DOCKET NO.: 1033300-00287

UNITED STATES PATENT AND TRADEMARK OFFICE

PATENT: 7,116,710

INVENTORS: HUI JIN, AAMOD KHANDEKAR, ROBERT J. MCELIECE

FILED: MAY 18, 2001

ISSUED: OCTOBER 3, 2006

TITLE: SERIAL CONCATENATION OF INTERLEAVED CONVOLUTIONAL CODES FORMING TURBO-LIKE CODES

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Apple Inc.
Petitioner

v.

California Institute of Technology
Patent Owner

Case IPR2017-00211

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 7,116,710 UNDER 35 U.S.C. § 312 AND 37 C.F.R. § 42.104

TABLE OF CONTENTS

I. MANDATORY NOTICES

A. Real Party-in-Interest

Apple Inc. ("Apple" or "Petitioner") and Broadcom Corp. are the real parties-in-interest.

B. Related Matters

U.S. Pat. No. 7,116,710 (the "'710 patent," Ex. 1101) is assigned to the California Institute of Technology ("Caltech" or "Patent Owner.") On May 26, 2016, Caltech sued Apple, Broadcom Corp., and Avago Technologies, Ltd. in the U.S. District Court for the Central District of California, claiming that Apple products compliant with the 802.11n and 802.11ac wireless communication standards infringe the '710 patent (along with three additional patents that claim priority to the '710 patent). On August 15, 2016, Caltech amended its complaint to add a claim of patent infringement against Cypress Semiconductor Corp. *See* Amended Complaint for Patent Infringement, *California Institute of Technology v. Broadcom, Ltd. et al.* (Case 2:16-cv-03714), Docket No. 36. The '710 patent was also asserted by Caltech against Hughes Communications Inc. in *California Institute of Technology v. Hughes Communs., Inc* (Case 2:13-cv-07245), and its claims were challenged in two prior petitions for *inter partes review*, IPR2015-00068 and IPR 2015-00067. Patents claiming priority to the '710 patent were

challenged in IPR2015-00060, IPR2015-00061, IPR-2015-00081, and IPR2015-00059.

C. Counsel

Lead Counsel: Richard Goldenberg (Registration No. 38,895)

Backup Counsel: Brian M. Seeve (Registration No. 71,721)

D. Service Information

Petitioner consents to electronic service.

E-mail: richard.goldenberg@wilmerhale.com

Post and Hand Delivery: WilmerHale, 60 State St., Boston MA 02109

Telephone: 617-526-6548

Fax No. 617-526-5000

II. CERTIFICATION OF GROUNDS FOR STANDING

Petitioner certifies pursuant to Rule 42.104(a) that the patent for which review is sought is available for *inter partes* review and that Petitioner is not barred or estopped from requesting an *inter partes* review challenging the patent claims on the grounds identified in this Petition.

III. OVERVIEW OF CHALLENGE AND RELIEF REQUESTED

Pursuant to Rules 42.22(a)(1) and 42.104(b)(1)-(2), Petitioner challenges claims 1-8, 10-17, and 19-33 of the '710 Patent ("the challenged claims") and requests that each challenged claim be canceled.

A. Prior Art Patents and Printed Publications

Petitioner relies upon the patents and printed publications listed in the Table of Exhibits, including:

1. Frey, B. J. and MacKay, D. J. C., "Irregular Turbocodes," *Proc. 37th Allerton Conf. on Comm., Control and Computing*, Monticello, Illinois, published on or before March 20, 2000 ("Frey"; Ex. 1102.)
2. Frey, B. J. and MacKay, D. J. C., "Irregular Turbo-Like Codes," *37th Allerton Conf. on Comm., Control and Computing*, Monticello, Illinois, published on or before September 24, 1999 ("Frey Slides"; Ex. 1113).
3. D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for "turbo-like" codes," *Proc. 36th Allerton Conf. on Comm., Control and Computing*, Allerton, Illinois, pp. 201-10, March, 1999 ("Divsalar"; Ex. 1103.)
4. Luby, M. *et al.*, "Practical Loss-Resilient Codes," *STOC '97*, pp. *150-159*, published in 1997 ("Luby97"; Ex. 1111).
5. Pfister, H. and Siegel, P, "The Serial Concatenation of Rate-1 Codes Through Uniform Random Interleavers," *37th Allerton Conf. on Comm., Control and Computing*, Monticello, Illinois, published on or before September 24, 1999 ("Pfister"; Ex. 1105.)

The ’710 patent claims priority to a provisional application filed on May 18, 2000, and to U.S. application Ser. No. 09/922,852, filed on Aug. 18, 2000. The Petitioner reserves the right to dispute that the challenged claims are entitled to a priority date of May 18, 2000. However, even if the challenged claims are entitled to the benefit of this priority date, the references relied upon in this Petition qualify as prior art under 35 U.S.C. §102.

B. Relief Requested

Petitioner requests that the Patent Trial and Appeal Board cancel the challenged claims because they are unpatentable under 35 U.S.C. §§ 102 and 103 as set forth in this Petition. This conclusion is further supported by the declaration of Professor James Davis, Ph.D. (“Davis Declaration,” Ex. 1106), filed herewith.

IV. OVERVIEW OF THE TECHNOLOGY

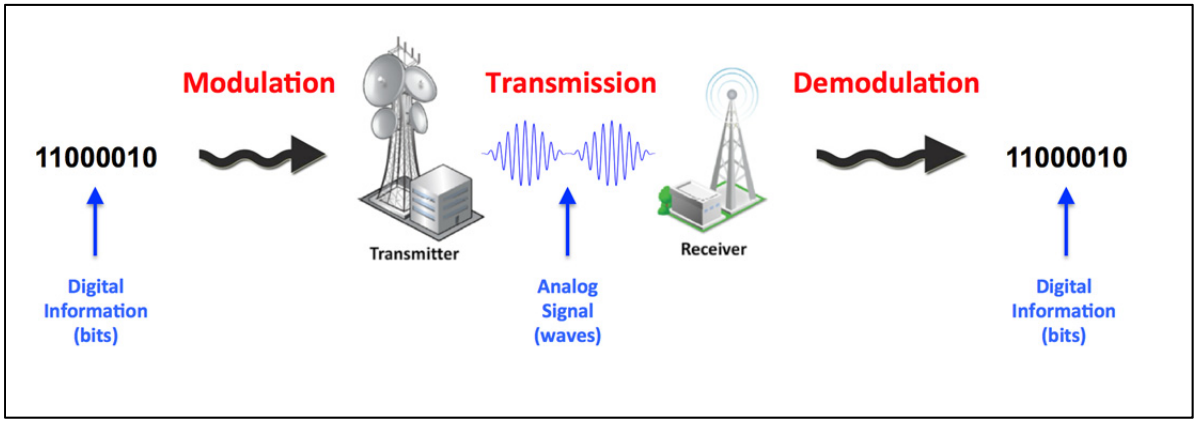
The ’710 patent relates to the field of channel coding and error-correcting codes. This section provides a brief introduction to channel coding and error-correcting codes, and highlights a few of the developments in the field that are relevant to the ’710 patent. (Ex. 1106, ¶21.)

A. Error-Correcting Codes in General

Most computing devices and other digital electronics use bits to represent information. A bit is a binary unit of information that may have one of two values:

1 or 0. Any type of information, including, *e.g.*, text, music, images and video information, can be represented digitally as a collection of bits. (Ex. 1106, ¶22.)

When transmitting binary information over an analog communication channel, the data bits representing the information to be communicated (also called "information bits") are converted into an analog signal that can be transmitted over the channel. This process is called *modulation*. The transmitted signal is then received by a receiving device and converted back into binary form. This process, in which a received analog waveform is converted into bits, is called *demodulation*. The steps of modulation and demodulation are shown in the figure below:



Modulation, Transmission, and Demodulation

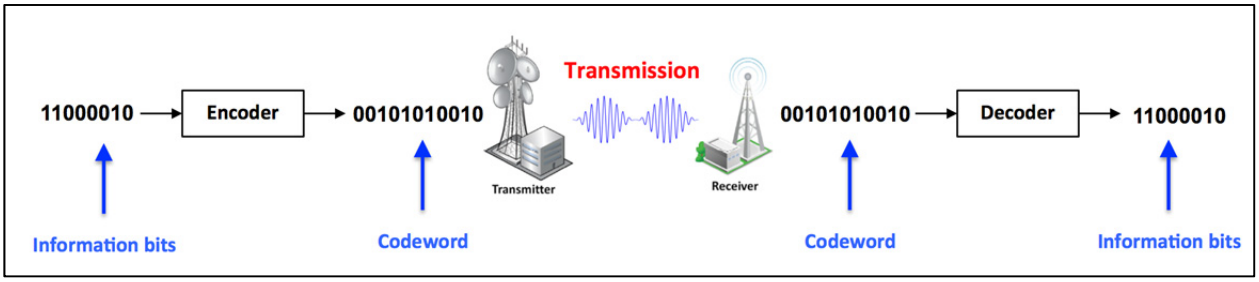
Transmission over physical channels is rarely 100% reliable. The transmitted signal can be corrupted during transmission by "noise" caused by, *e.g.*, obstructions in the signal path, interference from other signals, or electrical/magnetic disturbances. Noise can cause bits to "flip" during

transmission: for example, because of noise, a bit that was transmitted as a 1 can be corrupted during transmission and demodulated as 0, and vice versa. (Ex. 1106, ¶¶23-24.)

Error-correcting codes were developed to mitigate such transmission errors. Using the bits representing the information to be communicated (called "information bits") an error-correcting code generates "parity bits" that allow the receiver to verify that the bits were transmitted correctly, and to correct transmission errors that occurred. (Ex. 1106, ¶25.)

Bits are encoded by an *encoder*, which receives a sequence of information bits as input, generates parity bits based on the information bits according to a particular encoding algorithm, and outputs a sequence of encoded bits (or data elements) called a *codeword*. The codeword produced by the encoder is then modulated and transmitted as an analog signal. (Ex. 1106, ¶26.)

At the receiver, the signal is received and demodulated, and the codeword is passed to the *decoder*, which uses a decoding algorithm to recover the original information bits.



Encoding and Decoding

Error-correcting codes work by adding redundant information to the original message. Due to redundancy, the information represented by a given information bit is spread across multiple bits of the codeword. Thus, even if one of those bits is flipped during transmission, the original information bit can still be recovered from the others. (Ex. 1106, ¶¶27-28.)

As a simple example, consider an encoding scheme, called "repeat-three," that outputs three copies of each information bit. In this scheme, the information bits "1 0 1" would be encoded as "111 000 111." Upon receipt, the decoder converts instances of "111" into "1" and instances of "000" into "0" to produce the decoded bits "1 0 1," which match the original information bits. (Ex. 1106, ¶29.)

Suppose a bit is flipped during transmission, changing "000" to "010." The decoder will be able to detect a transmission error, because "010" is not a valid "repeat-three" codeword. Using a "majority vote" rule, the decoder can infer that the original information bit was a 0, correcting the transmission error. Thus, due to the redundancy incorporated into the codeword, no information was lost due to the transmission error. Error-correcting codes may be either *systematic* or *non-systematic*. In a systematic code, both the parity bits and the original information bits are included in the codeword. In a non-systematic code, the encoded data only includes the parity bits. Systematic and non-systematic codes had been known in

the art for decades prior to May 18, 2000, the claimed priority date of the '710 patent. (Ex. 1106, ¶30.)

B. Coding Rate

Many error-correcting codes encode information bits in groups, or *blocks* of fixed length *n*. An encoder receives a *k*-bit block of information bits as input, and produces a corresponding *n*-bit codeword. The ratio *k/n* is called the *rate* of the code. Because the codeword generally includes redundant information, *n* is generally greater than *k*, and the rate *k/n* of an error-correcting code is generally less than one. (Ex. 1106, ¶33.)

C. Performance of Error-Correcting Codes

The effectiveness of an error-correcting code may be measured using a variety of metrics.

One tool used to assess the performance of a code is its *bit-error rate* (BER.) The BER is defined as the number of corrupted information bits divided by the total number of information bits during a particular time interval. For example, if a decoder outputs one thousand bits in a given time period, and ten of those bits are corrupted (*i.e.*, they differ from the information bits originally received by the encoder), then the BER of the code during that time period is (10 bit errors) / (1000 total bits) = 0.01 or 1%. (Ex. 1106, ¶35.)

The BER of a transmission depends on the amount of noise that is present in the communication channel and the strength of the transmitted signal (*i.e.,* the power that is used to transmit the modulated waveform). An increase in noise tends to increase the error rate and an increase in signal strength tends to decrease the error rate. The ratio of the signal strength to the noise, called the "signal-to-noise ratio," is often used to characterize the channel over which the encoded signal is transmitted. The signal-to-noise ratio can be expressed mathematically as E_b/N_0, in which E_b is the amount of energy used to transmit each bit of the signal, and N_0 is the density of the noise on the channel. The BER of an error-correcting code is often measured for multiple values of E_b/N_0 to determine how the code performs under various channel conditions. (Ex. 1106, ¶36.)

D. LDPC Codes, Turbocodes, and Repeat-Accumulate Codes

In 1963, Robert Gallager described a set of error correcting codes called Low Density Parity Check ("LDPC") codes. Gallager described how LDPC codes provide one method of generating parity bits from information bits using a matrix populated with mostly 0s and relatively few 1s, as explained in more detail below. (*See* Gallager, R., *Low-Density Parity-Check Codes*, Monograph, M.I.T. Press, 1963; Ex. 1107.) (Ex. 1106, ¶38.)

Gallager's work was largely ignored over the following decades, as researchers continued to discover other algorithms for calculating parity bits. These

algorithms included, for example, convolutional encoding with Viterbi decoding and cyclic code encoding with bounded distance decoding. In many cases, these new codes could be decoded using low-complexity decoding algorithms. (Ex. 1106, ¶39.)

In 1993, researchers discovered "turbocodes," a class of error-correcting codes capable of transmitting information at a rate close to the so-called "Shannon Limit" – the maximum rate at which information can be transmitted over a channel. As explained further in the Davis Declaration, the main drawback of convolutional codes is that they do not perform well over channels in which errors are clustered tightly together. Turbocodes overcome this deficiency by encoding the input bits twice. The input bits are fed to a first convolutional encoder in their normal order to produce a first set of parity bits. The same input bits are also reordered by an interleaver (*i.e.*, a component that shuffles the input bits and outputs them in a different order) and then encoded by a second convolutional encoder to produce a second set of parity bits. The two sets of parity bits together with the information bits form a single codeword. Using a turbocode, a small number of errors will not result in loss of information unless the errors happen to fall close together in both the original data stream and in the permuted data stream, which is unlikely. (Ex. 1106, ¶¶40-41.)

