UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Apple Inc.,
Petitioner

v.

California Institute of Technology,
Patent Owner.

Case TBD

DECLARATION OF JAMES A. DAVIS, PH.D.
REGARDING U.S. PATENT NO. 7,116,710
CLAIMS 1-8. 10-17, and 19-33

TABLE OF CONTENTS

I, James A. Davis, Ph.D., declare as follows:

1. My name is James A. Davis.

II. BACKGROUND

2. I am a Professor of Mathematics at the University of Richmond in Richmond, Virginia.

3. I received a B.S. in Mathematics (with honors) from Lafayette College in 1983 and an M.S. and Ph.D. in Mathematics from the University of Virginia in 1985 and 1987, respectively.

4. After receiving my doctorate, I taught for one year at Lafayette College before accepting a position at the University of Richmond as an Assistant Professor of Mathematics in 1988. I became an Associate Professor of Mathematics in 1994 and a Full Professor of Mathematics in 2001.

5. Since joining the faculty of the University of Richmond in 1988, I have been engaged in research in Coding Theory, Algebra, and Combinatorics. My research has appeared in journals such as IEEE Transactions on Information Theory, the Journal of Combinatorial Theory Series A, Designs, Codes, and Cryptography, the Proceedings of the American Mathematical Society, and the Journal of Algebra.

6. I have made several major contributions to the field of coding theory in wireless communication and sequence design. I co-discovered the connection

between sequences with good power control and Reed-Muller codes, an important step in making OFDM communication practical. I co-discovered a technique for constructing difference sets that has been applied to constructions of bent functions. I co-wrote the paper on the non-existence of Barker arrays.

7. I was a co-Principal Investigator of a $1.5 million National Science Foundation grant designed to engage undergraduates in long-term research projects in mathematics.

8. I have taught mathematics courses in Calculus, Statistics, Linear Algebra, Abstract Algebra, Coding Theory, and Cryptography, among others. I have directed 12 honors projects and 76 summer research experiences for undergraduates in the general area of Coding Theory and Combinatorics.

9. I spent two years (academic years 1995-96 and 2000-01) working at Hewlett-Packard Laboratories in Bristol, England. I was in a communications lab during this time, an industrial research lab focused on applications of Coding Theory to wireless communication and storage devices. I am co-inventor on 16 patents based on my work during this time.

10. I served as Chair of the Department of Mathematics and Computer Science 1997-2000.

11. I have authored or co-authored over 50 peer-reviewed academic publications in the fields of Coding Theory, Combinatorics, Finite Geometry, and Algebra..

12. A copy of my curriculum vitae is attached as Appendix A.

13. I have reviewed the specification and claims of U.S. Patent No. 7,116,710 (the "'710 patent"; Ex. 1001). I have been informed that 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.

14. I have also reviewed the following references, all of which I understand to be prior art to the '710 patent:

- 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. 1002.)
- 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. 1013)
- 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. 1003.)
- Luby, M. et al., "Analysis of Low Density Codes and Improved Designs Using Irregular Graphs," STOC '98, pp. 249-59, published in 1998 ("Luby"; Ex. 1004.)

- Luby, M. et al., "Practical Loss-Resilient Codes," STOC '97, pp. 150-159, published in 1997 (Ex. 1011)
- 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. 1005.)

15. I am being compensated at my normal consulting rate for my work.

16. My compensation is not dependent on and in no way affects the substance of my statements in this Declaration.

17. I have no financial interest in Petitioners. I similarly have no financial interest in the '710 patent.

III. LEGAL PRINCIPLES

18. I have been informed that a claim is invalid as anticipated under Pre-AIA 35 U.S.C. § 102(a) if "the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for patent." I have also been informed that a claim is invalid as anticipated under Pre-AIA 35 U.S.C. § 102(b) if "the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of the application for patent in the United States." Further I have been informed that a claim is invalid as anticipated under Pre-AIA 35 U.S.C. § 102(e) if "the invention was described in … an application for patent, published under

section 122(b), by another filed in the United States before the invention by the applicant for patent ….” It is my understanding that for a claim to be anticipated, all of the limitations must be present in a single prior art reference, either expressly or inherently.

19. I have been informed that a claim is invalid as obvious under Pre-AIA 35 U.S.C. § 103(a):

> if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which [the] subject matter pertains.

20. I understand that a claimed invention would have been obvious, and therefore not patentable, if the subject matter claimed would have been considered obvious to a person of ordinary skill in the art at the time that the invention was made. I understand that when there are known elements that perform in known ways and produce predictable results, the combination of those elements is probably obvious. Further, I understand that when there is a predictable variation and a person would see the benefit of making that variation, implementing that predictable variation is probably not patentable. I have also been informed that obviousness does not require absolute predictability of success, but that what does

matter is whether the prior art gives direction as to what parameters are critical and which of many possible choices may be successful.

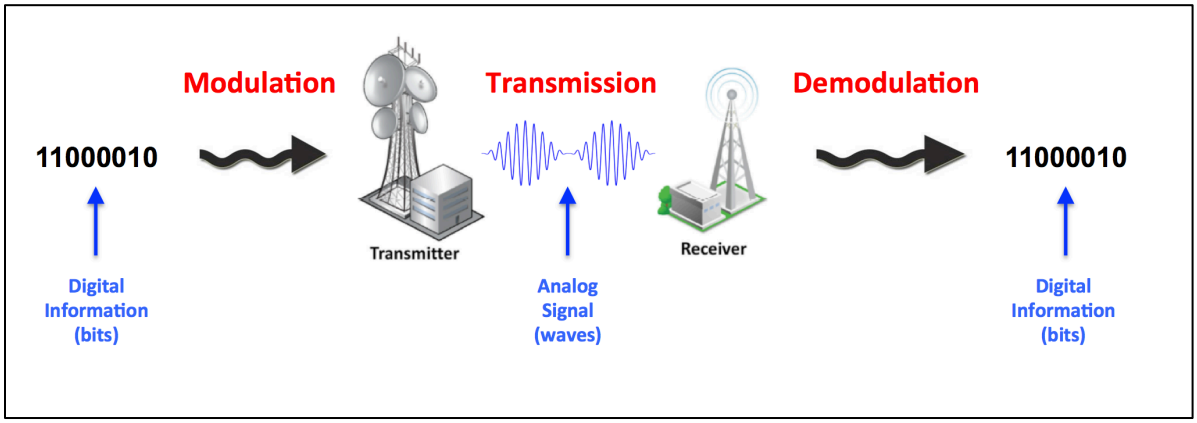
IV. OVERVIEW OF THE TECHNOLOGY

21. 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.

