Network Working Group
Request for Comments: 5372
Category: Standards Track

A. Leung
S. Futemma
E. Itakura
Sony

October 2008

Payload Format for JPEG 2000 Video: Extensions for Scalability and Main Header Recovery

Status of This Memo

Abstract

This memo describes extended uses for the payload header in "RTP Payload Format for JPEG 2000 Video Streams" as specified in RFC 5371, for better support of JPEG 2000 features such as scalability and main header recovery.

This memo must be accompanied with a complete implementation of "RTP Payload Format for JPEG 2000 Video Streams". That document is a complete description of the payload header and signaling, this document only describes additional processing for the payload header. There is an additional media type and Session Description Protocol (SDP) marker signaling for implementations of this document.

Table of Contents

1. Introduction 3
1.1. Description of the Mechanisms 3
1.1.1. Main Header Compensation 3
1.1.2. Priority Table 3
1.2. Motivations for Priority Field Coding 4
1.2.1. Scenario: Just Enough Resolution 4
1.2.2. Scenario: Multiple Clients, Single Source 4
1.3. Conventions Used in This Document 4
2. Payload Format Enhanced Processing 5
2.1. Enhanced Processing Markers 5
3. Priority Mapping Table 6
3.1. Packet-Number-Based Ordering 7
3.2. Progression-Based Ordering 7
3.3. Layer-Based Ordering 9
3.4. Resolution-Based Ordering 9
3.5. Component-Based Ordering 10
4. JPEG 2000 Main Header Compensation Scheme 10
4.1. Sender Processing 11
4.2. Receiver Processing 11
5. Media Type Registration 11
6. SDP Parameters 12
6.1. Mapping of the Optional Parameters to SDP 12
6.2. Usage with the SDP Offer/Answer Model 13
6.2.1. Examples 13
7. IANA Considerations 16
8. Security Considerations 16
9. Congestion Control 16
10. Normative References 16
Appendix A. Sample Headers in Detail 17
A.1. Sample 1: Progressive Image with Single Tile, 3500 Bytes (i.e., thumbnail) 17
A.2. Sample 2: Image with 4 Tiles 19
A.3. Sample 3: Packing Multiple Tiles in Single Payload, Fragmented Header. No Header Compensation, Progressive Image 20
A.4. Sample 4: Interlace Image, Single Tile 22

1. Introduction

This document is an extension of "RTP Payload Format for JPEG 2000 Video Streams" [RFC5371]. These are additional mechanisms that can be used with certain parts of the header in [RFC5371] to support JPEG 2000 features such as scalability and a main header compensation method. These mechanisms are described in detail in this document.

These are optional extensions to RFC 5371 [RFC5371], which one may use to make better use of JPEG 2000 features. These extensions are not required for any implementations of RFC 5371 [RFC5371].

1.1. Description of the Mechanisms

1.1.1. Main Header Compensation

JPEG 2000 image header contains essential decoding information for the decoder. If a JPEG 2000 decoder receives JPEG 2000 image data without a JPEG 2000 image header, it could not decode the JPEG 2000 image data properly. Encoders for JPEG 2000 video very rarely change encoding parameters between successive frames. So, the possibility of the decoder successively decoding the new JPEG 2000 image data using a prior JPEG 2000 image header is very high in this situation.

The main header compensation scheme used in this document exploits the fact that successive JPEG 2000 video images rarely change encoding parameters. It requires receivers to save past JPEG 2000 image headers to use in case of missing JPEG 2000 image headers in the future. Signaling of changes to the JPEG 2000 image header is done through the payload header. When the mh_id value of the payload header changes, receivers should save the new JPEG 2000 header to use for main header recovery.

1.1.2. Priority Table

JPEG 2000 codestream has rich functionality built into it so decoders can easily handle scalable delivery or progressive transmission. Progressive transmission allows images to be reconstructed with increasing pixel accuracy or spatial resolution. This feature allows the reconstruction of images with different resolutions and pixel accuracy, for different target devices. A single image source can provide a codestream that is easily processed for smaller image display devices.

JPEG 2000 packets contain all compressed image data from a specific layer, component, resolution level, and/or precinct. The order in which these JPEG 2000 packets are found in the codestream is called the progression order. The ordering of the JPEG 2000 packets can

progress along four axes: layer, component, resolution, and precinct (or position).

Providing a priority field to indicate the importance of data contained in a given RTP packet can aid in usage of JPEG 2000 progressive and scalable functions.

1.2. Motivations for Priority Field Coding

The JPEG 2000 coding scheme allows one to reorder the codestream in many ways. Even when the coding scheme is determined and arranged by the encoder, a decoder can still re-arrange the code stream on the fly to suit decode parameters such as re-arranging from resolution progressive to quality progressive.

Using the priority field coding, the decoder gains insight into the codestream without access to the full codestream and exposes features of JPEG 2000 to a higher level.

The authors found the scenarios below to utilize this field. The priority field allows more information about the image to be sent without more signaling between sender and receivers to leverage JPEG 2000 capabilities.

1.2.1. Scenario: Just Enough Resolution

The scenario is when rapid scene access is more important than higher image quality. By using the priority field, the receiver can decode for its own quality level. If the sender cannot determine the receiver's resolution, the receiver can select which parts of the codestream to be decoded or loaded by using the priority field.

1.2.2. Scenario: Multiple Clients, Single Source

In a multicast environment, there are clients with better visual capability than others (i.e., TV conference versus Mobile). The respective clients can use the priority field to determine which packets are vital for their own visual presentation. The sender only has to do work on the priority field to optimally serve all the clients while only managing a single visual stream.

1.3. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. [RFC2119].

2. Payload Format Enhanced Processing

2.1. Enhanced Processing Markers

This section of the document describes additional usage in the values of mh_id and priority fields and interpretation that differ from RFC 5371 [RFC5371]. Implementations of this document should follow RFC 5371 [RFC5371] first then add additional header processing as described in this document. Implementations following this document are expected to interoperate with implementations of [RFC5371] and this document as well.

The RTP payload header format for JPEG 2000 video stream is as follows:

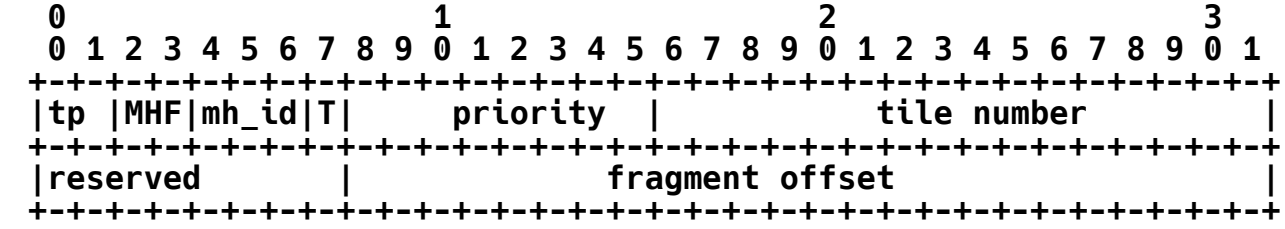

Figure 1: RTP Payload Header Format for JPEG 2000

mh_id (Main Header Identification): 3 bits

Main header identification value. This is used for JPEG 2000 main header recovery.

