

IEEE 1722 Control Format Generic Image Sensor Transport

2020-07-28, Rev 6 (Rev 5+graphics annotations)

Edited by Yong Kim

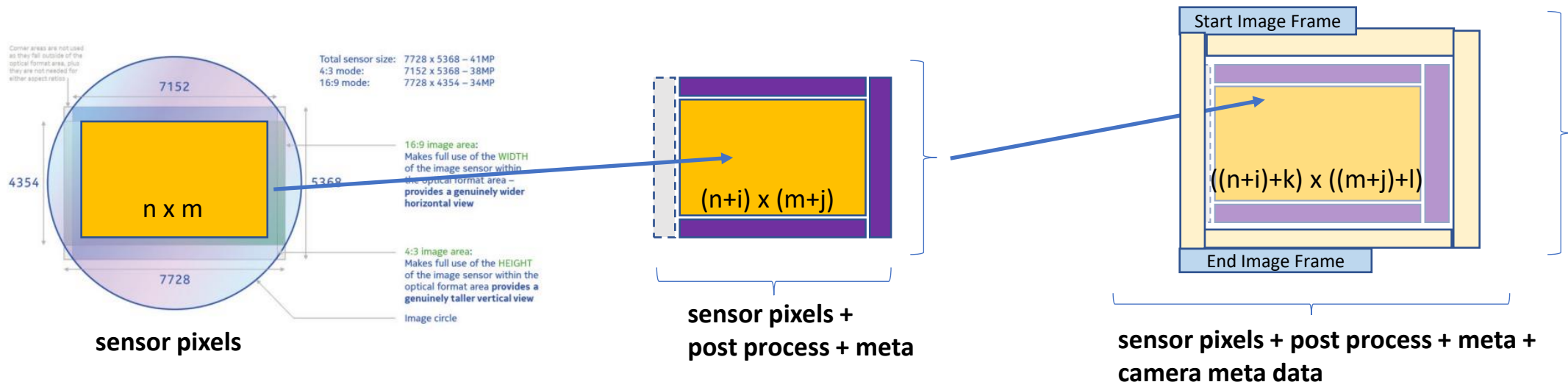
yongkim at axonne dot com

Motivation

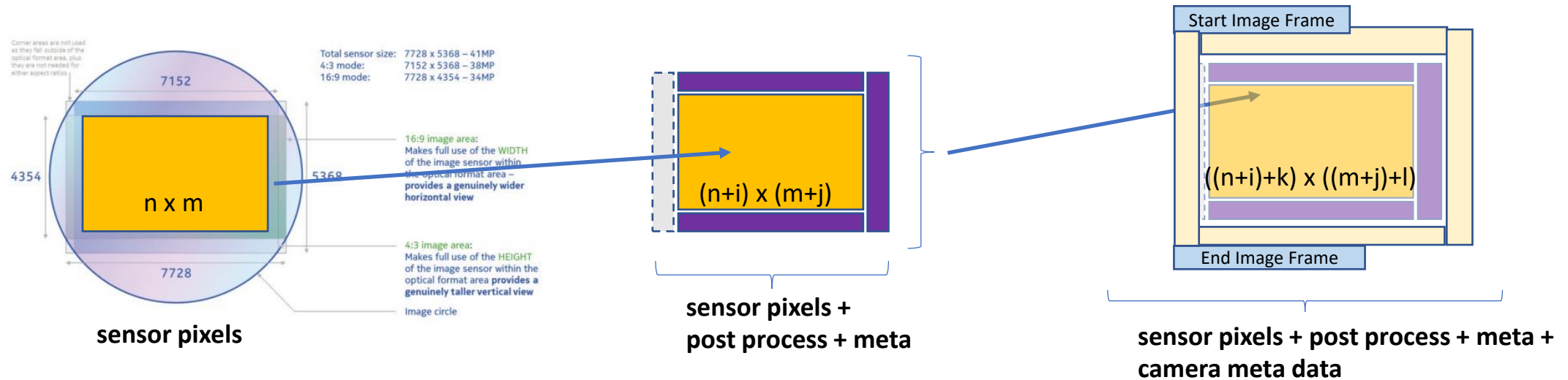
- One of the major justifications of automotive Multi-gig Ethernet PHY (IEEE 802.3ch) is network connection of high bandwidth autonomous drive (AD)/advanced driver assistance system (ADAS) sensors, such as uncompressed AD and parking cameras.
- Image sensors outputs $n \times m$ matrix of n pixels per line \times m lines.
- Image sensors meta-data outputs, often, are embedded in the extra row or columns.
- There is a need to transport image sensor outputs, row by row, over high-speed serial interface; P2P, or ideally real P2MP network, transport protocol that supports time sensitive networking, i.e. IEEE 1722 AVTP.
- Avoid pre-standard and proprietary implementation islands that form once system productions start; Avoid artificial turf defense when these form. There is no intrinsic value to having multiple different but functionally equivalent approaches.
- Avoid compatibility issues, implement transparent transport, i.e. bits in, bits out, layered.
- Stated goal of 1722b – serve the industry and perform necessary revision quickly – fits well with a goal of this proposal.