A. Error-Correcting Codes in General

22. 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, for example, text, music, images and video information, can be represented digitally as a collection of bits.

23. 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

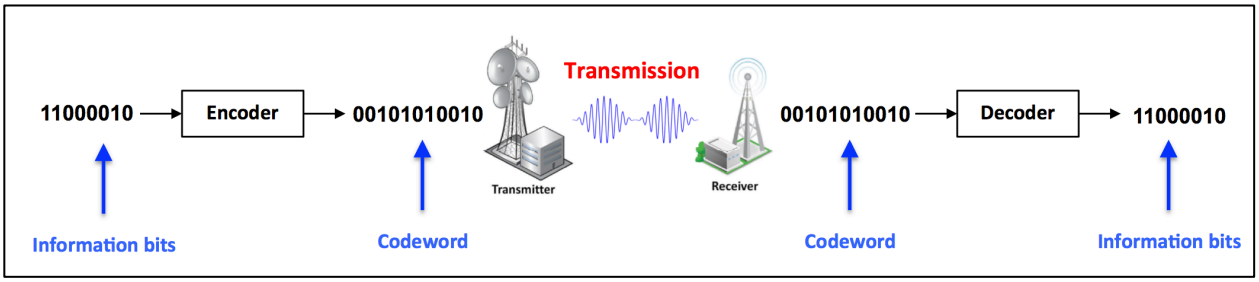
24. Transmission over physical channels is rarely 100% reliable. The transmitted signal can be corrupted during transmission by "noise" caused by, for example, obstructions in the signal path, interference from other signals, or electrical/magnetic disturbances. Noise can cause bits to "flip" during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

25. Error-correcting codes were developed to combat 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.

26. 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.

27. 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

28. 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.

29. 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.

30. 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.

31. 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.

32. 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.

B. Coding Rate

33. 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.

C. Performance of Error-Correcting Codes

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

35. 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 (meaning 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%.[1]

36. 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 (meaning 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

[1] Note that as used herein, BER refers to the *information* BER, which measures the percentage of bits that remain incorrect *after* decoding. This is different than the *transmission* BER, which measures the percentage of bits that are incorrect when *received* by the decoder, but before the decoding process begins.

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.

37. Error-correcting codes may also be assessed based on their computational complexity. The complexity of a code is a rough estimate of how many calculations are required for the encoder to generate the encoded parity bits and how many calculations are required for the decoder to reconstruct the information bits from the parity bits. If a code is too complex, it may be impractical to build encoders/decoders that are fast enough to use it.

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

38. 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. 1007.)

39. 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.

40. 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. A standard turbocoder encodes a sequence of information bits using two "convolutional" coders. The information bits are passed to the first convolutional coder in their original order. At the same time, a copy of the information bits that have been reordered by an interleaver is passed to the second convolutional coder. The figure below shows the structure of a typical turbocoder. *See* Berrou *et al.*, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," *ICC '93*, Technical Program, Conference Record 1064, Geneva 1993 (Ex. 1008)

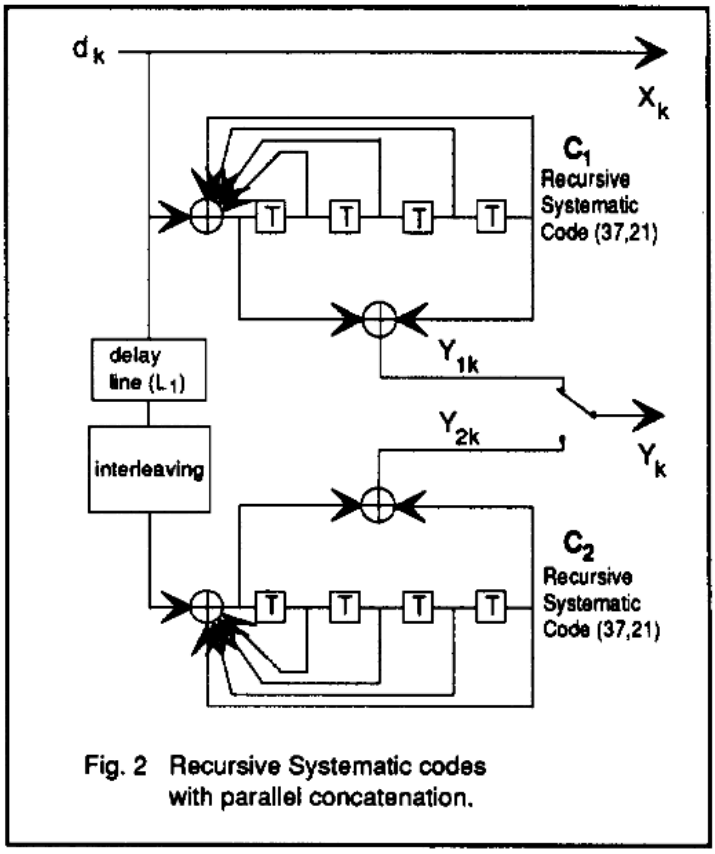


Fig. 2 Recursive Systematic codes with parallel concatenation.

41. The main drawback of convolutional codes is that they do not perform well over channels in which errors are clustered tightly together. Turbo codes 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 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 turbo code, 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.

42. 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. 1009.) This rediscovery was met with wide acclaim. Turbocodes and LDPC codes have some common characteristics: both codes use pseudo-random permutations to spread out redundancy, and both use iterative decoding algorithms.

43. 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. 1010.) While turbo codes 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.

44. In 1998, researchers developed "repeat-accumulate," or "RA codes," by simplifying the principles underlying turbocodes. (*See* Ex. 1003.) In RA codes, the information bits are first passed to a repeater that repeats (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.

45. 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. Accumulators can be implemented simply, allowing accumulate codes to be encoded quickly and cheaply.

46. Repetition and accumulation were well known in the art more than a year before the earliest claimed priority date of the '710 patent. (Ex. 1003, pp. 5-10; Ex. 1002, pp. 2-7; Ex. 1004, pp. 249-50.)

E. Mathematical Representations of Error-Correcting Codes

1. *Linear Transformations*

47. Coding theorists often think of error-correcting codes in linear-algebraic terms. For example, 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:

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

The *n*-dimensional vectors that can be written in the form $\mathbf{G}\boldsymbol{u}$ are valid codewords.

