeoTIFF 1.0 container
- Placement of SIDD product image data in GeoTIFF 1.0 container
- SIDD GeoTIFF 1.0 file container organization

3.1 Capabilities and Limitations

The GeoTIFF 1.0 container file format can support SIDD products with one or more product images and can be generated from a single or multiple SICD inputs. SIDD GeoTIFF 1.0 products use a raster image format.

The GeoTIFF 1.0 file format does have some limitations. The GeoTIFF 1.0 container file format is limited to 32-bit offsets, thus limiting the size of potential output products to 4 GB. Additionally, the GeoTIFF 1.0 file format does not readily support legends or other types of annotations.

3.1.1 Future Considerations

A future version of this document may allow for GeoTIFF 1.0 products to be generated with JPEG2000 compression enabled. JPEG2000 compression would allow for larger SIDD products (> 4 GB) to fit into a GeoTIFF 1.0 file container. Additionally, a future version of this document may allow for SIDD GeoTIFF 1.0 products to use tiling instead of strips to store the image pixel data thus enabling faster loading for display tools.

3.2 SIDD GeoTIFF 1.0 File Container Metadata

The purpose of this section is to specify all of the file container metadata that is required for populating a valid GeoTIFF 1.0 file. This section defines the Image File Header (IFH), Image File Directory (IFD), baseline TIFF tags, and GeoTIFF tags. Further information on the Image File Header, Image File Directory, and baseline TIFF tags is in the TIFF specification document listed in Table 1.2-2. More information on the GeoTIFF tags and keys can be found in the GeoTIFF specification listed in the same table.

3.2.1 Image File Header

The Image File Header is located in the first eight bytes of the SIDD GeoTIFF 1.0 products. The instructions for populating these eight bytes are shown in Table 3.2-1. This information is also in Section 2 of the TIFF specification document.

Table 3.2-1 SIDD GeoTIFF 1.0 Image File Header Definition

Purpose	Bytes	Value	Comment
Specify Byte Order	Bytes 0-1	“MM” or “II”	Big-endian products will contain “MM”, little-endian products contain “II”
File identifier	Bytes 2-3	“42”	Arbitrary but carefully chosen number that further indentifies the file as TIFF
IFD Offset	Bytes 4-7	Byte offset to first IFD	The offset in bytes of the first IFD. The byte offset is with respect to the start of the file

3.2.2 Image File Directory

An Image File Directory (IFD) exists for each product image in the SIDD GeoTIFF 1.0 file container. An IFD can be located almost anywhere in the file. The first IFD is located by using the IFD offset value in the Image File Header. When multiple IFDs exist in a file, the last 4 bytes of the N^{th} IFD point to the $N+1^{th}$ IFD. The length of the IFD is 2+12*A+4 where A is the number of entries in the IFD. The structure of the IFD is shown in Table 3.2-2. The TIFF tags are specified in sections 3.2.3 and 3.2.4. More information regarding the Image File Directory definition is in Section 2 of the TIFF specification document.

Table 3.2-2 SIDD GeoTIFF 1.0 Image File Directory Definition

Purpose	**Bytes**	**Value**	**Comment**
Number of Directory Entries	2	# of directory entries	
Repeat the next 4 fields for each dictionary entry specified in the first 2 bytes			
Tag	2	Each entry has a unique tag number, see sections below	Entries in an IFD must be sort ordered in ascending order by Tag.
Field Type	2	1-12, value depends on entry type	Specified in the sections below. Also see section 2 of TIFF specification
Number of Values	4	Number of values	
Value/Offset	4	If the *TotalSize* is larger than 4 bytes this field contains an byte offset, if not this field contains the value	*TotalSize* of the value is the size in bytes of *FieldType * Number of Values*
IFD entries complete			
Next IFD offset	4	NULL or byte offset	The NULL value (0000) is used to denote that the IFD is the last IFD in the file. Otherwise a byte offset to the next IFD is given

3.2.3 Baseline TIFF Tags

The purpose of this section is to specify the baseline TIFF tags that are required by the SIDD GeoTIFF 1.0 products. The columns in Table 3.2-3 refer back to the entries in Table 3.2-2. The baseline TIFF tag definitions are expanded upon in sections 2-8 of the TIFF specification document.