Cameras used in ADAS and Autonomous Drive

- 1 MP, 1.3MP, 2MP, and 8 MP, often leveraged from consumer electronics R&D, e.g. cell phones.
 - The same image sensors but instead of RGBG (Red, Green, Blue, Green) filter lens, it may use alternate color filters, RCCC (Red, Clear, Clear, Clear), or other combinations that are optimized for machine vision and object detection.
 - Camera sensor wrapped around camera system SoC may outputs to a number of standard interfaces.
 - Many of the ADAS/AD camera system leverages the mobile camera (integrated Sensor + system SoC) R&D.
- Transport Considerations
 - Camera sensors output what it sees in various configurable formats
 - Camera system SoC output what it processes in various configurable formats
 - Transport should just transport bits, bytes. In camera terms, [frame start] [lines]...[lines][frame end].



1722b to transport image data - image sizes

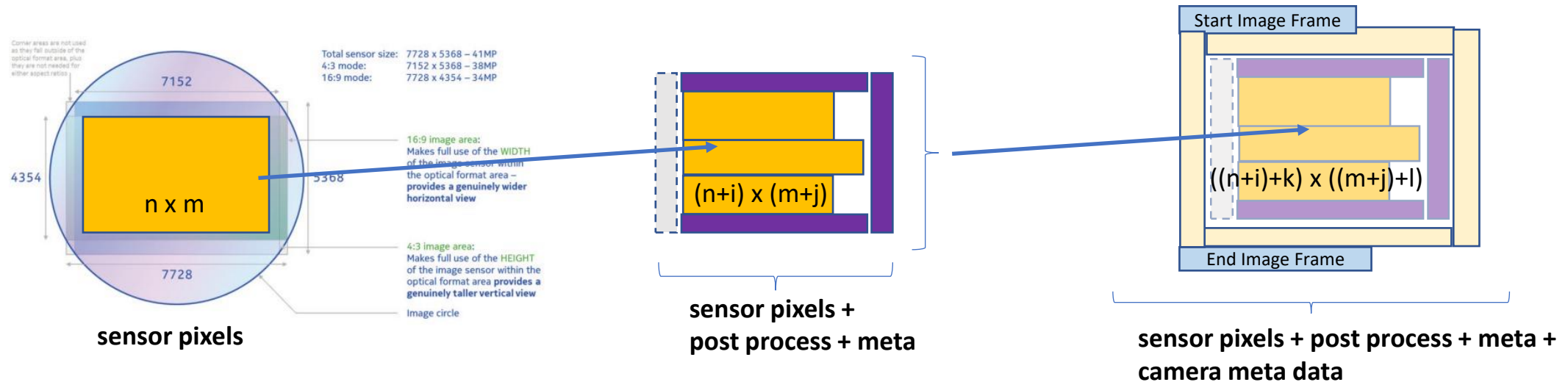


- Image sensor + lens + filter
- Just accept $n \times m$ pixels of some resolution
- No need to know, don't care about content (from transport perspective)

- Pixels has meaning (color space, depth, HDR processed, etc. whatever).
- Added pixel space may be added for meta data that increases $n \times m$ to be a bit bigger.
- No need to know, don't care about content (from transport perspective)

- Transparent transport of the previous content.
 - Start Image Frame
 - Line by line
 - End Image Frame.
 - May have added metadata but not distinguishable.

1722b to transport image data - discuss



- Image sensor or lens filter
- Just accept $n \times m$ pixels of some resolution
- No need to know, don't care about content (from transport perspective)

- Each image is on its own.
- Frame, Line, .. Line, Frame
- Don't assume fixed image size/depth.
- Each image type is separate stream (e.g allow for interspersed raw and HDR images, or dual image sensors onto single transport.)

- Each frame is on its own.
- FS, 'embedded', ... , line, ..., embedded', FE
- Don't assume fixed image size/depth.
- Transparent transport of packet header – issues?

