IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF MASSACHUSETTS

REALTIME ADAPTIVE STREAMING LLC, Plaintiff, v. ADOBE SYSTEMS INC., Defendant.	Case No. ____________________ JURY TRIAL DEMANDED

COMPLAINT FOR PATENT INFRINGEMENT

This is an action for patent infringement arising under the Patent Laws of the United States of America, 35 U.S.C. § 1 *et seq.* in which Plaintiff Realtime Adaptive Streaming LLC ("Plaintiff" or "Realtime") makes the following allegations against Defendant Adobe Systems Inc. ("Defendant" or "Adobe").

PARTIES

1. Realtime is a Texas limited liability company. Realtime has a place of business at 1828 E.S.E. Loop 323, Tyler, Texas 75701.

2. On information and belief, Defendant Adobe Systems Inc. is a Delaware corporation with its principal place of business at 345 Park Ave, San Jose, California 95110.

3. Defendant Adobe Systems Inc. has regular and established places of business in this District, specifically at least at One Newton Place, Newton, MA 02458 and One Broadway, Cambridge, MA 02142. Defendant Adobe Systems Inc. offers their products and/or services, including those accused herein of infringement, to customers and potential customers located in Massachusetts and in this District.

4. Defendant Adobe Systems Inc. may be served with process through its registered agent for service c/o Corporation Service Company at 84 State Street, Boston,

MA 02109.

JURISDICTION AND VENUE

5. This action arises under the patent laws of the United States, Title 35 of the United States Code. This Court has original subject matter jurisdiction pursuant to 28 U.S.C. §§ 1331 and 1338(a).

6. This Court has personal jurisdiction over Defendant in this action because Defendant has committed acts within the District of Massachusetts giving rise to this action and has established minimum contacts with this forum such that the exercise of jurisdiction over Defendant would not offend traditional notions of fair play and substantial justice. The Defendant has also committed and continues to commit acts of infringement in this District by, among other things, offering to sell and selling products and/or services that infringe the asserted patents.

7. Venue is proper in this district, *e.g.*, under 28 U.S.C. § 1400(b). Defendant has a regular and established place of business in this District, for example at One Newton Place, Newton, MA 02458 and One Broadway, Cambridge, MA 02142. Furthermore, upon information and belief, Defendant has transacted business in the District of Massachusetts and has committed acts of direct and indirect infringement in the District of Massachusetts.

THE PATENTS-IN-SUIT

8. This action arises under 35 U.S.C. § 271 for Adobe's infringement of Realtime's United States Patent Nos. 7,386,046 (the "'046 patent"), 8,634,462 (the "'462 patent"), 8,929,442 (the "'442 patent"), 8,934,535 (the "'535 patent"), 9,578,298 (the "'298 patent"), 9,762,907 (the "'907 patent"), and 9,769,477 (the "'477 patent") (collectively, the "Patents-In-Suit").

9. The ’046 patent, titled “Bandwidth Sensitive Data Compression and Decompression,” was duly and properly issued by the United States Patent and Trademark Office (“USPTO”) on June 10, 2008. A copy of the ’046 patent is attached hereto as Exhibit A. Realtime is the owner and assignee of the ’046 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

10. The ’462 patent, titled “Quantization for Hybrid Video Coding,” was duly and properly issued by the USPTO on January 21, 2014. A copy of the ’462 patent is attached hereto as Exhibit B. Realtime is the owner and assignee of the ’462 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

11. The ’442 patent, titled “System and method for video and audio data distribution,” was duly and legally issued by the USPTO on January 6, 2015. A true and correct copy of the ’442 patent is included as Exhibit C. Realtime is the owner and assignee of the ’442 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

12. The ’535 patent, titled “Systems and methods for video and audio data storage and distribution,” was duly and properly issued by the USPTO on January 13, 2015. A copy of the ’535 patent is attached hereto as Exhibit D. Realtime is the owner and assignee of the ’535 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

13. The ’298 patent, titled “Method for Decoding 2D-Compatible Stereoscopic Video Flows,” was duly and properly issued by the USPTO on February 21, 2017. A copy of the ’298 patent is attached hereto as Exhibit E. Realtime is the owner and assignee of

the ’298 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

14. The ’907 patent, titled “System and Methods for Video and Audio Data Distribution,” was duly and properly issued by the USPTO on September 12, 2017. A copy of the ’907 patent is attached hereto as Exhibit F. Realtime is the owner and assignee of the ’907 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

15. The ’477 patent, titled “Video data compression systems,” was duly and properly issued by the USPTO on September 19, 2017. A copy of the ’477 patent is attached hereto as Exhibit G. Realtime is the owner and assignee of the ’477 patent and holds the right to sue for and recover all damages for infringement thereof, including past infringement.

COUNT I

INFRINGEMENT OF U.S. PATENT NO. 7,386,046

16. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

17. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States Adobe products that infringe the ‘046 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe’s Adobe Media Encoder, Adobe Media Encoder Creative Cloud or CC, Adobe Media Encoder Creative Suite or CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Premiere Pro, Adobe Premiere Pro CC, Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Flash, Adobe Flash Player (e.g. Adobe Flash Player 9), Adobe Flash

Media Server, Adobe Flash Media Encoding Server, Adobe After Effects, Adobe After Effects CC, Adobe After Effects CS (e.g. CS4, CS5, CS5.5, CS6), Adobe HTTP Dynamic Streaming (HDS), and all versions and variations thereof since the issuance of the ’046 patent (“Accused Instrumentalities”).

18. On information and belief, Adobe has directly infringed and continues to infringe the ‘046 patent, for example, through its sale, offer for sale, importation, use and testing of the Accused Instrumentalities, which practices the system claimed by, for example, Claim 40 of the ‘046 patent, namely, a system, comprising: a data compression system for compressing and decompressing data input; a plurality of compression routines selectively utilized by the data compression system, wherein a first one of the plurality of compression routines includes a first compression algorithm and a second one of the plurality of compression routines includes a second compression algorithm; and a controller for tracking throughput and generating a control signal to select a compression routine based on the throughput, wherein said tracking throughput comprises tracking a number of pending access requests to a storage device; and wherein when the controller determines that the throughput falls below a predetermined throughput threshold, the controller commands the data compression engine to use one of the plurality of compression routines to provide a faster rate of compression so as to increase the throughput. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to Adobe’s customers.

19. The Accused Instrumentalities include, or practice a system, comprising: a

data compression system for compressing and decompressing data input. For example, the Accused Instrumentalities utilize H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec. H.264 is "the most widely used codec on the planet." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-H.264-74735.aspx. A "codec" is also a compression technology that has "two components, an encoder to compress the files, and a decoder to decompress. There are codecs for…video (Cinepark, MPEG-2, **H.264**, VP8)." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-a-Codec-74487.aspx; https://forums.adobe.com/thread/729526 ("Compression is essential for reducing the size of movies so that they can be stored, transmitted, and played back effectively. Compression is achieved by an encoder; decompression is achieved by a decoder. Encoders and decoders are known by the common term codec."). According to a website maintained by Adobe, "you **can produce H.264 video** with Adobe Media Encoder CS4 and Adobe Flash Media Encoding Server 3.5," "Adobe added **H.264 playback support** to Adobe Flash Player 9 Update 3 back in 2007," Adobe "Flash Media Server and Flash Player can **both stream and play back any H.264 file** in virtually any format," the encoding capabilities of the Adobe Flash Media Encoding Server using "**H.264**-related parameters" are described (e.g. using CAVLC, CABAC) and under a headline stating "Adobe Media Encoder," the website states: "Adobe significantly enhanced the Flash Video Encoder in Creative Suite 4. There is now both stand-alone operation and batch encoding capabilities. As before, you can **access H.264 encoding** by choosing different formats in the Format pop-up menu. When producing for Flash Player, you should always use the FLV|F4V option, which lets you produce both VP6- and

H.264-encoded files for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html. Another website maintained by Adobe mentions that "You have two primary options for using Adobe Media Encoder with After Effects to create videos in **H.264**." The website also mentions that "Since Adobe Media Encoder already has superior H.264…exporting capabilities, it was more prudent to rely on Adobe Media Encoder for export of these formats." *See* https://helpx.adobe.com/after-effects/kb/export-h264.html. Another website maintained by Adobe mentions that the "Adobe Flash Player 9 Update 3 is taking a step into the high-definition (HD) video realm in a major way by adding MPEG-4 video…[which] utilizes crisp, powerful **H.264 encoding**" and "The new Adobe Media Player…will **also support H.264**" and "The addition of **H.264**…**support** in Flash Player 9 Update 3 allows you to easily use high-definition, industry standard video and audio." *See* http://www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html. Adobe HTTP Dynamic Streaming (HDS) also supports additional data compression systems or codecs other than H.264 because it "enables high-quality (**H.264 or VP6**), network-efficient HTTP streaming for media delivery that is tightly integrated with Adobe® Access software for robust content protection in the Adobe Flash® Player 10.1 or later and Adobe AIR®2 or later runtimes," is also "built on standards and deployed using standard HTTP servers (Apache), standard media format (MP4 fragment) using standard codecs (**H.264/AAC**), open APIs (Flash Player), and an open source framework for building media players (OSMF)" and "like other Flash Player supported delivery methods," Adobe HDS "supports **H.264 video and VP6 codecs** required for the highest quality video on the platform." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS also allows users to encode content "using high-quality Flash Player compatible **codecs** (**VP6/MP3, H264/AAC**)," deliver "high-definition video up to 1080p, with bitrates from 700kbps up to and beyond 6Mbps, using either **H.264 or VP6 video codecs,"** and archive "live, high-definition streams on the server and enable HD DVR functionality (for example, instant replay and time shifting) with support for **H.264** stream recording for RTMP and HTTP Dynamic Streaming." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. Adobe Media Encoder also "lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html.

20. The Accused Instrumentalities also include, or practice a system comprising: a plurality of compression routines selectively utilized by the data compression system, wherein a first one of the plurality of compression routines includes a first compression algorithm and a second one of the plurality of compression routines includes a second compression algorithm. First, based on various parameters (e.g. throughput, bitrate, max video bitrate, resolution), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., "baseline," "extended," "main", or "high") corresponds with that parameter, whether if that parameter (e.g. throughput) meets some criteria, then select between at least two asymmetric compressors or plurality of compression routines, where one asymmetric compressor may serve as the first compression algorithm and another asymmetric compressor may serve as the second compression algorithm. If baseline or extended is the corresponding profile, then the system will select a Context-Adaptive Variable

Length Coding ("CAVLC") entropy encoder, which may serve as the first or second compression algorithm or asymmetric compressor. If main or high is the corresponding profile, then the system will select a Context-Adaptive Binary Arithmetic Coding ("CABAC") entropy encoder, which may serve as the first or second compression algorithm or asymmetric compressor. Both encoders are asymmetric compressors because it takes a longer period of time for them to compress data than to decompress data. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/.

21. The Accused Instrumentalities also include, or practice a system comprising: a controller for tracking throughput and generating a control signal to select a compression routine based on the throughput, wherein said tracking throughput comprises tracking a number of pending access requests to a storage device. For example, Adobe HTTP Dynamic Streaming (HDS) contains the "ability **to shift quality depending on bandwidth and computer power**. HTTP progressive delivery consumes more bandwidth because it's not intelligent enough to throttle the delivery (for example, a 30-minute video will be fully downloaded whether the user watches it or not)." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS can also **"[d]etect the client's bandwidth and computer resources** and serve them content fragments encoded at the most appropriate bitrate." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. On information and belief, the Accused Instrumentalities also include a storage device e.g., hard disk, disks, buffers, servers or other forms of memory/storage, that would receive pending access requests so the controller could track throughput by tracking a number of pending access requests to that storage device. *See* https://helpx.adobe.com/adobe-media-

server/dev/configure-dynamic-streaming-live-streaming.html (describing "Streams [from Adobe HDS] in a live event are packaged as fragments and written to **disk**," "The **IO buffer** loads the **disk file** into an **in-memory buffer**," and "To limit the amount of **storage the disk cache** uses," and a "Content storage (HDS and HLS)" header describing a section where a "media player **requests content from the server.**" As shown below, when the controller generates a control signal to select a compression routine based on the throughput, it is clear that the compression routines and the first and second compression algorithms all utilize various parameters to compress or decompress data input including, of course, throughput and bandwidth, but also bitrate (or max video bitrate), and resolution. Different parameters also correspond with different end applications. H.264, a data compression system, and compression routine provides for multiple different ranges of such parameters, each included in the "profiles" and "levels" as defined by the H.264 standard, from the below shown paragraphs from a white paper and Wikipedia. *See* http://www.axis.com/files/whitepaper/wp_h264_31669_en_0803_lo.pdf at 5:

4. H.264 profiles and levels

The joint group involved in defining H.264 focused on creating a simple and clean solution, limiting options and features to a minimum. An important aspect of the standard, as with other video standards, is providing the capabilities in profiles (sets of algorithmic features) and levels (performance classes) that optimally support popular productions and common formats.

H.264 has seven profiles, each targeting a specific class of applications. Each profile defines what feature set the encoder may use and limits the decoder implementation complexity.

Network cameras and video encoders will most likely use a profile called the baseline profile, which is intended primarily for applications with limited computing resources. The baseline profile is the most suitable given the available performance in a real-time encoder that is embedded in a network video product. The profile also enables low latency, which is an important requirement of surveillance video and also particularly important in enabling real-time, pan/tilt/zoom (PTZ) control in PTZ network cameras.

H.264 has 11 levels or degree of capability to limit performance, bandwidth and memory requirements. Each level defines the bit rate and the encoding rate in macroblock per second for resolutions ranging from QCIF to HDTV and beyond. The higher the resolution, the higher the level required.

See https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC:

Levels with maximum property values

Level	Max decoding speed		Max frame size		Max video bit rate for video coding layer (VCL) kbit/s			Examples for high resolution @ highest frame rate (max stored frames) Toggle additional details
	Luma samples/s	Macroblocks/s	Luma samples	Macroblocks	Baseline, Extended and Main Profiles	High Profile	High 10 Profile	
1	380,160	1,485	25,344	99	64	80	192	176×144@15.0 (4)
1b	380,160	1,485	25,344	99	128	160	384	176×144@15.0 (4)
1.1	768,000	3,000	101,376	396	192	240	576	352×288@7.5 (2)
1.2	1,536,000	6,000	101,376	396	384	480	1,152	352×288@15.2 (6)
1.3	3,041,280	11,880	101,376	396	768	960	2,304	352×288@30.0 (6)
2	3,041,280	11,880	101,376	396	2,000	2,500	6,000	352×288@30.0 (6)
2.1	5,068,800	19,800	202,752	792	4,000	5,000	12,000	352×576@25.0 (6)
2.2	5,184,000	20,250	414,720	1,620	4,000	5,000	12,000	720×576@12.5 (5)
3	10,368,000	40,500	414,720	1,620	10,000	12,500	30,000	720×576@25.0 (5)
3.1	27,648,000	108,000	921,600	3,600	14,000	17,500	42,000	1,280×720@30.0 (5)
3.2	55,296,000	216,000	1,310,720	5,120	20,000	25,000	60,000	1,280×1,024@42.2 (4)
4	62,914,560	245,760	2,097,152	8,192	20,000	25,000	60,000	2,048×1,024@30.0 (4)
4.1	62,914,560	245,760	2,097,152	8,192	50,000	62,500	150,000	2,048×1,024@30.0 (4)
4.2	133,693,440	522,240	2,228,224	8,704	50,000	62,500	150,000	2,048×1,080@60.0 (4)
5	150,994,944	589,824	5,652,480	22,080	135,000	168,750	405,000	3,672×1,536@26.7 (5)
5.1	251,658,240	983,040	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@26.7 (5)
5.2	530,841,600	2,073,600	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@56.3 (5)

22. A video data block is organized by the group of pictures (GOP) structure, which is a "collection of successive pictures within a coded video stream." *See* https://en.wikipedia.org/wiki/Group_of_pictures. A GOP structure can contain intra coded pictures (I picture or I frame), predictive coded pictures (P picture or P frame), bipredictive coded pictures (B picture or B frame) and direct coded pictures (D picture or D frames, or DC direct coded pictures which are used only in MPEG-1 video). *See* https://en.wikipedia.org/wiki/Video_compression_picture_types (for descriptions of I frames, P frames and B frames); https://en.wikipedia.org/wiki/MPEG-1#D-frames (for descriptions of D frames). Thus, at least a portion of a video data block would also make up a GOP structure and could also contain I frames, P frames, B frames and/or D frames.

The GOP structure also reflects the size of a video data block, and the GOP structure can be controlled and used to fine-tune other parameters (e.g. throughput/bandwidth, bitrate, max video bitrate and resolution parameters) or even be considered as a parameter by itself.