Table 3.2-3 SIDD GeoTIFF 1.0 Baseline TIFF Tag Definitions

Name	**Tag # (Dec.)**	**Field Type**	**Number of Values**	**Value/Offset**
ImageWidth	256	Short (3) or Long (4)	1	Value – Generate

Table 3.2-3 SIDD GeoTIFF 1.0 Baseline TIFF Tag Definitions				
Name	**Tag # (Dec.)**	**Field Type**	**Number of Values**	**Value/Offset**
ImageLength	257	Short (3) or Long (4)	1	Value – Generate
BitsPerSample	258	Short (3)	8 or 16	Value – See Table 3.2-4
Compression	259	Short (3)	1	Value – 1
PhotometricInterpretation	262	Short(3)	1	Value – See Table 3.2-4
ImageDescription	270	ASCII (2)	Based on length of string	Offset – "SIDD: <ProductName>" where ProductName comes from *SIDD.ProductCreation.ProductName*
StripOffsets	273	Short (3) or Long (4)	Based on stripping algorithm	Value – Generate
Orientation	274	Short (3)	1	Value – 1
SamplesPerPixel	277	Short (3)	See Table 3.2-4	Value – See Table 3.2-4
RowsPerStrip	278	Short (3) or Long (4)	Based on stripping algorithm	Value – Generate
StripByteCounts	279	Short (3) or Long (4)	Based on stripping algorithm	Value – Generate
XResolution	282	Rational (5)	1	Offset – 1,1
YResolution	283	Rational (5)	1	Offset – 1,1
PlanerConfiguration	284	Short (3)	1	Value – 1
ResolutionUnit	296	Short (3)	1	Value – 1 (no unit specified)
Software	305	ASCII (2)	Based on length of string	Offset – Populated equivalent to SIDD XML – *SIDD.ProductCreation.ProcessorInformation.Application*

Table 3.2-3 SIDD GeoTIFF 1.0 Baseline TIFF Tag Definitions				
Name	**Tag # (Dec.)**	**Field Type**	**Number of Values**	**Value/Offset**
DateTime	306	ASCII (2)	20	Populated equivalent to SIDD XML – *SIDD.ProductCreation.ProcessorInformation.ProcessingDateTime* Format - "YYYY:MM:DD HH:MM:SS" (19 bytes + terminating NULL)
Artist	315	ASCII (2)	Based on length of string	Populated equivalent to SIDD XML - *SIDD.ProductCreation.ProcessorInformation.Site*
ColorMap	320	Short (3)	See Table 3.2-4	Value – See Table 3.2-4

SIDD GeoTIFF 1.0 products can utilize five types of pixels as specified in the SIDD XML field *SIDD.Display.PixelType*. They are MONO8I, MONO8LU, MONO16I, RGB8LU, and RGB24I. The instructions for populating the baseline TIFF fields to support these five types are shown in Table 3.2-4. More information regarding the different pixel types is in the TIFF specification document in section 3–6.

Table 3.2-4 SIDD GeoTIFF 1.0 Baseline TIFF Tag Instructions for Supported Pixel Types					
Field Name	**MONO8I**	**MONO8LU**	**MONO16I**	**RGB8LU**	**RGB24I**
BitsPerSample (258)	# of Values: 1 Value: 8	# of Values: 1 Value: 8	# of Value: 1 Value: 16	# of Values: 1 Value: 8	# of Values: 3 Value: 8,8,8
PhotometricInterpretation (262)	Value: 1	Value: 1	Value: 1	Value: 3	Value: 2
SamplesPerPixel (277)	Not used	Not used	Value: 2	Not used	Value: 3
ColorMap (320)	Not used	Not used	Not used	# Values: 3*2^bitsPerSample :768 Value: User defined	Not used

3.2.4 GeoTIFF TIFF Tags

The purpose of this section is to specify the GeoTIFF TIFF tags that are required for SIDD GeoTIFF 1.0 products. The GeoTIFF TIFF tags are incorporated in the IFD with the same requirements (increasing order of tag number) as the baseline TIFF tags. For the purposes of this document the baseline and GeoTIFF TIFF tags are split up to show the differences between the GeoTIFF requirements and TIFF requirements. The four GeoTIFF TIFF tags required by SIDD GeoTIFF 1.0 products are shown in Table 3.2-5.