1722b to transport image data – requirements

Background

- Video Stream format is optimized for rendering device, e.g. TV
 - Presentation Time → renderer's use.
- Example: Brake actuator needs presentation time – currently in time-sensitive header.
- ACF format allows for input data timestamp, e.g. Lin message time stamp, camera shutter.
 - ACF time → sensor input's use.
- Example: Image sensor needs input time – currently only in ACF

Some Requirements for potential 1722 transport.

- Format with input time (e.g. as in existing ACF sub-type message timestamp)
- Raw video format that supports variable length lines (i.e multiple rectangles in a single frame).

1722b to transport image data – considerations

- Raw image : e.g. not i.e., 4:4:4, 4:4:4:4, ...,
- ‘Improved image’: e.g. 8b(4:2:2), 10b(4:4:2), 12b(4:4:4), 16b(5:6:5)...
- Resolutions: e.g. 1MP, 1.3MP, 2MP, 4MP, 8MP,...

- Line Lengths:

pixel/line	+8	12 bit/pixel in bytes	16 bit/pixel in bytes
1280	1288	1932	2576
1920	1928	2892	3856
2560	2568	3852	5136
3840	3848	5772	7696

- Should allow for transparent transport of a line of pixel bytes in multiple Ethernet frames and reduce max latency for other TSN flows w/o use of preemption.
- Be aware of robustness of CRC32 and payload – avoid use of jumbo.
- For those who loves linear math and error functions, see referenced in: http://www.ieee802.org/3/bj/public/jul12/cideciyan_01_0712.pdf

1722 Transport sub-type considerations - detail

[illegible]

-
- The diagram illustrates the structure of an AVTP packet, organized into several fields with their corresponding bit positions (0 to 31):
- subtype data** (bits 0-31): Contains fields like *subtype*, *sv*, *version*, *mr*, *f_s_d*, *tv*, *sequence_num*, and *format_specific_data_1*.
 - Stream ID** (bits 0-31): A field for identifying the stream.
 - AVTP Time** (bits 0-31): A field for timestamping.
 - Format Specific** (bits 0-31): A field for format-specific data, including *format_specific_data_2*.
 - Packet Info** (bits 0-31): Contains *stream_data_length (octets)* and *format_specific_data_3*.
 - AVTP Payload** (bits 0-31): The main data area, labeled *stream_data_payload (additional header and data - varies by format)*.

e.g. Audio, Video, TS Control Format

Diagram illustrating the AVTP Header structure (12 octets total):

- Octets 0-3: subtype data (0-3)
- Octets 4-5: Stream ID (4-5)
- Octets 6-9: control_data_payload (6-9)
- Octets 10-11: control_data_payload (10-11)

Additional information: control_data_payload (additional header and data - varies by format)

e.g. 1722.1 DCCC format

Diagram illustrating the Alternative Data Payload Format structure:

- The format is a 32-bit structure.
- The header consists of three fields:
 - subtype** (bits 0-7)
 - h** (bits 8-9)
 - version** (bits 10-15)
- The remaining 16 bits (bits 16-31) are labeled **alternative_data_payload** (additional header and data - varies by format).
- Examples of alternative data payload formats include:
 - e.g. Non-TS Control Format; encrypted; etc

Figure 13—Alternative AVTPDU header

IEEE 1722 Raw Video Sub-type – summary

- Excluding IEC 61883/IIDC format, Compressed Video Format (CVF), SDI Video Format (SVF), leaves just RAW Video Format (RVF)
 - IEC61883/IIDC format has color space and resolution – but camera use case no relevance to IEC61883.
 - Compressed Video Format has MJPEG, H.264, JPEG2000 (we should add H.265 in 1722b) -- mild relevancy to compressed (e.g. some Parking camera)
 - SDI Video Format has fixed raster structure (user space available but not good enough).
- RAW Video Format
 - Stream_id – G
 - Avtp_[presentation]timestamp – NM (not meaningful)
 - Pixel depth – 0x0, 0x5~0xE (reserved), 0xF (user) – total of 11 resv + 1 – NG but ok to not use.
 - Pixel format – 0x5, 0xC~0xE (reserved), 0xF (user) – total of 4 resv + 1 – NG but ok to not use.
 - Frame rate – 16 bit field (20 used, 1 user, rest reserved).
 - Color space - 0x0, 0xA~0xE (reserved), 0xF (user) – total of 6 resv + 1 – NM (not meaningful)
 - Num_lines – 4 bit -- # of whole lines in this PDU – N/A (not applicable)
 - Line number & total line – 16 bit (good for 64K lines x 113K across for 9x16) – G
 - Active Pixels – 16 bits – fixed for the frame (every line) – NG
 - i_seq_num – sequence of frames for partial lines – G
 - Flags: ap, f, ef, evt, pd, l – extend to other respective fields valid.