In 1995, David J. C. MacKay rediscovered Gallager's work from 1963 relating to low-density parity-check (LDPC) codes, and demonstrated that they have performance comparable to that of turbocodes. *See* MacKay, D. J. C, and Neal, R. M. "Near Shannon Limit Performance of Low Density Parity Check Codes," *Electronics Letters*, vol. 32, pp. 1645-46, 1996 (Ex. 1109.) This rediscovery was met with wide acclaim. Turbocodes and LDPC codes have some common characteristics: both codes use permutations to spread out redundancy, and both use iterative decoding algorithms. (Ex. 1106, ¶42.)

In 1995 and 1996, researchers began to explore "concatenated" convolutional codes. *See* Benedetto, S. et al., *Serial Concatenation of Block and Convolutional Codes*, 32.10 Electronics Letters 887-8, 1996 (Ex. 1110.) While turbocodes use two convolutional coders connected in parallel, concatenated convolutional codes use two convolutional coders connected in series: the information bits are encoded by a first encoder, the output of the first encoder is interleaved, and the interleaved sequence is encoded by a second, convolutional code. In such codes, the first and second encoders are often called the "outer coder" and the "inner coder," respectively. (Ex. 1106, ¶43.)

In 1998, researchers developed "repeat-accumulate," or "RA codes," by simplifying the principles underlying turbocodes. (*See* Ex. 1103.) In RA codes, the information bits are first passed to a repeater that repeats (*i.e.*, duplicates) the

information bits and outputs a stream of repeated bits (the encoder described above in the context of the "repeat three" coding scheme is one example of a repeater). The repeated bits are then optionally reordered by an interleaver, and are then passed to an accumulator where they are "accumulated" to form the parity bits. (Ex. 1106, ¶44.)

Accumulation is a running sum process whereby each input bit is added to the previous input bits to produce a sequence of running sums, each of which represents the sum of all input bits yet received. More formally, if an accumulator receives a sequence of input bits i_1, i_2, i_3, … i_n, it will produce output bits o_1, o_2, o_3, … o_n, such that:

$$\begin{aligned} o_1 &= i_1 \\ o_2 &= i_1 \oplus i_2 \\ o_3 &= i_1 \oplus i_2 \oplus i_3 \\ &\vdots \\ o_n &= i_1 \oplus i_2 \oplus i_3 \oplus \cdots \oplus i_n \end{aligned}$$

Where the ⊕ symbol denotes modulo-2 addition.[1] Accumulators can be implemented simply, allowing accumulate codes to be encoded quickly and cheaply. (Ex. 1106, ¶45.)

E. Mathematical Representations of Error-Correcting Codes

1. *Linear Transformations*

Coding theorists often think of error-correcting codes in linear-algebraic terms. For instance, a *k*-bit block of information bits is a *k*-dimensional vector of bits, and an *n*-bit codeword is an *n*-dimensional vector of bits (where an "*n* dimensional vector" of bits is a sequence of *n* bits). The encoding process, which converts blocks of information bits into codewords, is a linear transformation that maps *k*-dimensional bit vectors to *n*-dimensional bit vectors. This transformation is represented by an $n \times k$ matrix **G** called a *generator matrix*. For a vector of information bits $\boldsymbol{u}$, the corresponding codeword $\boldsymbol{x}$ is given by:

[1] Modulo-2 addition (also called "XOR") is an addition operation that is performed on bits, defined as follows: 1⊕1=0, 1⊕0=1, 0⊕1=1, and 0⊕0=0.

$$\boldsymbol{x} = \mathbf{G}\boldsymbol{u} = \mathbf{G} \cdot \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_k \end{bmatrix} = \begin{bmatrix} \sum_i^k g_{1,i} u_i \\ \sum_i^k g_{2,i} u_i \\ \vdots \\ \sum_i^k g_{n,i} u_i \end{bmatrix}$$

The *n*-dimensional vectors that can be written in the form **G*u*** are valid codewords.

Most *n*-dimensional vectors are *not* valid codewords. It is this property that allows a decoder to determine when there has been an error during transmission. This determination is made using an $(n - k) \times n$ matrix **H**, called a *parity check matrix*. Using a parity check matrix, a given vector ***x*** is a valid codeword if and only if $\mathbf{H}\boldsymbol{x} = \mathbf{0}$. As explained in the Davis declaration, if the parity check shows that $\mathbf{H}\boldsymbol{x}$ does not equal **0**, then the decoder knows there has been a transmission error (much like the "0**1**0" example in the "repeat three" code above). (Ex. 1106, ¶XXX.) In practice, parity-check matrices often have hundreds or thousands of rows, each of which represents an equation of the form $x_a + x_b + \ldots + x_z = 0$. These equations are called *parity-check* equations. (Ex. 1106, ¶¶47-50.)

2. *Repeating and Permuting as Examples of LDGM Codes*

As explained above, the encoding process can be represented using a matrix called a *generator matrix*. This is true of all linear codes, including the "repeat," "interleave," and "accumulate" phases of the "repeat-accumulate" codes discussed

above. More specifically, the "repeat" and interleave" phases can be represented using generator matrices whose entries are mostly zeros, with a relatively low proportion of 1s. Such generator matrices are called "low-density generator matrices" (LDGMs). The following is a generator matrix that can be used to repeat each information bit twice:

$$\mathbf{G}_{\mathrm{REPEAT2}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Using the generator matrix above, encoding a sequence of input bits "101010101" would result in the encoded sequence "110011001100110011" (the number of times each input bit is repeated is determined by the number of "1s" appearing in the column corresponding to that input bit). In the 18×9 matrix above, 11.1% of the entries are 1s, and the rest are 0s. The relative scarcity of 1s means that $\mathbf{G}_{\mathrm{REPEAT2}}$ is a low-density generator matrix (LDGM). (Ex. 1106, ¶¶51-52.)

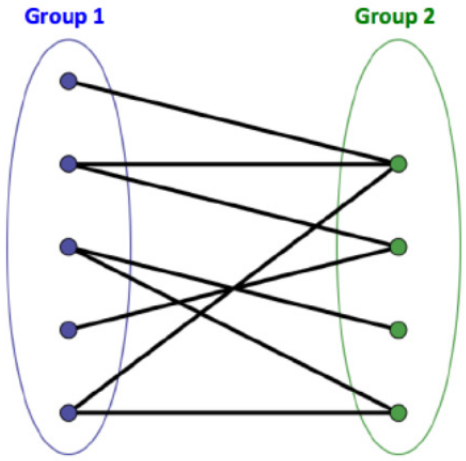
Permutation is another linear transform that is represented by a low-density generator matrix. For example, the matrix below is a permutation matrix that will output bits in a different order than bits are input:

$$\mathbf{G}_{\text{SHUFFLE}} = \begin{bmatrix} 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Permutation matrices, like $\mathbf{G}_{\text{SHUFFLE}}$ above, have exactly one 1 per row and one 1 per column. The order of the output bits is determined by the position of each of these 1s within its row/column. The matrix above, for example, would transform input bits i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8 into the output sequence i_2, i_1, i_7, i_5, i_4, i_8, i_6, i_3. Permutation matrices are square, with dimensions $n \times n$, because the input sequence and the output sequence have the same length. Because there is only one 1 per row/column of permutation matrices, the density of 1s is $1/n$ ($\mathbf{G}_{\text{SHUFFLE}}$, shown above, has a density of 1/8, or 12.5%). In the context of linear codes, which often operate on blocks of more than a thousand bits at a time, a permutation matrix might comprise 0.1% 1s or fewer, and, as further explained in the Davis Declaration, are therefore very low-density generator matrices (LDGMs). (Ex. 1106, ¶¶53-54.)

3. *Tanner Graphs*

Another widespread mathematical representation of error-correcting codes is the "Tanner Graph." Tanner graphs are bipartite graphs wherein nodes can be divided into two groups. Every edge connects a node in one group to a node in the other (*i.e.*, no two nodes in the same group are connected by an edge). A bipartite graph is shown below:



A Simple Bipartite Graph

A Tanner graph includes one group of nodes called *variable nodes* that correspond to the information and parity bits, and a second group of nodes called *check nodes* that represent constraints that parity and information bits must satisfy. In particular, when a set of variable nodes are connected to a particular check node, it means that the information and parity bits corresponding to those variable nodes must sum to 0. (Ex. 1106, ¶¶55-56.)

These two mathematical descriptions of linear codes – one using matrices, one using Tanner graphs – are two different ways of describing the same thing, in

much the same way that "0.5" and "½" are two different ways of describing the same number. Every generator matrix corresponds to a Tanner graph, and vice versa. (Ex. 1106, ¶57.)

F. Irregularity

Irregular LDPC codes were first introduced in a 1997 paper by Luby. *See* Luby, M. *et al.*, "Practical Loss-Resilient Codes," *STOC '97*, 1997 (Ex. 1111). The paper showed that irregular codes perform better than regular codes on certain types of noisy channels. (Ex. 1106, ¶58.)

Regular codes are codes in which each information bit contributes to the same number of parity bits, while *irregular* codes are codes in which different information bits or groups of information bits contribute to different numbers of parity bits. This is consistent with the Board's construction of "irregular" in prior proceedings. (*See*, *e.g.*, IPR2015-00060, Paper 18, p. 12.) Irregularity can also be defined in terms of Tanner graphs. A regular code is one with a Tanner graph in which each variable node corresponding to an information bit is connected to the same number of check nodes. Irregular codes are those with Tanner graphs in which some variable nodes corresponding to information bits are connected to more check nodes than others. These two formulations of irregularity are alternative (and well-known) ways of describing the same concept. (Ex. 1106, ¶59.)

After Luby's initial paper describing irregularity, the same team of researchers published a second paper, expanding the theory of irregularity in error-correcting codes. (*See* Luby, M. *et al*., "Analysis of Low Density Codes and Improved Designs Using Irregular Graphs," *STOC '98*, pp. 249-59, published in 1998; Ex. 1104.) At the time, both of these Luby papers were widely read by coding theorists, and gave rise to a great deal of research into irregular error-correcting codes. (Ex. 1106, ¶60.)

For example, the 1998 Luby paper was the first reference cited in a paper by Frey and MacKay titled "Irregular Turbocodes," presented at the 1999 Allerton Conference on Communications, Control, and Computing. (Ex. 1102.) The second author, MacKay, was the same researcher who had rediscovered LDPC codes in 1995. In this paper, the authors applied the concept of irregularity to turbocodes by explaining how to construct irregular turbocodes in which some information bits connect to more check nodes than others. The experimental results presented in the Frey paper demonstrated that these irregular turbocodes perform better than the regular turbocodes that were known in the art. (*See* Ex. 1102.) (Ex. 1106, ¶61.)

By May 18, 2000, the claimed priority date of the '710 patent, it was common knowledge that the performance of error-correcting code could be improved by adding irregularity. (Ex. 1106, ¶62.)

V. THE '710 PATENT

A. Claims

The '710 patent includes 33 claims, of which claims 1, 11, 15, and 25 are independent. Independent claims 1 and 11 are directed to methods of encoding a signal that include "first encoding" and "second encoding" steps. Independent claim 15 is directed to a "coder" for encoding bits that includes a "first coder" and a "second coder." Claim 25 is directed to a "coding system" that also encodes bits using a first and second coder, and further includes a decoder for decoding the encoded bits. (Ex. 1101, 7:15-8:64.) Claims 1-8, 10-17, and 19-33 are challenged in this Petition. (Ex. 1106, ¶96.)

B. Summary of the Specification

The specification of the '710 patent is generally directed to irregular RA codes (or "IRA" codes). Figure 2 of the specification, reproduced below, shows the structure of an IRA encoder:

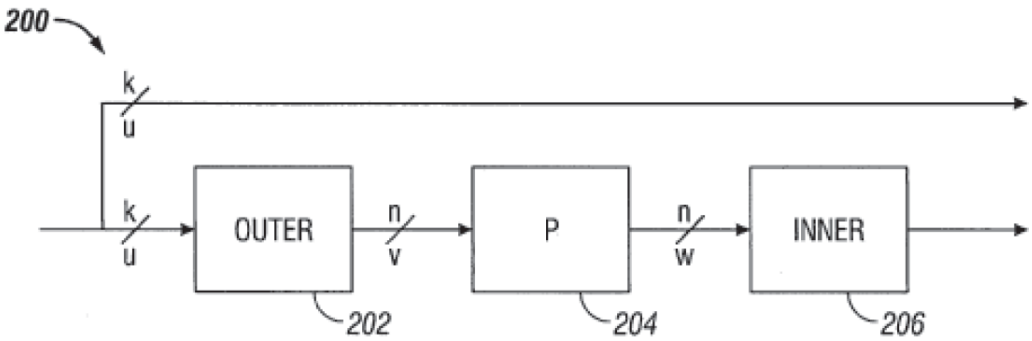


FIG. 2

Explaining this figure, the patent describes encoding data using an outer coder 202 connected to an inner coder 206 via an interleaver 204 (labeled "P") (Ex. 1101, 2:33-40.) (Ex. 1106, ¶¶97-98.)

Outer coder 202 receives a block of information bits and duplicates each of the bits in the block a given number of times, producing a sequence of repeated bits at its output. (Ex. 1101, 2:50-52.) The outer coder repeats bits irregularly – *i.e.*, it outputs more duplicates of some information bits than others. (*Id.*, 2:48-50.) (Ex. 1106, ¶99.)

The repeated bits are passed to an interleaver 204, where they are scrambled. (Ex. 1101, 3:18-22.) The scrambled bits are then passed to the inner coder 206, where they are accumulated to form parity bits. (*Id.*, 2:65-67, 2:33-38.) According to the specification:

> Such an accumulator may be considered a block coder whose input block $[x_1, \ldots, x_n]$ and output block $[y_1, \ldots, y_n]$ are related by the formula
>
> $y_1 = x_1$
>
> $y_2 = x_1 \oplus x_2$
>
> $y_3 = x_1 \oplus x_2 \oplus x_3$
>
> $y_n = x_1 \oplus x_2 \oplus x_3 \oplus \ldots \oplus x_n$.