Table 3.2-5 SIDD GeoTIFF 1.0 GeoTIFF Tag Definitions

Name	Tag # (Dec.)	Field Type	Number of Values	Value/Offset
ModelTiepointTag	33922	Double (12)	6*numTiePoints	Offset – Corner coordinates, Specifies the point (I,J,K) where the location (I,J) is in raster space with pixel-value K and (X,Y,Z) is a vector in model space.
GeoKeyDirectoryTag	34735	Short (3)	Variable	Offset - See section 3.2.4.1
GeoDoubleParamsTag	34736	Double (12)	Variable	Offset - See section 3.2.4.1
GeoAsciiParamsTag	34737	ASCII (2)	Variable	Offset - See section 3.2.4.1

3.2.4.1 GeoTIFF GeoKey Definition

The GeoTIFF 1.0 specification utilizes the four TIFF tags in Table 3.2-5 to geo-reference the raster image pixel data. The GeoTIFF 1.0 file format utilizes a sub-tagging structure, referred to as keys, that closely resembles TIFF tag structure. The four TIFF tags associated with GeoTIFF each have unique keys that are associated with the GeoTIFF format. The GeoKey key numbers are not related to the TIFF tag numbers.

The TIFF GeoKeyDirectoryTag is an array of shorts. The GeoKeyDirectoryTag identifies the layout of the GeoTIFF and references the keys provided in the GeoTIFF specification document found in Table 1.2-2. The four values in the GeoKeyDirectoryTag value indicate the KeyDirectoryVersion, KeyRevision, MinorRevision, and NumberOfKeys in the directory. The current values for KeyDirectoryVersion, KeyRevision, and MinorRevision are 1, 1, and 2 respectively. The NumberOfKeys is variable and identifies the keys contained in this directory.

GeoKeys are defined in the GeoKeyDirectoryTag. Each GeoKey is defined using four values that define sequentially the GeoKey number, TIFFTagLocation, Count, and Value/Offset. The TIFFTagLocation specifies the location of the GeoKey value which is in the GeoKeyDirectoryTag, GeoAsciiParamsTag, or GeoDoubleParamsTag depending on the value's type. If the TIFFTagLocation is 0 then the value is found in the GeoKeyDirectoryTag. Otherwise the TIFFTagLocation points to the TIFF Tag number (GeoAsciiParamsTag or GeoDoubleParamsTag). The Count provides the number of values for that key. The Value/Offset of each tag entry depends on the TIFFTagLocation and count. If the TIFFTagLocation is 0, then TIFFTagLocation is the location of the values for that key. If the TIFFTagLocation is non zero, then TIFFTagLocation is the offset from the start of the value/offset portion of the associated TIFFTag. For data stored in the GeoAsciiParamsTag, each entry is separated with the "|" character instead of the NULL. GeoTIFF readers are required to parse the "|" and return it is a NULL character. For more information regarding

Table 3.2-6 GeoKeyDirectoryTag Structure

Name	Key # (Dec.)	TIFFTagLocation	Count	Value/Offset
GTModelTypeGeoKey	1024	0	1	Value – 2 (Geodetic)
GTRasterTypeGeoKey	1025	0	1	Value – 1 (RasterPixelsArea)
GeographicTypeGeoKey	2048	0	1	Value – 4326 (GCS_WGS84)
GeogLinearUnitsGeoKey	2052	0	1	Value – 9001 (Linear_Meter)
GeoAngularUnitsGeoKey	2054	0	1	Value – 9102 (Angular Degree)
GeogEllipsoidGeoKey	2056	0	1	Value – 7030 (Ellipse_WGS_84)

3.3 SIDD GeoTIFF 1.0 SIDD Product Metadata

The purpose of this section is to specify how SIDD product metadata is included in the SIDD GeoTIFF 1.0 products. More specifically, this section specifies the placement of the SIDD XML and multiple SICD XML inputs. An existing GeoTIFF reader/writer can be updated to accommodate SIDD GeoTIFF 1.0 products by incorporating the information in this section.

3.3.1 SICD XML Field Definition

The purpose of this section is to define how the SICD XML metadata is included in the Image File Directory. Each IFD in a SIDD GeoTIFF 1.0 product should contain the SICDXMLTag, if the input was SICD. The value of the SICDXMLTag is the unaltered SICD XML. If the SIDD product image is generated from multiple SICD inputs each SICD XML must be present in the value. Each SICD XML is separated by the NULL character. For example, the value of the SICDXMLTag for a SIDD product generated from three SICD input products would look like the following: {<SICD_XML_Input_1>"NULL"<SICD_XML_Input_2>"NULL" <SICD_XML_Input_3>}. The SICD XML is defined in the Sensor Independent Complex Data (SICD) Design & Exploitation Description, see Table 1.2-1.

The SICDXMLTag TIFF tag number is 52766. The TIFF Tag number is not currently used according to the following tool: TIFF Tag Search. The field type is undefined which means that the data is an array of 8-bit units that are defined as UTF-8 in this specification. Table 3.3-1 summarizes the definition of the SICDXMLTag field definition.