The initial value of mh_id MUST be 1 at the beginning of the session.

The same mh_id value is used as long as the coding parameters described in the main header remains unchanged between frames.

The mh_id value MUST be incremented by 1 every time a new main header is transmitted. Once the mh_id value becomes greater than 7, it SHOULD roll over to 1.

When mh_id is 0, it has special usage for the receiver. This special usage is described in Section 4.2 of this document.

Senders should follow Section 4.1 of this document for proper mh_id assignment and usage.

priority: 8 bits

The priority field indicates the importance of the JPEG 2000 packet included in the payload. Typically, a higher priority value is set in the packets containing JPEG 2000 packets containing the lower sub-bands.

Special values of priority:

0: This is reserved for payloads that contain a header (main or tile part header). This is considered the most important.

1 to 255: These values decrease in importance as the values increase (i.e., 1 is more important than 2, etc.). Applying priority values should correlate directly to the JPEG 2000 codestream in importance.

The lower the priority value, the higher the importance. A priority value of 0 is the highest importance and 255 is the lowest importance. We define the priority value 0 as a special priority value for the headers (the main header or tile-part header). If any headers (the main header or tile-part header) are packed into the RTP payload, the sender MUST set the priority value to 0.

Assignment of the values is described in Section 3.

3. Priority Mapping Table

For the progression order, the priority value for each JPEG 2000 packet is given by the priority mapping table.

This document specify several commonly used priority mapping tables, pre-defined priority mapping tables: packet-number-based (default), progression-based, layer-based, resolution-based, and component-based.

Packet number priority mapping is REQUIRED to be supported by clients implementing this specification. Other priority mapping tables (progression, layer, resolution, and component-based) are OPTIONAL to implementations of this specification.

Rules that all implementations of this specification MUST follow in all priority modes:

- When there is a header in the packet with a JPEG 2000 packet, the sender MUST set the payload packet priority value to 0.

- When there are multiple JPEG 2000 packets in the same RTP payload packet, the sender MUST set the payload packet priority value to the lowest JPEG 2000 packet (i.e., if JPEG 2000 packets with priority: 5,6,7 are packed into a single payload, the priority value will be 5).

3.1. Packet-Number-Based Ordering

Packet-number-based ordering assigns the payload packet priority value from the "JPEG 2000 packet value" (note: JPEG 2000 codestreams are stored in units of packets and each packet has a value). This method is the default method for assigning priority value. All implementations of this specification MUST support this method.

If the JPEG 2000 codestream packet value is greater than 255, the sender MUST set the payload priority value to 255.

3.2. Progression-Based Ordering

The sender will assign the payload packet priority value only based on layer, resolution, and component ordering of the codestream.

Progression-based ordering can assign the different priority values in the same layer or the resolution level, which it cannot do in the layer-based ordering or resolution-based ordering.

Unlike the packet-number-based ordering, the progression-based ordering does not assign a value in the position level, which saves the priority values. The priority value in the position level is not so important because a receiver could recognize the position by checking the tile number field. Therefore, the progression-based ordering would be useful.

The general algorithm is that the ordering is based on the triple <layer, resolution, component> and the minimum priority is 1. So, if the codestream is constructed of L layers (layer value ranges from 0 to L-1), R resolutions (resolution value ranges from 0 to R-1), and C components (component value ranges from 0 to C-1), then for a triple <lval, rval, cval>:

the priority value of the codestream in LRCP order is calculated as:

```
priority = 1 + cval + (C * rval) + (C * R * lval)
```

the priority value of the codestream in RLCP order is calculated as:

priority = 1 + cval + (C * lval) + (C * L * rval)

the priority value of the codestream in RPCL order is calculated as:

priority = 1 + lval + (L * cval) + (L * C * rval)

the priority value of which codestream in PCRL order is calculated as:

priority = 1 + lval + (L * rval) + (L * R * cval)

the priority value of which codestream in CPRL order is calculated as:

priority = 1 + lval + (L * rval) + (L * R * cval)

For example:

If the codestream is ordered in LRCP (Layer, Resolution, Component, Position) with 1 layer (L=1), 2 resolutions (R=2), 3 components (C=3), and 2 positions, the priority value should be (1 + cval + 3*rval + 6*lval). Then an example would have packet numbering as so:

All the packets in:

layer.....0
resolution....0
component.....0
position......0 or 1

then the packet priority value : 1

All the packets in:

layer.....0
resolution....0
component.....1
position......0 or 1

then the packet priority value : 2

```
      All the packets in:

         layer.........0
         resolution....0
         component.....2
         position......0 or 1

      then the packet priority value : 3

      All the packets in:

         layer.........0
         resolution....1
         component.....0
         position......0 or 1

      then the packet priority value : 4

      All the packets in:

         layer.........0
         resolution....1
         component.....1
         position......0 or 1

      then the packet priority value : 5
```

3.3. Layer-Based Ordering

The layer-based priority mapping table simplifies the default mapping to just matching JPEG 2000 packets together from the same layer.

For example:

```
   All the packets in layer 0 : packet priority value : 1
   All the packets in layer 1 : packet priority value : 2
   All the packets in layer 2 : packet priority value : 3
   ...
   All the packets in layer n : packet priority value : n+1
   All the packets in layer 255 : packet priority value : 255
```

3.4. Resolution-Based Ordering

Resolution-based priority mapping table is similar to the layer-based order but for JPEG 2000 packets of the same resolution.

For example:

```
All the packets in resolution 0 : packet priority value : 1
All the packets in resolution 1 : packet priority value : 2
All the packets in resolution 2 : packet priority value : 3
...
All the packets in resolution n : packet priority value : n+1
All the packets in resolution 255 : packet priority value : 255
```

3.5. Component-Based Ordering

Component-based priority mapping table is mapping together JPEG 2000 components of the same component.

For example:

```
All the packets in component 0 : packet priority value : 1
All the packets in component 1 : packet priority value : 2
All the packets in component 2 : packet priority value : 3
...
All the packets in component n : packet priority value : n+1
All the packets in component 255 : packet priority value : 255
```

4. JPEG 2000 Main Header Compensation Scheme

The mh_id field of the payload header is used to indicate whether the encoding parameters of the main header are the same as the encoding parameters of the previous frame. The same value is set in mh_id of the RTP packet in the same frame. The mh_id and encode parameters are not associated with each other as 1:1, but they are used to indicate whether or not the encode parameters of the previous frame are the same in the event of a lost header.