Excellent template for the generic image sensor transport over AVTP Control Formats (ACF) [that supports origination timestamp]

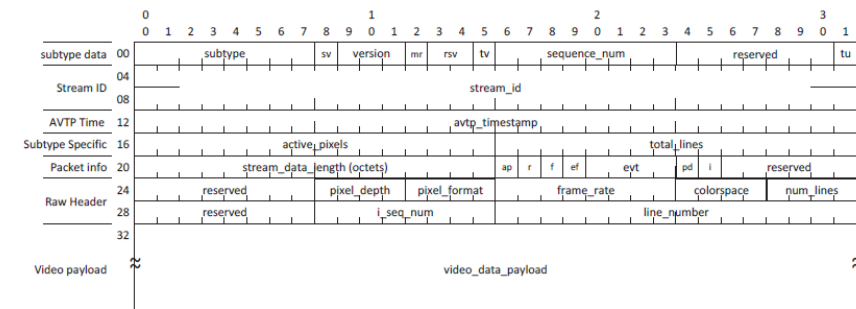


Figure 78—Raw Video PDU Format

AVTP Control Format Summary (9, Table 22)

- ACF Subtypes: [Stream | Alternative]
 - Stream:** stream_id, avtp_[presentation]timestamp, etc ← TS ACF (TSCF)
 - Alternative:** no stream_id, no avtp_[presentation]ts, etc. ← NTS ACF (NTSCF)
- Alternative Subtype: “Steering input – origination timestamp”,
Stream Subtype: “Wheel angle actuation – presentation timestamp”.
... but presentation timestamp could instead be in the alternative & ACF msg type.

Table 22—ACF message types

Value	Name	Description	Subclause
00 ₁₆	ACF_FLEXRAY	FlexRay™ message	9.4.2
01 ₁₆	ACF_CAN	Controller Area Network (CAN)/CAN with Flexible Data-Rate (CAN FD) message	9.4.3
02 ₁₆	ACF_CAN_BRIEF	Abbreviated CAN/CAN FD message	9.4.4
03 ₁₆	ACF_LIN	LIN® message	9.4.5
04 ₁₆	ACF_MOST	MOST® message	9.4.6
05 ₁₆	ACF_GPC	General purpose control message	9.4.7
06 ₁₆	ACF_SERIAL	Serial port message	9.4.8
07 ₁₆	ACF_PARALLEL	Parallel port message	9.4.9
08 ₁₆	ACF_SENSOR	Analog sensor message	9.4.10
09 ₁₆	ACF_SENSOR_BRIEF	Abbreviated sensor message	9.4.11
0A ₁₆	ACF_AECP	IEEE Std 1722.1 AECP message	9.4.12
0B ₁₆	ACF_ANCILLARY	Video ancillary data message	9.4.13
0C to 7F ₁₆	Reserved	Reserved	—
78 to 7F ₁₆	ACF_USER	User-defined ACF message	—

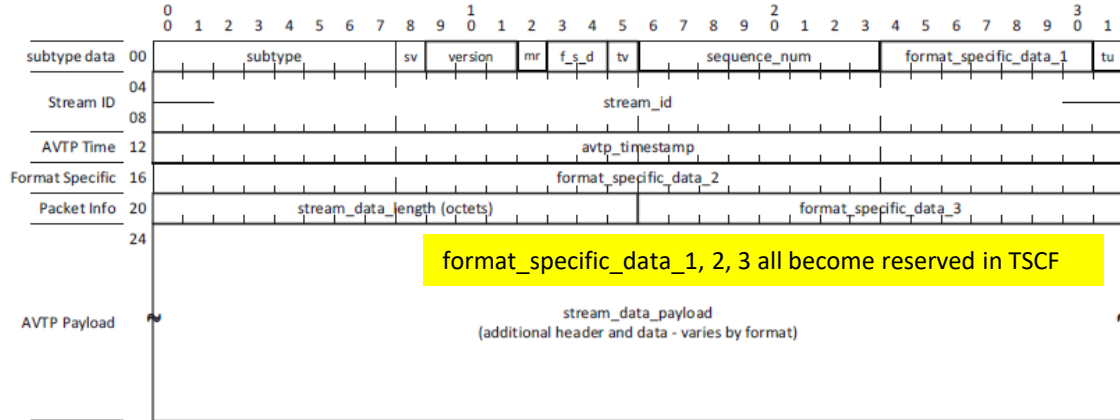


Figure 9—AVTPDU common stream header

e.g. Audio, Video, TS Control Format

AVTP TS and Non-TS Control Format subtypes has “origination” time-stamps, differentiating “presentation” time-stamps.