Table 3.3-1 SICDXMLTag Field Definition				
Name	**Tag # (Dec.)**	**Field Type**	**Number of Values**	**Value/Offset**
SICDXMLTag	52766	Undefined (7)	Variable	Offset – See SICD D&E Document

3.4 SIDD GeoTIFF 1.0 SIDD Product Image Pixel Data

The purpose of this section is to define how product image pixel data is stored in SIDD GeoTIFF 1.0 products. Product image pixel data is stored according to TIFF standards. SIDD GeoTIFF 1.0 products can use one of three TIFF pixel types: MONO, RGB/LUT, and RGB. Section 3.2.3 describes the metadata requirements for using these different data types. The TIFF 6.0 specification should be used as the primary reference for instructions in storing image pixel data in a SIDD GeoTIFF 1.0 product.

At present, the SIDD product image pixel data is stored in a single strip (RowsPerStrip = ImageLength). TIFF specifications allow for the data to be stored in strips or tiles but the current recommendation is to utilize a single strip.

SIDD GeoTIFF products require that the Orientation tag is set to 1, the tag indicates that the first pixel in the strip refers to the visual upper left corner of the image. For RGB image pixel data the PlanerConfiguration value of 1 indicates "chunky" storage. "Chunky" storage means that data is stored such that color channels are interleaved on a pixel by pixel basis, i.e., the 8 bit values would be read RGBRGBRGB where the first RGB corresponds to the first pixel; second RGB corresponds to the second pixel and so on.

3.5 SIDD GeoTIFF 1.0 File Container Organization

The purpose of this section is to define how the three basic components of the SIDD GeoTIFF 1.0 are organized, which are the GeoTIFF 1.0 file container metadata, SIDD product metadata, and SIDD product image data. Each of the three components is discussed in greater detail in sections 3.2, 3.3, and 3.4 respectively.

The GeoTIFF 1.0 file container organization is identical, at the highest level, to the TIFF. However, the GeoTIFF 1.0 product format defines an additional set of required tags. A TIFF file starts with an Image File Header. The Image File Header points to the first Image File Directory (IFD) which has no requirement to follow the Image File Header in the file. TIFF files can be organized in almost any manner because file offset pointers are used to reference the location of components.

For example, in a single image TIFF file the data could be organized with the Image File Header first followed by the image pixel data and ending with the IFD. The order could also be switched such that the IFD follows the Image File Header and the pixel data appears last. The IFD contains the metadata for its associated image. TIFF does not support sharing metadata across multiples images. Thus, the SICD input metadata is included for each image of a multi-image SIDD GeoTIFF 1.0 product.

The following sections provide examples of the organization of SIDD GeoTIFF 1.0 products.

3.5.1 Single Product Image

The simplest SIDD GeoTIFF 1.0 product is a single product image generated from a single input SICD image. The basic organization of a SIDD GeoTIFF 1.0 product with single product image is shown in Figure 3.5-1.

The file begins with the Image File Header which points to the Image File Directory. The baseline TIFF tags, GeoTIFF tags, SIDD XML tag, and SICD XML tag, if available, are stored in the IFD by increasing tag number. The entries in figure below are a collection of tags. The four types of tags in the IFD are described in section 3.2. The *StripOffsets* tag points to the product image pixel data.