Ex. 1101, 3:2-10

The specification teaches both systematic and non-systematic codes. In a systematic code, the encoder outputs a copy of the information bits in addition to the parity bits output by inner coder 206 (the systematic output is represented in Fig. 2. as an arrow running toward the right along the top of the figure). (Ex. 1106, ¶100-101.)

C. Level of Ordinary Skill in the Art

A person of ordinary skill in the art is a person with a Ph.D. in mathematics, electrical or computer engineering, or computer science with emphasis in signal processing, communications, or coding, or a master's degree in the above area with at least three years of work experience in this field at the time of the alleged invention. (Ex. 1106, ¶95.)

VI. CLAIM CONSTRUCTION

A claim in *inter partes* review is given the "broadest reasonable construction in light of the specification." 37 C.F.R. § 42.100(b). *See Cuozzo Speed Technologies. LLC v. Lee*, No. 15446, 579 U.S. ____, slip. op. at 13 (U.S. Jun. 20, 2016). Any claim term which lacks a definition in the specification is also given a broad interpretation. *In re ICON Health and Fitness, Inc.*, 496 F.3d 1374, 1379 (Fed. Cir. 2007.)

Should the Patent Owner, seeking to avoid the prior art, contend that the claims warrant a narrower construction, the appropriate course is for the Patent

Owner to seek to amend the claims to correspond expressly to its contentions in this proceeding. (*See* 77 Fed. Reg. 48764, Aug. 14, 2012.)

Consistent with this standard, this section proposes constructions of terms and provides support for these proposed constructions. Terms not included in this section have their broadest reasonable meaning in light of the specification as commonly understood by those of ordinary skill.

A. "close to one" (Claims 1 and 3)

Under the broadest reasonable construction standard, the term "close to one" should be construed to mean "within 50% of one." This construction is consistent with the specification of the '710 patent, which explains that the inner code 210 "can have a rate that is *close to one, e.g., within 50%,* more preferably 10% and perhaps even more preferably within 1% of 1." (Ex. 1101, 2:62-64, emphasis added.) (Ex. 1106, ¶¶102-103.)

This is also consistent with the prosecution history of the '710 patent, in which the Patent Owner attempted to overcome a prior art rejection by replacing the phrase "a rate close to one" with the limitation "a rate within 50% of one," in issued claim 15. (Ex. 1114, pp. 7-8.) This suggests that Patent Owner agrees that the term "close to one" encompasses rates "within 50% of one" (indeed, the fact that this amendment was made in order to narrow the claim suggests that Patent

Owner believes that the broadest reasonable interpretation of "close to one" is even broader). (Ex. 1106, ¶104.)

This is further consistent with the understanding of a person of ordinary skill, who would have understood that the term "close to one," as it is used in the claims of the '710 patent, encompasses rates within 50% of one. (Ex. 1106, ¶105.)

VII. OVERVIEW OF PRIMARY PRIOR ART REFERENCES

A. Frey

"Irregular Turbocodes," by Frey and MacKay ("Frey"; Ex. 1102) was published in the *Proceedings of the 37th Allerton Conference on Communication, Control and Computing*. The conference was held in Monticello, IL, from September 22-24, 1999, and the conference proceedings were published on or before March 20, 2000. (*See* Ex. 1102, showing a library stamp of March 20, 2000.) (Ex. 1106, ¶63.)

Frey applies the concept of irregularity (which was widely known in the coding community following the publication of the two Luby papers in 1997 and 1998, as explained above) to turbocodes. In particular, Frey explains how to construct irregular turbocodes, *i.e.*, turbocodes with Tanner graphs in which some information nodes are connected to more check nodes than others. Figure 2 of Frey, reproduced below, shows the structure of one of these irregular turbocodes:

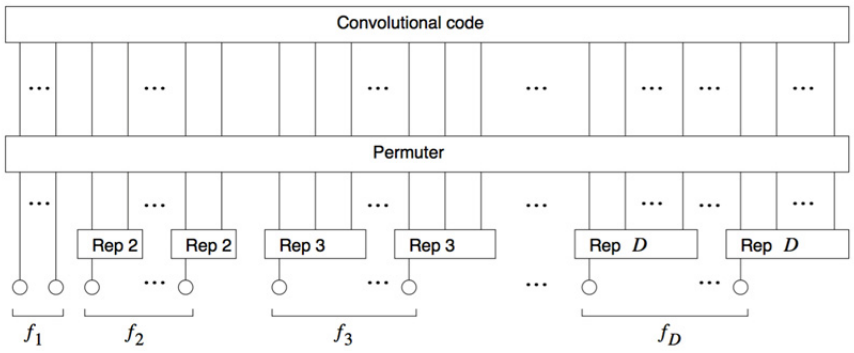


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1102, Figure 2

Applying Luby's teachings, Frey describes codes using a concept called a *degree profile*, which establishes the number of connections between groups of information bits and check nodes in the code's Tanner graph. (See, *e.g.*, Ex. 1102, p. 2.) More formally, a "degree profile" is a sequence of fractions f_d, each representing the proportion of information bits that have degree d. Here, d ranges from 1 to D, where D is the maximum degree. (Ex. 1106, ¶¶64-65.)

In Frey, a bit with "degree d" is a bit that is connected to d check nodes. In practical terms, as Frey explains, this means that "[e]ach codeword bit with degree d is repeated d times before being fed into the permuter." (Ex. 1102, p. 2.) This process, which involves repeating some bits more than others, is identical to the "irregular repetition" described in the '710 patent specification and claims. In the code illustrated in Figure 2, represented above, bits in the subset f_2 are repeated twice, bits in the subset f_3 are repeated three times, bits in the subset f_D are repeated

D times. The code of Figure 2 includes a subset of information bits f_1 that are not repeated at all. As shown in Figure 1, however, Frey also teaches that every bit may be repeated at least once. (Ex. 1106, ¶66.)

In Frey, the irregular repetition stage of encoding is followed by an interleaving stage, in which the irregularly repeated bits are interleaved, or reordered, using a "permuter." Frey's permuter changes the order of the repeated bits just like the "interleaver" described in the '710 patent's specification and claims. (Ex. 1106, ¶67.)

After interleaving, the permuted bits are encoded using a convolutional encoder. (Ex. 1106, ¶68.)

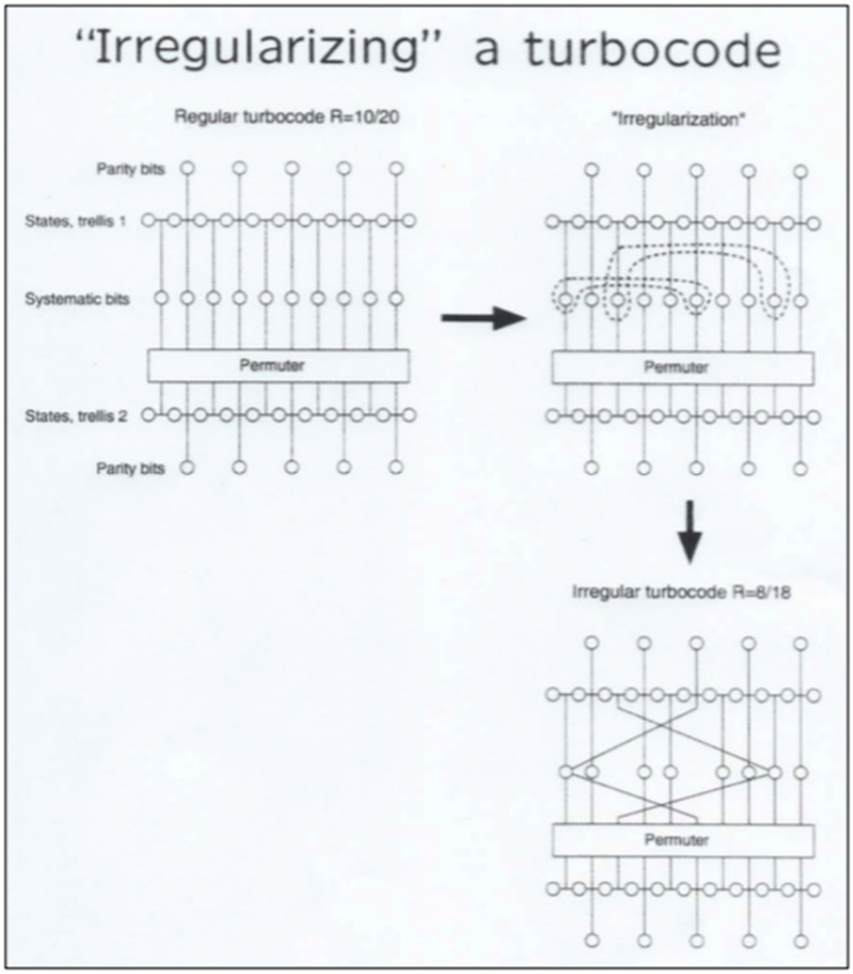
Frey reports on testing his irregular turbocodes by experiment. Sample "information bits" were encoded using irregular turbocodes. The resulting codewords were then sent across a simulated "noisy" channel, which introduced errors into the encoded bits similar to those that occur during transmission. The authors of Frey then decoded the "noisy" encoded bits using an iterative decoder, and compared the resulting decoded bits to the original message bits. In this way, the authors were able to measure the effect of irregularity on the bit-error rate (BER) of various turbocodes. The Frey paper concluded that in certain circumstances, "[t]he irregular turbocode clearly performs better than the regular turbocode." (Ex. 1102, p. 6.) (Ex. 1106, ¶69.)

Subjecting encoded bits to a simulated noisy channel before decoding them was a common way to evaluate the performance of error-correcting codes, and would have been known to a person of ordinary skill. As described below, the Divsalar, Luby and Pfister references each used the same experimental technique. (Ex. 1106, ¶70.)

B. Frey Slides

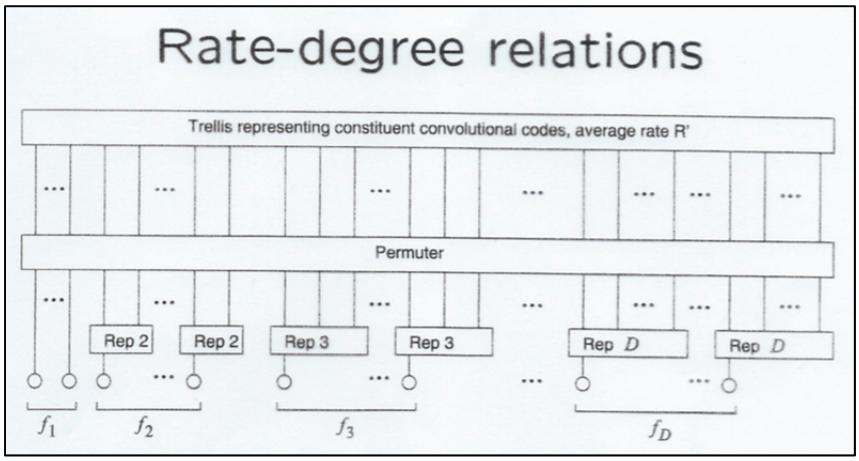
The Frey Slides (Ex. 1113) were presented by Brendan Frey at the Allerton Conference in September, 1999. The Frey Slides contain the material upon which the Frey paper, published in the Allerton 1999 conference proceedings, is based. (*See* Ex. 1117, p. 36). (Ex. 1106, ¶71.)

The Frey Slides describe how irregularity can improve code performance and introduce the concept of irregular turbocodes. Using the same procedure described in the Frey paper, the Frey Slides show how known, regular codes can be "irregularized," step by step:



Ex. 1113, p. 4

Also, using a diagram identical to Figure 2 of the Frey paper (described above), the Frey Slides show how irregular turbocodes can be implemented via irregular repetition:



Frey Slides at 5

The Frey Slides also describe selection of degree profiles (*see id.* at 6) and provide details regarding the rate of the resulting convolutional coder and the overall rate of the irregular turbocode (*id.* at 5-8, 13). (Ex. 1106, ¶74.)

As explained above, the Frey paper was published on March 20, 2000 and is thus prior art to the ’710 patent. However, in the event that the Board finds that the ’710 patent is entitled to a date of invention that predates the publication of the Frey paper, the Frey Slides will nonetheless qualify as prior art to the ’710 patent. Grounds using the Frey Slides, in place of the Frey paper, are accordingly presented below. (Ex. 1106, ¶75.)

C. Divsalar

“Coding Theorems for ‘Turbo-Like’ Codes,” by Divsalar *et al*., was published in March 1999, in the Proceedings of the 36th Allerton Conference on Communication, Control and Computing. (“Divsalar,” Ex. 1103.) As the Board

found in a prior IPR proceeding, Divsalar qualifies as prior art under 35 U.S.C. §102(b) because it was published "well before" the effective filing date of the '710 patent, which is May 18, 2000. (IPR2015-00059, Paper 42, pp. 13-22; *see also* Ex.1112, explaining that Divsalar was available to the public by March 30, 1999.) (Ex. 1106, ¶76.)

Divsalar teaches "repeat and accumulate" codes, which it describes as "a simple class of rate 1/*q* serially concatenated codes where the outer code is a *q*-fold repetition code and the inner code is a rate 1 convolutional code with transfer function 1/(1 + D)." (Ex. 1103, p. 1.) Figure 3 of Divsalar, reproduced below, shows an encoder for a repeat-accumulate code with rate *1/q*:

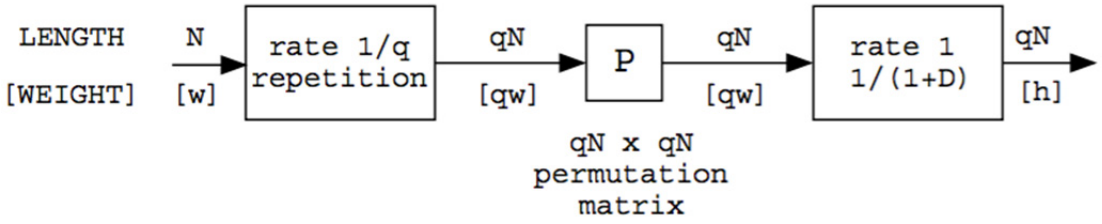


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1103, Figure 3

A block of *N* information bits enters the coder at the left side of the figure and is provided to the repeater (labeled "rate 1/q repetition"). (*See id.*, p. 5.) The repeater duplicates each of the *N* information bits *q* times and outputs the resulting $N \times q$ repeated bits, which are then "scrambled by an interleaver of size *qN*" (*id.*,

referring to the box labeled "P"). The scrambled bits are "then encoded by a rate 1 *accumulator*." (*Id.*, emphasis in original.) (Ex. 1106, ¶¶77-78.)