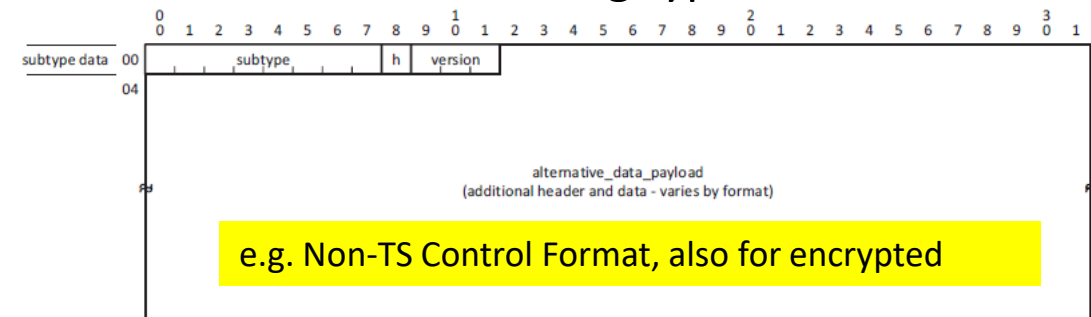


Figure 13—Alternative AVTPDU header

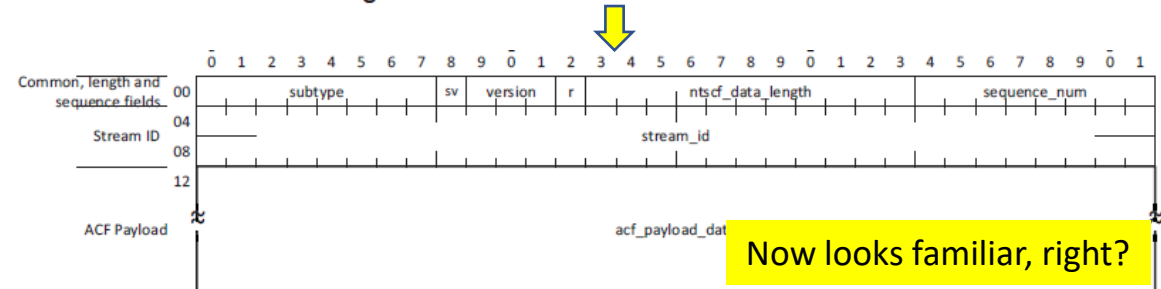
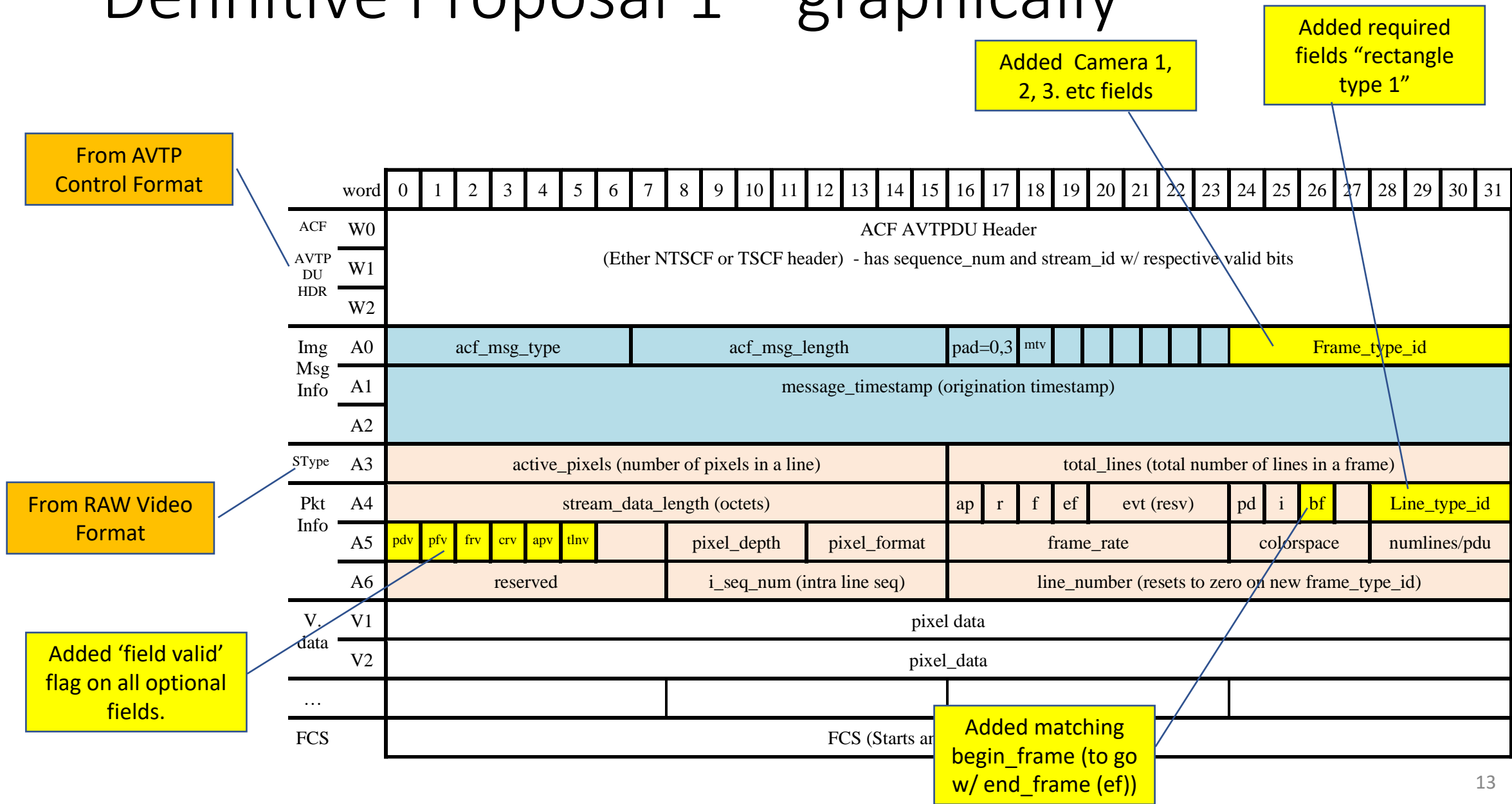