23. The Accused Instrumentalities also practice wherein when the controller determines that the throughput falls below a predetermined throughput threshold, the controller commands the data compression engine to use one of the plurality of compression routines to provide a faster rate of compression so as to increase the throughput. Based on the throughput, bitrate, and/or resolution parameter identified (e.g. throughput, bandwidth, bitrate, max video bitrate, resolution, GOP structure or frame type within a GOP structure), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., “baseline,” “extended,” “main”, or “high”) corresponds with that parameter, whether if that parameter (e.g. throughput) falls below a predetermined throughput threshold, then select between at least two asymmetric compressors to, for example, provide a faster rate of compression so as to increase the throughput. If baseline or extended is the corresponding profile, then the system will select a Context-Adaptive Variable Length Coding (“CAVLC”) entropy encoder. If main or high is the corresponding profile, then the system will select a Context-Adaptive Binary Arithmetic Coding (“CABAC”) entropy encoder. Both encoders are asymmetric compressors because it takes a longer period of time for them to compress data than to decompress data. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

	Baseline	Extended	Main	High	High 10
I and P Slices	Yes	Yes	Yes	Yes	Yes
B Slices	No	Yes	Yes	Yes	Yes
SI and SP Slices	No	Yes	No	No	No
Multiple Reference Frames	Yes	Yes	Yes	Yes	Yes
In-Loop Deblocking Filter	Yes	Yes	Yes	Yes	Yes
CAVLC Entropy Coding	Yes	Yes	Yes	Yes	Yes
CABAC Entropy Coding	No	No	Yes	Yes	Yes
Flexible Macroblock Ordering (FMO)	Yes	Yes	No	No	No
Arbitrary Slice Ordering (ASO)	Yes	Yes	No	No	No
Redundant Slices (RS)	Yes	Yes	No	No	No
Data Partitioning	No	Yes	No	No	No
Interlaced Coding (PicAFF, MBAFF)	No	Yes	Yes	Yes	Yes
4:2:0 Chroma Format	Yes	Yes	Yes	Yes	Yes
Monochrome Video Format (4:0:0)	No	No	No	Yes	Yes
4:2:2 Chroma Format	No	No	No	No	No
4:4:4 Chroma Format	No	No	No	No	No
8 Bit Sample Depth	Yes	Yes	Yes	Yes	Yes
9 and 10 Bit Sample Depth	No	No	No	No	Yes
11 to 14 Bit Sample Depth	No	No	No	No	No
8×8 vs. 4×4 Transform Adaptivity	No	No	No	Yes	Yes
Quantization Scaling Matrices	No	No	No	Yes	Yes
Separate Cb and Cr QP control	No	No	No	Yes	Yes
Separate Color Plane Coding	No	No	No	No	No
Predictive Lossless Coding	No	No	No	No	No

See http://web.cs.ucla.edu/classes/fall03/cs218/paper/H.264_MPEG4_Tutorial.pdf at 7:

The following table summarizes the two major types of entropy coding: Variable Length Coding (VLC) and Context Adaptive Binary Arithmetic Coding (CABAC). CABAC offers superior coding efficiency over VLC by adapting to the changing probability distribution of symbols, by exploiting correlation between symbols, and by adaptively exploiting bit correlations using arithmetic coding. H.264 also supports Context Adaptive Variable Length Coding (CAVLC) which offers superior entropy coding over VLC without the full cost of CABAC.

H.264 Entropy Coding – Comparison of Approaches

Characteristics	Variable Length Coding (VLC)	Context Adaptive Binary Arithmetic Coding(CABAC)
• Where it is used	MPEG-2, MPEG-4 ASP	H.264/MPEG-4 AVC (high efficiency option)
• Probability distribution	**Static** - Probabilities never change	**Adaptive** - Adjusts probabilities based on actual data
• Leverages correlation between symbols	**No** - Conditional probabilities ignored	**Yes** - Exploits symbol correlations by using "contexts"
• Non-integer code words	**No** - Low coding efficiency forhigh probability symbols	**Yes** - Exploits "arithmetic coding" which generates non-integer code words for higher efficiency

Moreover, the H.264 Standard requires a bit-flag descriptor, which is set to determine the correct decoder for the corresponding encoder. As shown below, if the flag = 0, then CAVLC must have been selected as the encoder; if the flag = 1, then CABAC must have been selected as the encoder. *See* https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.264-201304-S!!PDF-E&type=items (Rec. ITU-T H.264 (04/2013)) at 80:

entropy_coding_mode_flag selects the entropy decoding method to be applied for the syntax elements for which two descriptors appear in the syntax tables as follows:

– If entropy_coding_mode_flag is equal to 0, the method specified by the left descriptor in the syntax table is applied (Exp-Golomb coded, see clause 9.1 or CAVLC, see clause 9.2).

– Otherwise (entropy_coding_mode_flag is equal to 1), the method specified by the right descriptor in the syntax table is applied (CABAC, see clause 9.3).

24. The Accused Instrumentalities also practice wherein when the controller

determines that the throughput falls below a predetermined throughput threshold, the controller commands the data compression engine to use one of the plurality of compression routines to provide a faster rate of compression so as to increase the throughput. For example, after its selection, the asymmetric compressor (CAVLC or CABAC) then compresses the data input to provide compressed data blocks (which can also be organized in a GOP structure) so as to increase the throughput, as discussed previously above. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

Entropy Coding

For entropy coding, H.264 may use an enhanced VLC, a more complex context-adaptive variable-length coding (CAVLC) or an ever more complex Context-adaptive binary-arithmetic coding (CABAC) which are complex techniques to losslessly compress syntax elements in the video stream knowing the probabilities of syntax elements in a given context. The use of CABAC can improve the compression of around 5-7%. CABAC may requires a 30-40% of total processing power to be accomplished.

See

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.602.1581&rep=rep1&type=pdf at 13:

Typical compression ratios to maintain excellent quality are:

- 10:1 for general images using JPEG
- 30:1 for general video using H.263 and MPEG-2
- 60:1 for general video using H.264 and WMV9

See http://www.ijera.com/papers/Vol3_issue4/BM34399403.pdf at 2:

> Most visual communication systems today use Baseline Profile. Baseline is the simplest H.264 profile and defines, for example, zigzag scanning of the picture and using 4:2:0 (YUV video formats) chrominance sampling. In Baseline Profile, the picture is split in blocks consisting of 4x4 pixels, and each block is processed separately. Another important element of the Baseline Profile is the use of Universal Variable Length Coding (UVLC) and Context Adaptive Variable Length Coding (CAVLC) entropy coding techniques.
>
> The Extended and Main Profiles includes the functionality of the Baseline Profile and add improvements to the predictions algorithms. Since transmitting every single frame (think 30 frames per second for good quality video) is not feasible if you are trying to reduce the bit rate 1000-2000 times, temporal and motion prediction are heavily used in H.264, and allow transmitting only the difference between one frame and the previous frames. The result is spectacular efficiency gain, especially for scenes with little change and motion.
>
> The High Profile is the most powerful profile in H.264, and it allows most efficient coding of video. For example, large coding gain achieved through the use of Context Adaptive Binary Arithmetic Coding (CABAC) encoding which is more efficient than the UVLC/CAVLC used in Baseline Profile.
>
> The High Profile also uses adaptive transform that decides on the fly if 4x4 or 8x8-pixel blocks should be used. For example, 4x4 blocks are used for the parts of the picture that are dense with detail, while parts that have little detail are transformed using 8x8 blocks.

25. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the ’046 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the system claimed by, for example, Claim 40 of the ’046 patent, namely, a system, comprising: a data compression system for compressing and decompressing data input; a plurality of compression routines selectively utilized by the data compression system, wherein a first one of the plurality of compression routines includes a first compression algorithm and a second one of the plurality of compression routines includes a second compression algorithm; and a controller for

tracking throughput and generating a control signal to select a compression routine based on the throughput, wherein said tracking throughput comprises tracking a number of pending access requests to a storage device; and wherein when the controller determines that the throughput falls below a predetermined throughput threshold, the controller commands the data compression engine to use one of the plurality of compression routines to provide a faster rate of compression so as to increase the throughput. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers.

26. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’046 patent.

27. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the H.264 standard.

28. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the systems and/or methods claimed by the ’046 patent.

29. On information and belief, Adobe has had knowledge of the ’046 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’046 patent and knew of its infringement, including by way of this lawsuit. By the time of trial, Adobe will have known and intended (since receiving such notice) that its continued actions would actively induce and contribute to the infringement

of the claims of the ’046 patent.

30. Upon information and belief, Adobe’s affirmative acts of making, using, and selling the Accused Instrumentalities, and providing implementation services and technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’046 patent by practicing a system, comprising: a data compression system for compressing and decompressing data input; a plurality of compression routines selectively utilized by the data compression system, wherein a first one of the plurality of compression routines includes a first compression algorithm and a second one of the plurality of compression routines includes a second compression algorithm; and a controller for tracking throughput and generating a control signal to select a compression routine based on the throughput, wherein said tracking throughput comprises tracking a number of pending access requests to a storage device; and wherein when the controller determines that the throughput falls below a predetermined throughput threshold, the controller commands the data compression engine to use one of the plurality of compression routines to provide a faster rate of compression so as to increase the throughput. For example, Adobe adopted H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec as its video codec in its products/services, such as, e.g., Adobe’s media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused Instrumentalities to infringe other claims of the ’046 patent. Adobe specifically intended and was aware that these normal and customary activities

would infringe the ’046 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the ‘046 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continues to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the ‘046 patent, knowing that such use constitutes infringement of the ’046 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the ’046 patent, in violation of 35 U.S.C. § 271(b).

31. Adobe has also infringed, and continues to infringe, claims of the ’046 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the ’046 patent, and constitute a material part of the invention. Adobe knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the ’046 patent, not a staple article, and not a commodity of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the ’046 patent, in violation of 35 U.S.C. § 271(c).

32. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities’ compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the ’046 patent pursuant to 35 U.S.C. § 271.

33. As a result of Adobe’s infringement of the ’046 patent, Plaintiff Realtime is

entitled to monetary damages in an amount adequate to compensate for Adobe's infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

COUNT II

INFRINGEMENT OF U.S. PATENT NO. 8,634,462

34. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

35. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States products that infringe the '462 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe's Adobe Media Encoder, Adobe Media Encoder Creative Cloud or CC (e.g. CC 2015.1), Adobe Media Encoder Creative Suite or CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Premiere Pro, Adobe Premiere Pro CC (e.g. CC 2015.1), Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe After Effects, Adobe After Effects CC, Adobe After Effects CS (e.g. CS4, CS5, CS5.5, CS6), and all versions and variations thereof since the issuance of the '462 patent ("Accused Instrumentalities").

36. On information and belief, Adobe has directly infringed and continues to infringe the '462 patent, for example, through its sale, offer for sale, importation, use and testing of the Accused Instrumentalities, which practices the method claimed by, for example, Claim 1 of the '462 patent, namely, a method for coding a video signal using hybrid coding, comprising: reducing temporal redundancy by block based motion compensated prediction in order to establish a prediction error signal; performing quantization on samples of the prediction error signal or on coefficients resulting from a

transformation of the prediction error signal into the frequency domain to obtain quantized values, representing quantized samples or quantized coefficients respectively, wherein the prediction error signal includes a plurality of subblocks each including a plurality of quantized values; calculating a first quantization efficiency for the quantized values of at least one subblock of the plurality of subblocks; setting the quantized values of the at least one subblock to all zeroes; calculating a second quantization efficiency for the at least one subblock while all of the quantized values are zeroes; selecting which of the first and second quantization efficiencies is a higher efficiency; and selecting, for further proceeding, the at least one subblock with the quantized values prior to setting the quantized values of the at least one subblock to all zeroes if the first quantization efficiency is higher and selecting the at least one subblock with the quantized values set to zero, for further proceeding, if the second quantization efficiency is higher. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to Adobe's customers.

37. The Accused Instrumentalities utilize H.265 (HEVC) and/or its equivalents by practicing a method for coding a video signal using hybrid coding. "The video coding layer of **HEVC** employs **the same hybrid approach** (inter-/intrapicture prediction and 2-D transform coding) used in all video compression standards since H.261." Gary J Sullivan et al., Overview of the High Efficiency Video Coding (HEVC) Standard, 22 IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY 1650 (December 2012) ("IEEE HEVC"),

http://iphome.hhi.de/wiegand/assets/pdfs/2012_12_IEEE-HEVC-Overview.pdf; *see also id.* at 1654 ("As in all prior ITU-T and ISO/IEC JTC 1 video coding standards since H.261, **the HEVC design follows the classic block-based hybrid video coding approach** (as depicted in Fig. 1).") (citations omitted). Furthermore, the aim of the hybrid coding process is the production of a bitstream, as defined in definition 3.12 of the ITU-T H.265 Series H: Audiovisual and Multimedia Systems, "Infrastructure of audiovisual services – Coding of moving video" High efficiency video coding ("HEVC Spec"): "bitstream: A sequence of bits, in the form of a NAL unit stream or a byte stream, that forms the representation of coded pictures and associated data forming one or more coded video sequences (CVSs)." *See also, e.g.,* HEVC Spec at 0.7 "Overview of the design characteristics." Moreover, according to a website maintained by Adobe, describing an "Overview of What's New in Adobe Media Encoder CC 2015.1," the Adobe "Media Encoder will offer Expanded UHD capabilities with the addition of **support for…the new HEVC (H.265) codec,** which, at 60% the size of comparable H.264 files, is ideal for delivering pristine 4K content online." *See* https://blogs.adobe.com/creativecloud/adobe-media-encoder-cc-2015-1/. Also, according to another website maintained by Adobe discussing "Supported file formats" for Adobe Premiere Pro or specifically "Supported native video and audio formats for import" for Adobe Premiere Pro, the format of "**HEVC (H.265)"** is listed a long with the text "H.265 media with resolutions up to 8192x4320." *See* https://helpx.adobe.com/premiere-pro/using/supported-file-formats.html. According to another website, "One of the most significant new features in Adobe Premiere Pro CC 2015.1 is the expanded native support for 4K Ultra HD & HDR formats to include…**HEVC (H.265)**," Adobe "Media Encoder will offer Expanded UHD capabilities with the addition of support for…the new **HEVC/H.265 codec**, which, at 60% the size of comparable H.264 files, is ideal for delivering pristine 4K content online." *See* http://www.4kshooters.net/2015/12/04/adobe-premiere-pro-cc-2015-1-update-brings-h-265-more-4k-native-support/. The fact that both Adobe Media Encoder and Adobe Premiere Pro CC supported HEVC/H.265 as early as 2015 has also been confirmed by other websites. *See e.g.*, https://larryjordan.com/articles/adobe-media-encoder-and-h-265-not-ready-for-prime-time/ ("One of the exciting features of the Nov. 2015 update to Adobe Media Encoder is its support for **H.265, also known as HEVC**, short for High Efficiency Video Coding, compression.").

38. The Accused Instrumentalities also practice reducing temporal redundancy by block based motion compensated prediction in order to establish a prediction error signal. For example, clause 8.5.3 Decoding process for prediction units in inter prediction mode and the subclauses thereof of the HEVC Spec describe the block based motion compensation techniques used in the decoding process. *See also, e.g.,* IEEE HEVC at 1651-1652 6) Motion compensation: Quarter-sample precision is used for the MVs, and 7-tap or 8-tap filters are used for interpolation of fractional-sample positions (compared to six-tap filtering of half-sample positions followed by linear interpolation for quarter-sample positions in H.264/MPEG-4 AVC). Similar to H.264/MPEG-4 AVC, multiple reference pictures are used. For each PB, either one or two motion vectors can be transmitted, resulting either in unipredictive or bipredictive coding, respectively. As in H.264/MPEG-4 AVC, a scaling and offset operation may be applied to the prediction signal(s) in a manner known as weighted prediction.").

39. The Accused Instrumentalities also practice performing quantization on samples of the prediction error signal or on coefficients resulting from a transformation of the prediction error signal into the frequency domain to obtain quantized values, representing quantized samples or quantized coefficients respectively. For example, the quantization parameter and the scaling (inverse quantization) are defined in definitions 3.112 (page 10) and 3.131 (page 11), respectively, the usage of the scaling process in the decoding being described in clause and 8.6 Scaling, transformation and array construction process prior to deblocking filter process of the HEVC Spec. *See also, e.g.*, IEEE HEVC at 1652 ("8) Quantization control: As in H.264/MPEG-4 AVC, uniform reconstruction quantization (URQ) is used in HEVC, with quantization scaling matrices supported for the various transform block sizes.").

40. The Accused Instrumentalities also practice performing a method wherein the prediction error signal includes a plurality of subblocks each including a plurality of quantized values. For example, the quantized samples or transform coefficients from the subblock are scaled and transformed as described in above mentioned clause 8.6 of the HEVC Spec. *See also, e.g.*, IEEE HEVC at 1652 ("Prediction units and prediction blocks (PBs): The decision whether to code a picture area using interpicture or intrapicture prediction is made at the CU level. A PU partitioning structure has its root at the CU level. Depending on the basic prediction-type decision, the luma and chroma CBs can then be further split in size and predicted from luma and chroma prediction blocks (PBs). HEVC supports variable PB sizes from 64×64 down to 4×4 samples.").

41. The Accused Instrumentalities also practice performing a method of calculating a first quantization efficiency for the quantized values of at least one subblock

of the plurality of subblocks; setting the quantized values of the at least one subblock to all zeroes; calculating a second quantization efficiency for the at least one subblock while all of the quantized values are zeroes; selecting which of the first and second quantization efficiencies is a higher efficiency; and selecting, for further proceeding, the at least one subblock with the quantized values prior to setting the quantized values of the at least one subblock to all zeroes if the first quantization efficiency is higher and selecting the at least one subblock with the quantized values set to zero, for further proceeding, if the second quantization efficiency is higher. For example, the bitstream resulting from the encoding as described in this last item of the claim contains all the relevant information as needed by the decoder for proper decoding. If the coefficients of the subblock are set to zero as a consequence of the efficiency calculation, the coded_sub_block_flag, as described in clause 7.4.9.11 Residual coding semantics, HEVC Spec, is set to 0, indicating that all the 16 coefficients of the coded sub block have been set to 0: "coded_sub_block_flag[xS][yS] specifies the following for the sub-block at location (xS, yS) within the current transform block, where a sub-block is a (4x4) array of 16 transform coefficient levels: – If coded_sub_block_flag[xS][yS] is equal to 0, the 16 transform coefficient levels of the sub-block at location (xS, yS) are inferred to be equal to 0."

42. When coded_sub_block_flag[xS][yS] has not been set equal to 0, the position in the array of non 0 coefficients can be determined as follows:

– Otherwise (coded_sub_block_flag[xS][yS] is equal to 1), the following applies:

– If (xS, yS) is equal to (0, 0) and (LastSignificantCoeffX, LastSignificantCoeffY) is not equal to (0, 0), at least one of the 16 sig_coeff_flag syntax elements is present for the sub-block at location (xS, yS) .

– Otherwise, at least one of the 16 transform coefficient levels of the sub-block at location (xS, yS) has a non zero value.

When coded_sub_block_flag[xS][yS] is not present, it is inferred as follows:

– If one or more of the following conditions are true, coded_sub_block_flag[xS][yS] is inferred to be equal to 1:

– (xS, yS) is equal to (0, 0)

– (xS, yS) is equal to (LastSignificantCoeffX >> 2 , LastSignificantCoeffY >> 2)

– Otherwise, coded_sub_block_flag[xS][yS] is inferred to be equal to 0.