48. 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 $\boldsymbol{x}$ is a valid codeword if and only if $\mathbf{H}\boldsymbol{x} = \mathbf{0}$. 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).

49. Each of the $n - k$ rows of the parity-check matrix **H** represents an equation that a valid codeword must satisfy. For example, consider a codeword **x** and a parity check matrix **H** given as follows:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

50. If **x** is a valid codeword, the product $H\mathbf{x}$ must be equal to **0**, so we have:

$$\mathbf{H}\boldsymbol{x} = \begin{bmatrix} x_3 + x_4 \\ x_1 + x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \mathbf{0}$$

As this equation shows, the first row of **H** represents the constraint that $x_3 + x_4 = 0$, and the second row of **H** represents the constraint that $x_1 + x_2 = 0$. If the vector **x** satisfies both of these constraints, it is a valid codeword. In practice, parity-check

matrices often have hundreds or thousands of rows, each of which represents an equation of the form $x_a + x_b + \ldots + x_z = 0$, similar to those shown in the above example. These equations are called *parity-check* equations.

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

51. 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:

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

52. Using the generator matrix above, encoding a sequence of input bits "**1**0**1**0**1**0**1**" would result in the encoded sequence "**11**00**11**00**11**00**11**0011" (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 $\mathbf{G}_{\text{REPEAT2}}$ is a low-density generator matrix (LDGM).

53. 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:

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

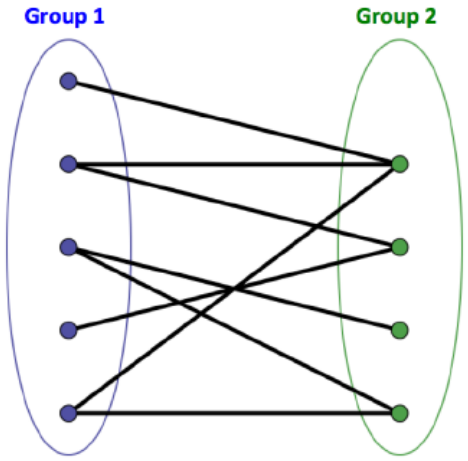
54. Permutation matrices, like $\mathbf{G}_{\text{SHUFFLE}}$ above, have exactly one 1 per row and one 1 per column. The order of the output bits is determined by the position of each of these 1s within its row/column. The matrix above, for example, would transform input bits i_1, i_2, i_3, i_4, i_5, 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 are therefore very low-density generator matrices (LDGMs).[2]

3. *Tanner Graphs*

55. Another popular 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:

[2] Indeed, for a given n and k, there are no viable codes whose generator matrices have a lower density than repeat-codes or permute-codes. Both of these codes have matrices with exactly one 1 per row – a matrix with lower density would necessarily have at least one row without any 1s at all, which would result in *every* codeword having a 0 in a particular bit position, conveying no information about the original message.



A Simple Bipartite Graph

56. 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/parity bits corresponding to those variable nodes must sum to 0.

57. These two mathematical descriptions of linear codes – one using matrices, one using Tanner graphs – are two different ways of describing the same thing. Matrices and Tanner graphs are two different ways of describing the same set of linear codes, 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.

F. Irregularity

58. 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. 1011). The paper showed that irregular codes perform better than regular codes on certain types of noisy channels.

59. *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 different (and well-known) ways of describing the same concept.

60. 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. 1004.) 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.

61. 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. (*See* Ex. 1002.) 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 turbo codes by explaining how to construct irregular turbo codes in which some information bits connect to more check nodes than others. The experimental results presented in the Frey paper demonstrated that these irregular turbo codes perform better than the regular turbo codes that were known in the art. (*See* Ex. 1002.)

62. 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.

V. Overview Of Primary Prior Art References

A. Frey

63. "Irregular Turbocodes," by Frey, B. J. and MacKay, D. J. C. ("Frey"; Ex. 1002) 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. 1015, showing a library stamp of March 20, 2000.)

64. 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 turbo codes. In particular, Frey explains how to construct irregular turbo codes, meaning turbo codes 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 turbo codes:

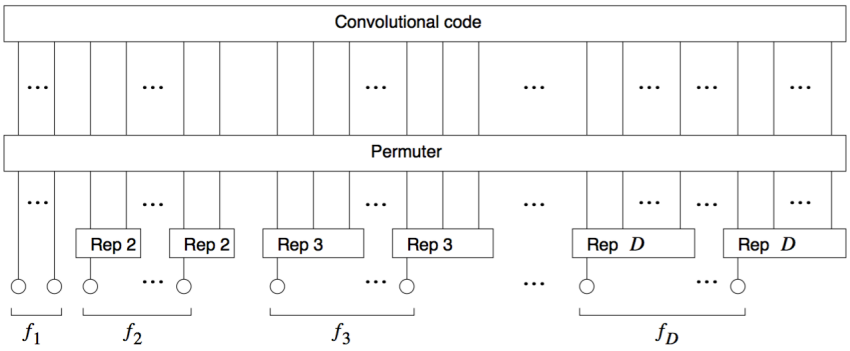


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

65. 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. 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.

66. 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. 1002, 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, though in other examples taught in Frey, such as the code shown in Figure 1, every bit is repeated at least once.

67. 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 exactly as described for the "interleaver" in the '710 patent's specification and claims.

68. After interleaving, the permuted bits are encoded using a convolutional encoder. An accumulator, like the one described in the '710 patent's specification and claims, is a simple convolutional encoder.