Divsalar describes the accumulator as follows:

> The accumulator can be viewed as a truncated rate-1 recursive convolutional encoder with transfer function $1/(1+D)$, but we prefer to think of it as a block code whose input block $[x_1, \ldots, x_n]$ and output block $[y_1, \ldots, y_n]$ are related by the formula
>
> $$\begin{aligned} y_1 &= x_1 \\ y_2 &= x_1 + x_2 \\ y_3 &= x_1 + x_2 + x_3 \\ &\vdots \\ y_n &= x_1 + x_2 + x_3 + \cdots + x_n. \end{aligned} \tag{5.1}$$

(Ex. 1103, p. 5.) Divsalar focuses on *binary* linear codes, in which data elements are bits, and thus the + symbols above are understood to denote modulo-2 addition. (Ex. 1103, p. 2, "Consider a binary linear (n,k) block code...".) (Ex. 1106, ¶¶78-79.)

Divsalar explains that RA codes have "very good" performance and that they can be efficiently decoded using a "message passing decoding algorithm." (Ex. 1103, pp. 9-10.) (Ex. 1106, ¶80.)

Further, Divsalar introduces this paper as relevant to "turbo-like" codes, which Divsalar states include classical turbocodes (the codes Frey examines), serially concatenated convolutional codes, and RA codes. "We call these systems 'turbo-like' codes and they include as special cases … the classical turbo codes," "the serial concatenation of interleaved convolutional codes," and "a special class

of turbo-like codes, the repeat-and-accumulate codes." (Ex. 1103, p. 1.) Divsalar also makes use of turbo-like decoding methods in the decoder: "an important feature of turbo-like codes is the availability of a simple iterative, message passing decoding algorithm that approximates ML decoding. We wrote a computer program to implement this 'turbo-like' decoding for RA codes with q = 3 (rate 1/3) and q = 4 (rate 1/4), and the results are shown in Figure 5." (*Id.*, p. 9.) (Ex. 1106, ¶81.)

As explained further below, Divsalar teaches all but one aspect of an IRA code: irregularity (the "**I**" in ***I***rregular ***R***epeat-***A***ccumulate). That is, Divsalar teaches regular repeat-accumulate (RA) codes rather than irregular repeat-accumulate codes as described and claimed in the '710 patent. A single modification to Divsalar – changing the repeat to being irregular instead of regular – results in the irregular codes that Caltech claims to have invented, as would have been obvious to one of skill in view of any of the multiple prior art publications teaching irregularity (including, *e.g.*, Luby, Luby97, and Frey, described below). (Ex. 1106, ¶82.)

D. Luby97

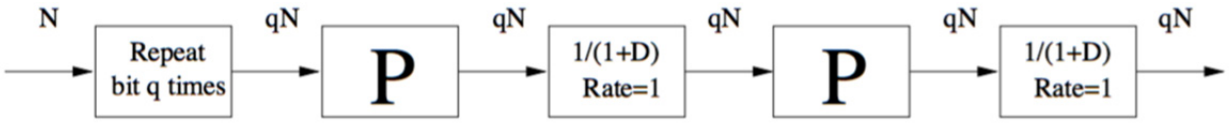
Luby, M. *et al.*, "Practical Loss-Resilient Codes," *STOC '97*, pp. *150-159*, published in 1997 ("Luby97"; Ex. 1111) is the paper in which Luby *et al.* first introduced irregularity. Luby97 is prior art to the '710 patent, as shown by the

library date stamp of June 9, 1998, on the cover of the reference. (Ex. 1111, p. 1.) Luby97 is further cited by a number of academic publications that were published before the claimed priority date of the '710 patent. (Ex. 1122, p. 482; Ex. 1123, p. 2552; Ex. 1124, p. 76.) Luby97 was also cited in the Ph.D. thesis of Aamod Khandekar, one of the inventors named on the face of the '710 patent (Ex. 1118, p. 103). The fact that multiple different authors cited Luby97 before the priority date of the '710 patent, and Luby97 was identified as a "1997" publication by Khandekar, shows that Luby97 was published, and generally available to the community of researchers studying error-correcting codes by the priority date of the '710 patent, and is therefore prior art to the '710 patent under §102. (Ex. 1106, ¶¶512-514.)

In addition to introducing irregularity, Luby97 describes receiving data to be encoded in a *stream* of data symbols (which could be, *e.g.*, bits), where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1111, p. 150, emphasis added.) One of ordinary skill would have understood that in practice, information bits are often received in a real-time *stream*. The encoder waits until it has received a certain number of information bits (*i.e.*, a "block" of information bits), which are then encoded all at once, producing a codeword of fixed size. (Ex. 1106, ¶91.)

E. Pfister

Pfister (Ex. 1105) was presented by Paul Siegel at the Allerton Conference in September 1999. (*See* Ex. 1120, p. 3.) Pfister relates to codes that are constructed as a serial chain of rate-1 encoders and interleavers. (*See* Ex. 1105, pp. 1-2.) Pfister explicitly builds on Divsalar and introduces Divsalar as the starting point for the findings presented therein. (*Id.*, p. 4.) In particular, Pfister discusses a class of codes called "RAA codes" (where "RAA" stands for "Repeat-Accumulate-Accumulate"), in which the outer code comprises a repeater, which is followed by two accumulators with Rate 1. (*Id.*, p. 1.) In particular, Pfister shows an RAA coder that uses two rate-1 accumulators, as shown below:



Ex. 1105, p. 20

In the figure above, the boxes labeled "1/(1+D) Rate=1" represent accumulators that are chained together in series after the repeater (represented by the box labeled "Repeat bit q times"). Thus, the RAA codes of Pfister are simply Divsalar's Repeat-Accumulate codes with an extra accumulator added to the end. Pfister compares the performance of Divsalar's RA codes to RAA codes, and concludes that adding an extra accumulator improves the codes' performance. (*See id.*, p. 21.) (Ex. 1106, ¶¶93-94.)

VIII. GROUNDS FOR CHALLENGE

This Petition, supported by the declaration filed herewith, demonstrates that there is a reasonable likelihood that Petitioner will prevail with respect to at least one challenged claim and that each of the challenged claims is not patentable. *See* 35 U.S.C. § 314(a).

Pursuant to Rule 42.104(b)(4)-(5), specific grounds for finding the challenged claims invalid are identified below and discussed in the Davis Declaration.

A. Ground 1: Claims 1 and 3 Are Anticipated by the Frey Slides

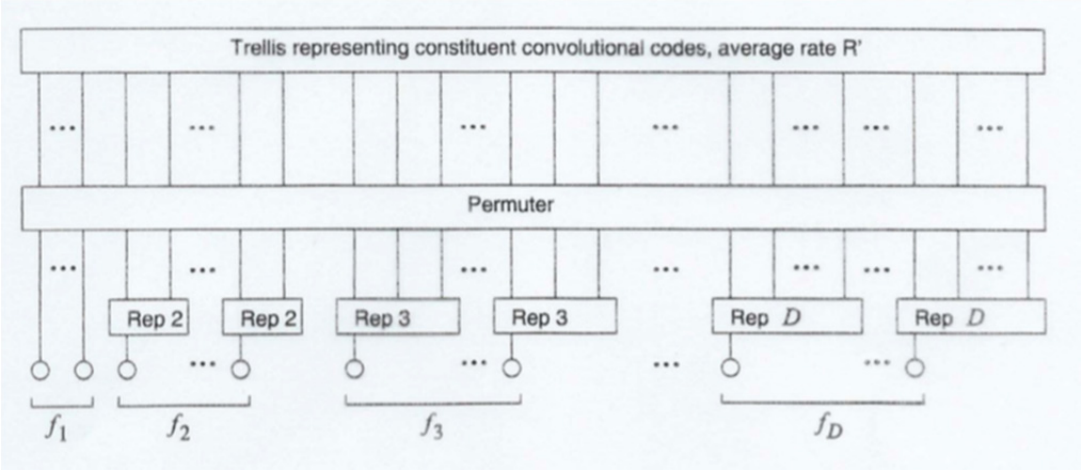
Claims 1 and 3 are anticipated by the Frey Slides. The Frey Slides were not considered by the Patent Office during prosecution of the ’710 patent. Also, as was explained above, grounds using the Frey Slides are not redundant of the grounds that use Frey. In the event the Board concludes that Frey does not qualify as prior art, the Frey Slides will nonetheless qualify as prior art to the ’710 patent and invalidate the claims. (Ex. 1106, ¶259.)

1. *Claim 1*

As described below, the Frey Slides teach every limitation of claim 1 of the ’710 patent.

(a) *"A method of encoding a signal"*

The Frey Slides teach the preamble.[2] As explained above, the Frey Slides deal with the construction of irregular turbocodes, which are used for encoding and decoding signals. In particular, the Frey Slides illustrate the encoding process graphically in a figure that is reproduced below:



Ex. 1113, p. 5

In the figure above, encoding is performed by repeating different groups of bits different numbers of times (the boxes labeled "Rep 2," "Rep 3," etc.), permuting the repeated bits (in the box labeled "Permuter"), and encoding them using a convolutional encoder. (*Id.*) (Ex. 1106, ¶261-262.)

---

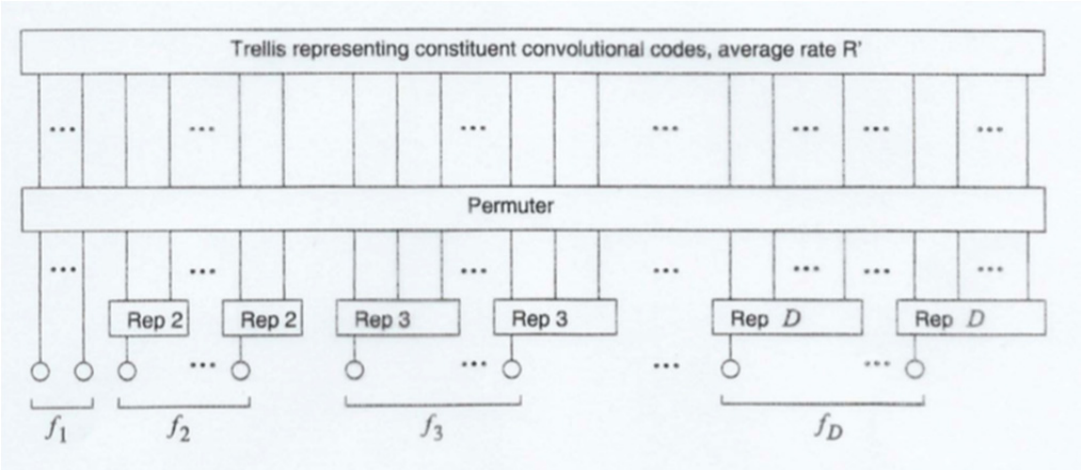
[2] Petitioner retains the right to establish that the preambles of the claims are not limiting, but nonetheless shows herein how the prior art teaches the claims' preambles.

(b) *"obtaining a block of data in the signal to be encoded"*

The Frey Slides disclose this limitation. The encoding methods taught in the Frey Slides operate on *blocks* of data, and the codes disclosed are described with reference to their "block length." For example, the Frey Slides recommend using two different types of decoding methods depending on the "block length" of the code being used, recommending one decoding method "[f]or long *block lengths*" and another "[f]or short *block lengths*." (Ex. 1113, p. 13, emphasis added.) (Ex. 1106, ¶263.)

(c) *"partitioning said data block into a plurality of sub-blocks, each sub-block including a plurality of data elements"*

The Frey Slides teach this limitation. As explained above, the Frey Slides illustrate the encoding process graphically, as shown below:

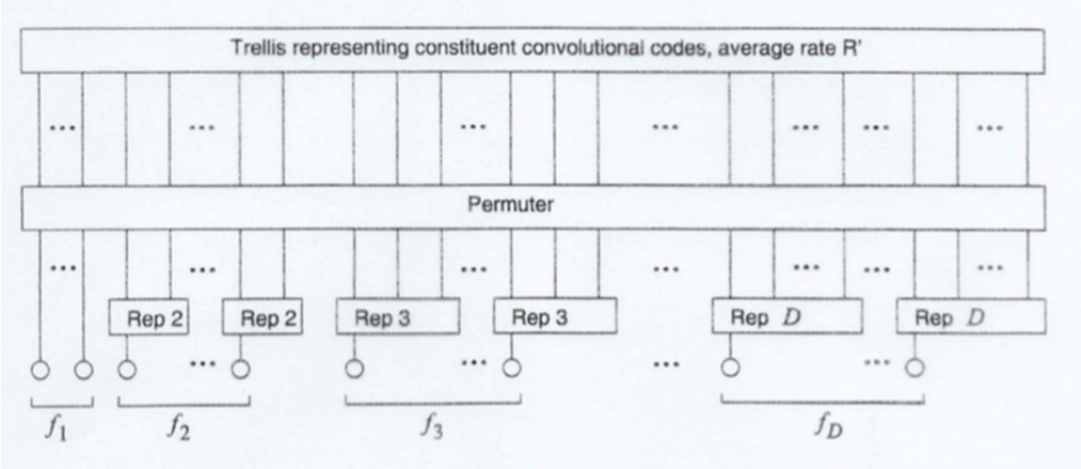


Ex. 1113, p. 5

As shown in the figure above, the Frey Slides teach partitioning the information bits into groups, where the bits in each group have the same degree (*i.e.*, bits within the same group are all repeated the same number of times). In the figure above, the circles at the bottom represent information bits. The groups of information bits labeled $f_2, f_3,\ldots, f_D$ represent sub-blocks into which the data block is partitioned. Thus, the bits that are repeated twice (the bits labeled f_2) constitute one sub-block, the bits that are repeated three times (the bits labeled f_3) constitute a second sub-block, and so on. As shown in the figure above, each of these sub-blocks contains a plurality of bits (or "data elements"), as required by the claim. (Ex. 1106, ¶¶264-264.)