HEVC Spec at 7.4.9.11 Residual coding semantics. Therefore, even though the coding algorithms than can be used for reaching specific efficiency targets are not specified by the HEVC Spec (as stated in clause 0.7), this particular combination of choices produces a valid bitstream that has to be decoded by a conformant decoder.

43. The infringement of the Accused Instrumentalities is also shown by way of considering the reference software (*see, e.g.*, https://hevc.hhi.fraunhofer.de/). Setting the flag RDOQ=true in the encoder configuration file enables rate-distortion-optimized quantization for transformed TUs. This feature is implemented in the HM reference software as function xRateDistOptQuant in file TComTrQuant.cpp. In the function xRateDistOptQuant, the efficiency for setting all quantized values to zero is calculated and stored in the variable d64BestCost. In the variable iBestLastIdxP1, a 0 is stored indicating that all values starting from the 0th position are set to zero. Afterwards, the

efficiency for keeping quantized values unequal to zero is calculated and stored in the variable totalCost. The variable iBestLastIdxP1 is adjusted correspondingly to values unequal to 0. The two efficiencies d64BestCost and totalCost are compared, and selecting for further proceeding either quantized values, which are all set to zero or quantized values, which are not all set to zero. All values starting from the position defined by the variable iBestLastIdxP1 are set to zero.

44. Calculation of the efficiency for setting all quantized values to zero and storing the result in the variable d64BestCost:

```
Double  d64BestCost        = 0;
Int     ui16CtxCbf         = 0;
Int     iBestLastIdxP1     = 0;
if( !pcCU->isIntra( uiAbsPartIdx ) && isLuma(compID) && pcCU->getTransformIdx( uiAbsPartIdx ) == 0 )
{
  ui16CtxCbf   = 0;
  d64BestCost  = d64BlockUncodedCost + xGetICost( m_pcEstBitsSbac->blockRootCbpBits[ ui16CtxCbf ][ 0 ] );
  d64BaseCost += xGetICost( m_pcEstBitsSbac->blockRootCbpBits[ ui16CtxCbf ][ 1 ] );
}
else
{
  ui16CtxCbf   = pcCU->getCtxQtCbf( rTu, channelType );
  ui16CtxCbf  += getCBFContextOffset(compID);
  d64BestCost  = d64BlockUncodedCost + xGetICost( m_pcEstBitsSbac->blockCbpBits[ ui16CtxCbf ][ 0 ] );
  d64BaseCost += xGetICost( m_pcEstBitsSbac->blockCbpBits[ ui16CtxCbf ][ 1 ] );
}
```

HEVC Reference Software (https://hevc.hhi.fraunhofer.de/).

45. Calculating the efficiency for keeping quantized values unequal to zero and storing the result in the variable totalCost:

```
Bool bFoundLast = false;
for (Int iCGScanPos = iCGLastScanPos; iCGScanPos >= 0; iCGScanPos--)
{
  UInt uiCGBlkPos = codingParameters.scanCG[ iCGScanPos ];

  d64BaseCost -= pdCostCoeffGroupSig [ iCGScanPos ];
  if (uiSigCoeffGroupFlag[ uiCGBlkPos ])
  {
    for (Int iScanPosinCG = uiCGSize-1; iScanPosinCG >= 0; iScanPosinCG--)
    {
      iScanPos = iCGScanPos*uiCGSize + iScanPosinCG;

      if (iScanPos > iLastScanPos) continue;
      UInt    uiBlkPos     = codingParameters.scan[iScanPos];

      if( piDstCoeff[ uiBlkPos ] )
      {
        UInt    uiPosY        = uiBlkPos >> uiLog2BlockWidth;
        UInt    uiPosX        = uiBlkPos - ( uiPosY << uiLog2BlockWidth );

        Double d64CostLast= codingParameters.scanType == SCAN_VER ? xGetRateLast( uiPosY, uiPosX, compID ) :
                                                                     xGetRateLast( uiPosX, uiPosY, compID );
        Double totalCost = d64BaseCost + d64CostLast - pdCostSig[ iScanPos ];
```

HEVC Reference Software (https://hevc.hhi.fraunhofer.de/).

46. Comparing the two efficiencies d64BestCost and totalCost:

```
if( totalCost < d64BestCost )
{
  iBestLastIdxP1   = iScanPos + 1;
  d64BestCost      = totalCost;
}
```

HEVC Reference Software (https://hevc.hhi.fraunhofer.de/).

47. Selecting for further proceeding either quantized values, which are all set to zero or quantized values, which are not all set to zero:

```
//===== clean uncoded coefficients =====
for ( Int scanPos = iBestLastIdxP1; scanPos <= iLastScanPos; scanPos++ )
{
  piDstCoeff[ codingParameters.scan[ scanPos ] ] = 0;
}
```

HEVC Reference Software (https://hevc.hhi.fraunhofer.de/).

48. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the '462 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the method claimed by, for example, Claim 1 of the '462 patent, namely, a method for coding a video signal using hybrid coding, comprising: reducing temporal redundancy by block based motion compensated prediction in order to establish a prediction error signal; performing quantization on samples of the prediction error signal or on coefficients resulting from a transformation of the prediction error signal into the frequency domain to obtain quantized values, representing quantized samples or quantized coefficients respectively, wherein the prediction error signal includes a plurality of subblocks each including a plurality of quantized values; calculating a first quantization efficiency for the quantized values of at least one subblock of the plurality of subblocks; setting the quantized values

of the at least one subblock to all zeroes; calculating a second quantization efficiency for the at least one subblock while all of the quantized values are zeroes; selecting which of the first and second quantization efficiencies is a higher efficiency; and selecting, for further proceeding, the at least one subblock with the quantized values prior to setting the quantized values of the at least one subblock to all zeroes if the first quantization efficiency is higher and selecting the at least one subblock with the quantized values set to zero, for further proceeding, if the second quantization efficiency is higher. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers.

49. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’462 patent.

50. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the HEVC (or H.265) standard.

51. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the methods and/or systems claimed by the ’462 patent.

52. On information and belief, Adobe has had knowledge of the ’462 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’462 patent and knew of its infringement, including by way of this lawsuit. By the time of trial, Adobe will have known and intended (since receiving

such notice) that its continued actions would actively induce and contribute to the infringement of the claims of the ’462 patent.

53. Upon information and belief, Adobe’s affirmative acts of making, using, and selling the Accused Instrumentalities, and providing implementation services and technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’462 patent by practicing a method for coding a video signal using hybrid coding, comprising: reducing temporal redundancy by block based motion compensated prediction in order to establish a prediction error signal; performing quantization on samples of the prediction error signal or on coefficients resulting from a transformation of the prediction error signal into the frequency domain to obtain quantized values, representing quantized samples or quantized coefficients respectively, wherein the prediction error signal includes a plurality of subblocks each including a plurality of quantized values; calculating a first quantization efficiency for the quantized values of at least one subblock of the plurality of subblocks; setting the quantized values of the at least one subblock to all zeroes; calculating a second quantization efficiency for the at least one subblock while all of the quantized values are zeroes; selecting which of the first and second quantization efficiencies is a higher efficiency; and selecting, for further proceeding, the at least one subblock with the quantized values prior to setting the quantized values of the at least one subblock to all zeroes if the first quantization efficiency is higher and selecting the at least one subblock with the quantized values set to zero, for further proceeding, if the second quantization efficiency is higher. For

example, Adobe adopted HEVC (or H.265) and/or its equivalents as its video codec in its products/services, such as in Adobe's media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused Instrumentalities to infringe other claims of the '462 patent. Adobe specifically intended and was aware that these normal and customary activities would infringe the '462 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the '462 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continues to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the '462 patent, knowing that such use constitutes infringement of the '462 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the '462 patent, in violation of 35 U.S.C. § 271(b).

54. Adobe has also infringed, and continues to infringe, claims of the '462 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the '462 patent, and constitute a material part of the invention. Adobe knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the '462 patent, not a staple article, and not a commodity of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the '462 patent, in violation of

35 U.S.C. § 271(c).

55. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities' compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the '462 patent pursuant to 35 U.S.C. § 271.

56. As a result of Adobe's infringement of the '462 patent, Plaintiff Realtime is entitled to monetary damages in an amount adequate to compensate for Adobe's infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

COUNT III

INFRINGEMENT OF U.S. PATENT NO. 8,929,442

57. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

58. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States Adobe products that infringe the '442 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe's Adobe Premiere Pro, Adobe Premiere Pro CC, Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Flash, Adobe Flash Player (e.g. Adobe Flash Player 9), Adobe Flash Media Server, Adobe After Effects, Adobe After Effects CC, Adobe After Effects CS (e.g. CS4, CS5, CS5.5, CS6), Adobe HTTP Dynamic Streaming (HDS), and all versions and variations thereof since the issuance of the '442 patent ("Accused Instrumentalities").

59. On information and belief, Adobe has directly infringed and continues to

infringe the ’442 patent, for example, through its sale, offer for sale, importation, use and testing of the Accused Instrumentalities, which practices the apparatus claimed by, for example, Claim 8 of the ’442 patent, namely, an apparatus, comprising: a data decompression system configured to decompress a compressed data block; and a storage medium configured to store at least a portion of the decompressed data block, wherein at least a portion of a data block having video or audio data was compressed with one or more compression algorithms selected from among a plurality of compression algorithms based upon a throughput of a communication channel and a parameter or an attribute of the at least the portion of the data block to create at least the compressed data block, and wherein at least one of the plurality of compression algorithms is asymmetric. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to Adobe’s customers.

60. The Accused Instrumentalities include, or practice an apparatus, comprising: a data decompression system configured to decompress a compressed data block. For example, the Accused Instrumentalities utilize H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec. H.264 is “the most widely used codec on the planet.” *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-H.264-74735.aspx. A “codec” is also a compression technology that has “two components, an encoder to compress the files, and **a decoder to decompress**. There are codecs for…video (Cinepark, MPEG-2, **H.264**, VP8).” *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-a-Codec-

74487.aspx; https://forums.adobe.com/thread/729526 ("Compression is essential for reducing the size of movies so that they can be stored, transmitted, and played back effectively. Compression is achieved by an encoder; decompression is achieved by a decoder. Encoders and decoders are known by the common term codec."). According to a website maintained by Adobe, "you **can produce H.264 video** with Adobe Media Encoder CS4 and Adobe Flash Media Encoding Server 3.5," "Adobe added **H.264 playback support** to Adobe Flash Player 9 Update 3 back in 2007," Adobe "Flash Media Server and Flash Player can **both stream and play back any H.264 file** in virtually any format," the encoding capabilities of the Adobe Flash Media Encoding Server using "**H.264**-related parameters" are described (e.g. using CAVLC, CABAC) and under a headline stating "Adobe Media Encoder," the website states: "Adobe significantly enhanced the Flash Video Encoder in Creative Suite 4. There is now both stand-alone operation and batch encoding capabilities. As before, you can **access H.264 encoding** by choosing different formats in the Format pop-up menu. When producing for Flash Player, you should always use the FLV|F4V option, which lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html. Another website maintained by Adobe mentions that "You have two primary options for using Adobe Media Encoder with After Effects to create videos in **H.264**." The website also mentions that "Since Adobe Media Encoder already has superior H.264…exporting capabilities, it was more prudent to rely on Adobe Media Encoder for export of these formats." *See* https://helpx.adobe.com/after-effects/kb/export-h264.html. Another website maintained by Adobe mentions that the "Adobe Flash Player 9 Update 3 is taking a step into the high-definition (HD) video realm in a major way by adding MPEG-4 video…[which] utilizes crisp, powerful **H.264 encoding**" and "The new Adobe Media Player…will **also support H.264**" and "The addition of **H.264**…**support** in Flash Player 9 Update 3 allows you to easily use high-definition, industry standard video and audio." *See* http://www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html. Adobe HTTP Dynamic Streaming (HDS) also supports additional data compression systems or codecs other than H.264 because it "enables high-quality (**H.264 or VP6**), network-efficient HTTP streaming for media delivery that is tightly integrated with Adobe® Access software for robust content protection in the Adobe Flash® Player 10.1 or later and Adobe AIR®2 or later runtimes," is also "built on standards and deployed using standard HTTP servers (Apache), standard media format (MP4 fragment) using standard codecs (**H.264/AAC**), open APIs (Flash Player), and an open source framework for building media players (OSMF)" and "like other Flash Player supported delivery methods," Adobe HDS "supports **H.264 video and VP6 codecs** required for the highest quality video on the platform." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS also allows users to encode content "using high-quality Flash Player compatible **codecs** (**VP6/MP3, H264/AAC**)," deliver "high-definition video up to 1080p, with bitrates from 700kbps up to and beyond 6Mbps, using either **H.264 or VP6 video codecs,"** and archive "live, high-definition streams on the server and enable HD DVR functionality (for example, instant replay and time shifting) with support for **H.264** stream recording for RTMP and HTTP Dynamic Streaming." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. Adobe Media Encoder also "lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html.

61. The Accused Instrumentalities also include or practice an apparatus comprising: a storage medium configured to store at least a portion of the decompressed data block. For example, the Accused Instrumentalities include volatile and non-volatile memory (e.g., RAM, flash, etc.) configured to store at least a portion of the decompressed data block. *See* https://streaminglearningcenter.com/articles/ram-requirements-for-adobe-cs5-5.html ("RAM Requirements for Adobe CS5.5"); https://helpx.adobe.com/adobe-media-server/dev/configure-dynamic-streaming-live-streaming.html ("Adobe HTTP Dynamic Streaming combines these approaches to introduce HTTP streaming to the Flash Platform. HTTP Dynamic Streaming packages media files into fragments that Flash Player clients can access instantly without downloading the entire file. Adobe HTTP Dynamic Streaming contains several components that work together to package media and stream it over HTTP to Flash Player and AIR. HTTP Dynamic Streaming supports multi-bitrate streaming, DVR, and Adobe® Flash® Access™ protection."). On information and belief, the Accused Instrumentalities also include a storage medium configured to store at least a portion of the decompressed data block e.g., hard disk, disks, buffers, servers or other forms of memory/storage. *See* https://helpx.adobe.com/adobe-media-server/dev/configure-dynamic-streaming-live-streaming.html (describing "Streams [from Adobe HDS] in a live event are packaged as fragments and written to **disk**," "The **IO buffer** loads the **disk file** into an **in-memory buffer**," and "To limit the amount of **storage the disk cache** uses," and a "Content storage (HDS and HLS)" header describing a section where a "media player **requests content from the server.**"

62. The Accused Instrumentalities also practice wherein at least a portion of a data block having video or audio data was compressed with one or more compression algorithms selected from among a plurality of compression algorithms based upon a throughput of a communication channel and a parameter or an attribute of the at least the portion of the data block to create at least the compressed data block. For throughput of a communication channel or bandwidth, Adobe HTTP Dynamic Streaming (HDS) contains the "ability **to shift quality depending on bandwidth and computer power**. HTTP progressive delivery consumes more bandwidth because it's not intelligent enough to throttle the delivery (for example, a 30-minute video will be fully downloaded whether the user watches it or not)." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS can also **"[d]etect the client's bandwidth and computer resources** and serve them content fragments encoded at the most appropriate bitrate." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. Examples of the aforementioned parameters or attributes of portions of data blocks include throughput or bandwidth, bitrate (or max video bitrate) and resolution parameters. Different parameters correspond with different end applications. H.264 provides for multiple different ranges of such parameters, each included in the "profiles" and "levels" as defined by the H.264 standard, from the below shown paragraphs from a white paper and Wikipedia. *See* http://www.axis.com/files/whitepaper/wp_h264_31669_en_0803_lo.pdf at 5:

4. H.264 profiles and levels

The joint group involved in defining H.264 focused on creating a simple and clean solution, limiting options and features to a minimum. An important aspect of the standard, as with other video standards, is providing the capabilities in profiles (sets of algorithmic features) and levels (performance classes) that optimally support popular productions and common formats.

H.264 has seven profiles, each targeting a specific class of applications. Each profile defines what feature set the encoder may use and limits the decoder implementation complexity.

Network cameras and video encoders will most likely use a profile called the baseline profile, which is intended primarily for applications with limited computing resources. The baseline profile is the most suitable given the available performance in a real-time encoder that is embedded in a network video product. The profile also enables low latency, which is an important requirement of surveillance video and also particularly important in enabling real-time, pan/tilt/zoom (PTZ) control in PTZ network cameras.

H.264 has 11 levels or degree of capability to limit performance, bandwidth and memory requirements. Each level defines the bit rate and the encoding rate in macroblock per second for resolutions ranging from QCIF to HDTV and beyond. The higher the resolution, the higher the level required.

See https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC:

Levels with maximum property values

Level	Max decoding speed		Max frame size		Max video bit rate for video coding layer (VCL) kbit/s			Examples for high resolution @ highest frame rate (max stored frames) Toggle additional details
	Luma samples/s	Macroblocks/s	Luma samples	Macroblocks	Baseline, Extended and Main Profiles	High Profile	High 10 Profile	
1	380,160	1,485	25,344	99	64	80	192	176×144@15.0 (4)
1b	380,160	1,485	25,344	99	128	160	384	176×144@15.0 (4)
1.1	768,000	3,000	101,376	396	192	240	576	352×288@7.5 (2)
1.2	1,536,000	6,000	101,376	396	384	480	1,152	352×288@15.2 (6)
1.3	3,041,280	11,880	101,376	396	768	960	2,304	352×288@30.0 (6)
2	3,041,280	11,880	101,376	396	2,000	2,500	6,000	352×288@30.0 (6)
2.1	5,068,800	19,800	202,752	792	4,000	5,000	12,000	352×576@25.0 (6)
2.2	5,184,000	20,250	414,720	1,620	4,000	5,000	12,000	720×576@12.5 (5)
3	10,368,000	40,500	414,720	1,620	10,000	12,500	30,000	720×576@25.0 (5)
3.1	27,648,000	108,000	921,600	3,600	14,000	17,500	42,000	1,280×720@30.0 (5)
3.2	55,296,000	216,000	1,310,720	5,120	20,000	25,000	60,000	1,280×1,024@42.2 (4)
4	62,914,560	245,760	2,097,152	8,192	20,000	25,000	60,000	2,048×1,024@30.0 (4)
4.1	62,914,560	245,760	2,097,152	8,192	50,000	62,500	150,000	2,048×1,024@30.0 (4)
4.2	133,693,440	522,240	2,228,224	8,704	50,000	62,500	150,000	2,048×1,080@60.0 (4)
5	150,994,944	589,824	5,652,480	22,080	135,000	168,750	405,000	3,672×1,536@26.7 (5)
5.1	251,658,240	983,040	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@26.7 (5)
5.2	530,841,600	2,073,600	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@56.3 (5)

63. A video data block is organized by the group of pictures (GOP) structure,

which is a "collection of successive pictures within a coded video stream." *See* https://en.wikipedia.org/wiki/Group_of_pictures. A GOP structure can contain intra coded pictures (I picture or I frame), predictive coded pictures (P picture or P frame), bipredictive coded pictures (B picture or B frame) and direct coded pictures (D picture or D frames, or DC direct coded pictures which are used only in MPEG-1 video). *See* https://en.wikipedia.org/wiki/Video_compression_picture_types (for descriptions of I frames, P frames and B frames); https://en.wikipedia.org/wiki/MPEG-1#D-frames (for descriptions of D frames). Thus, at least a portion of a video data block would also make up a GOP structure and could also contain I frames, P frames, B frames and/or D frames. The GOP structure also reflects the size of a video data block, and the GOP structure can be controlled and used to fine-tune other parameters (e.g. bitrate, max video bitrate and resolution parameters) or even be considered as a parameter by itself.