The mh_id field value SHOULD be saved from previous frames to be used to recover the current frame's main header. If the mh_id of the current frame has the same value as the mh_id value of the previous frame, the previous frame's main header MAY be used to decode the current frame, in case of a lost header in the current frame.

The sender MUST increment mh_id when parameters in the header change and send a new main header accordingly.

The receiver MAY use the mh_id and MAY retain the header for such compensation.

4.1. Sender Processing

The sender MUST transmit RTP packets with the same mh_id value if the encoder parameters of the current frame are the same as the previous frame. The encoding parameters are the fixed information marker segment (SIZ marker) and functional marker segments (COD, COC, RGN, QCD, QCC, and POC) specified in JPEG 2000 Part 1, Annex A [JPEG2000Pt_1].

If the encode parameters changes, the sender transmitting RTP packets MUST increment the mh_id value by one, but when the mh_id value becomes greater than 7, a sender MUST set the mh_id value back to 1.

4.2. Receiver Processing

When the receiver receives the main header completely, the RTP sequence number, the mh_id, and the main header should be saved. Only the last main header that was received completely SHOULD be saved. When the mh_id value is 0, the receiver SHOULD NOT save the header.

When the main header is not received, the receiver may compare the current payload header's mh_id value with the previous saved mh_id value. If the values match, decoding may be performed by using the previously saved main header.

If the mh_id field is set to 0, the receiver MUST NOT save the main header and MUST NOT compensate for lost headers.

If the mh_id value changes, receivers SHOULD save the current header and save the new mh_id value. The old saved header should be deleted from storage.

Also, if the decoder detects an inconsistency between the header and payload, it is recommended to clear the saved mh_id and the saved main header. Please see Section 8 for more explanation.

5. Media Type Registration

This document extends the associated media type "video/jpeg2000" from RFC 5371 [RFC5371]. Here are additional optional parameters.

Additional optional parameters:

mhc: Main Header Compensation. This option is used when a sender and/or receiver is utilizing the Main Header Compensation technique as specified in this document. Acceptable values when using the Main Header Compensation technique is "1"; otherwise, it should be "0".

This is a list of options to be included when the sender or receiver is utilizing the priority table option as specified in this document.

pt: Priority Table. This option is followed by a comma-separated list of pre-defined priority table definitions to be used by sender or receiver.

The option appearing front most in the option line is the most important and the next are of decreasing importance.

Acceptable values:

progression: this table follows the progression ordering of the codestream.

layer: this table follows the layer ordering of the codestream.

resolution: this table follows the resolution ordering of the codestream.

component: this table follows the component ordering of the codestream.

default: this table follows the packet ordering of the codestream.

6. SDP Parameters

6.1. Mapping of the Optional Parameters to SDP

The new optional parameters mhc and pt, if presented, MUST be included in the "a=fmtp" line of SDP. These parameters are expressed as a media type string, in the form of a semicolon-separated list of parameter=value pairs.

6.2. Usage with the SDP Offer/Answer Model

In addition to the SDP Offer/Answer section in RFC 5371 [RFC5371]:

When offering JPEG 2000 over RTP using SDP in an Offer/Answer model [RFC3264], the following rules and limitations apply:

- All parameters MUST have an acceptable value for that parameter.
- All parameters MUST correspond to the parameters of the payload.
- If the optional parameter "mhc" is used, it MUST appear in the offer with value "1", and if accepted, it SHOULD appear in the answer.
- If the optional parameter "pt" is used, it MUST appear in the offer containing a complete comma-separated list indicating which priority table definitions the sender supports. If accepted, it SHOULD appear in the answer containing a single priority table definition selected from the offer.
- If the optional parameter "mhc" is used, it MUST appear in the offer with value "1", and if accepted, it MUST appear in the answer. If the optional parameter "pt" is used, it MUST appear in the offer containing a complete comma-separated list indicating which priority table definitions the sender supports. If accepted, it MUST appear in the answer containing a single priority table definition selected from the offer.
- In a multicast environment:
  - Senders should send out one option for a priority table definition for everyone in the group.
  - Even if a single client in the group does not support the extensions outlined in this document, senders MAY use these mechanisms. A receiver that doesn't support the mechanisms would safely ignore the values.in mh_id and priority field.

6.2.1. Examples

Offer/Answer example exchanges are provided.

6.2.1.1. Example 1

Alice offers Main Header Compensation functionality, YCbCr 422 color space, interlace image with 720-pixel width and 480-pixel height, and several priority table options (default, progression, layer, resolution, component) as below:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.example
s=
c=IN IP4 host.example
t=0 0
m=video 49170 RTP/AVP 98
a=rtpmap:98 jpeg2000/90000
a=fmtp:98 mhc=1; sampling=YCbCr-4:2:2; interlace=1;
pt=default,progression,layer,resolution, component;
width=720;height=480
```

Bob accepts Main Header Compensation functionality, YCbCr-4:2:2 color space, interlace image, default mapping table and replies:

```
v=0
o=bob 2890844730 2890844731 IN IP4 host.example
s=
c=IN IP4 host.example
t=0 0
m=video 49920 RTP/AVP 98
a=rtpmap:98 jpeg2000/90000
a=fmtp:98 mhc=1; sampling=YCbCr-4:2:2;interlace=1;
pt=default;width=720;height=480
```

6.2.1.2. Example 2

Alice offers Main Header Compensation, YCbCr 420 color space, progressive image with 320-pixel width and 240-pixel height, and layer priority table options as below:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.example
s=
c=IN IP4 host.example
t=0 0
m=video 49170 RTP/AVP 98
a=rtpmap:98 jpeg2000/90000
a=fmtp:98 mhc=1; sampling=YCbCr-4:2:0;
pt=layer;width=320;height=240
```

Bob does not accept Main Header Compensation functionality but accepts YCbCr-4:2:0 color space, layer-based priority mapping and replies:

```
v=0
o=bob 2890844730 2890844731 IN IP4 host.example
s=
c=IN IP4 host.example
t=0 0
m=video 49920 RTP/AVP 98
a=rtpmap:98 jpeg2000/90000
a=fmtp:98 mhc=0; sampling=YCbCr-4:2:0;
pt=layer;width=320;height=240
```

6.2.1.3. Example 3

Alice offers 27 MHz timestamp, Main Header Compensation, YCbCr 420 color space, progressive image with 320-pixel width and 240-pixel height, and layer priority table options as below:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.example
s=
c=IN IP4 host.example
t=0 0
m=video 49170 RTP/AVP 98 99
a=rtpmap:98 jpeg2000/27000000
a=rtpmap:99 jpeg2000/90000
a=fmtp:98 mhc=1; sampling=YCbCr-4:2:0;
pt=layer;width=320;height=240
a=fmtp:99 mhc=1; sampling=YCbCr-4:2:0;
pt=layer;width=320;height=240
```

Bob can accept payload type with 27 MHz timestamp, and does not accept Main Header Compensation functionality but accepts YCbCr-4:2:0 color space, layer-based priority mapping and replies:

```
v=0
o=bob 2890844730 2890844731 IN IP4 host.example
s=
c=IN IP4 host.example
t=0 0
m=video 49920 RTP/AVP 98
a=rtpmap:98 jpeg2000/27000000
a=fmtp:98 mhc=0; sampling=YCbCr-4:2:0;
pt=layer;width=320;height=240
```

7. IANA Considerations

This document extends the associated media type "video/jpeg2000" from 5371 [RFC5371]. Additional parameters are specified in Section 5 of this document.

8. Security Considerations

Please refer to Section 6 of RFC 5371 [RFC5371] for Security Considerations regarding this RTP format. The security issues regarding enhanced mechanisms presented in this document are discussed in that section.

The mh_id field can identify a maximum of 7 different main headers. If severe packet loss (either random or intentionally introduced by an attacker) causes 6 successive updates to the main header to be lost, the decoder will attempt decompression using an incorrect main header. Even if the incorrect main header is passed, the standard JPEG 2000 decoder could detect inconsistency of the codestream and process it properly. It is recommended to clear the saved mh_id and the saved main header if the decoder detects such an inconsistency.

9. Congestion Control

Please refer to Section 7 of RFC 5371 [RFC5371] for Congestion Control regarding this RTP format.

10. Normative References

[RFC5371] Futemma, S., Leung, A., and E. Itakura, "RTP Payload Format for JPEG 2000 Video Streams", RFC 5371, October 2008.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[JPEG2000Pt_1] ISO/IEC JTC1/SC29, ISO/IEC 15444-1 | ITU-T Rec. T.800, "Information Technology - JPEG 2000 Image Coding System - Part 1: Core Coding System", December 2000.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

Appendix A. Sample Headers in Detail

The following figures are sample RTP headers demonstrating values that should appear in the RTP header. The packet priority is Packet-Number-Based Priority.

For reference, the payload header is as follows:

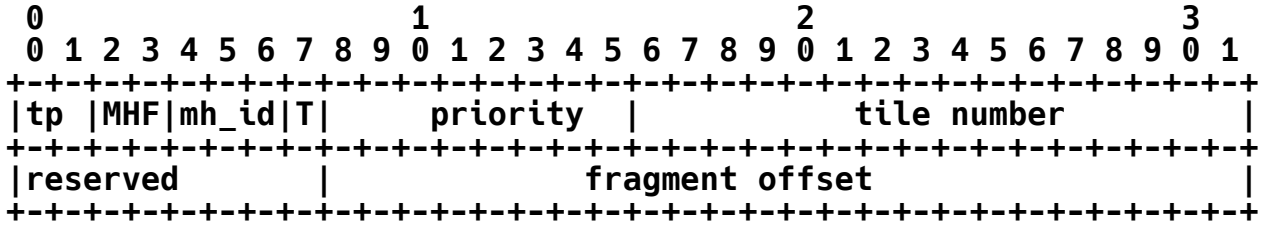

Figure 2: JPEG 2000 Payload Header

A.1. Sample 1: Progressive Image with Single Tile, 3500 Bytes (i.e., thumbnail)

First Packet: This packet will have the whole main header 210 bytes

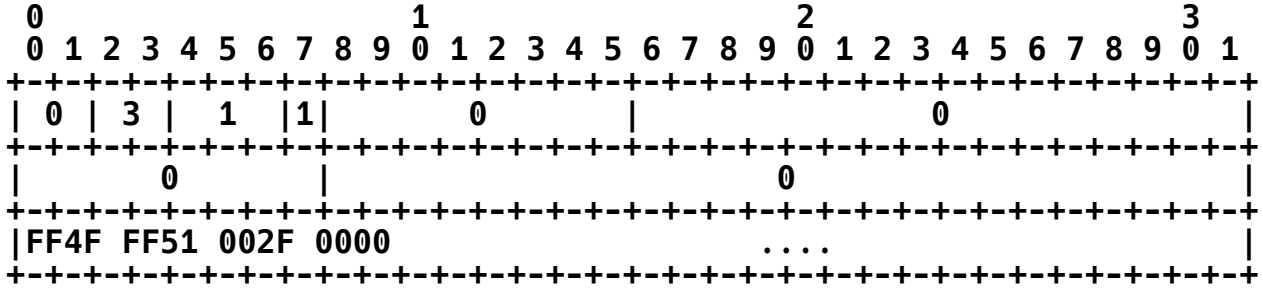

Figure 3: Header Sample 1-1 (First Packet)

Second Packet: This packet will have a tile header and the first tile part LLband 1500 bytes

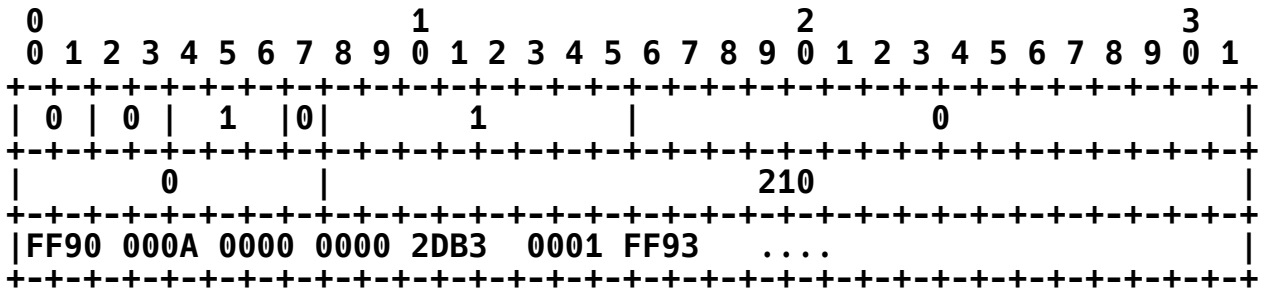

Figure 4: Header Sample 1-2 (Second Packet)

Third Packet: This packet will have the next part in the tile, no tile header 1500 bytes