(d) *"first encoding the data block to from [sic] a first encoded data block, said first encoding including repeating the data elements in different sub-blocks a different number of times"*

The Frey Slides teach repeating the data elements in different sub-blocks a different number of times (known to those of ordinary skill as "irregular repetition"). See, for example, the following figure:



Ex. 1113, p. 5

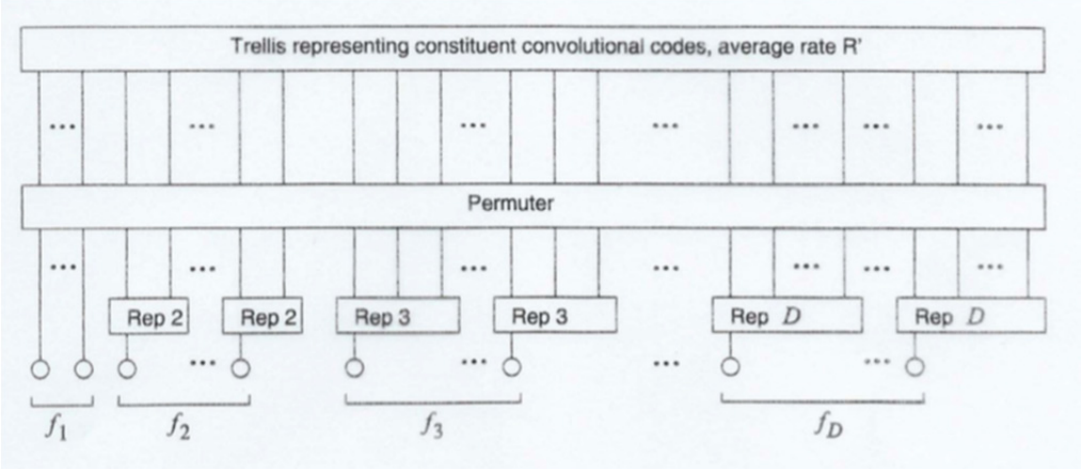
The figure reproduced above shows that the data elements in each sub-block are repeated a different number of times. The circles at the bottom represent information bits in the data block. The groups of information bits labeled $f_2, f_3, \ldots, f_D$ represent sub-blocks into which the data block is partitioned. The blocks labeled "Rep 2," "Rep 3," and "Rep *D*" represent the step of repetition. An information bit that is connected to a box labeled "Rep 2" is repeated twice, a bit connected to a box labeled "Rep 3" is repeated three times, etc. The data elements in sub-block "f_2" are thus repeated a different number of times than the data elements in sub-block "f_3," for example. In Figure 2, the repeated bits are represented by the vertical lines connecting the "Rep" boxes to the box labeled "Permuter." (*See id.*) (Ex. 1106, ¶266.)

The Frey slides also teach the first part of this limitation, "first encoding the data block to [form] a first encoded data block." In the figure above, the

information bits (shown at the bottom as open circles) are encoded, by repeating, to form a "first encoded data block" (shown as vertical lines extending from the repeaters into the permuter), which is then applied to the permuter. (Ex. 1106, ¶267.)

(e) *"interleaving the repeated data elements in the first encoded data block"*

The Frey Slides teach this limitation. The figure below includes a block labeled "Permuter" that performs the claimed interleaving:



Ex. 1113, p. 5

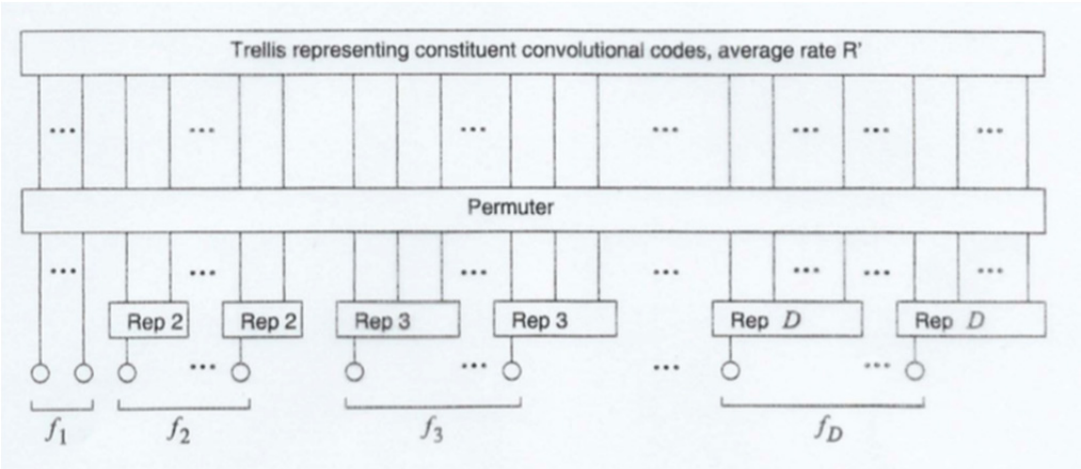
The "Permuter" shown above receives the repeated bits (produced by the blocks labeled "Rep 1," "Rep 2," …, and "Rep *D*") and permutes them. (Ex. 1106, ¶¶268-269.)

One of ordinary skill would have understood that "permuting" means "interleaving," and that a "permuter" is an "interleaver," as Patent Owner has admitted in prior litigation. (*See* Joint Claim Construction Statement, Ex. 1116, pp.

2-3, construing both "interleaver" and "permutation module" to mean "a module that changes the order of data elements.") Divsalar provides further evidence of the general level of knowledge of a person of ordinary skill in the art, showing that such a person would have understood that "permuting" means "interleaving." (Ex. 1103 at p. 2, "…and the ith encoder is preceded by an interleaver (permuter)…" (emphasis added)). Thus, the "Permuter" shown in the figure above satisfies the "interleaving" requirement of claim 1. (Ex. 1106, ¶270.)

(f) *"second encoding said first encoded data block using an encoder that has a rate close to one"*

The Frey Slides teach this limitation. The "second encoding" taught by the Frey Slides is a convolutional encoder, which accepts irregularly repeated and permuted bits as input and encodes these bits to produce parity bits, as shown in the figure below (showing a block labeled "Trellis representing constituent *convolutional codes*, average rate R'," emphasis added):



Ex. 1113, p. 5

Further, the convolutional coders of the Frey Slides have a rate "close to one," as required by the claim. The Frey Slides define the variable $\bar{R}$ as the "[average] rate of convolutional codes" used in the topmost box in the above figure, while R is the "Rate of [the] irregular turbocode" (*Id.*, p. 5.) The relationship between $\bar{R}$ and R is described as follows:

$$1-\bar{R}=\frac{1-R}{(1-R)+2(R-f_e)+d_e f_e}$$

Ex. 1113, p. 6

In the above equation, f_e is the fraction of information bits that has degree d_e (meaning that f_e is the proportion of information bits that are repeated d_e times). (*Id.*). One possible value for R is 1/2. (*Id.*, p. 7.) Good values for f_e and d_e are shown in a graph in the Frey Slides to be 0.03 and 12, respectively. (*Id.*, p. 10.) Plugging the values $f_e = 0.03$, $d_e = 12$, and $R = 1/2$ into the above equation and solving for $\bar{R}$ yields:

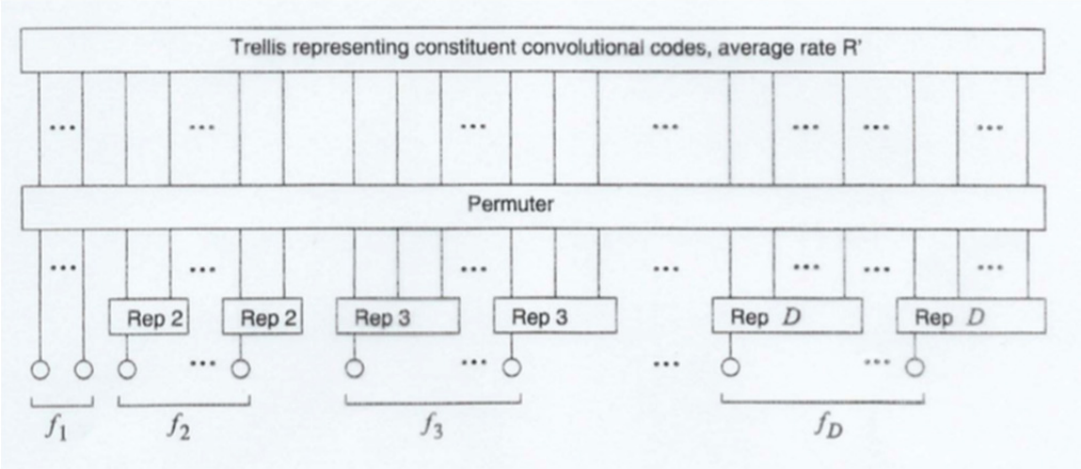
$$\bar{R}=1-\frac{1-R}{(1-R)+2(R-f_e)+d_e f_e}\approx \mathbf{0.72}$$

A rate of 0.72 is within 50% of one, and therefore, as explained above in the claim construction section, one of ordinary skill would recognize that 0.72 is "close to one," as required by the claim. (Ex. 1106, ¶¶271-272.)

2. *Claim 3*

Claim 3 of the '710 patent depends from claim 1, and further recites "wherein said first encoding is carried out by a first coder with a variable rate less than one, and said second encoding is carried out by a second coder with a rate substantially close to one." (Ex. 1101, 7:28-32.)

The Frey Slides teach "a first coder with a variable rate less than one" as recited in claim 3. The first coder is the collection of blocks labeled "Rep2," "Rep 3," to "Rep D" in the figure below:



Ex. 1113, p. 5

The rate of that encoder is a "variable rate" because the number of times bits are repeated varies from 1 to *D*. The rate of the first encoder thus varies within a block between 1 and 1/*D*. Also, in the irregular turbocodes disclosed in the Frey Slides, some bits are duplicated at least once, so the first encoder produces more

output bits than it receives input bits. Thus it is, by definition, an encoder with a "rate less than one," as required by claim 3. (Ex. 1106, ¶¶274-275.)

The Frey Slides also teach a "second coder with a rate substantially close to one" as recited in claim 3. As explained above in the claim construction section, the specification and file history of the '710 patent demonstrate that a rate "close to one" encompasses rates "within 50% of one." (Ex. 1106, ¶276.)

Also, as explained above for claim 1, the Frey Slides teach a convolutional coder with a rate of 0.72. A person of ordinary skill would understand a rate of 0.72 (which is almost *twice* as close to one as 50%) is "substantially close to one," as required by the claim. (Ex. 1106, ¶277.)

B. Ground 2: Claims 1-8 and 11-14 Are Obvious Over Divsalar in View of the Frey Slides

Claims 1-8 and 11-14 are obvious over the combination of Divsalar in view of the Frey Slides. While Divsalar was before the Patent Office during prosecution of the '710 patent, the Frey Slides were not. (Ex. 1106, ¶278.)

1. *Motivation to Combine the Divsalar and the Frey Slides*

The Frey Slides and Divsalar are both directed to the same field (*i.e.*, the field of error-correcting codes), and they are both specifically directed to variations on turbocodes. Specifically, the Frey Slides are directed to introducing irregularity into turbocodes. (*See generally* Ex. 1113, titled "Irregular Turbo-Like Codes.") Similarly, Divsalar, titled "Coding Theorems for 'Turbo-Like' Codes," is related to

a class of codes, called "turbo-like" codes, that have properties similar to those of turbocodes, and teaches that all turbocodes are "turbo-like" codes, as well as some serially concatenated codes. For example, Divsalar states, "We call these systems 'turbo-like' codes and they include as special cases both the classical turbo codes [] and the serial concatenation of interleaved convolutional codes []." (Ex. 1103 at Abstract; *see also id*., Title, p. 2, explaining that the paper "define[s] the class of '*turbo-like*' codes," emphasis added.) (Ex. 1106, ¶279.)

The Frey Slides teach that making a turbocode irregular can lead to codes (*i.e.*, "irregular turbocodes") with better performance than classical turbocodes. (*See* Ex. 1113, p. 11, including a graph showing that irregular codes have lower BERs than regular codes.) To a person of ordinary skill, this would have supplied a clear motivation to apply irregularity to Divsalar's "turbo-like" RA codes by, *e.g.*, combining the irregular repetition of the Frey Slides with the repeat-accumulate codes of Divsalar, resulting in an irregular repeat-accumulate (or "IRA") code, as discussed in more detail below. (Ex. 1106, ¶280.)

The motivation to combine the teachings of the Frey Slides and Divsalar would have been especially strong in view of the similarities between the coding schemes taught by the two references. For example, the Frey Slides teach a convolutional encoder as a second coder (Ex. 1113, p. 5), and Divsalar's second coder is also a convolutional encoder (specifically, a "truncated rate-1 *recursive*

convolutional encoder with transfer function 1/(1 + D.)") (Ex. 1103, p. 5, emphasis added.) Thus, one of ordinary skill would have understood the Frey Slides and Divsalar to disclose coding methods and systems with components (*e.g.*, convolutional encoders) that could be substituted for one another. (Ex. 1106, ¶281.)

Indeed, Frey *himself* suggested in an email to Dariush Divsalar that he make his RA codes irregular, months before the priority date of the patent at issue (*see* Ex. 1117, p. 52, showing an email from Brendan Frey to Dariush Divsalar dated Dec. 8, 1999, explaining that "it would [be] interesting to extend the work that you and Bob [McEliece] have done to the case of irregular turbocodes"). As Dr. Frey explains, "[c]onsistent with the email I sent to Dr. Divsalar, making RA codes irregular was merely an obvious application of my earlier work on irregular turbocodes, which I presented at the Allerton 1999 conference …" (*Id*., p. 40.) (Ex. 1106, ¶¶281-282.)

Incorporating the irregular repetition of the Frey Slides into the RA codes of Divslar would have required only a trivial change to the implementation of the encoder. Irregularity could be introduced into the coding schemes of Divsalar simply by modifying the Divsalar repeater, which repeats every information bit the same number of times, with the repeater of the Frey Slides, which repeats different information bits different numbers of times. This would have been a trivial

modification for a person of ordinary skill to make to an existing RA coder. (Ex. 1106, ¶282.)

Further, one of ordinary skill would have understood that introducing irregularity into the RA codes of Divsalar would yield codes with higher performance. As explained above, the conclusion of the Frey Slides was that introducing irregularity into turbocodes improved their performance. (*See, e.g.*, Ex. 1113, p. 11.) Divsalar teaches a class of "turbo-like" codes that it calls RA codes. Upon reading that the authors of the Frey Slides had improved the performance of turbocodes by introducing irregularity, one of ordinary skill would have found it obvious to try improving RA codes by introducing irregularity. (Ex. 1106, ¶283.)