Figure 50—Non-Time-Synchronous Control Stream PDU Format

Definitive Proposal – toward agreement

- Subtype: ACF – NTSCF. Discuss whether to allow TSCF.
- Indicate: Frame Start, Frame End, Line Start (implicit), Line # (w/ its valid flag), Line Sequence #, Line End (implicit).
 - Indicate: Frame Type (if multiple frame types are interspersed in a same avtp stream), for example single stream that may serve multiple inter-related camera outputs on single avtp stream.
 - Indicate: Line Type (“rectangle type 1”, “rectangle type 3”, etc)
- Indicate: Frame Origination Time Stamp – must be valid on Frame Start. May increment and vary for each of the lines (for non-global shutter imagers (rare))
- Color space, pixel depth, pixel format, -- copy from RAW format but add respective valid flags. -- thus making them allowed and optional -- further definition may not add further value.
- Controversial?
 - **Proposal**: Do NOT allow ACF_payload concatenation of more ACF_payload and must be the 1st ACF_payload_data. For ease of implementation and optimum MTU size consideration.
 - **Counter Proposal**: If ACF_payload concatenation is not allowed (TLV fashion), then why NOT use RAW as the basis and add origination timestamp and may any other required modifications?
 - **Discuss**.

Definitive Proposal 1 – graphically



Definitive Proposal 1 – annotated

12.2.9 pd (pull-down) field

The **pd** (pull-down) field is used to indicate a pull-down of the video frame rate. When **pd** is set to one (1), a multiplier ratio of 1000/1001 is applied to the frame rate specified by the **frame_rate** (12.2.13) field. When **pd** is zero (0) there is no pull-down applied to the frame rate (i.e., a 1/1 ratio). For example when **pd** is set to one (1) and the video frame rate is 30 Hz, the frame rate becomes $30 \text{ Hz} \times 1000/1001 = 29.97003 \text{ Hz}$.

12.2.5 ap (active pixels) field

The **ap** (active pixels) bit is used as an identifier to mark the video lines that contain active pixels. The **ap** bit is set to one (1) for all RVF AVTPDUs that contain active area pixels. The **ap** bit is set to zero (0) for all RVF AVTPDUs that contain pixels outside the active area (see Figure 79).

Each RVF AVTPDU shall have either all active pixels (**ap** is set) or all vertical blanking pixels (**ap** is not set). A mixture of active and blanking lines within the same RVF AVTPDU is not permitted.