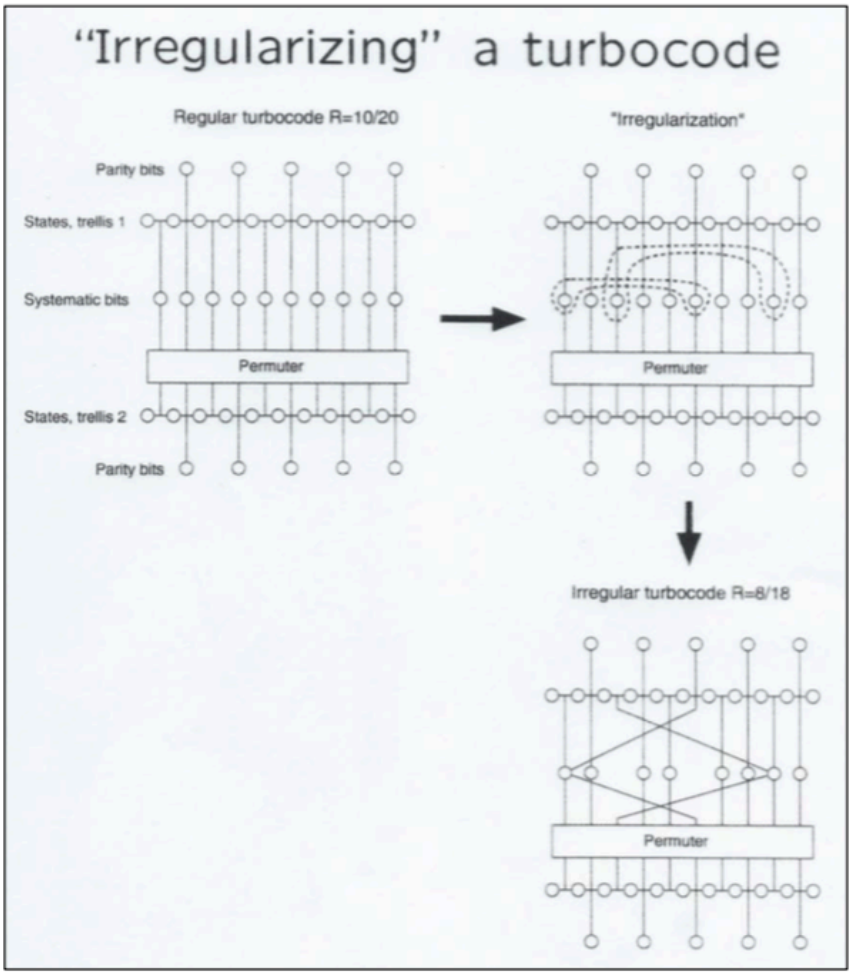
69. Frey reports on testing his irregular turbo codes by experiment. Sample "information bits" were encoded using irregular turbo codes. The resulting codeword was then sent across a simulated "noisy" channel, which introduced errors into the encoded bits similar to those that occur during transmission. 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 turbo codes. The Frey paper concluded that in certain circumstances, "[t]he irregular turbocode clearly performs better than the regular turbocode." (Ex. 1002, p. 6.)

70. 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 in the art. As described below, the Divsalar, Luby and Pfister references each used the same experimental technique.

B. Frey Slides

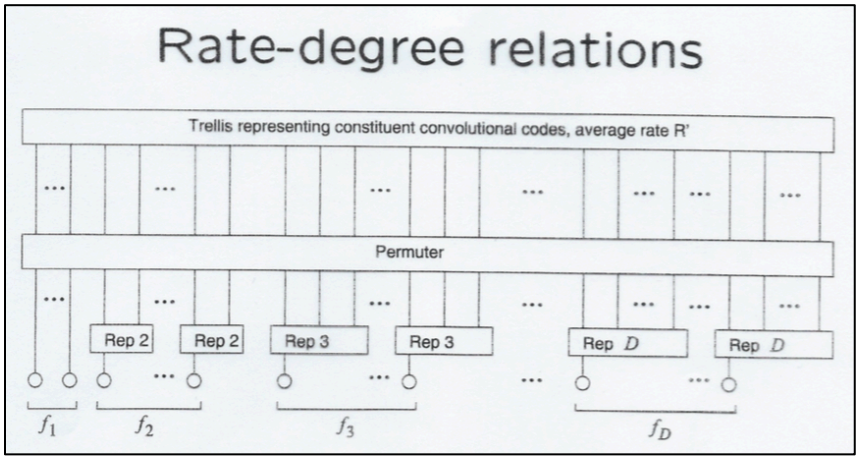
71. The Frey Slides (Ex. 1013) were presented by Brendan Frey at the Allerton Conference in 1999. The Frey Slides contain the material upon which the Frey paper, published in the Allerton 1999 conference proceedings, is based. Dr. Frey presented the Frey Slides at the Allerton Conference in September, 1999. (*See* Ex. 1017, p. 36).

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



Ex. 1013, p. 4

73. 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

74. 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).

75. As explained above, the Frey paper was published on March 20, 2000 and is thus prior art to the ’710 patent. But 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.

C. Divsalar

76. “Coding Theorems for ‘Turbo-Like’ Codes,” by D. Divsalar, H. Jin, and R. J. McEliece, was published in March 1999, in the Proceedings of the 36th

Allerton Conference on Communication, Control and Computing. ("Divsalar," Ex. 1003.)

77. 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)." (Divsalar at 1, Ex. 1003.) Figure 3 of Divsalar, reproduced below, shows an encoder for a repeat-accumulate code with rate *1/q*:

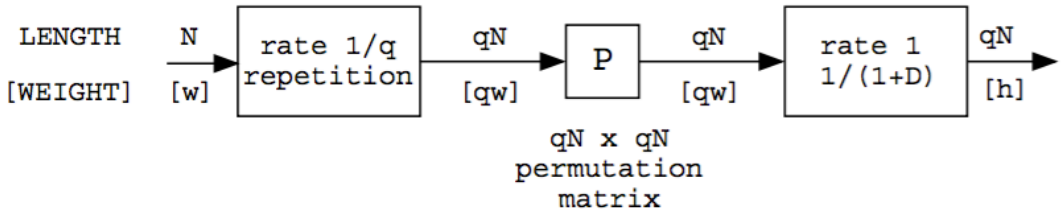


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

78. As shown in the figure above, 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.)

79. 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. 1003, 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. 1003, p. 2 "Consider a binary linear (n,k) block code...")

80. Divsalar explains that RA codes have "very good" performance and that they can be efficiently decoded using a "message passing decoding algorithm." (*Id.* at 9-10.)

81. Further, Divsalar introduces this paper as relevant to "turbo-like" codes, which Divsalar states include classical turbo codes (for example 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." (*Id.* at Abstract, 1.) Divsalar also makes use of turbo-like decoding methods in the decoder: "But 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.)

82. 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.

D. Luby

83. Building on their seminal 1997 paper (discussed briefly above), which showed that irregular codes perform better than regular codes on certain types of noisy channels, Luby et al. published a paper in 1998 titled "Analysis of Low density Codes and Improved Designs Using Irregular Graphs" that examined certain types of irregular codes in more detail. (*See* Ex. 1004; "Luby".) Luby was published on May 23, 1998, and qualifies as prior art to the '710 patent under 35 U.S.C. §102(a) and (b.) (*See* Ex. 1004.)