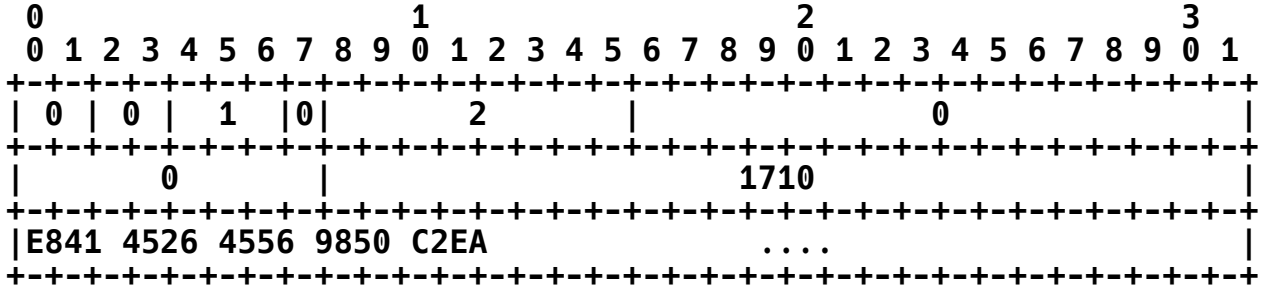

Figure 5: Header Sample 1-3 (Third Packet)

Fourth Packet: Last packet for the image 290 bytes

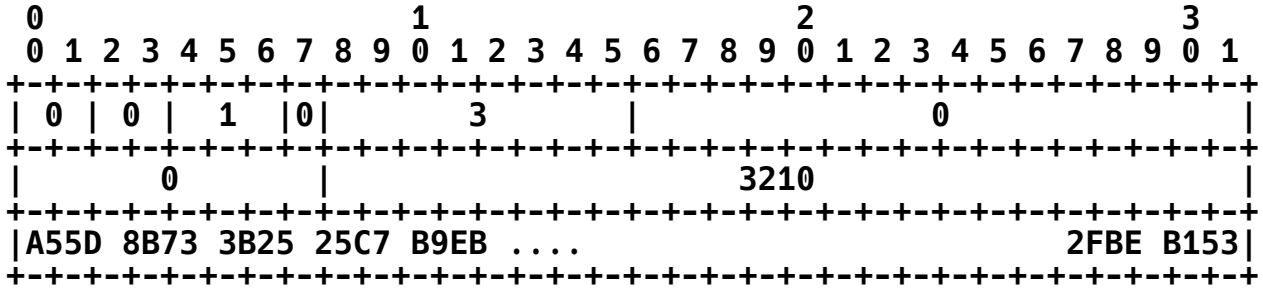

Figure 6: Header Sample 1-4 (Fourth Packet)

A.2. Sample 2: Image with 4 Tiles

First Packet: This packet will have the whole main header 210 bytes

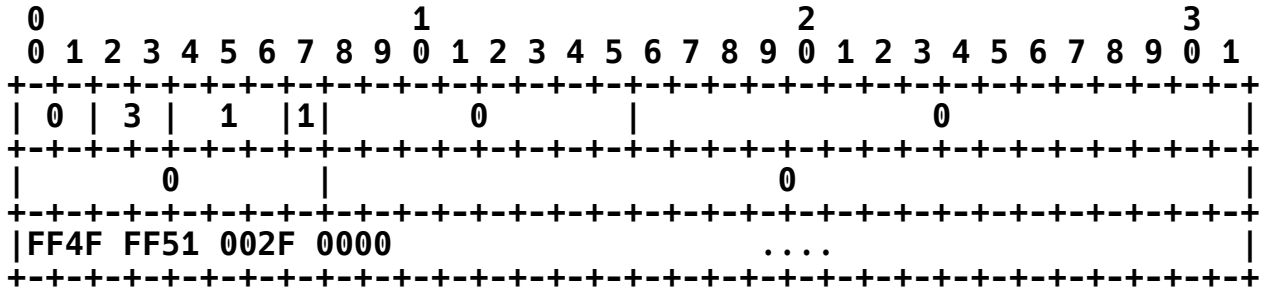

Figure 7: Header Sample 2-1 (First Packet)

Second Packet: This packet will have a first tile part (tile 0) 1400 bytes

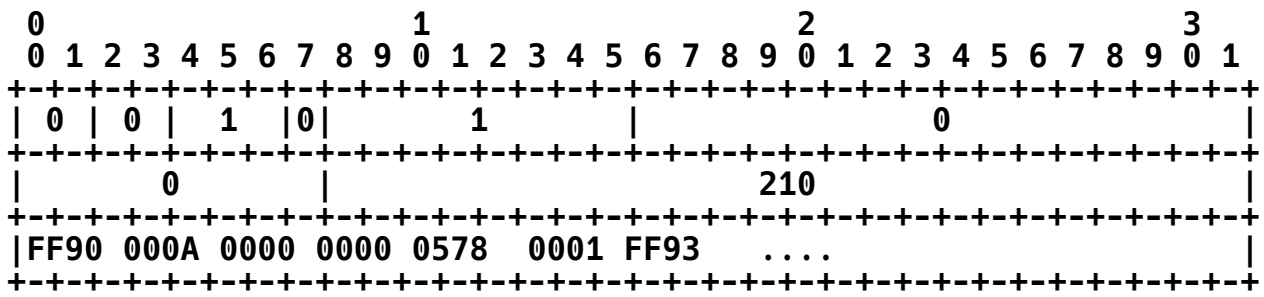

Figure 8: Header Sample 2-2 (Second Packet)

Third Packet: This packet will have a second tile part (tile 1) 1423 bytes

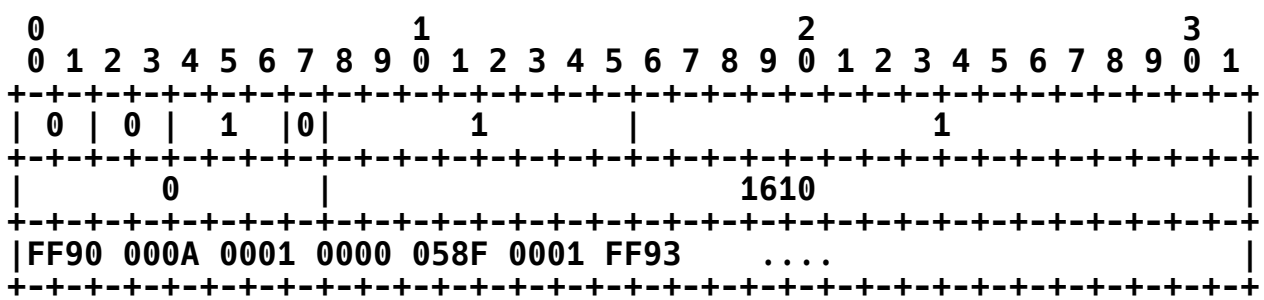

Figure 9: Header Sample 2-3 (Third Packet)

Fourth Packet: This packet will have a third tile part (tile 2) 1355 bytes

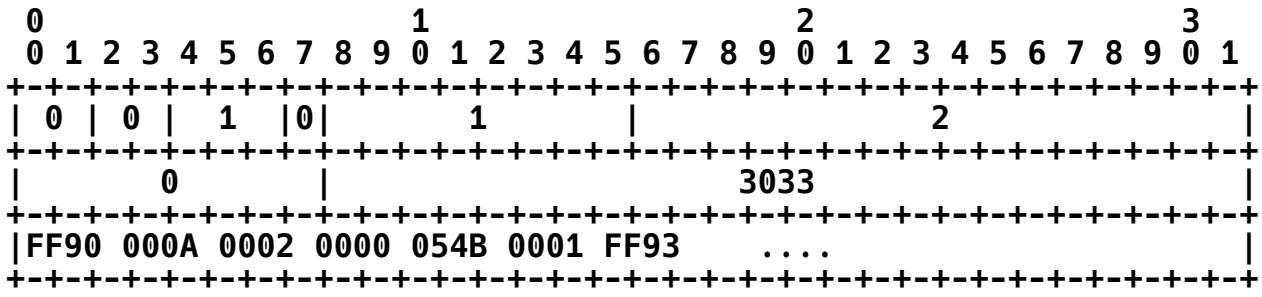

Figure 10: Header Sample 2-4 (4th Packet)

Fifth Packet: This packet will have a fourth tile part (tile 3) 1290 bytes

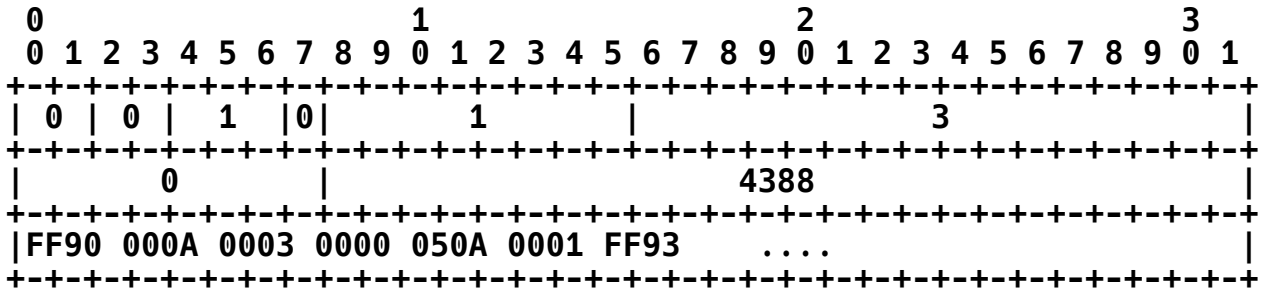

Figure 11: Header Sample 2-5 (5th Packet)

A.3. Sample 3: Packing Multiple Tiles in Single Payload, Fragmented Header. No Header Compensation, Progressive Image

First Packet: This packet will have the first part of the main header 110 bytes

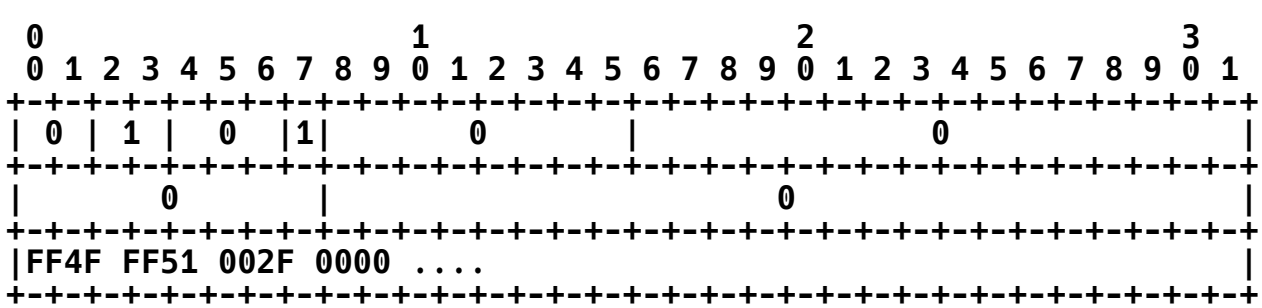