12.2.4 total_lines field

The **total_lines** field indicates the total number of lines of the video frame. For interlaced formats, the total number of lines should be the total of both video fields. The total number of lines can be higher than the active video lines to include vertical blanking lines.

word	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
ACF	W0																																
AVTP	W1																																
DU	W1																																
ACF AVTPDU Header																																	
(Ether NTSCF or TSCF header) - has sequence_num and stream_id w/ respective valid bits																																	
active pixels) field																																	
ve pixels) bit is used as an identifier to mark the video lines that contain active pixels. The ap																																	
ne (1) for all RVF AVTPDUs that contain active area pixels. The ap bit is set to zero (0) for all																																	
DUs that contain pixels outside the active area (see Figure 79).																																	
AVTPDU shall have either all active pixels (ap is set) or all vertical blanking pixels (ap is not																																	
re of active and blanking lines within the same RVF AVTPDU is not permitted.																																	
12.2.4 total_lines field																																	
The total_lines field indicates the total number of lines of the video frame. For interlaced fo																																	
number of lines should be the total of both video fields. The total number of lines can be																																	
active video lines to include vertical blanking lines.																																	
SType	A3	active_pixels (number of pixels in a line)															total_lines (total number of lines in a frame)																
Pkt	A4	stream_data_length (octets)															ap	r	f	ef	evt (resv)					pd	i	bf	Line_type_id				
Info	A5	pdv	pfv	frv	crv	apv	tlv						pixel_depth			pixel_format			frame_rate					colorspace					numlines/pdu				
	A6																																
V. data	V1																																
	V2																																
...																																	
FCS																																	

Table 40 —pixel_depth values			
Value	Pixel depth (bits)		
0 ₁₆	Reserved		
1 ₁₆	8		
2 ₁₆	10		
3 ₁₆	12		
4 ₁₆	16		
5 to E ₁₆	Reserved		
F ₁₆	User defined		

Table 41 —pixel_format values	
Value	Pixel format
0 ₁₆	Monochrome
1 ₁₆	4:1:1
2 ₁₆	4:2:0
3 ₁₆	4:2:2
4 ₁₆	4:4:4
5 ₁₆	Reserved
6 ₁₆	4:2:2:4
7 ₁₆	4:4:4:4
8 ₁₆	Bayer grbg
9 ₁₆	Bayer rggb
A ₁₆	Bayer gbgr
B ₁₆	Bayer bgrg

Table 42 —frame_rate field values	
Value	Video frame rate (Hz)
0 ₁₆	Reserved
1 ₁₆	1
2 ₁₆	2
3 ₁₆	5
4 to F ₁₆	Reserved
10 ₁₆	10
11 ₁₆	15
12 ₁₆	20
13 ₁₆	24
14 ₁₆	25
15 ₁₆	30

Table 43 —colorspace field values	
Value	Description
0 ₁₆	Reserved
1 ₁₆	YCbCr [B19], [B21], [B22]
2 ₁₆	sRGB [B10]
3 ₁₆	YCgCo
4 ₁₆	Grayscale
5 ₁₆	XYZ [B14]
6 ₁₆	YCM
7 ₁₆	BT Rec.601 [B19]
8 ₁₆	BT Rec.709 [B21]
9 ₁₆	ITU BT 2020 [B22]

Table 40 —pixel_depth values

Value	Pixel depth (bits)
0 ₁₆	Reserved
1 ₁₆	8
2 ₁₆	10
3 ₁₆	12
4 ₁₆	16
5 to E ₁₆	Reserved
F ₁₆	User defined

Table 41 —pixel_format values

Value	Pixel format
0 ₁₆	Monochrome
1 ₁₆	4:1:1
2 ₁₆	4:2:0
3 ₁₆	4:2:2
4 ₁₆	4:4:4
5 ₁₆	Reserved
6 ₁₆	4:2:2:4
7 ₁₆	4:4:4:4
8 ₁₆	Bayer grbg
9 ₁₆	Bayer rggb
A ₁₆	Bayer bggr
B ₁₆	Bayer gbrg
C to E ₁₆	Reserved
F ₁₆	User defined