84. Luby explains that introducing irregularity into Gallager's low-density parity-check (LDPC) codes resulted in codes with markedly improved error-correcting capabilities. (*See generally id.*)

85. Specifically, Luby concludes:

> **In summary, irregular codes Code 14 and Code 22 appear superior to any regular code in practice, and irregular codes Code 10' and Code 14' are far superior to any regular code. We have similarly found irregular codes that perform well at other rates.**

(Ex. 1004, p. 257, emphasis added.) Luby's discovery that irregularity could be used to generate codes that were "far superior to any regular code," generated widespread interest in irregularity among computer scientists, mathematicians, and coding theorists. Like the earlier 1997 paper that introduced the concept of irregularity, the 1998 Luby reference is now widely regarded as a groundbreaking paper in the field.

86. Luby represents irregular codes using Tanner graphs (which Luby calls "bipartite graphs"). As explained above, a bipartite graph is one whose nodes can be divided into two groups, such that no two nodes in the same group are connected. As described above with reference to Tanner graphs, these two groups of nodes correspond to message nodes and check nodes. (Ex. 1004, p. 250.)

87. Luby describes a decoding algorithm in which values are passed between nodes along the edges of a bipartite graph. At the beginning of the process,

each message node receives a bit that is purported to be the correct message bit, but which has some probability *p* of being incorrect. The decoding process then proceeds in *rounds*, in which messages are passed back and forth between message nodes and check nodes, allowing each message node to refine its guess as to the correct message bit. (*Id.*, p. 253.) Each decoding round consists of the following steps:

- For all edges (m, c) do the following in parallel:
  - If this is the zeroth round, then set $g_{m,c}$ to r_m.
  - If this is a subsequent round, then $g_{m,c}$ is computed as follows:
    - if all the check nodes of m excluding c sent the same value to m in the previous round, then set $g_{m,c}$ to this value,
    - else set $g_{m,c}$ to r_m.
  - In either case, m sends $g_{m,c}$ to c.
- For all edges (m, c) do the following in parallel:
  - the check node c sends to m the exclusive-or of the values it received in this round from its adjacent message nodes excluding m.

Ex. 1004, p. 251

88. The procedure continues until the proposed values for the message nodes satisfy all the check nodes, at which point the algorithm terminates with the belief that it has successfully decoded the message. This process is commonly known as a "belief-propagation" algorithm. (Ex. 1004, 250-52.)

89. As Luby explains, irregular codes are those represented by bipartite graphs in which different message nodes have different *degrees* (*i.e.*, where different message nodes are connected to different numbers of check nodes). Luby demonstrates that such codes perform dramatically better than codes in which the message nodes all have the same degree. (*Id.*, p. 257.)

90. The improved performance of irregular codes, Luby explains, "arises from varying the degrees of the message nodes. Message nodes with high degree tend to their correct value quickly. These nodes then provide good information to the check nodes, which subsequently provide better information to lower degree message nodes. Irregular graph constructions thus lead to a wave effect, where high degree message nodes tend to get corrected first, and then message nodes with slightly smaller degree, and so on down the line." (*Id.*, p. 253.)

E. Luby97

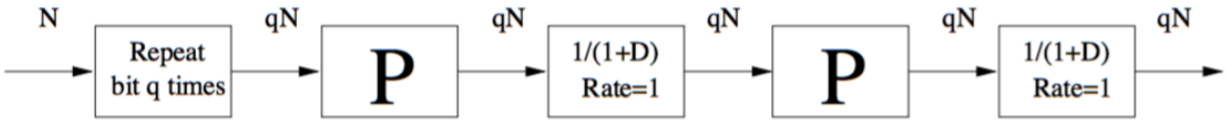
91. Luby, M. *et al.*, "Practical Loss-Resilient Codes," *STOC '97*, pp. *150-159*, published in 1997 ("Luby97"; Ex. 1011) is the paper in which Luby *et al.* first introduced irregularity, as explained above. In addition to introducing irregularity, Luby97 describes receiving data to be encoded in a *stream* of data symbols (which could be, for example, bits), where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (*Id.*, p. 150, emphasis added.) One of ordinary skill in the art 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 (meaning a "block" of information bits), which are then encoded all at once, producing a codeword of fixed size.

F. Pfister Slides

92. Pfister (Ex. 1005) was presented by Paul Siegel at the Allerton Conference in 1999. (*See* 1020, p. 3.)

93. Pfister relates to codes that are constructed as a serial chain of rate-1 encoders and random interleavers. (*See* Ex. 1005, pp. 1-2.) Pfister explicitly builds on Divsalar and introduce 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. In particular, page 20 of Pfister illustrates an RAA coder that uses two rate-1 accumulators, as shown below:



Ex. 1005, p. 20

94. 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.)

VI. PERSON OF ORDINARY SKILL IN THE ART

95. A person of ordinary skill in the art at the time of the alleged invention of the '710 patent would have had 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.

VII. OVERVIEW OF THE '710 PATENT

96. 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. 1001 at 7:15-8:64.) Claims 1-8, 10-17, and 19-33 are challenged in this Petition.

97. The specification of the ’710 patent is generally directed to irregular RA codes (or “IRA” codes). Figure 2 of the specification, reproduced below, illustrates the structure of an IRA encoder:

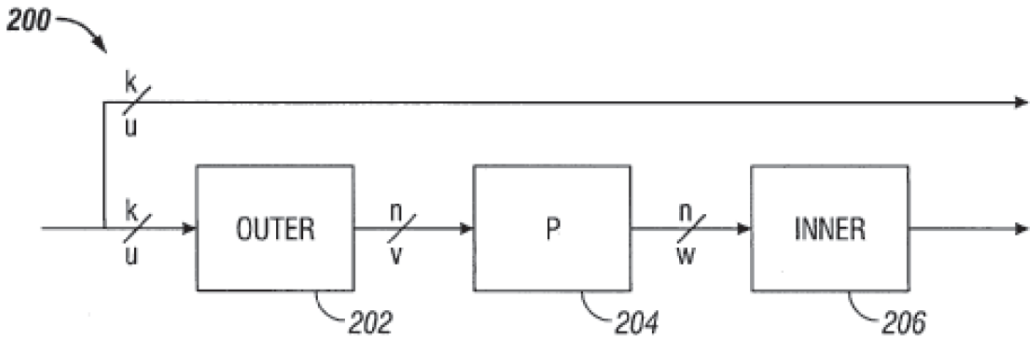


FIG. 2

98. 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. 1001 at 2:33-40.)