Figure 12: Header Sample 3-1 (First Packet)

Second Packet: This packet has the second part of the main header. 1400 bytes

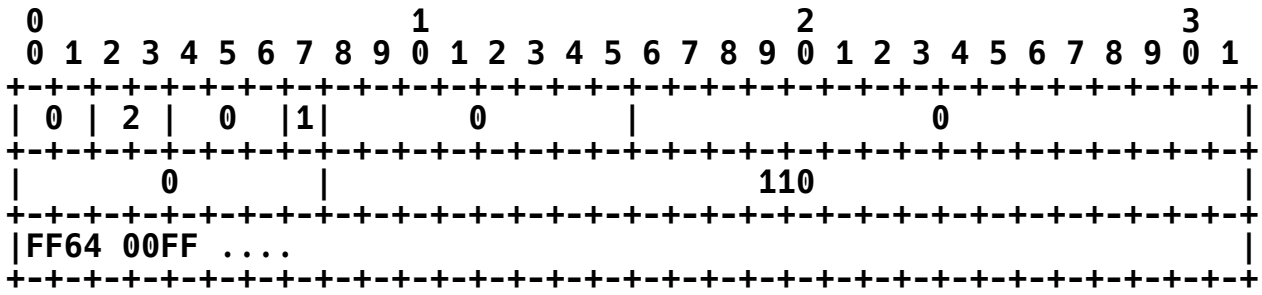

Figure 13: Header Sample 3-2 (Second Packet)

Third Packet: This packet has two tiles, tile 0 and tile 1 1400 bytes

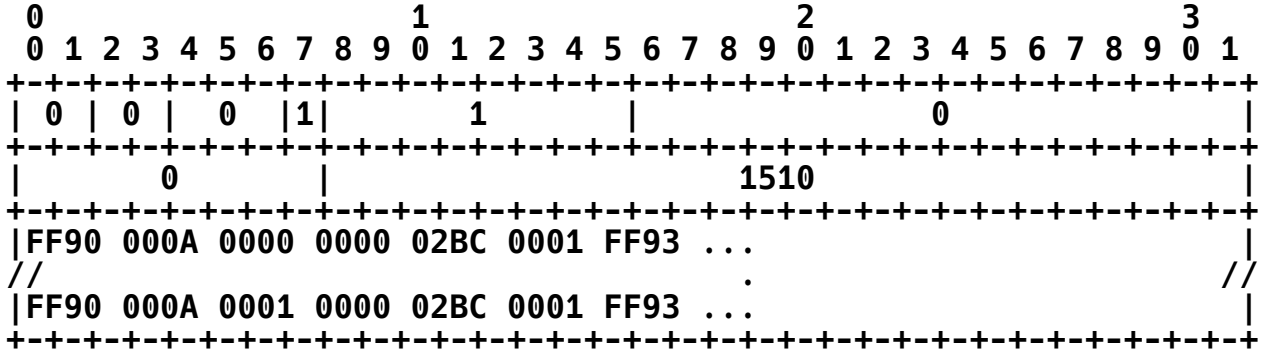

Figure 14: Header Sample 3-3 (Third Packet)

Fourth Packet: This packet has one tile, tile 2 1395 bytes

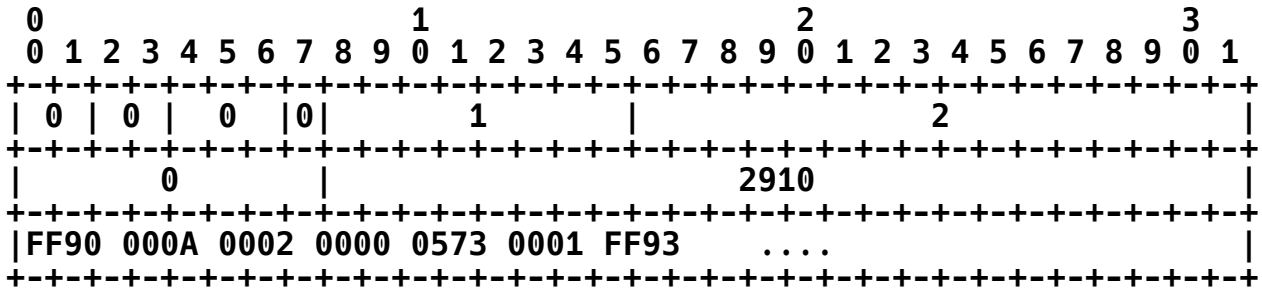

Figure 15: Header Sample 3-4 (4th Packet)

A.4. Sample 4: Interlace Image, Single Tile

The codestream of each image is ordered in LRCP (Layer, Resolution, Component, Position) with 1 layer, 3 resolutions, 3 components and 1 position.

First packet: This packet will have the whole main header for the odd field 210 bytes

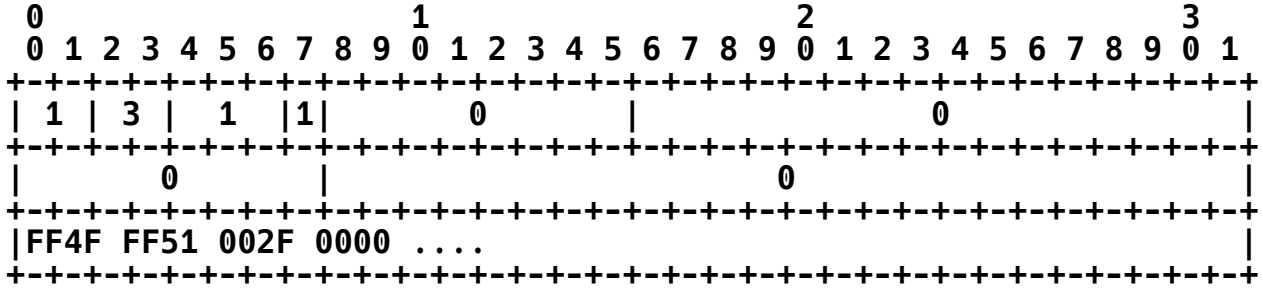

Figure 16: Header Sample 4-1 (First Packet)

Second packet: This packet will have the first part of the odd field's tile where three jp2-packets are included 1400 bytes

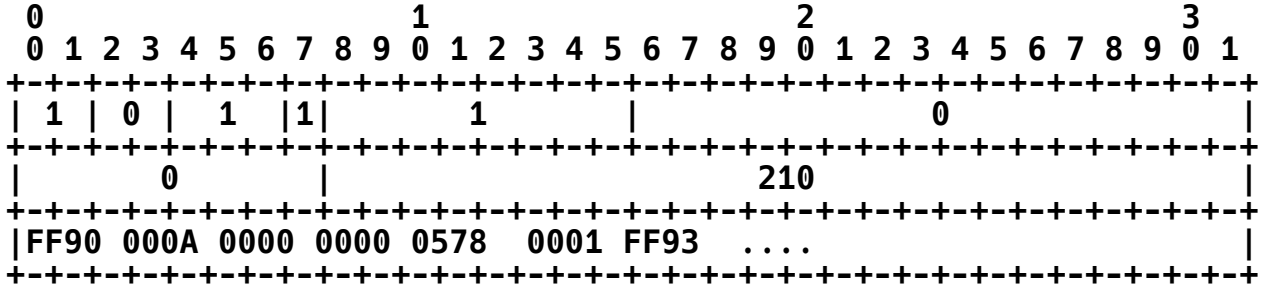

Figure 17: Header Sample 4-2 (Second Packet)

Third packet: This packet will have the second part of the odd field's tile 1400 bytes

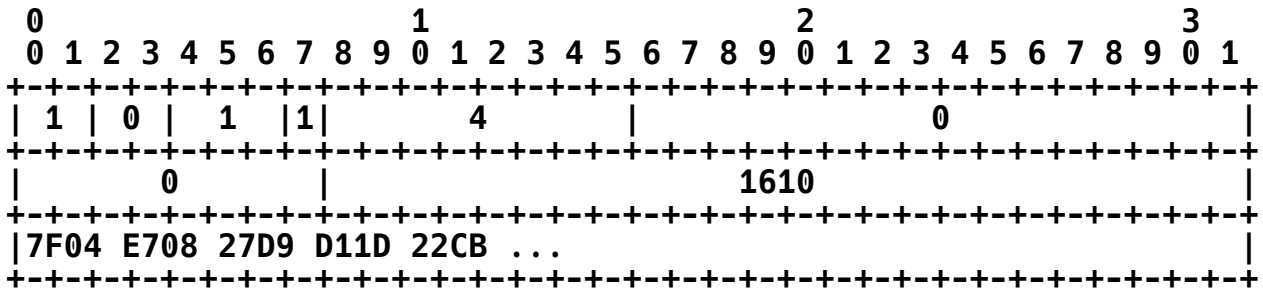

Figure 18: Header Sample 4-3 (Third Packet)