One of ordinary skill would have been further motivated to incorporate the irregularity of the Frey Slides into the RA codes of Divsalar because, by the priority date of the patent at issue, the benefits of adding irregularity to regular codes had been demonstrated by numerous researchers in the field. As described above, the benefits of irregularity were first explained in Luby's 1997 paper, and expanded in a subsequent paper by Luby and Mitzenmacher in 1998. (Ex. 1104, p. 257; Ex. 1111.) Further benefits of irregularity were documented by Richardson and Urbanke. (Ex. 1119, p. 38.) Based on these repeated demonstrations of the benefits of irregularity, one of ordinary skill would have been motivated to incorporate the Frey Slide's irregular repetition into Divsalar's repeat-accumulate

codes. (Ex. 1106, ¶284.) As explained in detail below, the similarity and combinability of the Frey Slides and Divsalar are further evidenced by the claim limitations they both teach.

2. *Claim 1*

As explained above, the Frey Slides teach every limitation recited by claim 1. Divsalar teaches every limitation of claim 1, except for the irregularity of the "first encoding" step, which the Frey Slides explicitly teach. (Ex. 1106, ¶286.)

(a) *"A method of encoding a signal"*

The preamble is taught by Divsalar. Divsalar describes a "turbo-like" code called a repeat-accumulate code. The purpose of the disclosed repeat-accumulate code is for the encoding and decoding of signals. In particular, Figure 3 of Divsalar, reproduced below, shows an "encoder" for a "repeat and accumulate code":

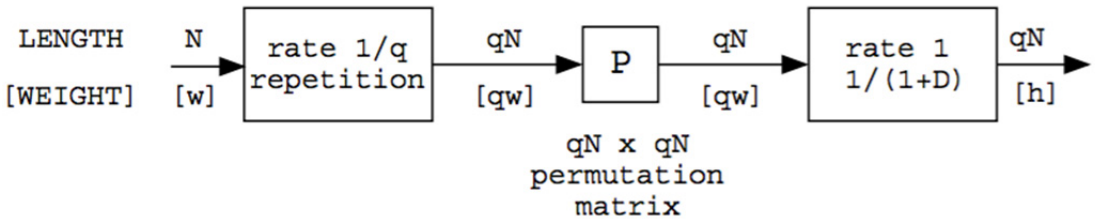


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1103, Figure 3

Divsalar explains that the encoder shown above performs the following encoding method: "An information block of length *N* is repeated *q* times, scrambled by an interleaver of size *qN*, and then encoded by a rate 1 *accumulator*." (*Id.*, p. 5, emphasis in original.) (Ex. 1106, ¶¶287-288.)

(b) *"obtaining a block of data in the signal to be encoded"*

Divsalar teaches block codes. The repeat-accumulate codes introduced by Divsalar are encoded by receiving an "input block" or "information block of length *N*" and passing the block to the repeater. (Ex. 1103, p. 5.) (Ex. 1106, ¶289.)

Figure 3 of Divsalar, reproduced below, shows that each step of the encoding process is associated with a "block" length:

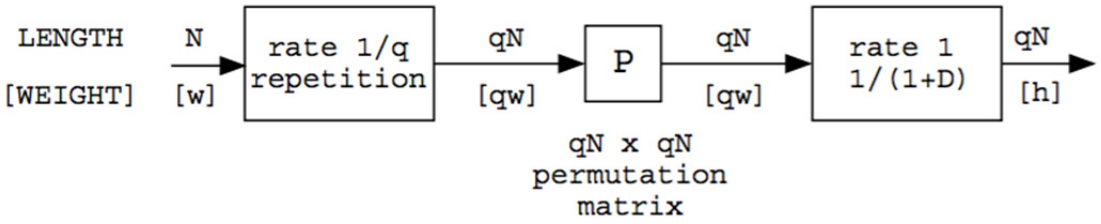


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1103, Figure 3

As the caption of the above figure explains, the numbers above and below the arrows connecting the components of the encoder indicate the length and weight of the corresponding "block." (*Id.*) (Ex. 1106, ¶¶290-291.)

(c) *"partitioning said data block into a plurality of sub-blocks, each sub-block including a plurality of data elements" and "first encoding the data block to from [sic] a first encoded data block, said first encoding including repeating the data elements in different sub-blocks a different number of times"*

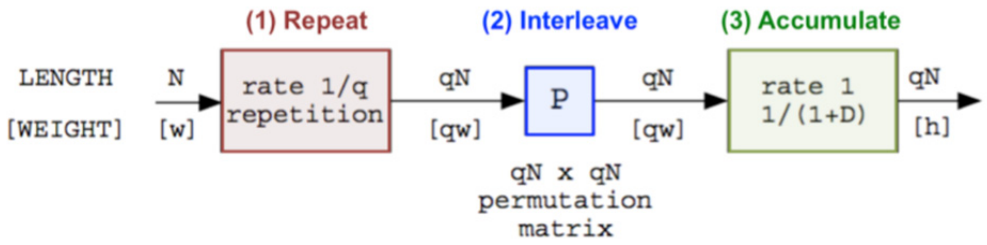
As explained above, Divsalar teaches an RA code that uses three steps:

(1) repeat bits q times;

(2) interleave the repeated bits (with the block labeled "P" in Figure 3); and

(3) accumulate the repeated-interleaved bits with the *rate 1* accumulator.

Each of these steps is represented by a block in Figure 3 (with added highlighting and labels), reproduced below:



Ex. 1103, Figure 3 (annotations added)

A block of N information bits enters the coder at the left side of the figure and is provided to the repeater (labeled "rate 1/q repetition.") (*Id.*, p. 5.) The repeater duplicates each of the *N* information bits *q* times and outputs the resulting $N \times q$ repeated bits. (Ex. 1106, ¶¶292-293.)

While Divsalar does not teach partitioning the data block into a plurality of sub-blocks and repeating information bits in different sub-blocks "a different

number of times" (*i.e.*, irregular repetition), the Frey Slides teach these limitations, as explained above in the anticipation section. Note in particular that an encoding method that meets the "different number of times" limitation of claim 1 necessarily meets the "partitioning" limitation. Any coding scheme that repeats different information bits different numbers of times (such as that taught in the Frey Slides) will *de facto* partition information bits into sub-blocks (*i.e.*, with bits in one sub-block being repeated one number of times and with bits in another sub-block being repeated a different number of times.) As explained in the Davis Declaration, applying different degrees of repetition to different bits is what *defines* the different sub-blocks of the partition. (Ex. 1106, ¶294.)

(d) *"interleaving the repeated data elements in the first encoded data block"*

Divsalar teaches this limitation. Figure 3 of Divsalar, reproduced above, shows a "permutation matrix" (the box labeled "P".) (Ex. 1103, Figure 3.) After the repeater duplicates each of the N information bits q times and outputs $N \times q$ repeated bits, the repeated bits are "scrambled by an interleaver of size qN." (*Id*., p. 5.) (Ex. 1106, ¶295.)

(e) *"second encoding said first encoded data block using an encoder that has a rate close to one"*

Divsalar teaches this limitation. The "second encoding" in Divsalar is the "accumulate" step, shown in Figure 3 of Divsalar, reproduced above. Divsalar

explains the accumulate step as follows: "The accumulator can be viewed as a truncated rate-1 recursive convolutional encoder with transfer function 1/(1 + D), but we prefer to think of it as a block coder" with "input block [x_1, …, x_n] and output block [y_1, …, y_n]." (Ex. 1103, p. 5.) An encoder that outputs n bits (*i.e.*, "output block [y_1, …, y_n]") for every *n* bits of input (*i.e.*, "input block [x_1, …, x_n]) has a rate of *n/n* = 1. Thus, the "second encoder" taught by Divsalar has a rate of 1 (and it is described in Fig. 3 of Divsalar as a "rate 1" encoder.) (*Id.*) Divsalar's accumulator therefore teaches the second encoding limitation of claim 1. (Ex. 1106, ¶¶296-297.)

3. *Claim 2*

Dependent claim 2 depends from independent claim 1, and further requires that the second encoding be performed "via a rate 1 linear transformation." (Ex. 1101, 7:26-7.) As explained below, the accumulator of Divsalar meets this additional requirement. (Ex. 1106, ¶¶298-299.)

Divsalar teaches that the second encoding is performed by a "rate 1 accumulator." (Ex. 1103, p. 5.) As the Davis Declaration explains in more detail, because it transforms bits in a way that can be represented using a matrix, the "rate 1 accumulator" taught by Divsalar performs a "rate 1 linear transformation," as claim 2 requires. (Ex. 1106, ¶¶300-302.)

4. *Claim 3*

Claim 3 depends from claim 1, and further recites "wherein said first encoding is carried out by a first coder with a variable rate less than one, and said second encoding is carried out by a second coder with a rate substantially close to one." (Ex. 1101, 7:28-31.) As explained above in the anticipation ground, the Frey Slides teach the additional limitation imposed by claim 3. Also, as was explained above for claim 2, Divsalar's second encoder has a rate equal to one. While the rate of Divsalar's first encoder is not variable, it does have a rate less than one, as do all repeaters. It would have been obvious to incorporate the irregularity of the Frey Slides into Divsalar, thus making the rate of the first encoder variable. (Ex. 1106, ¶303.)

5. *Claim 4*

Claim 4 depends from claim 3, and further requires that "the second coder comprise[] an accumulator." (Ex. 1101, 7:32-3.) Divsalar teaches this additional limitation. As explained above in the context of claim 2, Divsalar teaches a second coder that is an accumulator. (Ex. 1106, ¶304.)

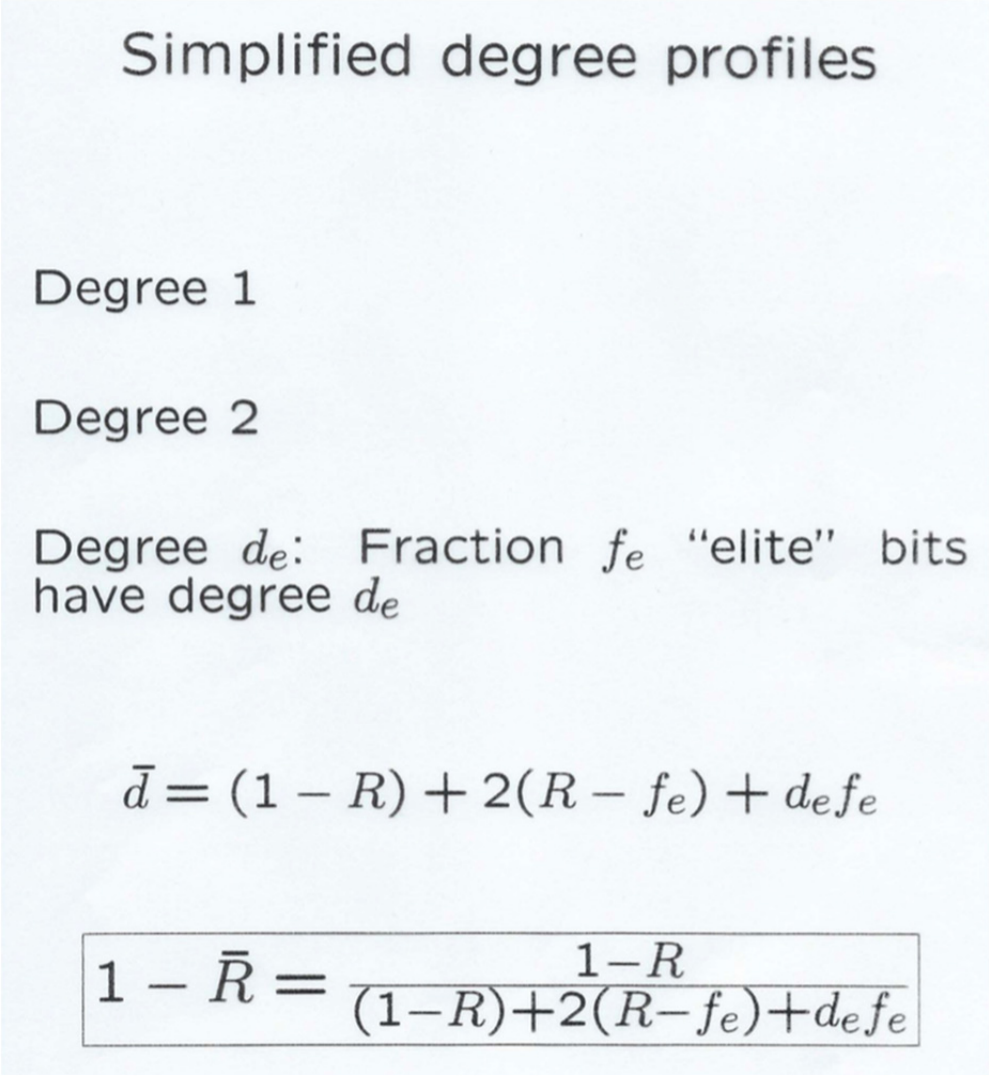
6. *Claim 5*

Claim 5 depends from claim 4, and further requires that "the data elements comprise[] bits." (Ex. 1101, 7:34-5.) Both the Frey Slides and Divsalar teach methods of encoding signals in which the "data elements" comprise bits. For

example, the Frey Slides refer to the "*bits* in a transmitted codeword" (Ex. 1113, p. 2, emphasis added; *see also id.*, p. 4, referring to "systematic bits," which a person of ordinary skill would understand refers to the information bits to be encoded.) Similarly, Divsalar teaches a "*binary* linear (*n, k*) block code." (Ex. 1103, p. 2, emphasis added.) One of ordinary skill would understand Divsalar's "binary" block code to be a code in which the input data elements are "binary digits" or "bits." (Ex. 1106, ¶305.)

7. *Claim 6*

Claim 6 depends from claim 5, and further requires that "the first coder comprise[] a repeater operable to repeat different sub-blocks a different number of times in response to a selected degree profile." (Ex. 1101, 7:37-40.) The Frey Slides teach the use of a degree profile that defines how many times each sub-block is repeated, as shown below:



Ex. 1113, p. 6

Here, the degree profile specifies what fraction f_e of information bits has degree d_e (meaning that f_e is the proportion of information bits that are repeated d_e times). (*Id.*) (Ex. 1106, ¶¶306-309.)