99. 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. (*Id.* at 2:50-52.) The outer coder repeats bits irregularly, meaning it outputs more duplicates of some information bits than others. (*Id.* at 2:48-50.)

100. The repeated bits are passed to an interleaver 204, where they are scrambled. (*Id.* at 3:18-22.) The scrambled bits are then passed to the inner coder

206, where they are accumulated to form parity bits. (*Id.* at 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. 1001 at 3:2-10

101. 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).

VIII. CLAIM CONSTRUCTION

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

102. The term "close to one" should be construed to mean "within 50% of one."

103. This construction is supported by 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. 1001 at 2:62-64, emphasis added.)

104. This is also supported by 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. 1014, pp. 7-8.) This indicates that Patent Owner agrees that the term "close to one" encompasses rates "within 50% of one" (in fact, the fact that this amendment was made in order to narrow the claim indicates that Patent Owner believes that the broadest reasonable interpretation of "close to one" is even broader).

105. This is also supported by the understanding of a person of ordinary skill in the art, who would have known that the term "close to one," as it is used in the claims of the '710 patent, encompasses rates within 50% of one.

IX. THE CHALLENGED CLAIMS ARE INVALID

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

106. Claims 1 and 3 are anticipated by Frey, which was not considered by the Patent Office during prosecution of the '710 patent.

1. *Claim 1*

107. As described below, Frey teaches all the limitations of claim 1 of the '710 patent.

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

108. To the extent that the preamble limits the claim, it is taught by Frey. As explained above, Frey deals with the construction of irregular turbo codes, which are used for encoding and decoding signals. Specifically, Section 2 of Frey describes how a turbo code is "a code that copies the systematic bits, permutes both sets of these bits, and then feeds them into a convolutional code." (Ex. 1002 at 2.) Frey also explains that irregularity can be introduced into this encoding process by "having some systematic bits replicated more than once." *Id*. at 3.

109. Frey also illustrates the encoding process graphically in Figure 2, reproduced below:

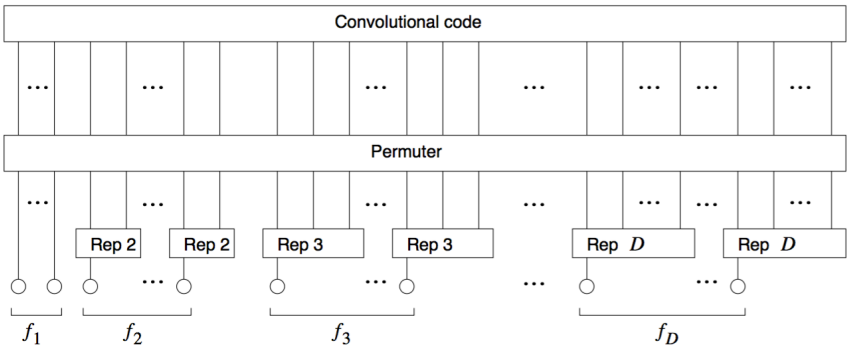


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

110. As explained in the caption, the figure illustrates that encoding is performed by repeating different groups of bits different numbers of times,

permuting the repeated bits, and encoding them using a convolutional encoder. (*See id*.)

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

111. Frey discloses obtaining data in blocks to be encoded. The encoding methods taught in Frey operate on *blocks* of data, and the codes Frey discloses are often described with reference to their "block length." For example, Frey includes experimental results comparing a regular code and an irregular code, both having "block length N = 131,082," meaning that the encoding method operates on blocks of data, each having 131,082 bits (Ex. 1002, p. 6.) Also, Frey describes experimental results relating to, for example, the "block length" of irregular turbo codes. In selecting a coding profile, Frey teaches "making small changes to a *block length N = 10,000* version of the original rate $R = 1/2$ turbo code proposed by Berrou et al." (Ex. 1002, p. 5.)

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

112. Frey teaches this limitation in describing irregular turbo codes: "[A]n irregular turbocode has the form shown in Fig. 2, which is a type 'trellis-constrained code' as described in [7]. We specify a degree profile, $f_d \in [0, 1]$, $d \in \{1, 2, \ldots, D\}$. *$f_d$ is the fraction of codeword bits that have degree d and D is the*

maximum degree. Each codeword bit with degree d is repeated d times before being fed into the permuter." (Ex. 1002, p. 2, emphasis added.)

113. As described above, Frey partitions the information bits into groups, where the bits in each group have the same degree (meaning bits within the same subgroup are all repeated the same number of times). Frey also illustrates this operation graphically in Figure 2, reproduced below:

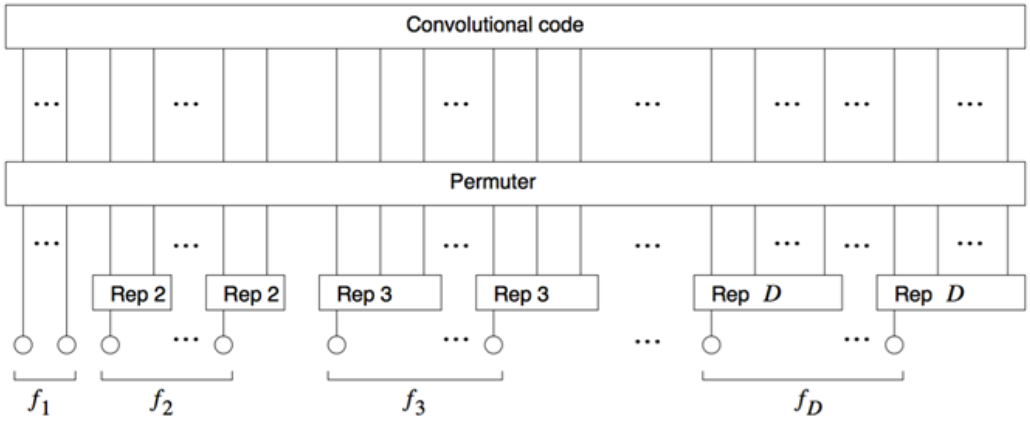


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

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. Accordingly, 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 forth. As shown in Figure 2 of Frey, each

of these sub-blocks contains a plurality of bits (or "data elements"), as required by claim 1 of the '710 patent.

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"*

114. Frey teaches repeating the data elements in different sub-blocks a different number of times (known to those of ordinary skill in the art as "irregular repetition").