Fourth packet: This packet will have the third part of the odd field's tile 1300 bytes

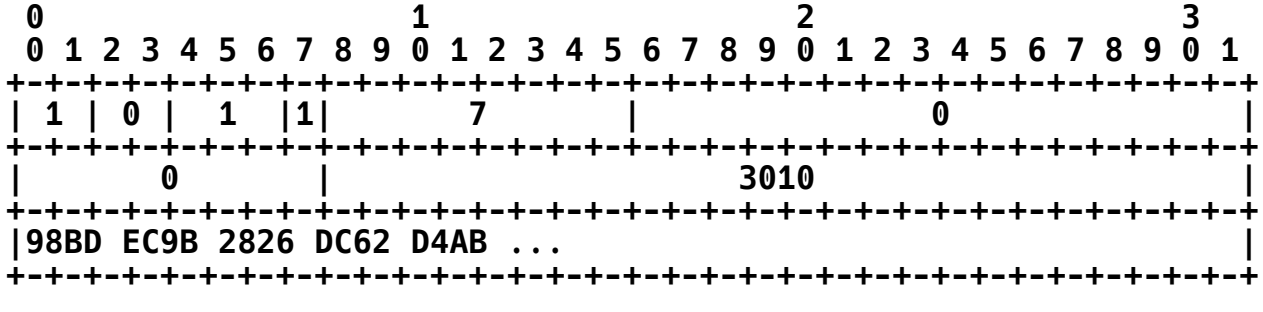

Figure 19: Header Sample 4-4 (4th Packet)

Fifth packet: This packet will have the whole main header for the even field 210 bytes

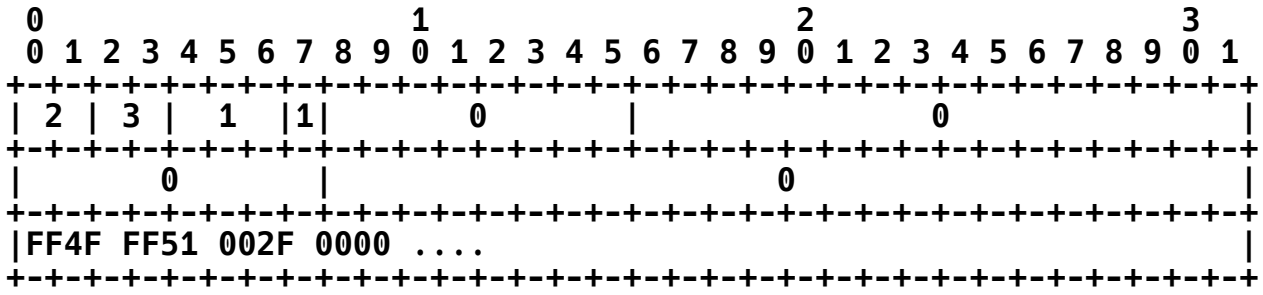

Figure 20: Header Sample 4-5 (5th Packet)

Sixth packet: This packet will have the first part of the even field's tile 1400 bytes

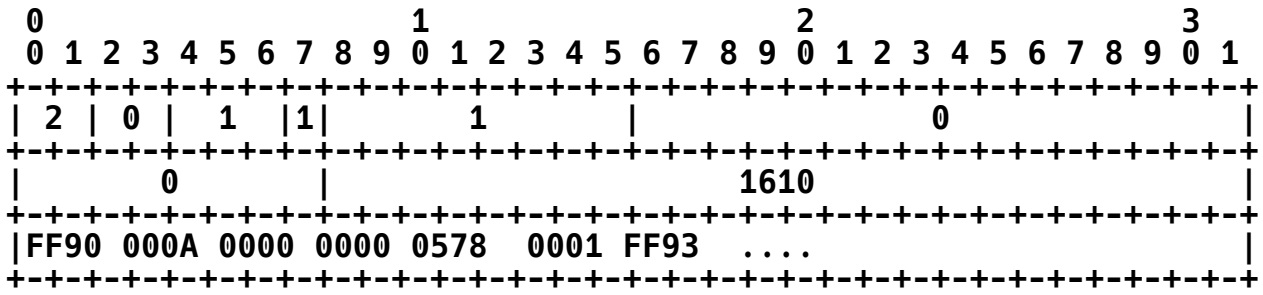

Figure 21: Header Sample 4-6 (6th Packet)

Seventh packet: This packet will have the second part of the even field's tile 1400 bytes

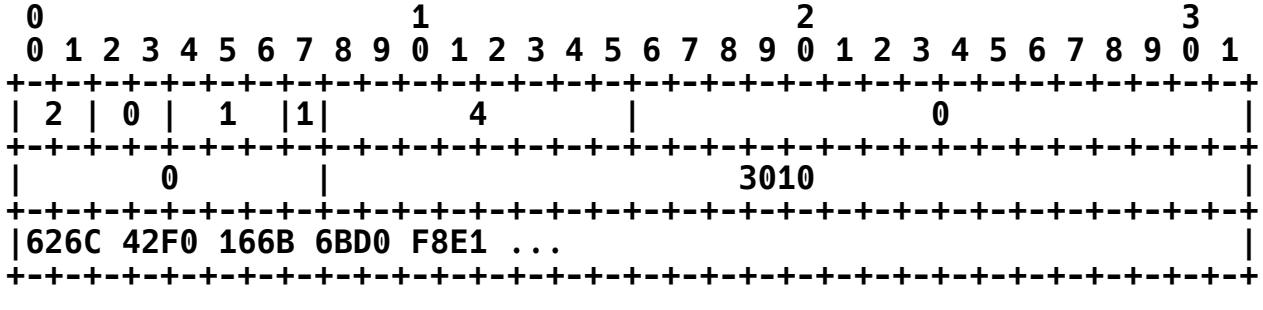

Figure 22: Header Sample 4-7 (7th Packet)

Eighth packet: This packet will have the third part of the even field's tile 1300 bytes

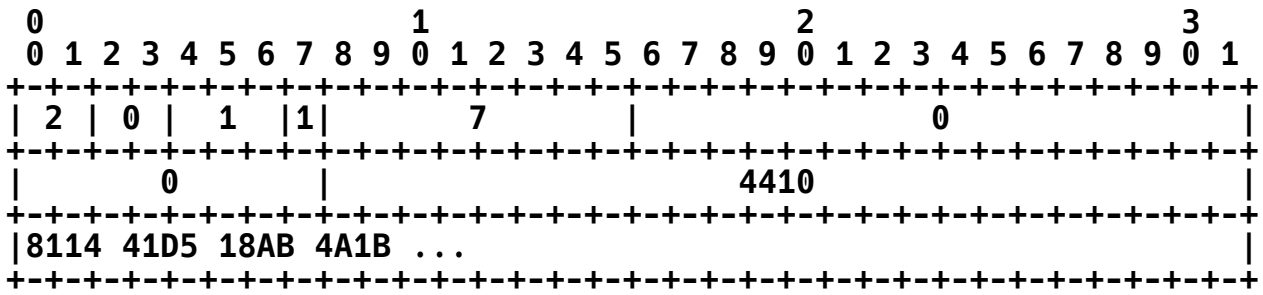

Figure 23: Header Sample 4-8 (8th Packet)

Authors' Addresses

Andrew Leung
Sony Corporation

EMail: andrew@ualberta.net

Satoshi Futemma
Sony Corporation
1-7-1 Konan
Minato-ku
Tokyo 108-0075
Japan

Phone: +81 3 6748-2111
EMail: satosi-f@sm.sony.co.jp
URI: http://www.sony.net/

Eisaburo Itakura
Sony Corporation
1-7-1 Konan
Minato-ku
Tokyo 108-0075
Japan

Phone: +81 3 6748-2111
EMail: itakura@sm.sony.co.jp
URI: http://www.sony.net/

Full Copyright Statement

Intellectual Property