This is consistent with the description of "degree profile" given in the '710 patent specification, which explains that "a fraction of the bits in the block may be repeated two times, a fraction of bits may be repeated three times, and the

remainder of bits may be repeated four times. *These fractions define a degree sequence, or degree profile, of the code.*" (Ex. 1101, 2:55-58, emphasis added.) (Ex. 1106, ¶310.)

8. *Claim 7*

Claim 7 depends from claim 4, and further requires that the first coder comprise "a low-density generator matrix coder." (Ex. 1101, 7:41-3.) Both the Frey Slides and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder. (Ex. 1106, ¶¶311-312.)

As explained above in the background section, a generator matrix is a mathematical representation of an encoder that represents how information bits are transformed into encoded bits. A generator matrix is a two-dimensional array of 1s and 0s. A "low-density" generator matrix is a matrix with a relatively small number of 1s compared to the number of 0s. (Ex. 1106, ¶314.)

As explained in the background section above and in the Davis Declaration, the generator matrix associated with a "repeat" encoder (whether regular, as taught by Divsalar, or irregular, as taught by the Frey Slides) is a low-density generator matrix. The first encoding step taught by both the Frey Slides and Divsalar is implemented using a repeater, and therefore both the Frey Slides and Divsalar teach first encoders that "comprise a low-density generator matrix coder" that implements a low-density generator matrix transformation. Also, because they

have only one "1" per row, a generator matrix for a repeat code has as low a density as any practical code. Therefore, both the Frey Slides and Divsalar teach the additional requirement imposed by claim 7. (Ex. 1106, ¶¶315-316.)

9. *Claim 8*

Claim 8 depends from claim 1, and further requires that "the second encoding use[] a transfer function of 1/(l+D.)" (Ex. 1101, 7:43-4.) Divsalar teaches this additional limitation. As explained above in the context of claim 2, the second encoder of Divsalar is an accumulator. (*See* Ex. 1103, p. 5.) Divsalar further explains that "[t]he accumulator can be viewed as a truncated rate-1 recursive convolutional encoder *with transfer function 1/(1+D).*" (*Id.*, emphasis added.) (Ex. 1106, ¶¶317-319.)

10. *Claim 11*

The combination of Divsalar and the Frey Slides teaches every limitation of independent claim 11.

(a) *"A method of encoding a signal"*

Both the Frey Slides and Divsalar teach the preamble, as explained above in the context of claim 1. (Ex. 1106, ¶321.)

(b) *"receiving a block of data in the signal to be encoded, the data block including a plurality of bits"*

Both the Frey Slides and Divsalar teach receiving a block of data to be encoded, as explained above in connection with claim 1. The Frey Slides teach

further that the data block includes "a plurality of bits." (*See, e.g.*, Ex. 1113, p. 11, showing an input block size K of "8920".) Divsalar also teaches that the block of data to be encoded includes a plurality of bits. (Ex. 1103, p. 2. Referring to "the signal-to-noise ratio *per bit*," emphasis added.) (Ex. 1106, ¶322.)

(c) *"first encoding the data block such that each bit in the data block is repeated and two or more of said plurality of bits are repeated a different number of times in order to form a first encoded data block"*

As explained above with reference to claim 1, the Frey Slides teach irregularly repeating the bits in the data block in order to form a first encoded data block, with some bits being repeated more than others. Thus, the Frey Slides teach systems in which "two or more of said plurality of bits are repeated a different number of times," as the claim requires. (Ex. 1106, ¶323.)

Further, as explained above with reference to claim 1, Divsalar teaches codes in which each bit in the data block is repeated. (*See* Ex. 1103, p. 5, "An information block of length *N* is repeated *q* times ….") It would have been obvious to incorporate the irregularity of the Frey Slides into Divsalar, thus making Divsalar's repetition irregular. (Ex. 1106, ¶324.)

(d) *"second encoding the first encoded data block in such a way that bits in the first encoded data block are accumulated"*

As explained above in connection with claim 1, Divsalar teaches "second encoding the first encoded data block" using an accumulator. (Ex. 1103, Fig. 3.) (Ex. 1106, ¶325.)

11. *Claim 12*

Claim 12 depends from claim 11, and further requires that the second encoding be performed "via a rate 1 linear transformation." As explained above for claim 2, the accumulator of Divsalar meets this additional requirement. (Ex. 1103, p. 5.) (Ex. 1106, ¶¶326-327.)

12. *Claim 13*

Claim 13 depends from independent claim 11, and further requires that the first coding be "via a low-density generator matrix transformation." (Ex. 1101, 7:60-1.) As explained above in the context of claim 7, both the Frey Slides and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder. (Ex. 1106, ¶¶328-329.)

13. *Claim 14*

Claim 14 depends from independent claim 11, and further requires that "the signal to be encoded comprise[] a plurality of data blocks of fixed size." (Ex. 1101, 7:62-3.) This additional limitation is taught by Divsalar. For example, Figure 3 of

Divsalar, reproduced below, shows that each step of the encoding process is associated with a "block" length:

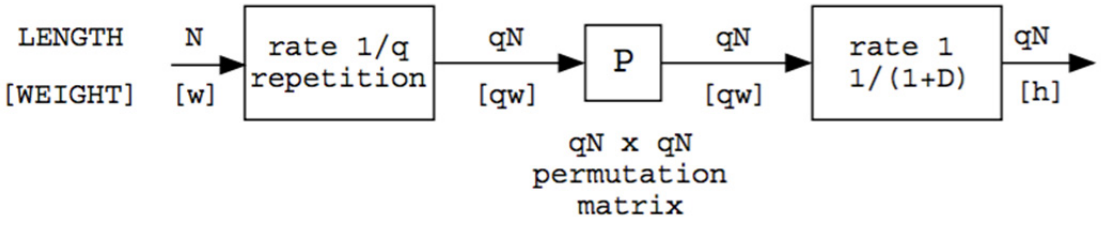


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1103, Figure 3

As this figure shows, the signal to be encoded, represented by the leftmost arrow, is associated with a fixed block length of *N*, satisfying the limitations of claim 14. (Ex. 1103, Fig. 3.) (Ex. 1106, ¶¶330-331.)

C. Ground 3: Claims 15-17, 19-22, and 24-33 Are Obvious Over Divsalar in View of the Frey Slides, and Further in View of Luby97

Claims 15-17, 19-22, and 24-33 are obvious over the combination of Divsalar, the Frey Slides, and Luby97. While Divsalar was before the Patent Office during prosecution of the '710 patent, Frey and Luby97 were not. (Ex. 1106, ¶184.)

1. *Motivations to combine Divsalar and the Frey Slides with Luby97*

As explained above, one of ordinary skill would have been motivated to incorporate the Frey Slides' teaching of irregularity into Divsalar. One of ordinary skill would also have been motivated to combine Divsalar with Luby97, as both relate to error-correcting codes and the performance thereof. (*See generally* Ex. 1103, Ex. 1111.) In particular, a person of ordinary skill would have been motivated to modify the encoder of Divsalar, using the teachings of Luby97, to receive a "stream" of bits, where the "stream" of bits comprises one or more blocks that are encoded separately. (Ex. 1106, ¶332.)

As explained above, Luby97 describes receiving data to be encoded in a *stream* of data symbols (which could be, *e.g.*, bits), where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1111, p. 150, emphasis added.) One of ordinary skill would have understood that in practice, information bits are often received in a real-time *stream*. The encoder waits until it has received a certain number of information bits (*i.e.*, a "block" of information bits), which are then encoded all at once, producing a codeword of fixed size. (Ex. 1106, ¶333.)

Coders that receive "streams" of bits that comprise one or more "blocks" of data were common, and would have been familiar to a person of ordinary skill. To the extent that "streams" were not obvious in view of Divsalar and/or the Frey

Slides alone, it would have been obvious to use Luby97's teaching of "stream" in Divsalar and/or Divsalar in view of the Frey Slides, to make the encoder capable of receiving and processing "streams" as opposed to just blocks. Thus, it would have been obvious to a person of ordinary skill to incorporate the "stream" teachings of Luby97 above into the encoder of Divsalar. (Ex. 1106, ¶334.)

2. *Claim 15*

The combination of Divsalar, the Frey Slides, and Luby97 teaches every limitation in independent claim 15. (Ex. 1106, ¶335.)

(a) *"A coder comprising"*

The preamble is taught by both the Frey Slides and Divsalar, as explained above in the context of claim 1. (Ex. 1106, ¶¶336-339.)

(b) *"a first coder having an input configured to receive a stream of bits, said first coder operative to repeat said stream of bits irregularly and scramble the repeated bits"*

As explained above in the context of claim 1, the combination of Divsalar in view of the Frey Slides teaches all of these limitations except for the possible exception of the "stream of bits." Luby97 teaches this "stream of bits" limitation and, as explained above regarding motivation to combine Luby97 with Divsalar, it would have been obvious to incorporate Luby's teaching of "streams of bits" into Divsalar and Frey. Luby97 teaches a coder configured to receive a "stream" of bits. (*See* Ex. 1111, p. 150). As explained above, one of ordinary skill would understand

that the "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar. (Ex. 1106, ¶¶340-343.)

(c) *"a second coder operative to further encode bits output from the first coder at a rate within 10% of one"*

As explained above in connection with claim 2, Divsalar teaches "a second coder operative to further encode bits output from the first coder at a rate within 10% of one" because Divsalar teaches a second encoder that has a rate equal to one. (Ex. 1106, ¶¶344-348.)

3. *Claim 16*

Claim 16 depends from independent claim 15, and it further requires that "the stream of bits includes a data block, and wherein the first coder is operative to apportion said data block into a plurality of sub-blocks and to repeat bits in each sub-block a number of times, wherein bits in different sub-blocks are repeated a different number of times." (Ex. 1101, 8:7-12.)

As explained above for claims 1 and 6, Divsalar in view of the Frey Slides teaches every limitation of claim 15 except for the possible exception of the "stream of bits includes a data block." Luby97 teaches a coder configured to receive a "stream" of bits, where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1111, p. 150, emphasis added.) As explained above, one of ordinary skill would have been motivated to incorporate

these "stream" teachings of Luby97 into the coders taught by Divsalar. (Ex. 1106, ¶¶350-351.)

4. *Claim 17*

Claim 17 depends from claim 16, and further requires that "the second coder comprise[] a recursive convolutional encoder with a transfer function of 1/(l+D)." (Ex. 1101, 8:13-15.) Divsalar teaches this additional limitation. As explained above for claim 4, the second encoder of Divsalar is an accumulator, and as Divsalar explains, "*[t]he accumulator can be viewed as a truncated rate-1 recursive convolutional encoder with transfer function 1/(1+D.)*" (Ex. 1103, p. 5, emphasis added.) (Ex. 1106, ¶¶352-354.)

5. *Claim 20*

Claim 20 depends from claim 15, and further requires that the first coder comprise "a low-density generator matrix coder." (Ex. 1101, 8:21-2.) As explained above in the context of claim 7, both the Frey Slides and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder. (Ex. 1106, ¶¶355-356.)

6. *Claim 21*

Dependent claim 21 depends from claim 15, and further requires that the second encoder comprise "a rate 1 linear encoder." (Ex. 1101, 8:23-4.) As

explained above in the context of claim 2, Divsalar teaches this additional limitation. (Ex. 1106, ¶¶357-359.)

7. *Claim 22*

Claim 22 depends from claim 21, and further requires that "the second coder comprise[] an accumulator." (Ex. 1101, 8: 25-6.) As explained above for claim 4, Divsalar teaches this additional limitation. (Ex. 1106, ¶¶360-362.)

8. *Claims 24 and 33*

Claims 24 and 33 depend from independent claims 15 and 25, respectively, and further require that the second coder comprise a coder "operative to further encode bits output from the first coder at a rate within 1% of one." As explained above, Divsalar teaches a second coder, (*i.e.*, an accumulator), that has a rate equal to 1 and thus has a rate that is "within 1% of one," as claims 24 and 33 require. (Ex. 1106, ¶363.)

As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches every limitation of claims 15 and 25, from which claims 24 and 33 depend. Thus, for the reasons given above, claims 24 and 33 are obvious over Divsalar, the Frey Slides, and Luby97. (Ex. 1106, ¶364.)

9. *Claim 25*

The combination of Divsalar, in view of the Frey Slides, and further in view of Luby97 teaches every limitation of independent claim 25.

(a) *"A coding system comprising"*

The preamble is taught in both the Frey Slides and Divsalar, as explained above in the context of claims 1, 11, and 15. (Ex. 1106, ¶366.)

(b) *"a first coder having an input configured to receive a stream of bits, said first coder operative to repeat said stream of bits irregularly and scramble the repeated bits"*

As explained above in the context of claim 15, this limitation would have been obvious over Divsalar, the Frey Slides, and Luby97. (Ex. 1106, ¶367.)

(c) *"a second coder operative to further encode bits output from the first coder at a rate within 10% of one in order to form an encoded data stream"*

As explained above in the context of claim 15, Divsalar teaches this limitation. (Ex. 1106, ¶368.)

(d) *"a decoder operative to receive the encoded data stream and decode the encoded data stream using an iterative decoding technique"*

Divsalar teaches this limitation. As Divsalar explains, "an important feature of turbo-like codes is the availability of a simple *iterative*, message passing decoding algorithm that approximates ML decoding. *We wrote a computer program to implement this "turbo-like" decoding for RA codes* … and the results are shown in Figure 5." (Ex. 1103, p. 9, emphasis added; *see also id.*, Figure 5.) One of ordinary skill would understand that the computer program mentioned in this passage is "a decoder operative to receive the encoded data stream and decode

the encoded data stream using an iterative decoding technique," as claim 25 requires. (Ex. 1106, ¶369.)

10. *Claim 26*

Claim 26 depends from claim 25, and further requires that "the first coder comprise[] a repeater operative to receive a data block including a plurality of bits from said stream of bits and to repeat bits in the data block a different number of times according to a selected degree profile." (Ex. 1101, 8:42-46.)

As explained in the background section, Luby97 teaches a coder configured to receive a "stream" of bits, where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1111, p. 150, emphasis added.) As explained above, one of ordinary skill would understand that these "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar. (Ex. 1106, ¶371.)

Further, as explained above in the context of claim 6, the Frey Slides teach "repeat[ing] bits in the data block a different number of times according to a selected degree profile." Claim 26 is therefore obvious over Divsalar, in view of the Frey Slides, and further in view of Luby97. (Ex. 1106, ¶372.)