115. For example, Figure 2 of Frey, reproduced above, illustrates that the data elements in each sub-block are repeated a different number of times. The circles at the bottom of Figure 2 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. For example, 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, and so forth. In Figure 2, the repeated bits are represented by the vertical lines connecting the "Rep *n*" boxes to the box labeled "Permuter." (*See* Ex. 1002, Figure 2; *see also id.*, p. 4, explaining that "For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times ….") Those vertical lines extending from the "Reb n" boxes to the "Permuter" represent the "first encoded block."

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

116. Frey teaches this limitation. In Frey, "[e]ach codeword bit with degree *d* is repeated *d* times *before being fed into the permuter.*" (Ex. 1002, p. 2, emphasis added.) Similarly, Frey explains that "a turbocode can be viewed as a code that copies the systematic bits, *permutes both sets of these bits* and then feeds them into a convolutional code." (*Id.*, p. 3, emphasis added; *see also id.*, Figs. 1 and 2, showing repeated bits connected to blocks labeled "Permuter".)

117. The "Permuter" shown in Figure 2 of Frey, reproduced above, receives the repeated bits (produced by the blocks labeled "Rep 1," "Rep 2," …, and "Rep *D*") and permutes them. (*See* Ex. 1002, Figure 2.) As the caption of Figure 2 explains, "fraction f_d of the codeword bits are repeated *d* times, *permuted* and connected to a convolutional code." (*Id*., p. 4, emphasis added.)

118. One of ordinary skill in the art around the priority date of the '710 patent would have known 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. 1016, pp. 2-3, construing both "interleaver" and "permutation module" to mean "a module that changes the order of data elements.") Divsalar also shows that one of ordinary skill would have understood the terms "permuter" and "interleaver" to be synonymous. (Ex. 1003,

p.2 "…the *i*th encoder is preceded by an interleaver (permuter)…") Accordingly, the "Permuter" shown in Fig. 2 of Frey satisfies the "interleaving" requirement of claim 1.

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

119. Frey teaches this limitation. The "second encoding" taught by Frey is a convolutional encoder, which accepts irregularly repeated and permuted bits as input and encodes these bits to produce parity bits, as shown in Figure 2, reproduced below:

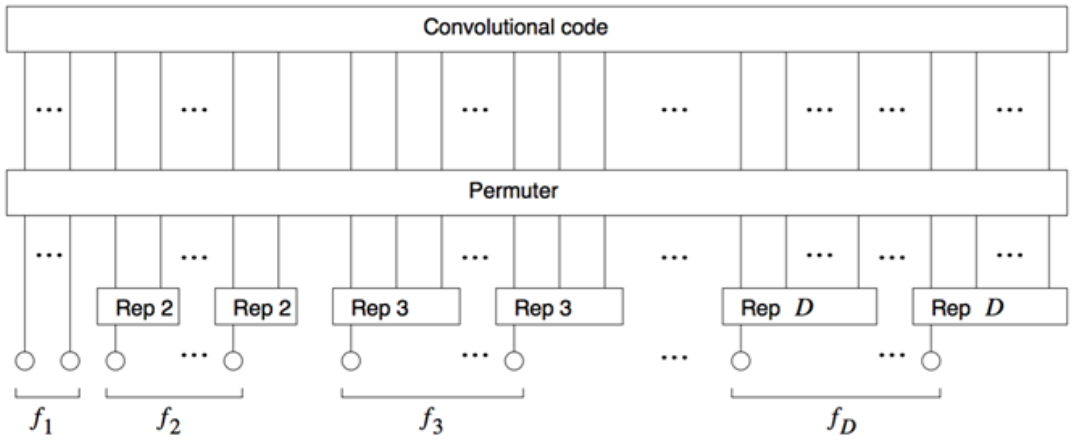


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

120. Also, the convolutional coders of Frey have a rate "close to one," as required by the claim. Specifically, the second encoding of Frey has a

"convolutional code rate of *R'* = 2/3." (Ex. 1002, p. 5..) A rate of 2/3 is within 50% of one, and is therefore "close to one," as explained above.

121. In addition, Frey also teaches a second encoder with a rate even closer to one. Repetition increases the number of bits input to the convolutional encoder, but the number of bits sent across the channel can be decreased by "puncturing," a process in which some of the encoded bits are removed so that the resulting codeword has a predetermined length. (*Id*.) Because the rate of an encoder is equal to the ratio between the number of bits input to the encoder and the total number of bits output by the encoder, puncturing the convolutional code lowers the number of output bits, reducing the denominator of the ratio and therefore raising the rate of the code. For example, Frey teaches that "[f]or a rate 1/2 turbocode, each constituent convolutional code should be rate 2/3 (which may, for example, *be obtained by puncturing a lower-rate convolutional code*.)" (Ex. 1002, p. 2, emphasis added.)

122. Frey also teaches puncturing a convolutional code to obtain the following rate (for the convolutional code):

$$R' = 1 - \frac{1-R}{\bar{d}} = 1 - \frac{1/2}{1/2 + 2(1/2 - f_e) + ef_e}$$

(Ex. 1002, p. 5.) This equation includes two variables, e, and f_e. Frey presents results that "show that for $e = 10$, $f_e = 0.05$ is a good fraction, and that for $f_e = 0.05$,

$e = 10$ is a good degree." (*Id.*) Plugging the values $e = 10$ and $f_e = 0.05$ into the equation above yields:

$$R' = 1 - \frac{\frac{1}{2}}{\frac{1}{2} + 2(\frac{1}{2} - f_e) + e \cdot f_e} = \frac{1.4}{1.9} \approx \mathbf{0.74}$$

One of ordinary skill in the art would know that 0.74, which is even closer to one than the rate of 2/3 discussed above, is "close to one," as required by the claim.

2. *Claim 3*

123. Claim 3 of the '710 patent depends from claim 1, and also 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." Frey teaches this additional limitation.

124. Frey teaches "a first coder with a variable rate less than one" as recited in claim 3. Frey's first coder is the collection of blocks labeled "Rep 2," "Rep 3," to "Rep D" in Figure 2. 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 therefore varies within a block between 1 and 1/*D*, where *D* may be set as high as desired. (Ex. 1002, Figure 2.) Also, the irregular turbocodes disclosed in Frey, at a minimum, have at least one bit that is duplicated at least once. Even at this minimum, the first encoder produces more output bits than it

receives input bits. Accordingly, it is, by definition, an encoder with a "rate less than one," as required by claim 3.