64. The Accused Instrumentalities also practice wherein at least one of the plurality of compression algorithms is asymmetric. Based on the bitrate and/or resolution parameter identified (e.g. bitrate, max video bitrate, resolution, GOP structure or frame type within a GOP structure), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., "baseline," "extended," "main", or "high") corresponds with that parameter, then select between at least two asymmetric compressors. If baseline or extended is the corresponding profile, then the system will select a Context-Adaptive Variable Length Coding ("CAVLC") entropy encoder. If main or high is the corresponding profile, then the system will select a Context-Adaptive Binary Arithmetic Coding ("CABAC") entropy encoder. Both encoders are asymmetric compressors because it takes a longer period of time for them to compress data than to

decompress data. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

	Baseline	Extended	Main	High	High 10
I and P Slices	Yes	Yes	Yes	Yes	Yes
B Slices	No	Yes	Yes	Yes	Yes
SI and SP Slices	No	Yes	No	No	No
Multiple Reference Frames	Yes	Yes	Yes	Yes	Yes
In-Loop Deblocking Filter	Yes	Yes	Yes	Yes	Yes
CAVLC Entropy Coding	Yes	Yes	Yes	Yes	Yes
CABAC Entropy Coding	No	No	Yes	Yes	Yes
Flexible Macroblock Ordering (FMO)	Yes	Yes	No	No	No
Arbitrary Slice Ordering (ASO)	Yes	Yes	No	No	No
Redundant Slices (RS)	Yes	Yes	No	No	No
Data Partitioning	No	Yes	No	No	No
Interlaced Coding (PicAFF, MBAFF)	No	Yes	Yes	Yes	Yes
4:2:0 Chroma Format	Yes	Yes	Yes	Yes	Yes
Monochrome Video Format (4:0:0)	No	No	No	Yes	Yes
4:2:2 Chroma Format	No	No	No	No	No
4:4:4 Chroma Format	No	No	No	No	No
8 Bit Sample Depth	Yes	Yes	Yes	Yes	Yes
9 and 10 Bit Sample Depth	No	No	No	No	Yes
11 to 14 Bit Sample Depth	No	No	No	No	No
8×8 vs. 4×4 Transform Adaptivity	No	No	No	Yes	Yes
Quantization Scaling Matrices	No	No	No	Yes	Yes
Separate Cb and Cr QP control	No	No	No	Yes	Yes
Separate Color Plane Coding	No	No	No	No	No
Predictive Lossless Coding	No	No	No	No	No

See http://web.cs.ucla.edu/classes/fall03/cs218/paper/H.264_MPEG4_Tutorial.pdf at 7:

The following table summarizes the two major types of entropy coding: Variable Length Coding (VLC) and Context Adaptive Binary Arithmetic Coding (CABAC). CABAC offers superior coding efficiency over VLC by adapting to the changing probability distribution of symbols, by exploiting correlation between symbols, and by adaptively exploiting bit correlations using arithmetic coding. H.264 also supports Context Adaptive Variable Length Coding (CAVLC) which offers superior entropy coding over VLC without the full cost of CABAC.

H.264 Entropy Coding – Comparison of Approaches

Characteristics	Variable Length Coding (VLC)	Context Adaptive Binary Arithmetic Coding(CABAC)
• Where it is used	MPEG-2, MPEG-4 ASP	H.264/MPEG-4 AVC (high efficiency option)
• Probability distribution	**Static** - Probabilities never change	**Adaptive** - Adjusts probabilities based on actual data
• Leverages correlation between symbols	**No** - Conditional probabilities ignored	**Yes** - Exploits symbol correlations by using "contexts"
• Non-integer code words	**No** - Low coding efficiency forhigh probability symbols	**Yes** - Exploits "arithmetic coding" which generates non-integer code words for higher efficiency

Moreover, the H.264 Standard requires a bit-flag descriptor, which is set to determine the correct decoder for the corresponding encoder. As shown below, if the flag = 0, then CAVLC must have been selected as the encoder; if the flag = 1, then CABAC must have been selected as the encoder. *See* https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.264-201304-S!!PDF-E&type=items (Rec. ITU-T H.264 (04/2013)) at 80:

entropy_coding_mode_flag selects the entropy decoding method to be applied for the syntax elements for which two descriptors appear in the syntax tables as follows:

– If entropy_coding_mode_flag is equal to 0, the method specified by the left descriptor in the syntax table is applied (Exp-Golomb coded, see clause 9.1 or CAVLC, see clause 9.2).

– Otherwise (entropy_coding_mode_flag is equal to 1), the method specified by the right descriptor in the syntax table is applied (CABAC, see clause 9.3).

65. The Accused Instrumentalities practice at least a portion of a data block having video or audio data that was compressed with one or more compression algorithms selected from among a plurality of compression algorithms…and wherein at least one of the plurality of compression algorithms is asymmetric. The compressed data blocks can also be organized in a GOP structure (see above). After its selection, the **asymmetric compressor** (CAVLC or CABAC) will compress the video or audio data to provide various compressed data blocks, which can also be organized in a GOP structure, as discussed previously above. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

Entropy Coding

For entropy coding, H.264 may use an enhanced VLC, a more complex context-adaptive variable-length coding (CAVLC) or an ever more complex Context-adaptive binary-arithmetic coding (CABAC) which are complex techniques to losslessly compress syntax elements in the video stream knowing the probabilities of syntax elements in a given context. The use of CABAC can improve the compression of around 5-7%. CABAC may requires a 30-40% of total processing power to be accomplished.

See http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.602.1581&rep=rep1&type=pdf at 13:

Typical compression ratios to maintain excellent quality are:

- 10:1 for general images using JPEG
- 30:1 for general video using H.263 and MPEG-2
- 60:1 for general video using H.264 and WMV9

See http://www.ijera.com/papers/Vol3_issue4/BM34399403.pdf at 2:

Most visual communication systems today use Baseline Profile. Baseline is the simplest H.264 profile and defines, for example, zigzag scanning of the picture and using 4:2:0 (YUV video formats) chrominance sampling. In Baseline Profile, the picture is split in blocks consisting of 4x4 pixels, and each block is processed separately. Another important element of the Baseline Profile is the use of Universal Variable Length Coding (UVLC) and Context Adaptive Variable Length Coding (CAVLC) entropy coding techniques.

The Extended and Main Profiles includes the functionality of the Baseline Profile and add improvements to the predictions algorithms. Since transmitting every single frame (think 30 frames per second for good quality video) is not feasible if you are trying to reduce the bit rate 1000-2000 times, temporal and motion prediction are heavily used in H.264, and allow transmitting only the difference between one frame and the previous frames. The result is spectacular efficiency gain, especially for scenes with little change and motion.

The High Profile is the most powerful profile in H.264, and it allows most efficient coding of video. For example, large coding gain achieved through the use of Context Adaptive Binary Arithmetic Coding (CABAC) encoding which is more efficient than the UVLC/CAVLC used in Baseline Profile.

The High Profile also uses adaptive transform that decides on the fly if 4x4 or 8x8-pixel blocks should be used. For example, 4x4 blocks are used for the parts of the picture that are dense with detail, while parts that have little detail are transformed using 8x8 blocks.

66. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the ’442 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the apparatus claimed by, for example, Claim 8 of the ’442 patent, namely, an apparatus, comprising: a data decompression system configured to decompress a compressed data block; and a storage medium configured to store at least a portion of the decompressed data block, wherein at least a portion of a data block having video or audio data was compressed with one or more compression algorithms selected from among a plurality of compression algorithms based

upon a throughput of a communication channel and a parameter or an attribute of the at least the portion of the data block to create at least the compressed data block, and wherein at least one of the plurality of compression algorithms is asymmetric. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers.

67. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’442 patent.

68. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the H.264 standard.

69. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the systems and/or methods claimed by the ’442 patent.

70. On information and belief, Adobe has had knowledge of the ’442 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’442 patent and knew of its infringement, including by way of this lawsuit. By the time of trial, Adobe will have known and intended (since receiving such notice) that its continued actions would actively induce and contribute to the infringement of the claims of the ’442 patent.

71. Upon information and belief, Adobe’s affirmative acts of making, using, and selling the Accused Instrumentalities, and providing implementation services and

technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’442 patent by practicing, for example, an apparatus, comprising: a data decompression system configured to decompress a compressed data block; and a storage medium configured to store at least a portion of the decompressed data block, wherein at least a portion of a data block having video or audio data was compressed with one or more compression algorithms selected from among a plurality of compression algorithms based upon a throughput of a communication channel and a parameter or an attribute of the at least the portion of the data block to create at least the compressed data block, and wherein at least one of the plurality of compression algorithms is asymmetric. For example, Adobe adopted H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec as its video codec in its products/services, such as, e.g., Adobe’s media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused Instrumentalities to infringe other claims of the ’442 patent. Adobe specifically intended and was aware that these normal and customary activities would infringe the ’442 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the ’442 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continues to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the ’442 patent, knowing that such use constitutes infringement of the ’442 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the ’442 patent, in violation of 35 U.S.C. § 271(b).

72. Adobe has also infringed, and continues to infringe, claims of the ’442 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the ’442 patent, and constitute a material part of the invention. Adobe knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the ’442 patent, not a staple article, and not a commodity of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the ’442 patent, in violation of 35 U.S.C. § 271(c).

73. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities’ compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the ’442 patent pursuant to 35 U.S.C. § 271.

74. As a result of Adobe’s infringement of the ’442 patent, Plaintiff Realtime is entitled to monetary damages in an amount adequate to compensate for Adobe’s infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

COUNT IV

INFRINGEMENT OF U.S. PATENT NO. 8,934,535

75. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

76. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States Adobe products that infringe the ’535 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe’s Adobe Media Encoder, Adobe Media Encoder Creative Cloud or CC, Adobe Media Encoder Creative Suite or CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Premiere Pro, Adobe Premiere Pro CC, Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Flash, Adobe Flash Player (e.g. Adobe Flash Player 9), Adobe Flash Media Server, Adobe Flash Media Encoding Server, Adobe After Effects, Adobe After Effects CC, Adobe After Effects CS (e.g. CS4, CS5, CS5.5, CS6), Adobe HTTP Dynamic Streaming (HDS), and all versions and variations thereof since the issuance of the ’535 patent (“Accused Instrumentalities”).

77. On information and belief, Adobe has directly infringed and continues to infringe the ‘535 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the method claimed by, for example, Claim 15 of the ’535 patent, namely, a method, comprising: determining a parameter of at least a portion of a data block; selecting one or more asymmetric compressors from among a plurality of compressors based upon the determined parameter or attribute; compressing the at least the portion of the data block with the selected one or more asymmetric compressors to provide one or more compressed data blocks; and storing at least a portion of the one or more compressed data blocks. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to

Adobe's customers.

78. The Accused Instrumentalities include, or practice a method involving data compression with asymmetric compressors. For example, the Accused Instrumentalities utilize H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec. H.264 is "the most widely used codec on the planet." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-H.264-74735.aspx. A "codec" is also a compression technology that has "two components, an **encoder to compress the files**, and a decoder to decompress. There are codecs for…video (Cinepark, MPEG-2, **H.264**, VP8)." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-a-Codec-74487.aspx; https://forums.adobe.com/thread/729526 ("Compression is essential for reducing the size of movies so that they can be stored, transmitted, and played back effectively. Compression is achieved by an encoder; decompression is achieved by a decoder. Encoders and decoders are known by the common term codec."). According to a website maintained by Adobe, "you **can produce H.264 video** with Adobe Media Encoder CS4 and Adobe Flash Media Encoding Server 3.5," "Adobe added **H.264 playback support** to Adobe Flash Player 9 Update 3 back in 2007," Adobe "Flash Media Server and Flash Player can **both stream and play back any H.264 file** in virtually any format," the encoding capabilities of the Adobe Flash Media Encoding Server using "**H.264-**related parameters" are described (e.g. using CAVLC, CABAC) and under a headline stating "Adobe Media Encoder," the website states: "Adobe significantly enhanced the Flash Video Encoder in Creative Suite 4. There is now both stand-alone operation and batch encoding capabilities. As before, you can **access H.264 encoding** by

choosing different formats in the Format pop-up menu. When producing for Flash Player, you should always use the FLV|F4V option, which lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html. Another website maintained by Adobe mentions that "You have two primary options for using Adobe Media Encoder with After Effects to create videos in **H.264**." The website also mentions that "Since Adobe Media Encoder already has superior H.264…exporting capabilities, it was more prudent to rely on Adobe Media Encoder for export of these formats." *See* https://helpx.adobe.com/after-effects/kb/export-h264.html. Another website maintained by Adobe mentions that the "Adobe Flash Player 9 Update 3 is taking a step into the high-definition (HD) video realm in a major way by adding MPEG-4 video…[which] utilizes crisp, powerful **H.264 encoding**" and "The new Adobe Media Player…will **also support H.264**" and "The addition of **H.264**…**support** in Flash Player 9 Update 3 allows you to easily use high-definition, industry standard video and audio." *See* http://www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html. Adobe HTTP Dynamic Streaming (HDS) also supports methods involving data compression with asymmetric compressors or codecs other than H.264 because it "enables high-quality (**H.264 or VP6**), network-efficient HTTP streaming for media delivery that is tightly integrated with Adobe® Access software for robust content protection in the Adobe Flash® Player 10.1 or later and Adobe AIR®2 or later runtimes," is also "built on standards and deployed using standard HTTP servers (Apache), standard media format (MP4 fragment) using standard codecs (**H.264/AAC**), open APIs (Flash Player), and an open source framework for building media players (OSMF)" and "like other Flash Player supported delivery methods," Adobe HDS "supports **H.264 video and VP6 codecs** required for the highest quality video on the platform." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS also allows users to encode content "using high-quality Flash Player compatible **codecs** (**VP6/MP3, H264/AAC**)," deliver "high-definition video up to 1080p, with bitrates from 700kbps up to and beyond 6Mbps, using either **H.264 or VP6 video codecs,"** and archive "live, high-definition streams on the server and enable HD DVR functionality (for example, instant replay and time shifting) with support for **H.264** stream recording for RTMP and HTTP Dynamic Streaming." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. Adobe Media Encoder also "lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html.

79. The Accused Instrumentalities also practice determining a parameter of at least a portion of a video data block. As shown below, examples of such parameters include bitrate (or max video bitrate) and resolution parameters. Different parameters correspond with different end applications. H.264 provides for multiple different ranges of such parameters, each included in the "profiles" and "levels" as defined by the H.264 standard, from the below shown paragraphs from a white paper and Wikipedia. *See* http://www.axis.com/files/whitepaper/wp_h264_31669_en_0803_lo.pdf at 5:

4. H.264 profiles and levels

The joint group involved in defining H.264 focused on creating a simple and clean solution, limiting options and features to a minimum. An important aspect of the standard, as with other video standards, is providing the capabilities in profiles (sets of algorithmic features) and levels (performance classes) that optimally support popular productions and common formats.

H.264 has seven profiles, each targeting a specific class of applications. Each profile defines what feature set the encoder may use and limits the decoder implementation complexity.

Network cameras and video encoders will most likely use a profile called the baseline profile, which is intended primarily for applications with limited computing resources. The baseline profile is the most suitable given the available performance in a real-time encoder that is embedded in a network video product. The profile also enables low latency, which is an important requirement of surveillance video and also particularly important in enabling real-time, pan/tilt/zoom (PTZ) control in PTZ network cameras.

H.264 has 11 levels or degree of capability to limit performance, bandwidth and memory requirements. Each level defines the bit rate and the encoding rate in macroblock per second for resolutions ranging from QCIF to HDTV and beyond. The higher the resolution, the higher the level required.