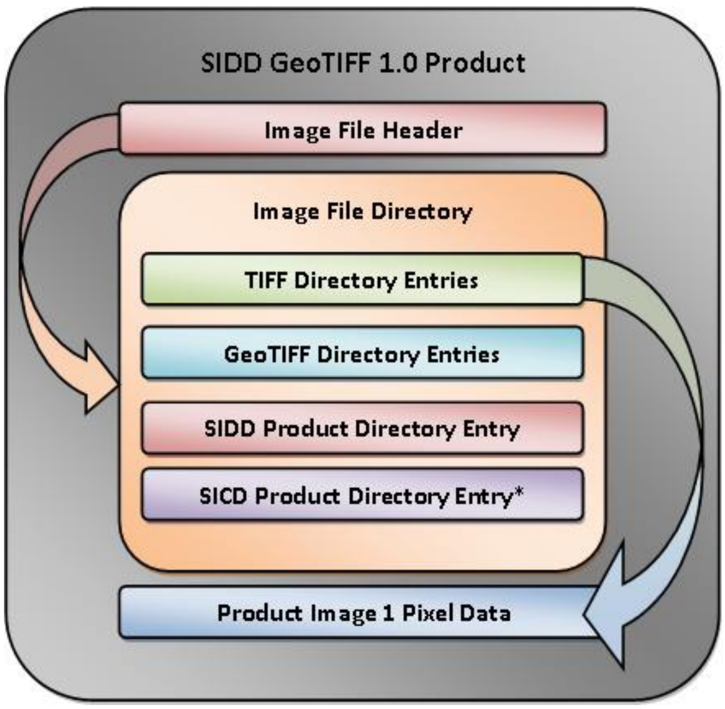

Figure 3.5-1 SIDD GeoTIFF 1.0 Single Product Image Example

3.5.2 Multiple Product Images

A slightly more advanced SIDD GeoTIFF 1.0 product may consist of multiple product images. In the TIFF specification, having multiple images in a single TIFF is referred to as having multiple subfiles, or pages. An example file organization of a SIDD GeoTIFF 1.0 with multiple product images is shown in Figure 3.5-2.

All of the concepts that are related to a product with a single image apply to the multiple product image SIDD GeoTIFF file. The primary difference is that at the end of the first IFD an offset is given to the beginning of the next IFD. In addition, each product image generated also contains a SICDXMLTag, if the input was SICD. This behavior leads to duplication of SICD information but is required because there is no standard means by which to associate IFD entries with multiple product images in the TIFF file format.

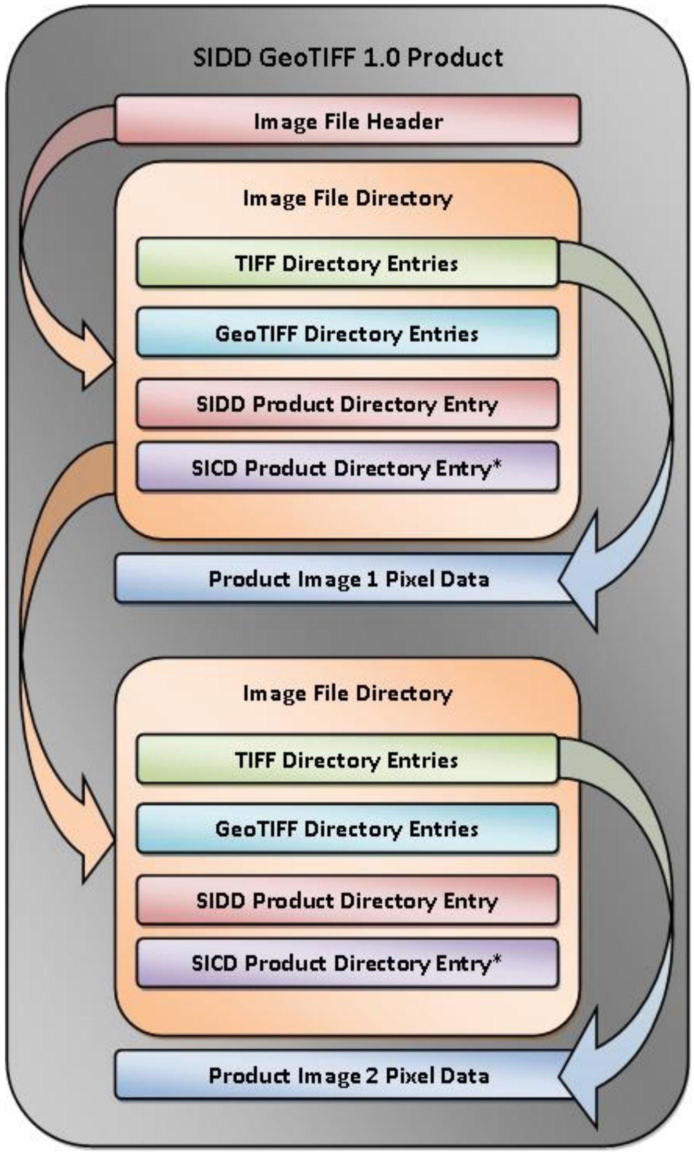

Figure 3.5-2 SIDD GeoTIFF 1.0 Multiple Product Image Example (* = if SICD available)

Appendix A – Terms & Definitions

Terms & Definitions	
Term	**Definition**
CCS	Common Coordinate System
DES	Data Extension Segments
GeoTIFF	Standard for providing georeferencing data within a TIFF file
IFD	Image File Directory (GeoTIFF)
LUT	Lookup Table
NITF	National Imagery Transmission Format
RGB	Red-Green-Blue
SICD	Sensor Independent Complex Data.
TIFF	Tagged Image File Format – a public standard image container
XML	Extensible Markup Language