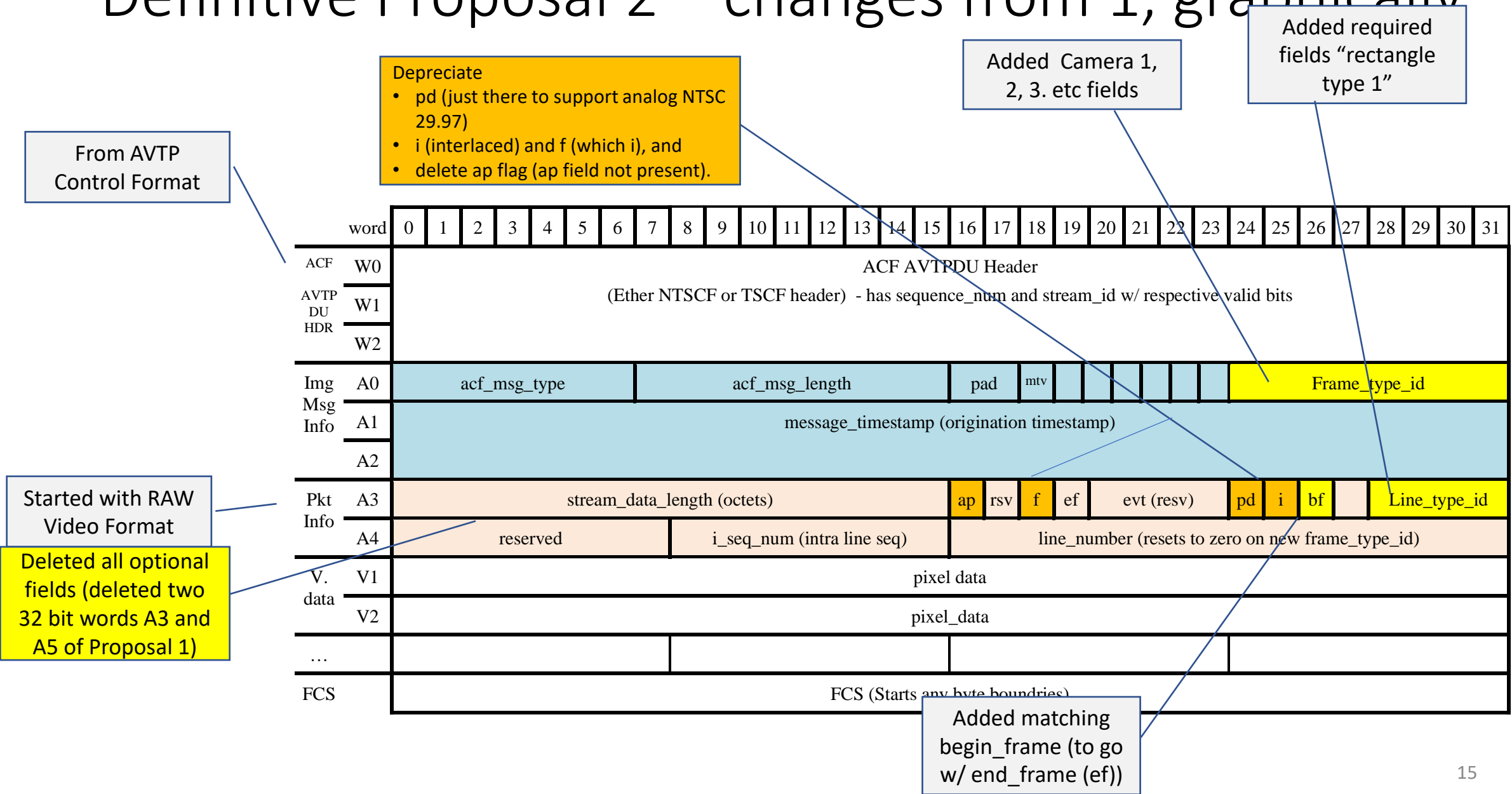
Table 42 —frame_rate field values

Value	Video frame rate (Hz)
0 ₁₆	Reserved
1 ₁₆	1
2 ₁₆	2
3 ₁₆	5
4 to F ₁₆	Reserved
10 ₁₆	10
11 ₁₆	15
12 ₁₆	20
13 ₁₆	24
14 ₁₆	25
15 ₁₆	30
16 ₁₆	48
17 ₁₆	50
18 ₁₆	60

Table 43 —colorspace field values

Value	Description
0 ₁₆	Reserved
1 ₁₆	YCbCr [B19], [B21], [B22]
2 ₁₆	sRGB [B10]
3 ₁₆	YCbCo
4 ₁₆	Grayscale
5 ₁₆	XYZ [B14]
6 ₁₆	YCM
7 ₁₆	BT Rec.601 [B19]
8 ₁₆	BT Rec.709 [B21]
9 ₁₆	ITU BT 2020 [B22]
A to E ₁₆	Reserved
F ₁₆	User defined

Definitive Proposal 2 – changes from 1, graphically



Discuss and build consensus

- AVTP timestamp always has been used as a presentation timestamp. Changing this may be confusing.
- AVTP origination timestamp had been introduced for automotive (and industrial) sensor data over AVTP. Therefore AVTP Control Formats were defined.
- RAW Video format has many relevant fields, but some are not appropriate (as in number of pixels per line shall not change in a frame).
- Introduce a new image sensor format that parallels RAW Video format but under AVTP Control Format, where it is easy to be found (sensors).
- Discuss -