See https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC:

Levels with maximum property values

Level	Max decoding speed		Max frame size		Max video bit rate for video coding layer (VCL) kbit/s			Examples for high resolution @ highest frame rate (max stored frames) Toggle additional details
	Luma samples/s	Macroblocks/s	Luma samples	Macroblocks	Baseline, Extended and Main Profiles	High Profile	High 10 Profile	
1	380,160	1,485	25,344	99	64	80	192	176×144@15.0 (4)
1b	380,160	1,485	25,344	99	128	160	384	176×144@15.0 (4)
1.1	768,000	3,000	101,376	396	192	240	576	352×288@7.5 (2)
1.2	1,536,000	6,000	101,376	396	384	480	1,152	352×288@15.2 (6)
1.3	3,041,280	11,880	101,376	396	768	960	2,304	352×288@30.0 (6)
2	3,041,280	11,880	101,376	396	2,000	2,500	6,000	352×288@30.0 (6)
2.1	5,068,800	19,800	202,752	792	4,000	5,000	12,000	352×576@25.0 (6)
2.2	5,184,000	20,250	414,720	1,620	4,000	5,000	12,000	720×576@12.5 (5)
3	10,368,000	40,500	414,720	1,620	10,000	12,500	30,000	720×576@25.0 (5)
3.1	27,648,000	108,000	921,600	3,600	14,000	17,500	42,000	1,280×720@30.0 (5)
3.2	55,296,000	216,000	1,310,720	5,120	20,000	25,000	60,000	1,280×1,024@42.2 (4)
4	62,914,560	245,760	2,097,152	8,192	20,000	25,000	60,000	2,048×1,024@30.0 (4)
4.1	62,914,560	245,760	2,097,152	8,192	50,000	62,500	150,000	2,048×1,024@30.0 (4)
4.2	133,693,440	522,240	2,228,224	8,704	50,000	62,500	150,000	2,048×1,080@60.0 (4)
5	150,994,944	589,824	5,652,480	22,080	135,000	168,750	405,000	3,672×1,536@26.7 (5)
5.1	251,658,240	983,040	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@26.7 (5)
5.2	530,841,600	2,073,600	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@56.3 (5)

80. A video data block is organized by the group of pictures (GOP) structure, which is a "collection of successive pictures within a coded video stream." *See*

https://en.wikipedia.org/wiki/Group_of_pictures. A GOP structure can contain intra coded pictures (I picture or I frame), predictive coded pictures (P picture or P frame), bipredictive coded pictures (B picture or B frame) and direct coded pictures (D picture or D frames, or DC direct coded pictures which are used only in MPEG-1 video). *See* https://en.wikipedia.org/wiki/Video_compression_picture_types (for descriptions of I frames, P frames and B frames); https://en.wikipedia.org/wiki/MPEG-1#D-frames (for descriptions of D frames). Thus, at least a portion of a video data block would also make up a GOP structure and could also contain I frames, P frames, B frames and/or D frames. The GOP structure also reflects the size of a video data block, and the GOP structure can be controlled and used to fine-tune other parameters (e.g. bitrate, max video bitrate and resolution parameters) or even be considered as a parameter by itself.

81. The Accused Instrumentalities also practice selecting one or more asymmetric compressors from among a plurality of compressors based upon the determined parameter or attribute. Based on the bitrate and/or resolution parameter identified (e.g. bitrate, max video bitrate, resolution, GOP structure or frame type within a GOP structure), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., "baseline," "extended," "main", or "high") corresponds with that parameter, then select between at least two asymmetric compressors. If baseline or extended is the corresponding profile, then the system will select a Context-Adaptive Variable Length Coding ("CAVLC") entropy encoder. If main or high is the corresponding profile, then the system will select a Context-Adaptive Binary Arithmetic Coding ("CABAC") entropy encoder. Both encoders are asymmetric compressors because it takes a longer period of time for them to compress data than to

decompress data. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

	Baseline	Extended	Main	High	High 10
I and P Slices	Yes	Yes	Yes	Yes	Yes
B Slices	No	Yes	Yes	Yes	Yes
SI and SP Slices	No	Yes	No	No	No
Multiple Reference Frames	Yes	Yes	Yes	Yes	Yes
In-Loop Deblocking Filter	Yes	Yes	Yes	Yes	Yes
CAVLC Entropy Coding	Yes	Yes	Yes	Yes	Yes
CABAC Entropy Coding	No	No	Yes	Yes	Yes
Flexible Macroblock Ordering (FMO)	Yes	Yes	No	No	No
Arbitrary Slice Ordering (ASO)	Yes	Yes	No	No	No
Redundant Slices (RS)	Yes	Yes	No	No	No
Data Partitioning	No	Yes	No	No	No
Interlaced Coding (PicAFF, MBAFF)	No	Yes	Yes	Yes	Yes
4:2:0 Chroma Format	Yes	Yes	Yes	Yes	Yes
Monochrome Video Format (4:0:0)	No	No	No	Yes	Yes
4:2:2 Chroma Format	No	No	No	No	No
4:4:4 Chroma Format	No	No	No	No	No
8 Bit Sample Depth	Yes	Yes	Yes	Yes	Yes
9 and 10 Bit Sample Depth	No	No	No	No	Yes
11 to 14 Bit Sample Depth	No	No	No	No	No
8×8 vs. 4×4 Transform Adaptivity	No	No	No	Yes	Yes
Quantization Scaling Matrices	No	No	No	Yes	Yes
Separate Cb and Cr QP control	No	No	No	Yes	Yes
Separate Color Plane Coding	No	No	No	No	No
Predictive Lossless Coding	No	No	No	No	No

See http://web.cs.ucla.edu/classes/fall03/cs218/paper/H.264_MPEG4_Tutorial.pdf at 7:

The following table summarizes the two major types of entropy coding: Variable Length Coding (VLC) and Context Adaptive Binary Arithmetic Coding (CABAC). CABAC offers superior coding efficiency over VLC by adapting to the changing probability distribution of symbols, by exploiting correlation between symbols, and by adaptively exploiting bit correlations using arithmetic coding. H.264 also supports Context Adaptive Variable Length Coding (CAVLC) which offers superior entropy coding over VLC without the full cost of CABAC.

H.264 Entropy Coding – Comparison of Approaches

Characteristics	Variable Length Coding (VLC)	Context Adaptive Binary Arithmetic Coding(CABAC)
• Where it is used	MPEG-2, MPEG-4 ASP	H.264/MPEG-4 AVC (high efficiency option)
• Probability distribution	**Static** - Probabilities never change	**Adaptive** - Adjusts probabilities based on actual data
• Leverages correlation between symbols	**No** - Conditional probabilities ignored	**Yes** - Exploits symbol correlations by using "contexts"
• Non-integer code words	**No** - Low coding efficiency forhigh probability symbols	**Yes** - Exploits "arithmetic coding" which generates non-integer code words for higher efficiency

Moreover, the H.264 Standard requires a bit-flag descriptor, which is set to determine the correct decoder for the corresponding encoder. As shown below, if the flag = 0, then CAVLC must have been selected as the encoder; if the flag = 1, then CABAC must have been selected as the encoder. *See* https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.264-201304-S!!PDF-E&type=items (Rec. ITU-T H.264 (04/2013)) at 80:

entropy_coding_mode_flag selects the entropy decoding method to be applied for the syntax elements for which two descriptors appear in the syntax tables as follows:

– If entropy_coding_mode_flag is equal to 0, the method specified by the left descriptor in the syntax table is applied (Exp-Golomb coded, see clause 9.1 or CAVLC, see clause 9.2).

– Otherwise (entropy_coding_mode_flag is equal to 1), the method specified by the right descriptor in the syntax table is applied (CABAC, see clause 9.3).

82. The Accused Instrumentalities also practice compressing the at least the portion of the data block with the selected one or more asymmetric compressors to provide one or more compressed data blocks. The one or more compressed data blocks can also be organized in a GOP structure (see above). After its selection, the asymmetric compressor (CAVLC or CABAC) will compress the video data to provide various compressed data blocks, which can also be organized in a GOP structure, as discussed previously above. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

> **Entropy Coding**
> For entropy coding, H.264 may use an enhanced VLC, a more complex context-adaptive variable-length coding (CAVLC) or an ever more complex Context-adaptive binary-arithmetic coding (CABAC) which are complex techniques to losslessly compress syntax elements in the video stream knowing the probabilities of syntax elements in a given context. The use of CABAC can improve the compression of around 5-7%. CABAC may requires a 30-40% of total processing power to be accomplished.

See http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.602.1581&rep=rep1&type=pdf at 13:

> Typical compression ratios to maintain excellent quality are:
> - 10:1 for general images using JPEG
> - 30:1 for general video using H.263 and MPEG-2
> - 60:1 for general video using H.264 and WMV9

See http://www.ijera.com/papers/Vol3_issue4/BM34399403.pdf at 2:

> Most visual communication systems today use Baseline Profile. Baseline is the simplest H.264 profile and defines, for example, zigzag scanning of the picture and using 4:2:0 (YUV video formats) chrominance sampling. In Baseline Profile, the picture is split in blocks consisting of 4x4 pixels, and each block is processed separately. Another important element of the Baseline Profile is the use of Universal Variable Length Coding (UVLC) and Context Adaptive Variable Length Coding (CAVLC) entropy coding techniques.
>
> The Extended and Main Profiles includes the functionality of the Baseline Profile and add improvements to the predictions algorithms. Since transmitting every single frame (think 30 frames per second for good quality video) is not feasible if you are trying to reduce the bit rate 1000-2000 times, temporal and motion prediction are heavily used in H.264, and allow transmitting only the difference between one frame and the previous frames. The result is spectacular efficiency gain, especially for scenes with little change and motion.
>
> The High Profile is the most powerful profile in H.264, and it allows most efficient coding of video. For example, large coding gain achieved through the use of Context Adaptive Binary Arithmetic Coding (CABAC) encoding which is more efficient than the UVLC/CAVLC used in Baseline Profile.
>
> The High Profile also uses adaptive transform that decides on the fly if 4x4 or 8x8-pixel blocks should be used. For example, 4x4 blocks are used for the parts of the picture that are dense with detail, while parts that have little detail are transformed using 8x8 blocks.

83. The Accused Instrumentalities also practice storing at least a portion of the one or more compressed data blocks. For example, the Accused Instrumentalities include volatile and non-volatile memory (e.g., RAM, flash, etc.) configured to store at least a portion of the decompressed data block. *See* https://streaminglearningcenter.com/articles/ram-requirements-for-adobe-cs5-5.html ("RAM Requirements for Adobe CS5.5"); https://helpx.adobe.com/adobe-media-server/dev/configure-dynamic-streaming-live-streaming.html ("Adobe HTTP Dynamic Streaming combines these approaches to introduce HTTP streaming to the Flash

Platform. HTTP Dynamic Streaming packages media files into fragments that Flash Player clients can access instantly without downloading the entire file. Adobe HTTP Dynamic Streaming contains several components that work together to package media and stream it over HTTP to Flash Player and AIR. HTTP Dynamic Streaming supports multi-bitrate streaming, DVR, and Adobe® Flash® Access™ protection."). On information and belief, the Accused Instrumentalities also include a storage medium configured to store at least a portion of the decompressed data block e.g., hard disk, disks, buffers, servers or other forms of memory/storage. *See* https://helpx.adobe.com/adobe-media-server/dev/configure-dynamic-streaming-live-streaming.html (describing "Streams [from Adobe HDS] in a live event are packaged as fragments and written to **disk**," "The **IO buffer** loads the **disk file** into an **in-memory buffer**," and "To limit the amount of **storage the disk cache** uses," and a "Content storage (HDS and HLS)" header describing a section where a "media player **requests content from the server.**"

84. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the '535 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the method claimed by, for example, Claim 15 of the '535 patent, namely, a method, comprising: determining a parameter of at least a portion of a data block; selecting one or more asymmetric compressors from among a plurality of compressors based upon the determined parameter or attribute; compressing the at least the portion of the data block with the selected one or more asymmetric compressors to provide one or more compressed data blocks; and storing at least a portion of the one or more compressed data blocks. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing

business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers.

85. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’535 patent.

86. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the H.264 standard.

87. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the systems and/or methods claimed by the ’535 patent.

88. On information and belief, Adobe has had knowledge of the ’535 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’535 patent and knew of its infringement, including by way of this lawsuit. By the time of trial, Adobe will have known and intended (since receiving such notice) that its continued actions would actively induce and contribute to the infringement of the claims of the ’535 patent.

89. Upon information and belief, Adobe’s affirmative acts of making, using, and selling the Accused Instrumentalities, and providing implementation services and technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’535 patent by practicing, for example, a method, comprising: determining a parameter of at least a portion of a data block; selecting one or more

asymmetric compressors from among a plurality of compressors based upon the determined parameter or attribute; compressing the at least the portion of the data block with the selected one or more asymmetric compressors to provide one or more compressed data blocks; and storing at least a portion of the one or more compressed data blocks. For example, Adobe adopted H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec as its video codec in its products/services, such as, e.g., Adobe's media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused Instrumentalities to infringe other claims of the '535 patent. Adobe specifically intended and was aware that these normal and customary activities would infringe the '535 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the '535 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continues to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the '535 patent, knowing that such use constitutes infringement of the '535 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the '535 patent, in violation of 35 U.S.C. § 271(b).

90. Adobe has also infringed, and continues to infringe, claims of the '535 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the '535 patent, and constitute a material part of the invention. Adobe

knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the ’535 patent, not a staple article, and not a commodity of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the ’535 patent, in violation of 35 U.S.C. § 271(c).

91. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities’ compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the ’535 patent pursuant to 35 U.S.C. § 271.

92. As a result of Adobe’s infringement of the ’535 patent, Plaintiff Realtime is entitled to monetary damages in an amount adequate to compensate for Adobe’s infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

COUNT V

INFRINGEMENT OF U.S. PATENT NO. 9,578,298

93. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

94. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States products that infringe the ’298 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe’s Adobe Media Encoder, Adobe Media Encoder Creative Cloud or CC (e.g. CC 2015.1), Adobe Media Encoder Creative Suite or CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Premiere Pro, Adobe Premiere Pro CC (e.g. CC 2015.1), Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe After Effects, Adobe After Effects CC, Adobe After

Effects CS (e.g. CS4, CS5, CS5.5, CS6), and all versions and variations thereof since the issuance of the ’298 patent (“Accused Instrumentalities”).

95. On information and belief, Adobe has directly infringed and continues to infringe the ’298 patent, for example, through its sale, offer for sale, importation, use and testing of the Accused Instrumentalities, which practices the method claimed by, for example, Claim 1 of the ’298 patent, namely, a method for processing a video stream of digital images, the method comprising the steps of: receiving the video stream which comprises at least one composite frame (FC), each composite frame containing a pair of stereoscopic digital images (L,R) according to a predetermined frame packing format; generating an output video stream which can be reproduced on a visualization apparatus, receiving metadata which determine an area occupied by one of the two images within said composite frame (FC), said metadata indicating either a geometry of the frame packing format or a frame packing type of said composite frame (FC); determining the area in the composite frame (FC) which is occupied by said one image of the stereoscopic pair within the composite frame based on said metadata; decoding only that part of the composite frame (FC) which contains said one image to be displayed, and generating an output frame containing said decoded image. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to Adobe’s customers.

96. The Accused Instrumentalities practice a method for processing a video stream of digital images by utilizing H.265 (HEVC) and/or its equivalents. One of the aims of HEVC is the production of a bitstream, as defined in definition 3.12 of the ITU-T

H.265 Series H: Audiovisual and Multimedia Systems, "Infrastructure of audiovisual services – Coding of moving video" High efficiency video coding ("HEVC Spec"): "bitstream: A sequence of bits, in the form of a NAL unit stream or a byte stream, that forms the representation of coded pictures and associated data forming one or more coded video sequences (CVSs)." Furthermore, according to a website maintained by Adobe, describing an "Overview of What's New in Adobe Media Encoder CC 2015.1," the Adobe "Media Encoder will offer Expanded UHD capabilities with the addition of **support for…the new HEVC (H.265) codec,** which, at 60% the size of comparable H.264 files, is ideal for delivering pristine 4K content online." *See* https://blogs.adobe.com/creativecloud/adobe-media-encoder-cc-2015-1/. Also, according to another website maintained by Adobe discussing "Supported file formats" for Adobe Premiere Pro or specifically "Supported native video and audio formats for import" for Adobe Premiere Pro, the format of "**HEVC (H.265)"** is listed a long with the text "H.265 media with resolutions up to 8192x4320." *See* https://helpx.adobe.com/premiere-pro/using/supported-file-formats.html. According to another website, "One of the most significant new features in Adobe Premiere Pro CC 2015.1 is the expanded native support for 4K Ultra HD & HDR formats to include…**HEVC (H.265)**," Adobe "Media Encoder will offer Expanded UHD capabilities with the addition of support for…the new **HEVC/H.265 codec**, which, at 60% the size of comparable H.264 files, is ideal for delivering pristine 4K content online." *See* http://www.4kshooters.net/2015/12/04/adobe-premiere-pro-cc-2015-1-update-brings-h-265-more-4k-native-support/. The fact that both Adobe Media Encoder and Adobe Premiere Pro CC supported HEVC/H.265 as early as 2015 has also been confirmed by other websites. *See e.g.*, https://larryjordan.com/articles/adobe-media-encoder-and-h-265-not-ready-for-prime-time/ ("One of the exciting features of the Nov. 2015 update to Adobe Media Encoder is its support for H.265, also known as HEVC, short for High Efficiency Video Coding, compression.").

97. The Accused Instrumentalities also practice receiving the video stream which comprises at least one composite frame (FC), each composite frame containing a pair of stereoscopic digital images (L,R) according to a predetermined frame packing format. For example, the coded bitstream when it contains a stereoscopic video in one of the frame packing arrangements such as side-by-side or top-and-bottom or segmented rectangular frame packing format as defined in the following sections of the ITU-T H.265 Series H: Audiovisual and Multimedia Systems, "Infrastructure of audiovisual services – Coding of moving video" High efficiency video coding ("HEVC Spec"): D.2.16 Frame packing arrangement SEI message syntax, D.3.16 Frame packing arrangement SEI message semantics, D.2.29 Segmented rectangular frame packing arrangement SEI message syntax, D.3.29 Segmented rectangular frame packing arrangement SEI message semantics.

98. The Accused Instrumentalities also practice generating an output video stream which can be reproduced on a visualization apparatus. For example, the output of the decoding process as defined above is a sequence of decoded pictures. *See, e.g.,* HEVC Spec at 3.39 ("3.39 decoded picture: A decoded picture is derived by decoding a coded picture"). Decoded pictures are the input of the display process. *Id.* at 3.47 ("3.47 display process: A process not specified in this Specification having, as its input, the cropped decoded pictures that are the output of the decoding process.").

99. The Accused Instrumentalities also practice receiving metadata which determine an area occupied by one of the two images within said composite frame, said metadata indicating either a geometry of the frame packing format or a frame packing type of said composite frame. For example, the HEVC spec provides the default display window parameter to support 2D compatible decoding of stereo formats. *See, e.g.*, HEVC Spec ("NOTE 9 – The default display window parameters in the VUI parameters of the SPS can be used by an encoder to indicate to a decoder that does not interpret the frame packing arrangement SEI message that the default display window is an area within only one of the two constituent frames.").