11. *Claims 19 and 27*

Claims 19 and 27 depend from claims 15 and 26, respectively, and both further require that the first coder comprise "an interleaver." Claim 19 also requires

that the first coder comprise "a repeater having a variable rate." (Ex. 1101, 8:19-48.) Divsalar and the Frey Slides both teach the claimed "interleaver" and the Frey Slides teach "a repeater having a variable rate." (Ex. 1106, ¶373.)

As explained above in the context of claim 3, the Frey Slides teach a first coder having a variable rate. The first coder in the Frey Slides is also a repeater, because it repeats different codeword bits different number of times. (Ex. 1113, p. 5.) Thus, the Frey Slides teach a first coder comprising "a repeater having a variable rate," as claim 19 requires. (Ex. 1106, ¶374.)

The Frey Slides also teach a first coder that comprises an "interleaver," as explained above in the context of claim 1 (*See*, *e.g.*, Ex. 1113, p. 5, reproduced above.) (Ex. 1106, ¶375.)

As explained above with reference to claim 1, the "permuter" shown in the figure is "an interleaver," as claims 19 and 27 require. Moreover, as explained above in the context of claim 1, Divsalar also teaches a first coder comprising "an interleaver." Accordingly, Divsalar and the Frey Slides teach these further limitations of claims 19 and 27. (Ex. 1106, ¶376.)

As explained above, the combination of Divsalar, the Frey Slides, and Luby97 discloses each claim limitation in claims 15 and 26, from which claims 19 and 27 depend. Therefore, claims 19 and 27 are obvious over Divsalar, in view of the Frey Slides, and further in view of Luby97. (Ex. 1106, ¶377.)

12. *Claim 28*

Claim 28 depends from claim 25, and further requires that the first coder comprise "a low-density generator matrix coder." (Ex. 1101, 8:49-50.) As explained above in the context of claim 7, both the Frey Slides and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) coder. (Ex. 1106, ¶¶378-379.)

13. *Claim 29*

Claim 29 depends from claim 25, and further requires that "the second coder comprise[] a rate 1 accumulator." (Ex. 1101, 8:51-2.) As explained above in the context of claims 2, 4, and 22, the accumulator of Divsalar meets all these additional requirements. (Ex. 1106, ¶¶380-381.)

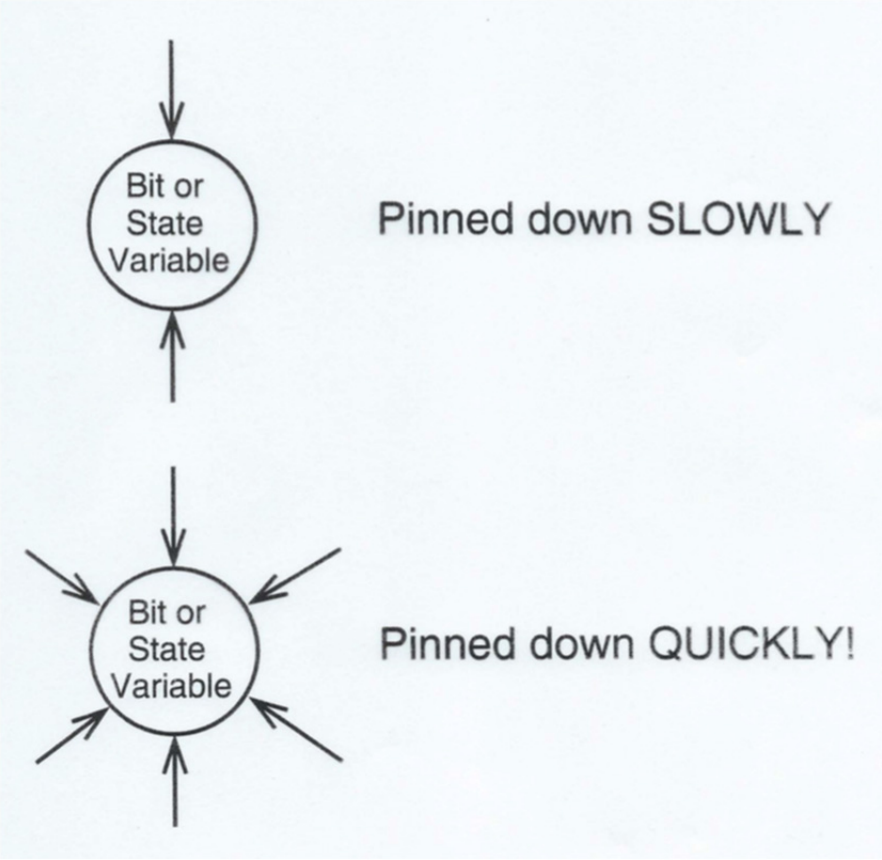
14. *Claim 30*

Claim 30 depends from claim 25, and further requires that "the decoder [be] operative to decode the encoded data stream using a posterior [sic] decoding techniques." (Ex. 1101, 8:53-4.) The Frey Slides disclose this additional limitation. The Frey Slides describe a process for decoding irregular turbocodes using "density evolution." (Ex. 1113, p. 13.) A person of ordinary skill would understand that the "density evolution" algorithm makes use of *a posteriori* decoding techniques, because it determines the value of a given message bit based on the observed values of other message bits. Thus, the decoder in the Frey Slides

decodes the encoded data using "a posterior[i] decoding techniques," as claim 30 requires. (Ex. 1106, ¶¶382-383.)

15. *Claim 31*

Claim 31 depends from claim 25, and further requires that "the decoder [be] operative to decode the encoded data stream based on a Tanner graph representation." (Ex. 1101, 8:56-8.) The Frey Slides teach this limitation. The Frey Slides use a graphical model to explain how irregularity allows some bits to be "pinned down quickly" while other bits are "pinned down slowly":



Ex. 1113, p. 3

A person of ordinary skill would have understood the process of "pinning down" the value of a "bit or state variable," as shown in the figure above, refers to the decoding process. A person of ordinary skill would further understand that the circles and arrows shown in the figure above represent portions of a Tanner graph, in which the "bit or state variable" that is "pinned down quickly" is one that is connected to a large number of additional nodes, while the "bit or state variable" that is "pinned down slowly" is connected to fewer additional nodes. (Ex. 1106, ¶¶384-387.)

16. *Claim 32*

Claim 32 depends from claim 25, and further requires that "the decoder [be] operative to decode the encoded data stream in linear time." (Ex. 1101, 8:59-60.) Luby97 teaches this additional limitation. In particular, Luby97 discloses "randomized constructions of *linear-time* encodable and *decodable* codes." (Ex. 1111, p. 150, emphasis added.) A person of ordinary skill would have been motivated to adopt Luby97's decoding technique, which provides linear-time decoding, into the systems taught by Divsalar and Frey, as decoding in linear time makes the decoding process faster and more efficient. (Ex. 1106, ¶247.)

Divsalar also teaches this limitation. As Divsalar points out, "the complexity of [maximum likelihood] decoding of RA codes, like that of all turbo-like codes, is prohibitively large. But an important feature of turbo-like codes is the availability

of a simple iterative, message passing decoding algorithm that approximates ML decoding." (Ex. 1103, pp. 8-9.) A person of ordinary skill in the art would have understood that the "simple iterative, message passing algorithm" to which Divsalar refers could be a linear-time decoding algorithm. To the extent that Divslar does not explicitly teach a linear-time decoder, it would have been obvious to incorporate Luby97's decoding techniques into Divsalar, which are explicitly taught to be linear time. (Ex. 1106, ¶248.)

17. *Claims 24 and 33*

Claims 24 and 33 depend from claims 15 and 25, respectively, and further require that the second coder comprise a coder "operative to further encode bits output from the first coder at a rate within 1% of one." (Ex. 1101, 8:29-64.) As explained above for claims 2 and 4, Divsalar teaches a second coder (*i.e.*, an accumulator) that has a rate equal to 1, and which is therefore "within 1% of one," as claims 24 and 33 require. (Ex. 1106, ¶249.)

D. Ground 4: Claim 10 Is Obvious in View of Divsalar and the Frey Slides, and Further in View of Pfister

Claim 10 depends from claim 1, and further requires that the second encoding "utilize[] two accumulators." (Ex. 1101, 7:48-9.) While Divsalar and the Frey Slides do not explicitly disclose a second coder comprising two accumulators, Pfister does. As described above, Pfister teaches "RAA codes," (where "RAA"

stands for "Repeat-Accumulate-Accumulate"). (Ex. 1105, p. 1.) (Ex. 1106, ¶¶390-391.)

The RAA codes of Pfister are simply Divsalar's Repeat-Accumulate codes with an extra accumulator added to the end. (Ex. 1105, p. 20) (Ex. 1106, ¶392.)

A person of ordinary skill would have been motivated to incorporate the two-accumulator design of Pfister into Divsalar. Both references are directed to the same field (*i.e.*, the field of error-correcting codes), and both are specifically directed to error-correcting codes that use rate-1 inner coders. (*See, e.g.*, Ex. 1103, Figure 3; *see also* Ex. 1105, pp. 1-2.) (Ex. 1106, ¶393.)

Further, incorporating a second accumulator into Divsalar's RA codes would have been obvious to one of ordinary skill. As explained above, Pfister's RAA codes are merely RA codes with additional accumulators added to the end of the encoding chain in series. (*See* Ex. 1105, Figure 2.) Indeed, Pfister compares the performance of Divsalar's RA codes to RAA codes, and concludes that adding an extra accumulator improves the performance of Divsalar's RA codes, providing an explicit suggestion to combine the extra accumulator of Pfister with the RA codes of Divsalar. (*See id.*, p. 21.) (Ex. 1106, ¶394.)

As explained above, the combination of Divsalar and the Frey Slides discloses every limitation of claim 1, from which claim 10 depends. Claim 10 is

therefore obvious over Divsalar and the Frey Slides, further in view of Pfister. (Ex. 1106, ¶395.)

E. Ground 5: Claim 23 Is Obvious in View of Divsalar, the Frey Slides, and Luby97, and Further in View of Pfister

Claim 23 depends from claim 22, and further requires that the second coder comprise "a second accumulator." (Ex. 1101, 8:27-8.) As described above in the context of claim 10, Pfister teaches this limitation. As further explained above, the combination of Divsalar, the Frey Slides, and Luby97 discloses every limitation of claim 22, from which claim 23 depends. Claim 23 is therefore obvious over Divsalar, the Frey Slides, and Luby97, further in view of Pfister. (Ex. 1106, ¶¶396-397.)

IX. CONCLUSION

Based on the foregoing, it is clear that the challenged claims of the ’710 patent are both anticipated by and obvious in view of the prior art references cited in this Petition. Accordingly, the Petitioner requests institution of an *inter partes* review to cancel those claims.

Respectfully Submitted,

_/Richard Goldenberg/

Richard Goldenberg
Registration No. 38,895

Brian M. Seeve
Registration No. 71,721

TABLE OF EXHIBITS

Exhibit	Description
1101	U.S. Patent 7,116,710
1102	Frey, B. J. and MacKay, D. J. C., "Irregular Turbocodes," Proc. 37th Allerton Conf. on Comm., Control and Computing, Monticello, Illinois, published on or before March 20, 2000
1103	D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for "turbo-like" codes," Proc. 36th Allerton Conf. on Comm., Control and Computing, Allerton, Illinois, pp. 201-10, March, 1999
1104	Luby, M. et al., "Analysis of Low Density Codes and Improved Designs Using Irregular Graphs," STOC '98, pp. 249-59, published in 1998
1105	Pfister et al., "The Serial Concatenation of Rate-1 Codes Through Uniform Random Interleavers" (Exhibit 1 to the Siegel Declaration)
1106	Declaration of Professor James Davis, Ph.D. ("Davis Declaration")
1107	Gallager, R., Low-Density Parity-Check Codes, Monograph, M.I.T. Press, 1963
1108	Berrou et al., "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," ICC '93, Technical Program, Conference Record 1064, Geneva 1993
1109	MacKay, D. J. C, and Neal, R. M. "Near Shannon Limit Performance of Low Density Parity Check Codes," Electronics Letters, vol. 32, pp. 1645-46, 1996
1110	Benedetto, S. et al., Serial Concatenation of Block and Convolutional Codes, 32.10 Electronics Letters 887-8, 1996
1111	Luby, M. et al., "Practical Loss-Resilient Codes," STOC '97, 1997
1112	Declaration of Robin Fradenburgh Concerning the "Proceedings, 36th Allerton Conference on Communications, Control, and Computing" Reference
1113	Frey, B. J. and MacKay, D. J. C., "Irregular Turbo-Like Codes" presented at the 1999 Allerton Conference on Communications, Control, and Computing

1114	Prosecution History of the '710 Patent, Response Dated May 5, 2005
1115	Table of Contents of Proceedings of the 37th Allerton Conference on Communication, Control and Computing from September 22-24, 1999
1116	Joint Claim Construction Statement (Case No. 2:13-cv-07245, Dkt. No. 47)
1117	Expert Report of Dr. Brendan Frey (Case No. 2:13-cv-07245)
1118	Aamod Khandekar, "Graph-based Codes and Iterative Decoding," Thesis submitted June 10, 2002
1119	Richardson, Shokrollahi, and Urbanke, "Design of Provably Good Low-Density Parity Check Codes"
1120	Declaration of Paul H. Siegel
1121	U.S. Provisional Application No. 60/205,095
1122	Mitzenmacher, M. "Studying balanced allocations with differential equations," *Combinatorics, Probability & Computing* 8.5, pp. 473-482, September 1999
1123	Barg *et al.*, "Linear-time Binary Codes Correcting Localized Structures," *IEEE Transactions on Information Theory* 45.7, pp. 2545-2550, November 1999
1124	Shokrollahi, "New Sequences of Linear Time Erasure Codes Approaching the Channel Capacity," *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pp. 65-76, 1999

CERTIFICATE OF SERVICE

I hereby certify that on November 15, 2016, I caused a true and correct copy of the following materials

- Petition for *Inter Partes* Review of U.S. Patent No. 7,116,710
- Exhibits 1101-1124
- Fee Summary Page
- Apple Inc. Power of Attorney
- Broadcom Corp. Power of Attorney
- Certificate of Compliance

to be served via Federal Express on the following attorney of record as listed on PAIR:

CALIFORNIA INSTITUTE OF TECHNOLOGY
1200 E. CALIFORNIA BLVD.
MC 6 – 32
PASADENA, CA 91125

Respectfully Submitted,

/Brian M. Seeve/
Brian M. Seeve
Registration No. 71,721