100. The Accused Instrumentalities also practice determining the area in the composite frame (FC) which is occupied by said one image of the stereoscopic pair within the composite frame based on said metadata. For example, the default display window parameter has been defined to support this application. The parameter syntax is defined in clause E.2.1 VUI parameters syntax, the semantics thereof being described in clause E.3.1 VUI parameters semantics. The usage of the Default Display Window for signaling the 2D single view in a stereoscopic frame packing format is illustrated in Note 9 of clause D.3.16 and Note 3 in Clause D.3.29 cited above.

101. The Accused Instrumentalities also practice decoding only that part of the composite frame which contains said one image to be displayed. For example, tiles are intended to support independent decoding of different picture regions. Clause 7.4.3.2.1 cited above illustrates the process to convert CTB picture scan in CTB tile scan to enable independent decoding of the tile. *See also* HEVC Spec:

> **row_height_minus1**[i] plus 1 specifies the height of the i-th tile row in units of coding tree blocks.
>
> The following variables are derived by invoking the coding tree block raster and tile scanning conversion process as specified in clause 6.5.1:
>
> – The list CtbAddrRsToTs[ctbAddrRs] for ctbAddrRs ranging from 0 to PicSizeInCtbsY − 1, inclusive, specifying the conversion from a CTB address in the CTB raster scan of a picture to a CTB address in the tile scan,
> – the list CtbAddrTsToRs[ctbAddrTs] for ctbAddrTs ranging from 0 to PicSizeInCtbsY − 1, inclusive, specifying the conversion from a CTB address in the tile scan to a CTB address in the CTB raster scan of a picture,
> – the list TileId[ctbAddrTs] for ctbAddrTs ranging from 0 to PicSizeInCtbsY − 1, inclusive, specifying the conversion from a CTB address in tile scan to a tile ID,
> – the list ColumnWidthInLumaSamples[i] for i ranging from 0 to num_tile_columns_minus1, inclusive, specifying the width of the i-th tile column in units of luma samples,
> – the list RowHeightInLumaSamples[j] for j ranging from 0 to num_tile_rows_minus1, inclusive, specifying the height of the j-th tile row in units of luma samples.
>
> The values of ColumnWidthInLumaSamples[i] for i ranging from 0 to num_tile_columns_minus1, inclusive, and RowHeightInLumaSamples[j] for j ranging from 0 to num_tile_rows_minus1, inclusive, shall all be greater than 0.
>
> The array MinTbAddrZs with elements MinTbAddrZs[x][y] for x ranging from 0 to (PicWidthInCtbsY << (CtbLog2SizeY − MinTbLog2SizeY)) − 1, inclusive, and y ranging from 0 to (PicHeightInCtbsY << (CtbLog2SizeY − MinTbLog2SizeY)) − 1, inclusive, specifying the conversion from a location (x, y) in units of minimum transform blocks to a transform block address in z-scan order, is derived by invoking the z-scan order array initialization process as specified in clause 6.5.2.

102. The Accused Instrumentalities also practice generating an output frame containing said extracted image. For example, there is an output of the tile decoding process. *See, e.g.,* HEVC Spec at 8.1.1 ("8.1.1 General…Input to this process is a bitstream. Output of this process is a list of decoded pictures.").

103. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the '298 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the method claimed by, for example, Claim 1 of the '298 patent, namely, a method for processing a video stream of digital images, the method comprising the steps of: receiving the video stream which comprises at least one composite frame (FC), each composite frame containing a pair of stereoscopic

digital images (L,R) according to a predetermined frame packing format; generating an output video stream which can be reproduced on a visualization apparatus, receiving metadata which determine an area occupied by one of the two images within said composite frame (FC), said metadata indicating either a geometry of the frame packing format or a frame packing type of said composite frame (FC); determining the area in the composite frame (FC) which is occupied by said one image of the stereoscopic pair within the composite frame based on said metadata; decoding only that part of the composite frame (FC) which contains said one image to be displayed, and generating an output frame containing said decoded image. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers.

104. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’298 patent.

105. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the HEVC (or H.265) standard.

106. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the methods claimed by the ’298 patent.

107. On information and belief, Adobe has had knowledge of the ’298 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’298 patent and knew of its infringement, including by way of this

lawsuit. By the time of trial, Adobe will have known and intended (since receiving such notice) that its continued actions would actively induce and contribute to the infringement of the claims of the ’298 patent.

108. Upon information and belief, Adobe’s affirmative acts of making, using, and selling the Accused Instrumentalities, and providing implementation services and technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’298 by practicing a method for processing a video stream of digital images, the method comprising the steps of: receiving the video stream which comprises at least one composite frame (FC), each composite frame containing a pair of stereoscopic digital images (L,R) according to a predetermined frame packing format; generating an output video stream which can be reproduced on a visualization apparatus, receiving metadata which determine an area occupied by one of the two images within said composite frame (FC), said metadata indicating either a geometry of the frame packing format or a frame packing type of said composite frame (FC); determining the area in the composite frame (FC) which is occupied by said one image of the stereoscopic pair within the composite frame based on said metadata; decoding only that part of the composite frame (FC) which contains said one image to be displayed, and generating an output frame containing said decoded image. For example, Adobe adopted HEVC (or H.265) and/or its equivalents as its video codec in its products/services, such as in Adobe’s media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused

Instrumentalities to infringe other claims of the ’298 patent. Adobe specifically intended and was aware that these normal and customary activities would infringe the ’298 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the ’298 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continue to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the ’298 patent, knowing that such use constitutes infringement of the ’298 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the ’298 patent, in violation of 35 U.S.C. § 271(b).

109. Adobe has also infringed, and continues to infringe, claims of the ’298 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the ’298 patent, and constitute a material part of the invention. Adobe knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the ’298 patent, not a staple article, and not a commodity of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the ’298 patent, in violation of 35 U.S.C. § 271(c).

110. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities’ compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the ’298 patent pursuant to 35 U.S.C. § 271.

111. As a result of Adobe's infringement of the '298 patent, Plaintiff Realtime is entitled to monetary damages in an amount adequate to compensate for Adobe's infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

COUNT VI

INFRINGEMENT OF U.S. PATENT NO. 9,762,907

112. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

113. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States Adobe products that infringe the '907 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe's Adobe Media Encoder, Adobe Media Encoder Creative Cloud or CC, Adobe Media Encoder Creative Suite or CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Premiere Pro, Adobe Premiere Pro CC, Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Flash, Adobe Flash Player (e.g. Adobe Flash Player 9), Adobe Flash Media Server, Adobe Flash Media Encoding Server, Adobe After Effects, Adobe After Effects CC, Adobe After Effects CS (e.g. CS4, CS5, CS5.5, CS6), Adobe HTTP Dynamic Streaming (HDS), and all versions and variations thereof since the issuance of the '907 patent ("Accused Instrumentalities").

114. On information and belief, Adobe has directly infringed and continues to infringe the '907 patent, for example, through its sale, offer for sale, importation, use and testing of the Accused Instrumentalities, which practices the system claimed by, for example, Claim 1 of the '907 patent, namely, a system comprising: one or more different

asymmetric data compression algorithms, wherein each algorithm of the one or more different asymmetric data compression algorithms utilizes one or more asymmetric data compression routines of a plurality of different asymmetric data compression routines, wherein a first asymmetric data compression routine of the plurality of different asymmetric data compression routines is configured to produce compressed data with a higher data rate for a given data throughput than a second asymmetric data compression routine of the plurality of different asymmetric data compression routines; and a processor configured: to analyze one or more data parameters from one or more data blocks containing video data, wherein at least one data parameter relates to an expected or anticipated throughput of a communications channel; and to select two or more different data compression routines from among a plurality of different data compression routines based upon, at least in part, the one or more data parameters relating to the expected or anticipated throughput of the communications channel. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to Adobe's customers.

115. The Accused Instrumentalities include, or practice a system involving data compression with asymmetric compressors. For example, the Accused Instrumentalities utilize H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent

codec. H.264 is "the most widely used codec on the planet." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-H.264-74735.aspx. A "codec" is also a compression technology that has "two components, an **encoder to compress the files**, and a decoder to decompress. There are codecs for…video (Cinepark, MPEG-2, **H.264**, VP8)." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-a-Codec-74487.aspx; https://forums.adobe.com/thread/729526 ("Compression is essential for reducing the size of movies so that they can be stored, transmitted, and played back effectively. Compression is achieved by an encoder; decompression is achieved by a decoder. Encoders and decoders are known by the common term codec."). According to a website maintained by Adobe, "you **can produce H.264 video** with Adobe Media Encoder CS4 and Adobe Flash Media Encoding Server 3.5," "Adobe added **H.264 playback support** to Adobe Flash Player 9 Update 3 back in 2007," Adobe "Flash Media Server and Flash Player can **both stream and play back any H.264 file** in virtually any format," the encoding capabilities of the Adobe Flash Media Encoding Server using "**H.264**-related parameters" are described (e.g. using CAVLC, CABAC) and under a headline stating "Adobe Media Encoder," the website states: "Adobe significantly enhanced the Flash Video Encoder in Creative Suite 4. There is now both stand-alone operation and batch encoding capabilities. As before, you can **access H.264 encoding** by choosing different formats in the Format pop-up menu. When producing for Flash Player, you should always use the FLV|F4V option, which lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html. Another website maintained by Adobe mentions that "You have two primary options for using Adobe Media Encoder with After Effects to create videos in **H.264**." The website also mentions that "Since Adobe Media Encoder already has superior H.264…exporting capabilities, it was more prudent to rely on Adobe Media Encoder for export of these formats." *See* https://helpx.adobe.com/after-effects/kb/export-h264.html. Another website maintained by Adobe mentions that the "Adobe Flash Player 9 Update 3 is taking a step into the high-definition (HD) video realm in a major way by adding MPEG-4 video…[which] utilizes crisp, powerful **H.264 encoding**" and "The new Adobe Media Player…will **also support H.264**" and "The addition of **H.264**…**support** in Flash Player 9 Update 3 allows you to easily use high-definition, industry standard video and audio." *See* http://www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html. Adobe HTTP Dynamic Streaming (HDS) also supports methods involving data compression with asymmetric compressors or codecs other than H.264 because it "enables high-quality (**H.264 or VP6**), network-efficient HTTP streaming for media delivery that is tightly integrated with Adobe® Access software for robust content protection in the Adobe Flash® Player 10.1 or later and Adobe AIR®2 or later runtimes," is also "built on standards and deployed using standard HTTP servers (Apache), standard media format (MP4 fragment) using standard codecs (**H.264/AAC**), open APIs (Flash Player), and an open source framework for building media players (OSMF)" and "like other Flash Player supported delivery methods," Adobe HDS "supports **H.264 video and VP6 codecs** required for the highest quality video on the platform." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS also allows users to encode content "using high-quality Flash Player compatible **codecs**

(**VP6/MP3, H264/AAC**)," deliver "high-definition video up to 1080p, with bitrates from 700kbps up to and beyond 6Mbps, using either **H.264 or VP6 video codecs,"** and archive "live, high-definition streams on the server and enable HD DVR functionality (for example, instant replay and time shifting) with support for **H.264** stream recording for RTMP and HTTP Dynamic Streaming." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. Adobe Media Encoder also "lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html.

116. The Accused Instrumentalities also include and practice a system comprising: one or more different asymmetric data compression algorithms, wherein each algorithm of the one or more different asymmetric data compression algorithms utilizes one or more asymmetric data compression routines of a plurality of different asymmetric data compression routines. The one or more asymmetric data compression algorithms can be, for example, H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec. Based on the bitrate and/or resolution parameter identified (e.g. bitrate, max video bitrate, resolution, group of pictures (GOP) structure or frame type within a GOP structure), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., "baseline," "extended," "main", or "high") corresponds with that parameter, then select between at least two asymmetric compressors, which may serve as the one or more different asymmetric data compression routines. If baseline or extended is the corresponding profile, then the system will select a Context-Adaptive Variable Length Coding ("CAVLC") entropy encoder, which may serve as the first or second asymmetric data compression routine or asymmetric

compressor. If main or high is the corresponding profile, then the system will select a Context-Adaptive Binary Arithmetic Coding ("CABAC") entropy encoder, which may serve as the first or second asymmetric data compression routine or asymmetric compressor. Both encoders are asymmetric compressors because it takes a longer period of time for them to compress data than to decompress data. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

	Baseline	Extended	Main	High	High 10
I and P Slices	Yes	Yes	Yes	Yes	Yes
B Slices	No	Yes	Yes	Yes	Yes
SI and SP Slices	No	Yes	No	No	No
Multiple Reference Frames	Yes	Yes	Yes	Yes	Yes
In-Loop Deblocking Filter	Yes	Yes	Yes	Yes	Yes
CAVLC Entropy Coding	Yes	Yes	Yes	Yes	Yes
CABAC Entropy Coding	No	No	Yes	Yes	Yes
Flexible Macroblock Ordering (FMO)	Yes	Yes	No	No	No
Arbitrary Slice Ordering (ASO)	Yes	Yes	No	No	No
Redundant Slices (RS)	Yes	Yes	No	No	No
Data Partitioning	No	Yes	No	No	No
Interlaced Coding (PicAFF, MBAFF)	No	Yes	Yes	Yes	Yes
4:2:0 Chroma Format	Yes	Yes	Yes	Yes	Yes
Monochrome Video Format (4:0:0)	No	No	No	Yes	Yes
4:2:2 Chroma Format	No	No	No	No	No
4:4:4 Chroma Format	No	No	No	No	No
8 Bit Sample Depth	Yes	Yes	Yes	Yes	Yes
9 and 10 Bit Sample Depth	No	No	No	No	Yes
11 to 14 Bit Sample Depth	No	No	No	No	No
8x8 vs. 4x4 Transform Adaptivity	No	No	No	Yes	Yes
Quantization Scaling Matrices	No	No	No	Yes	Yes
Separate Cb and Cr QP control	No	No	No	Yes	Yes
Separate Color Plane Coding	No	No	No	No	No
Predictive Lossless Coding	No	No	No	No	No

See http://web.cs.ucla.edu/classes/fall03/cs218/paper/H.264_MPEG4_Tutorial.pdf at 7:

The following table summarizes the two major types of entropy coding: Variable Length Coding (VLC) and Context Adaptive Binary Arithmetic Coding (CABAC). CABAC offers superior coding efficiency over VLC by adapting to the changing probability distribution of symbols, by exploiting correlation between symbols, and by adaptively exploiting bit correlations using arithmetic coding. H.264 also supports Context Adaptive Variable Length Coding (CAVLC) which offers superior entropy coding over VLC without the full cost of CABAC.

H.264 Entropy Coding – Comparison of Approaches

Characteristics	Variable Length Coding (VLC)	Context Adaptive Binary Arithmetic Coding(CABAC)
• Where it is used	MPEG-2, MPEG-4 ASP	H.264/MPEG-4 AVC (high efficiency option)
• Probability distribution	**Static** - Probabilities never change	**Adaptive** - Adjusts probabilities based on actual data
• Leverages correlation between symbols	**No** - Conditional probabilities ignored	**Yes** - Exploits symbol correlations by using "contexts"
• Non-integer code words	**No** - Low coding efficiency forhigh probability symbols	**Yes** - Exploits "arithmetic coding" which generates non-integer code words for higher efficiency

Moreover, the H.264 Standard requires a bit-flag descriptor, which is set to determine the correct decoder for the corresponding encoder. As shown below, if the flag = 0, then CAVLC must have been selected as the encoder; if the flag = 1, then CABAC must have been selected as the encoder. *See* https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.264-201304-S!!PDF-E&type=items (Rec. ITU-T H.264 (04/2013)) at 80:

entropy_coding_mode_flag selects the entropy decoding method to be applied for the syntax elements for which two descriptors appear in the syntax tables as follows:

– If entropy_coding_mode_flag is equal to 0, the method specified by the left descriptor in the syntax table is applied (Exp-Golomb coded, see clause 9.1 or CAVLC, see clause 9.2).

– Otherwise (entropy_coding_mode_flag is equal to 1), the method specified by the right descriptor in the syntax table is applied (CABAC, see clause 9.3).

117. The Accused Instrumentalities also practice wherein a first asymmetric data compression routine of the plurality of different asymmetric data compression routines is configured to produce compressed data with a higher data rate for a given data throughput than a second asymmetric data compression routine of the plurality of different asymmetric data compression routines. For example, Adobe HTTP Dynamic Streaming (HDS) contains the "ability **to shift quality depending on bandwidth and computer power**. HTTP progressive delivery consumes more bandwidth because it's not intelligent enough to throttle the delivery (for example, a 30-minute video will be fully downloaded whether the user watches it or not)." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS can also **"[d]etect the client's bandwidth and computer resources** and serve them content fragments encoded at the most appropriate bitrate." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html.

118. A video data block is organized by the group of pictures (GOP) structure, which is a "collection of successive pictures within a coded video stream." *See* https://en.wikipedia.org/wiki/Group_of_pictures. A GOP structure can contain intra coded pictures (I picture or I frame), predictive coded pictures (P picture or P frame), bipredictive coded pictures (B picture or B frame) and direct coded pictures (D picture or D frames, or DC direct coded pictures which are used only in MPEG-1 video). *See* https://en.wikipedia.org/wiki/Video_compression_picture_types (for descriptions of I frames, P frames and B frames); https://en.wikipedia.org/wiki/MPEG-1#D-frames (for descriptions of D frames). Thus, at least a portion of a video data block would also make up a GOP structure and could also contain I frames, P frames, B frames and/or D frames. The GOP structure also reflects the size of a video data block, and the GOP structure can

be controlled and used to fine-tune other parameters (e.g. bitrate, max video bitrate and resolution parameters) or even be considered as a parameter by itself.

119. The produced compressed data formed by the Accused Instrumentalities via the first asymmetric data compression routine of the plurality of different asymmetric data compression routines can also be organized in a GOP structure (see above). After its selection, the asymmetric compressor or the first/second asymmetric data compression routine (CAVLC or CABAC) will compress the video data to provide various compressed data blocks, which can also be organized in a GOP structure, as discussed previously above. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

> **Entropy Coding**
> For entropy coding, H.264 may use an enhanced VLC, a more complex context-adaptive variable-length coding (CAVLC) or an ever more complex Context-adaptive binary-arithmetic coding (CABAC) which are complex techniques to losslessly compress syntax elements in the video stream knowing the probabilities of syntax elements in a given context. The use of CABAC can improve the compression of around 5-7%. CABAC may requires a 30-40% of total processing power to be accomplished.

See
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.602.1581&rep=rep1&type=pdf at 13:

> Typical compression ratios to maintain excellent quality are:
> - 10:1 for general images using JPEG
> - 30:1 for general video using H.263 and MPEG-2
> - 60:1 for general video using H.264 and WMV9

See http://www.ijera.com/papers/Vol3_issue4/BM34399403.pdf at 2:

Most visual communication systems today use Baseline Profile. Baseline is the simplest H.264 profile and defines, for example, zigzag scanning of the picture and using 4:2:0 (YUV video formats) chrominance sampling. In Baseline Profile, the picture is split in blocks consisting of 4x4 pixels, and each block is processed separately. Another important element of the Baseline Profile is the use of Universal Variable Length Coding (UVLC) and Context Adaptive Variable Length Coding (CAVLC) entropy coding techniques.

The Extended and Main Profiles includes the functionality of the Baseline Profile and add improvements to the predictions algorithms. Since transmitting every single frame (think 30 frames per second for good quality video) is not feasible if you are trying to reduce the bit rate 1000-2000 times, temporal and motion prediction are heavily used in H.264, and allow transmitting only the difference between one frame and the previous frames. The result is spectacular efficiency gain, especially for scenes with little change and motion.

The High Profile is the most powerful profile in H.264, and it allows most efficient coding of video. For example, large coding gain achieved through the use of Context Adaptive Binary Arithmetic Coding (CABAC) encoding which is more efficient than the UVLC/CAVLC used in Baseline Profile.

The High Profile also uses adaptive transform that decides on the fly if 4x4 or 8x8-pixel blocks should be used. For example, 4x4 blocks are used for the parts of the picture that are dense with detail, while parts that have little detail are transformed using 8x8 blocks.

120. The Accused Instrumentalities also include and practice a system comprising: a processor configured: to analyze one or more data parameters from one or more data blocks containing video data, wherein at least one data parameter relates to an expected or anticipated throughput of a communications channel; and to select two or more different data compression routines from among a plurality of different data compression routines based upon, at least in part, the one or more data parameters relating to the expected or anticipated throughput of the communications channel. As discussed briefly above and in more detail below, examples of such data parameters include throughput or

bandwidth, bitrate (or max video bitrate), and resolution. For the at least one data parameter relating to an expected or anticipated throughout of a communications channel and selecting data compression routine(s) based on those data parameter(s), Adobe HTTP Dynamic Streaming (HDS) contains the "ability **to shift quality depending on bandwidth and computer power**. HTTP progressive delivery consumes more bandwidth because it's not intelligent enough to throttle the delivery (for example, a 30-minute video will be fully downloaded whether the user watches it or not)." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS can also **"[d]etect the client's bandwidth and computer resources** and serve them content fragments encoded at the most appropriate bitrate." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. As to data parameters overall, different data parameters correspond with different end applications. H.264 provides for multiple different ranges of such data parameters, each included in the "profiles" and "levels" as defined by the H.264 standard, from the below shown paragraphs from a white paper and Wikipedia. *See* http://www.axis.com/files/whitepaper/wp_h264_31669_en_0803_lo.pdf at 5:

4. H.264 profiles and levels

The joint group involved in defining H.264 focused on creating a simple and clean solution, limiting options and features to a minimum. An important aspect of the standard, as with other video standards, is providing the capabilities in profiles (sets of algorithmic features) and levels (performance classes) that optimally support popular productions and common formats.

H.264 has seven profiles, each targeting a specific class of applications. Each profile defines what feature set the encoder may use and limits the decoder implementation complexity.

Network cameras and video encoders will most likely use a profile called the baseline profile, which is intended primarily for applications with limited computing resources. The baseline profile is the most suitable given the available performance in a real-time encoder that is embedded in a network video product. The profile also enables low latency, which is an important requirement of surveillance video and also particularly important in enabling real-time, pan/tilt/zoom (PTZ) control in PTZ network cameras.

H.264 has 11 levels or degree of capability to limit performance, bandwidth and memory requirements. Each level defines the bit rate and the encoding rate in macroblock per second for resolutions ranging from QCIF to HDTV and beyond. The higher the resolution, the higher the level required.

See https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC:

Levels with maximum property values

Level	Max decoding speed		Max frame size		Max video bit rate for video coding layer (VCL) kbit/s			Examples for high resolution @ highest frame rate (max stored frames) Toggle additional details
	Luma samples/s	Macroblocks/s	Luma samples	Macroblocks	Baseline, Extended and Main Profiles	High Profile	High 10 Profile	
1	380,160	1,485	25,344	99	64	80	192	176×144@15.0 (4)
1b	380,160	1,485	25,344	99	128	160	384	176×144@15.0 (4)
1.1	768,000	3,000	101,376	396	192	240	576	352×288@7.5 (2)
1.2	1,536,000	6,000	101,376	396	384	480	1,152	352×288@15.2 (6)
1.3	3,041,280	11,880	101,376	396	768	960	2,304	352×288@30.0 (6)
2	3,041,280	11,880	101,376	396	2,000	2,500	6,000	352×288@30.0 (6)
2.1	5,068,800	19,800	202,752	792	4,000	5,000	12,000	352×576@25.0 (6)
2.2	5,184,000	20,250	414,720	1,620	4,000	5,000	12,000	720×576@12.5 (5)
3	10,368,000	40,500	414,720	1,620	10,000	12,500	30,000	720×576@25.0 (5)
3.1	27,648,000	108,000	921,600	3,600	14,000	17,500	42,000	1,280×720@30.0 (5)
3.2	55,296,000	216,000	1,310,720	5,120	20,000	25,000	60,000	1,280×1,024@42.2 (4)
4	62,914,560	245,760	2,097,152	8,192	20,000	25,000	60,000	2,048×1,024@30.0 (4)
4.1	62,914,560	245,760	2,097,152	8,192	50,000	62,500	150,000	2,048×1,024@30.0 (4)
4.2	133,693,440	522,240	2,228,224	8,704	50,000	62,500	150,000	2,048×1,080@60.0 (4)
5	150,994,944	589,824	5,652,480	22,080	135,000	168,750	405,000	3,672×1,536@26.7 (5)
5.1	251,658,240	983,040	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@26.7 (5)
5.2	530,841,600	2,073,600	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@56.3 (5)

121. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the ’907 patent, for example, through its own use and testing of the

Accused Instrumentalities, which when used, practices the system claimed by, for example, Claim 1 of the ’907 patent, namely, a system comprising: one or more different asymmetric data compression algorithms, wherein each algorithm of the one or more different asymmetric data compression algorithms utilizes one or more asymmetric data compression routines of a plurality of different asymmetric data compression routines, wherein a first asymmetric data compression routine of the plurality of different asymmetric data compression routines is configured to produce compressed data with a higher data rate for a given data throughput than a second asymmetric data compression routine of the plurality of different asymmetric data compression routines; and a processor configured: to analyze one or more data parameters from one or more data blocks containing video data, wherein at least one data parameter relates to an expected or anticipated throughput of a communications channel; and to select two or more different data compression routines from among a plurality of different data compression routines based upon, at least in part, the one or more data parameters relating to the expected or anticipated throughput of the communications channel.

122. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’907 patent.

123. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the H.264 standard.

124. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the systems and/or methods claimed by the ’907 patent.

125. On information and belief, Adobe has had knowledge of the ’907 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’907 patent and knew of its infringement, including by way of this lawsuit. By the time of trial, Adobe will have known and intended (since receiving such notice) that its continued actions would actively induce and contribute to the infringement of the claims of the ’907 patent.

126. Upon information and belief, Adobe’s affirmative acts of making, using, and selling the Accused Instrumentalities, and providing implementation services and technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’907 patent by practicing a system comprising: one or more different asymmetric data compression algorithms, wherein each algorithm of the one or more different asymmetric data compression algorithms utilizes one or more asymmetric data compression routines of a plurality of different asymmetric data compression routines, wherein a first asymmetric data compression routine of the plurality of different asymmetric data compression routines is configured to produce compressed data with a higher data rate for a given data throughput than a second asymmetric data compression routine of the plurality of different asymmetric data compression routines; and a processor configured: to analyze one or more data parameters from one or more data blocks containing video data, wherein at least one data parameter relates to an expected or anticipated throughput of a communications channel; and to select two or more different data compression routines from among a plurality of different data compression routines

based upon, at least in part, the one or more data parameters relating to the expected or anticipated throughput of the communications channel. For example, Adobe adopted H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec as its video codec in its products/services, such as, e.g., Adobe's media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused Instrumentalities to infringe other claims of the '907 patent. Adobe specifically intended and was aware that these normal and customary activities would infringe the '907 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the '907 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continues to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the '907 patent, knowing that such use constitutes infringement of the '907 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the '907 patent, in violation of 35 U.S.C. § 271(b).

127. Adobe has also infringed, and continues to infringe, claims of the '907 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the '907 patent, and constitute a material part of the invention. Adobe knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the '907 patent, not a staple article, and not a commodity

of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the ’907 patent, in violation of 35 U.S.C. § 271(c).

128. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities’ compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the ’907 patent pursuant to 35 U.S.C. § 271.

129. As a result of Adobe’s infringement of the ’907 patent, Plaintiff Realtime is entitled to monetary damages in an amount adequate to compensate for Adobe’s infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

COUNT VII

INFRINGEMENT OF U.S. PATENT NO. 9,769,477

130. Plaintiff re-alleges and incorporates by reference the foregoing paragraphs, as if fully set forth herein.

131. On information and belief, Adobe has made, used, offered for sale, sold and/or imported into the United States Adobe products that infringe the ‘477 patent, and continues to do so. By way of illustrative example, these infringing products include, without limitation, Adobe’s Adobe Media Encoder, Adobe Media Encoder Creative Cloud or CC, Adobe Media Encoder Creative Suite or CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Premiere Pro, Adobe Premiere Pro CC, Adobe Premiere Pro CS (e.g. CS4, CS5, CS5.5, CS6), Adobe Flash, Adobe Flash Player (e.g. Adobe Flash Player 9), Adobe Flash Media Server, Adobe Flash Media Encoding Server, Adobe After Effects, Adobe After Effects CC, Adobe After Effects CS (e.g. CS4, CS5, CS5.5, CS6), Adobe HTTP

Dynamic Streaming (HDS), and all versions and variations thereof since the issuance of the ’477 patent (“Accused Instrumentalities”).

132. On information and belief, Adobe has directly infringed and continues to infringe the ’477 patent, for example, through its sale, offer for sale, importation, use and testing of the Accused Instrumentalities that practice, for example, Claim 1 of the ’477 patent, namely, a system, comprising: a plurality of different asymmetric data compression encoders, wherein each asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to utilize one or more data compression algorithms, and wherein a first asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to compress data blocks containing video or image data at a higher data compression rate than a second asymmetric data compression encoder of the plurality of different asymmetric data compression encoders; and one or more processors configured to: determine one or more data parameters, at least one of the determined one or more data parameters relating to a throughput of a communications channel measured in bits per second; and select one or more asymmetric data compression encoders from among the plurality of different asymmetric data compression encoders based upon, at least in part, the determined one or more data parameters. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to Adobe’s customers.

133. The Accused Instrumentalities include, or practice a system involving data compression with asymmetric compressors. For example, the Accused Instrumentalities

utilize H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec. H.264 is "the most widely used codec on the planet." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-H.264-74735.aspx. A "codec" is also a compression technology that has "two components, an **encoder to compress the files**, and a decoder to decompress. There are codecs for…video (Cinepark, MPEG-2, **H.264**, VP8)." *See* http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-a-Codec-74487.aspx; https://forums.adobe.com/thread/729526 ("Compression is essential for reducing the size of movies so that they can be stored, transmitted, and played back effectively. Compression is achieved by an encoder; decompression is achieved by a decoder. Encoders and decoders are known by the common term codec."). According to a website maintained by Adobe, "you **can produce H.264 video** with Adobe Media Encoder CS4 and Adobe Flash Media Encoding Server 3.5," "Adobe added **H.264 playback support** to Adobe Flash Player 9 Update 3 back in 2007," Adobe "Flash Media Server and Flash Player can **both stream and play back any H.264 file** in virtually any format," the encoding capabilities of the Adobe Flash Media Encoding Server using "**H.264-**related parameters" are described (e.g. using CAVLC, CABAC) and under a headline stating "Adobe Media Encoder," the website states: "Adobe significantly enhanced the Flash Video Encoder in Creative Suite 4. There is now both stand-alone operation and batch encoding capabilities. As before, you can **access H.264 encoding** by choosing different formats in the Format pop-up menu. When producing for Flash Player, you should always use the FLV|F4V option, which lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html. Another website maintained by Adobe mentions that "You have two primary options for using Adobe Media Encoder with After Effects to create videos in **H.264**." The website also mentions that "Since Adobe Media Encoder already has superior H.264…exporting capabilities, it was more prudent to rely on Adobe Media Encoder for export of these formats." *See* https://helpx.adobe.com/after-effects/kb/export-h264.html. Another website maintained by Adobe mentions that the "Adobe Flash Player 9 Update 3 is taking a step into the high-definition (HD) video realm in a major way by adding MPEG-4 video…[which] utilizes crisp, powerful **H.264 encoding**" and "The new Adobe Media Player…will **also support H.264**" and "The addition of **H.264**…**support** in Flash Player 9 Update 3 allows you to easily use high-definition, industry standard video and audio." *See* http://www.adobe.com/devnet/flashplayer/articles/hd_video_flash_player.html. Adobe HTTP Dynamic Streaming (HDS) also supports methods involving data compression with asymmetric compressors or codecs other than H.264 because it "enables high-quality (**H.264 or VP6**), network-efficient HTTP streaming for media delivery that is tightly integrated with Adobe® Access software for robust content protection in the Adobe Flash® Player 10.1 or later and Adobe AIR®2 or later runtimes," is also "built on standards and deployed using standard HTTP servers (Apache), standard media format (MP4 fragment) using standard codecs (**H.264/AAC**), open APIs (Flash Player), and an open source framework for building media players (OSMF)" and "like other Flash Player supported delivery methods," Adobe HDS "supports **H.264 video and VP6 codecs** required for the highest quality video on the platform." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS also allows users to encode content "using high-quality Flash Player compatible **codecs** (**VP6/MP3, H264/AAC**)," deliver "high-definition video up to 1080p, with bitrates from 700kbps up to and beyond 6Mbps, using either **H.264 or VP6 video codecs,"** and archive "live, high-definition streams on the server and enable HD DVR functionality (for example, instant replay and time shifting) with support for **H.264** stream recording for RTMP and HTTP Dynamic Streaming." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html. Adobe Media Encoder also "lets y**ou produce both VP6- and H.264-encoded files** for Flash Player distribution." *See* http://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html.

134. The Accused Instrumentalities also include or practice a system, comprising: a plurality of different asymmetric data compression encoders, wherein each asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to utilize one or more data compression algorithms. The one or more asymmetric data compression encoders can be, for example, H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec. Based on a bitrate, throughput and/or resolution parameter identified (e.g. bitrate, max video bitrate, resolution, group of pictures (GOP) structure or frame type within a GOP structure), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., "baseline," "extended," "main", or "high") corresponds with that parameter, then select between at least two asymmetric compressors, which may serve as an asymmetric data compression encoder or the one or more data compression algorithms. If baseline or extended is the corresponding profile, then the system will select a Context-Adaptive Variable Length Coding ("CAVLC") entropy encoder, which may serve as an asymmetric

data compression encoder, data compression algorithm or asymmetric compressor. If main or high is the corresponding profile, then the system will select a Context-Adaptive Binary Arithmetic Coding ("CABAC") entropy encoder, which may serve as another asymmetric data compression encoder, data compression algorithm or asymmetric compressor. Both encoders are asymmetric compressors because it takes a longer period of time for them to compress data than to decompress data. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

	Baseline	Extended	Main	High	High 10
I and P Slices	Yes	Yes	Yes	Yes	Yes
B Slices	No	Yes	Yes	Yes	Yes
SI and SP Slices	No	Yes	No	No	No
Multiple Reference Frames	Yes	Yes	Yes	Yes	Yes
In-Loop Deblocking Filter	Yes	Yes	Yes	Yes	Yes
CAVLC Entropy Coding	Yes	Yes	Yes	Yes	Yes
CABAC Entropy Coding	No	No	Yes	Yes	Yes
Flexible Macroblock Ordering (FMO)	Yes	Yes	No	No	No
Arbitrary Slice Ordering (ASO)	Yes	Yes	No	No	No
Redundant Slices (RS)	Yes	Yes	No	No	No
Data Partitioning	No	Yes	No	No	No
Interlaced Coding (PicAFF, MBAFF)	No	Yes	Yes	Yes	Yes
4:2:0 Chroma Format	Yes	Yes	Yes	Yes	Yes
Monochrome Video Format (4:0:0)	No	No	No	Yes	Yes
4:2:2 Chroma Format	No	No	No	No	No
4:4:4 Chroma Format	No	No	No	No	No
8 Bit Sample Depth	Yes	Yes	Yes	Yes	Yes
9 and 10 Bit Sample Depth	No	No	No	No	Yes
11 to 14 Bit Sample Depth	No	No	No	No	No
8x8 vs. 4x4 Transform Adaptivity	No	No	No	Yes	Yes
Quantization Scaling Matrices	No	No	No	Yes	Yes
Separate Cb and Cr QP control	No	No	No	Yes	Yes
Separate Color Plane Coding	No	No	No	No	No
Predictive Lossless Coding	No	No	No	No	No

See http://web.cs.ucla.edu/classes/fall03/cs218/paper/H.264_MPEG4_Tutorial.pdf

at 7:

The following table summarizes the two major types of entropy coding: Variable Length Coding (VLC) and Context Adaptive Binary Arithmetic Coding (CABAC). CABAC offers superior coding efficiency over VLC by adapting to the changing probability distribution of symbols, by exploiting correlation between symbols, and by adaptively exploiting bit correlations using arithmetic coding. H.264 also supports Context Adaptive Variable Length Coding (CAVLC) which offers superior entropy coding over VLC without the full cost of CABAC.

H.264 Entropy Coding – Comparison of Approaches

Characteristics	Variable Length Coding (VLC)	Context Adaptive Binary Arithmetic Coding(CABAC)
• Where it is used	MPEG-2, MPEG-4 ASP	H.264/MPEG-4 AVC (high efficiency option)
• Probability distribution	**Static** - Probabilities never change	**Adaptive** - Adjusts probabilities based on actual data
• Leverages correlation between symbols	**No** - Conditional probabilities ignored	**Yes** - Exploits symbol correlations by using "contexts"
• Non-integer code words	**No** - Low coding efficiency forhigh probability symbols	**Yes** - Exploits "arithmetic coding" which generates non-integer code words for higher efficiency

Moreover, the H.264 Standard requires a bit-flag descriptor, which is set to determine the correct decoder for the corresponding encoder. As shown below, if the flag = 0, then CAVLC must have been selected as the encoder; if the flag = 1, then CABAC must have been selected as the encoder. *See* https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.264-201304-S!!PDF-E&type=items (Rec. ITU-T H.264 (04/2013)) at 80:

entropy_coding_mode_flag selects the entropy decoding method to be applied for the syntax elements for which two descriptors appear in the syntax tables as follows:

– If entropy_coding_mode_flag is equal to 0, the method specified by the left descriptor in the syntax table is applied (Exp-Golomb coded, see clause 9.1 or CAVLC, see clause 9.2).

– Otherwise (entropy_coding_mode_flag is equal to 1), the method specified by the right descriptor in the syntax table is applied (CABAC, see clause 9.3).

135. The Accused Instrumentalities also practice wherein a first asymmetric

data compression encoder of the plurality of different asymmetric data compression encoders is configured to compress data blocks containing video or image data at a higher data compression rate than a second asymmetric data compression encoder of the plurality of different asymmetric data compression encoders. For example, Adobe HTTP Dynamic Streaming (HDS) contains the "ability **to shift quality [and compression rate] depending on bandwidth and computer power**. HTTP progressive delivery consumes more bandwidth because it's not intelligent enough to throttle the delivery (for example, a 30-minute video will be fully downloaded whether the user watches it or not)." *See* https://www.adobe.com/products/hds-dynamic-streaming/faq.html. Adobe HDS can also **"[d]etect the client's bandwidth and computer resources** and serve them content fragments encoded at the most appropriate bitrate." *See* https://www.adobe.com/products/hds-dynamic-streaming/features.html; https://www.adobe.com/devnet/adobe-media-server/articles/h264_encoding.html ("Like MPEG-2, H.264 uses three types of frames, meaning that each group of pictures (GOP) is comprised of I-, B-, and P-frames, with I-frames like the DCT-based compression used in DV and B- and P-frames referencing redundancies in other frames **to increase compression**…Finally, though it's not technically related to B-frames, consider the number of Slices per picture, which can be 1, 2, or 4. At a value of 4, the encoder divides each frame into four regions and searches for redundancies in other frames only within the respective region. This can **accelerate encoding** on multicore computers because the encoder can assign the regions to different cores.").

136. A video data block is organized by the group of pictures (GOP) structure, which is a "collection of successive pictures within a coded video stream." *See*

https://en.wikipedia.org/wiki/Group_of_pictures. A GOP structure can contain intra coded pictures (I picture or I frame), predictive coded pictures (P picture or P frame), bipredictive coded pictures (B picture or B frame) and direct coded pictures (D picture or D frames, or DC direct coded pictures which are used only in MPEG-1 video). *See* https://en.wikipedia.org/wiki/Video_compression_picture_types (for descriptions of I frames, P frames and B frames); https://en.wikipedia.org/wiki/MPEG-1#D-frames (for descriptions of D frames). Thus, at least a portion of a video data block would also make up a GOP structure and could also contain I frames, P frames, B frames and/or D frames. The GOP structure also reflects the size of a video data block, and the GOP structure can be controlled and used to fine-tune other parameters (e.g. bitrate, max video bitrate and resolution parameters) or even be considered as a parameter by itself.

137. The compressed data blocks that the different asymmetric data compression encoders are configured to compress can also be organized in a GOP structure (see above). After its selection, the asymmetric compressor (CAVLC or CABAC) will compress the video data to provide various compressed data blocks, which can also be organized in a GOP structure, as discussed previously above. *See* https://sonnati.wordpress.com/2007/10/29/how-h-264-works-part-ii/:

> **Entropy Coding**
> For entropy coding, H.264 may use an enhanced VLC, a more complex context-adaptive variable-length coding (CAVLC) or an ever more complex Context-adaptive binary-arithmetic coding (CABAC) which are complex techniques to losslessly compress syntax elements in the video stream knowing the probabilities of syntax elements in a given context. The use of CABAC can improve the compression of around 5-7%. CABAC may requires a 30-40% of total processing power to be accomplished.

See http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.602.1581&rep=rep1&type=pdf at 13:

Typical compression ratios to maintain excellent quality are:

- 10:1 for general images using JPEG
- 30:1 for general video using H.263 and MPEG-2
- 60:1 for general video using H.264 and WMV9

See http://www.ijera.com/papers/Vol3_issue4/BM34399403.pdf at 2:

Most visual communication systems today use Baseline Profile. Baseline is the simplest H.264 profile and defines, for example, zigzag scanning of the picture and using 4:2:0 (YUV video formats) chrominance sampling. In Baseline Profile, the picture is split in blocks consisting of 4x4 pixels, and each block is processed separately. Another important element of the Baseline Profile is the use of Universal Variable Length Coding (UVLC) and Context Adaptive Variable Length Coding (CAVLC) entropy coding techniques.

The Extended and Main Profiles includes the functionality of the Baseline Profile and add improvements to the predictions algorithms. Since transmitting every single frame (think 30 frames per second for good quality video) is not feasible if you are trying to reduce the bit rate 1000-2000 times, temporal and motion prediction are heavily used in H.264, and allow transmitting only the difference between one frame and the previous frames. The result is spectacular efficiency gain, especially for scenes with little change and motion.

The High Profile is the most powerful profile in H.264, and it allows most efficient coding of video. For example, large coding gain achieved through the use of Context Adaptive Binary Arithmetic Coding (CABAC) encoding which is more efficient than the UVLC/CAVLC used in Baseline Profile.

The High Profile also uses adaptive transform that decides on the fly if 4x4 or 8x8-pixel blocks should be used. For example, 4x4 blocks are used for the parts of the picture that are dense with detail, while parts that have little detail are transformed using 8x8 blocks.

138. The Accused Instrumentalities also include or practice a system, comprising: one or more processors configured to: determine one or more data parameters, at least one of the determined one or more data parameters relating to a throughput of a communications channel measured in bits per second; and select one or more asymmetric data compression encoders from among the plurality of different asymmetric data compression encoders based upon, at least in part, the determined one or more data

parameters. As shown below, examples of such data parameters include throughput or bandwidth, bitrate (or max video bitrate), and resolution. Different parameters correspond with different end applications. The asymmetric data compression encoder will then be selected based on the determined one or more data parameters, as discussed above. For example, based on the determined one or more data parameters, such as a bitrate, throughput and/or resolution parameter that is determined (e.g. bitrate, max video bitrate, resolution, group of pictures (GOP) structure or frame type within a GOP structure), any H.264-compliant system such as the Accused Instrumentalities would determine which profile (e.g., "baseline," "extended," "main", or "high") corresponds with that data parameter, then accordingly select an asymmetric data compression encoder based on that data parameter. H.264 provides for multiple different ranges of such data parameters, each included in the "profiles" and "levels" as defined by the H.264 standard, from the below shown paragraphs from a white paper and Wikipedia. *See* http://www.axis.com/files/whitepaper/wp_h264_31669_en_0803_lo.pdf at 5:

4. H.264 profiles and levels

The joint group involved in defining H.264 focused on creating a simple and clean solution, limiting options and features to a minimum. An important aspect of the standard, as with other video standards, is providing the capabilities in profiles (sets of algorithmic features) and levels (performance classes) that optimally support popular productions and common formats.

H.264 has seven profiles, each targeting a specific class of applications. Each profile defines what feature set the encoder may use and limits the decoder implementation complexity.

Network cameras and video encoders will most likely use a profile called the baseline profile, which is intended primarily for applications with limited computing resources. The baseline profile is the most suitable given the available performance in a real-time encoder that is embedded in a network video product. The profile also enables low latency, which is an important requirement of surveillance video and also particularly important in enabling real-time, pan/tilt/zoom (PTZ) control in PTZ network cameras.

H.264 has 11 levels or degree of capability to limit performance, bandwidth and memory requirements. Each level defines the bit rate and the encoding rate in macroblock per second for resolutions ranging from QCIF to HDTV and beyond. The higher the resolution, the higher the level required.

See https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC:

Levels with maximum property values

Level	Max decoding speed		Max frame size		Max video bit rate for video coding layer (VCL) kbit/s			Examples for high resolution @ highest frame rate (max stored frames) Toggle additional details
	Luma samples/s	Macroblocks/s	Luma samples	Macroblocks	Baseline, Extended and Main Profiles	High Profile	High 10 Profile	
1	380,160	1,485	25,344	99	64	80	192	176×144@15.0 (4)
1b	380,160	1,485	25,344	99	128	160	384	176×144@15.0 (4)
1.1	768,000	3,000	101,376	396	192	240	576	352×288@7.5 (2)
1.2	1,536,000	6,000	101,376	396	384	480	1,152	352×288@15.2 (6)
1.3	3,041,280	11,880	101,376	396	768	960	2,304	352×288@30.0 (6)
2	3,041,280	11,880	101,376	396	2,000	2,500	6,000	352×288@30.0 (6)
2.1	5,068,800	19,800	202,752	792	4,000	5,000	12,000	352×576@25.0 (6)
2.2	5,184,000	20,250	414,720	1,620	4,000	5,000	12,000	720×576@12.5 (5)
3	10,368,000	40,500	414,720	1,620	10,000	12,500	30,000	720×576@25.0 (5)
3.1	27,648,000	108,000	921,600	3,600	14,000	17,500	42,000	1,280×720@30.0 (5)
3.2	55,296,000	216,000	1,310,720	5,120	20,000	25,000	60,000	1,280×1,024@42.2 (4)
4	62,914,560	245,760	2,097,152	8,192	20,000	25,000	60,000	2,048×1,024@30.0 (4)
4.1	62,914,560	245,760	2,097,152	8,192	50,000	62,500	150,000	2,048×1,024@30.0 (4)
4.2	133,693,440	522,240	2,228,224	8,704	50,000	62,500	150,000	2,048×1,080@60.0 (4)
5	150,994,944	589,824	5,652,480	22,080	135,000	168,750	405,000	3,672×1,536@26.7 (5)
5.1	251,658,240	983,040	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@26.7 (5)
5.2	530,841,600	2,073,600	9,437,184	36,864	240,000	300,000	720,000	4,096×2,304@56.3 (5)

139. Therefore, from at least the above, Adobe has directly infringed and continues to infringe the '477 patent, for example, through its own use and testing of the Accused Instrumentalities, which when used, practices the system claimed by, for example, Claim 1 of the '477 patent, namely, a system, comprising: a plurality of different asymmetric data compression encoders, wherein each asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to utilize one or more data compression algorithms, and wherein a first asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to compress data blocks containing video or image data at a higher data compression rate than a second asymmetric data compression encoder of the plurality of different asymmetric data compression encoders; and one or more processors configured

to: determine one or more data parameters, at least one of the determined one or more data parameters relating to a throughput of a communications channel measured in bits per second; and select one or more asymmetric data compression encoders from among the plurality of different asymmetric data compression encoders based upon, at least in part, the determined one or more data parameters. Upon information and belief, Adobe uses the Accused Instrumentalities to practice infringing methods for its own internal non-testing business purposes, while testing the Accused Instrumentalities, and while providing technical support and repair services for the Accused Instrumentalities to their customers.

140. On information and belief, Adobe also directly infringes and continues to infringe other claims of the ’477 patent.

141. On information and belief, all of the Accused Instrumentalities perform the claimed methods in substantially the same way, e.g., in the manner specified in the H.264 standard.

142. On information and belief, use of the Accused Instrumentalities in their ordinary and customary fashion results in infringement of the systems and/or methods claimed by the ’477 patent.

143. On information and belief, Adobe has had knowledge of the ’477 patent since at least the filing of this Complaint or shortly thereafter, and on information and belief, Adobe knew of the ’477 patent and knew of its infringement, including by way of this lawsuit. By the time of trial, Adobe will have known and intended (since receiving such notice) that its continued actions would actively induce and contribute to the infringement of the claims of the ’477 patent.

144. Upon information and belief, Adobe’s affirmative acts of making, using,

and selling the Accused Instrumentalities, and providing implementation services and technical support to users of the Accused Instrumentalities, including, e.g., through training, demonstrations, brochures, installation and user guides, have induced and continue to induce users of the Accused Instrumentalities to use them in their normal and customary way to infringe the ’477 patent by practicing a system, comprising: a plurality of different asymmetric data compression encoders, wherein each asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to utilize one or more data compression algorithms, and wherein a first asymmetric data compression encoder of the plurality of different asymmetric data compression encoders is configured to compress data blocks containing video or image data at a higher data compression rate than a second asymmetric data compression encoder of the plurality of different asymmetric data compression encoders; and one or more processors configured to: determine one or more data parameters, at least one of the determined one or more data parameters relating to a throughput of a communications channel measured in bits per second; and select one or more asymmetric data compression encoders from among the plurality of different asymmetric data compression encoders based upon, at least in part, the determined one or more data parameters. For example, Adobe adopted H.264 and/or H.264 with Scalable Video Coding (SVC) and/or an equivalent codec as its video codec in its products/services, such as, e.g., Adobe’s media encoder, Flash player, Flash media server, Flash media server encoder, special effects and video editing software. For similar reasons, Adobe also induces its customers to use the Accused Instrumentalities to infringe other claims of the ’477 patent. Adobe specifically intended and was aware that these normal and customary activities would infringe the ’477 patent. Adobe performed the acts that constitute induced infringement, and would induce actual infringement, with the knowledge of the ’477 patent and with the knowledge, or willful blindness to the probability, that the induced acts would constitute infringement. On information and belief, Adobe engaged in such inducement to promote the sales of the Accused Instrumentalities. Accordingly, Adobe has induced and continues to induce users of the Accused Instrumentalities to use the Accused Instrumentalities in their ordinary and customary way to infringe the ’477 patent, knowing that such use constitutes infringement of the ’477 patent. Accordingly, Adobe has been, and currently is, inducing infringement of the ’477 patent, in violation of 35 U.S.C. § 271(b).

145. Adobe has also infringed, and continues to infringe, claims of the ’477 patent by offering to commercially distribute, commercially distributing, making, and/or importing the Accused Instrumentalities, which are used in practicing the process, or using the systems, of the ’477 patent, and constitute a material part of the invention. Adobe knows the components in the Accused Instrumentalities to be especially made or especially adapted for use in infringement of the ’477 patent, not a staple article, and not a commodity of commerce suitable for substantial noninfringing use. Accordingly, Adobe has been, and currently is, contributorily infringing the ’477 patent, in violation of 35 U.S.C. § 271(c).

146. By making, using, offering for sale, selling and/or importing into the United States the Accused Instrumentalities, and touting the benefits of using the Accused Instrumentalities’ compression features, Adobe has injured Realtime and is liable to Realtime for infringement of the ’477 patent pursuant to 35 U.S.C. § 271.

147. As a result of Adobe’s infringement of the ’477 patent, Plaintiff Realtime is entitled to monetary damages in an amount adequate to compensate for Adobe’s

infringement, but in no event less than a reasonable royalty for the use made of the invention by Adobe, together with interest and costs as fixed by the Court.

PRAYER FOR RELIEF

WHEREFORE, Plaintiff Realtime respectfully requests that this Court enter:

a. A judgment in favor of Plaintiff that Defendant has infringed, literally and/or under the doctrine of equivalents, the ’046, ’462, ’442, ’535, ’298, ’907, and ’477 patents (the “Asserted Patents”);

b. A judgment and order requiring Defendant to pay Plaintiff its damages, costs, expenses, and prejudgment and post-judgment interest for its infringement of the Asserted Patents, as provided under 35 U.S.C. § 284;

c. A judgment and order requiring Defendant to provide an accounting and to pay supplemental damages to Realtime, including without limitation, prejudgment and post-judgment interest;

d. A permanent injunction prohibiting Defendant from further acts of infringement of the Asserted Patents;

e. A judgment and order finding that this is an exceptional case within the meaning of 35 U.S.C. § 285 and awarding to Plaintiff its reasonable attorneys’ fees against Defendant; and

f. Any and all other relief as the Court may deem appropriate and just under the circumstances.

DEMAND FOR JURY TRIAL

Plaintiff, under Rule 38 of the Federal Rules of Civil Procedure, requests a trial by jury of any issues so triable by right.

Dated: February 23, 2018

Respectfully submitted,

REALTIME ADAPTIVE STREAMING LLC,
By its attorneys,

/s/ David S. Godkin

David S. Godkin (BBO#196530)
James E. Kruzer (BBO# 670827)
BIRNBAUM & GODKIN, LLP
280 Summer Street
Boston, MA 02210
Tel: (617) 307-6100
Fax: (617) 307-6101
Email: godkin@birnbaumgodkin.com
Email: kruzer@birnbaumgodkin.com

Attorneys for Realtime Adaptive Streaming LLC

Of Counsel:

Marc A. Fenster (CA SBN 181067)
Reza Mirzaie (CA SBN 246953)
Brian D. Ledahl (CA SBN 186579)
C. Jay Chung (CA SBN 252794)
Philip X. Wang (CA SBN 262239)
Timothy T. Hsieh (CA SBN 255953)
RUSS, AUGUST & KABAT
12424 Wilshire Boulevard, 12th Floor
Los Angeles, California 90025
Telephone: (310) 826-7474
Facsimile: (310) 826-6991
Email: mfenster@raklaw.com
Email: rmirzaie@raklaw.com
Email: bledahl@raklaw.com
Email: jchung@raklaw.com
Email: pwang@raklawcom
Email: thsieh@raklaw.com