125. Frey also teaches 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 show that a rate "close to one" encompasses rates "within 50% of one."

126. Frey teaches a convolutional coder with a rate $R' \approx 0.74$. A person of ordinary skill in the art would know a rate of 0.74 (which is almost *twice* as close to one as 50%) is "substantially close to one," as required by the claim.

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

127. Claims 1-8 and 11-14 are obvious over the combination of Divsalar and Frey. Although Divsalar was before the Patent Office during prosecution of the '710 patent, Frey was not.

1. *Motivation to Combine Divsalar and Frey*

128. Divsalar and Frey are both directed to the same field (the field of error-correcting codes), and they are both specifically directed to variations on turbo codes. Frey is directed to introducing irregularity into turbo codes. (*See generally* Ex. 1002, titled "Irregular Turbocodes"; *see also id*, p. 2, explaining that "by tweaking a turbocode so that it is irregular, we obtain a coding gain ….").
Similarly, Divsalar is entitled "Coding Theorems for 'Turbo-Like' Codes," and

relates (as this title indicates) to codes that have properties like those of turbo codes. Divsalar's "turbo-like" codes specifically include not only turbo codes (like those examined in Frey), but also repeat-accumulate codes. (*See* Ex. 1003, Title, Abstract; *see also id.*, p. 2, explaining that the paper "define[s] the class of '*turbo-like*' codes" and states "… a conjecture … about the ML decoder performance of *turbo-like* codes," emphasis added.)

129. Frey teaches that making a turbo code irregular can lead to codes ("irregular turbocodes") with better performance than classical turbo codes. (*See, e.g.*, Ex. 1002, p. 6, explaining that "[t]he irregular turbocode clearly performs better than the regular turbocode for BER > 10^{-4}".) To a person of ordinary skill in the art, this would have provided a significant motivation to apply irregularity to Divsalar's "turbo-like" RA codes by, for example, combining the irregular repetition of Frey with the repeat-accumulate codes of Divsalar, resulting in an irregular repeat-accumulate (or "IRA") code, as discussed in more detail below.

130. The motivation to combine the teachings of Frey and Divsalar would have been particularly strong because of the similarities between the coding schemes taught by the two references. For example both Frey and Divsalar use a repeater as their first (outer) encoder. (*Compare* 1002, Fig. 2 *with* 1003, Fig. 3.) Both Frey and Divsalar use an interleaver between their first (outer) encoder and their second (inner) encoder. (*Id.*) And both Frey and Divsalar use a convolutional

encoder as their second (inner) encoder. (*Id.*) Specifically, where Frey teaches using a convolutional encoder as a second coder (Ex. 1002, Figure 2), Divsalar's second coder is a "truncated rate-1 recursive *convolutional encoder* with transfer function 1/(1 + D.)") (Ex. 1003, p. 5, emphasis added.) Accordingly, one of ordinary skill in the art would have understood Frey and Divsalar to disclose coding methods and systems with components (for example, repeaters or convolutional encoders) that could be substituted for one another.

Indeed, Frey *himself* suggested this combination to Dariush Divsalar in an email sent months before the priority date of the patent at issue. Specifically, Dr. Frey suggested applying the irregularity disclosed in his "Irregular Turbocodes" paper to the RA codes discussed in Dr. Divsalar's "Coding Theorems for 'Turbo-Like' Codes" paper. (*See* Ex. 1017, 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.)

131. Incorporating the irregular repetition of Frey into the RA codes of Divslar would have required only a minor change to the implementation of the

Divsalar 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 Frey, which repeats different information bits different numbers of times. This would have been a trivial modification for a person of ordinary skill in the art to make to an existing RA coder.

132. Also, incorporating the irregular repetition of Frey into the RA codes of Divsalar would have required only a minor change to the code itself. Below is a Tanner graph of an RA code that was included in the thesis of Aamod Khandekar, one of the named inventors on the patent at issue:

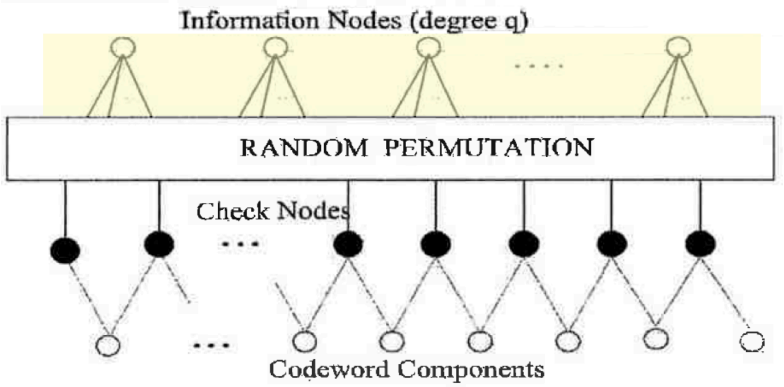
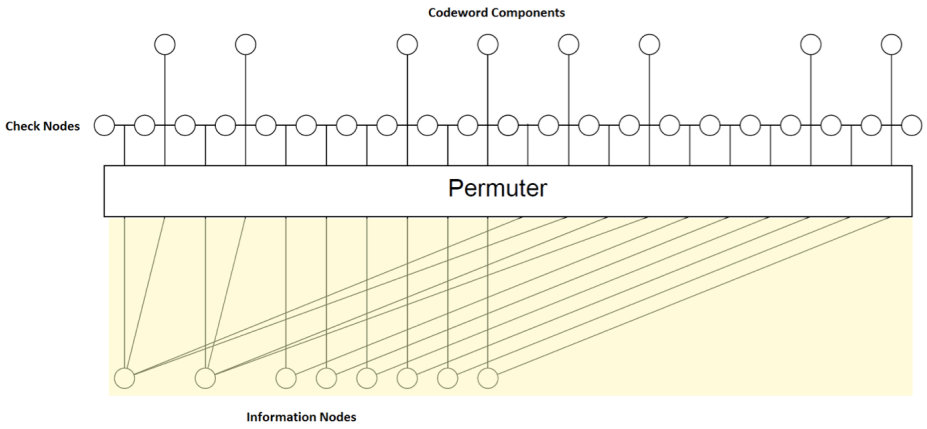


Figure 1.6: The Tanner graph of an RA code.

Ex. 1018, Fig. 1.6 (annotations added)

In the figure above, the information nodes and the edges leading from them to the permutation block are highlighted in yellow. As shown in the figure above, each information node is connected to exactly three check nodes.

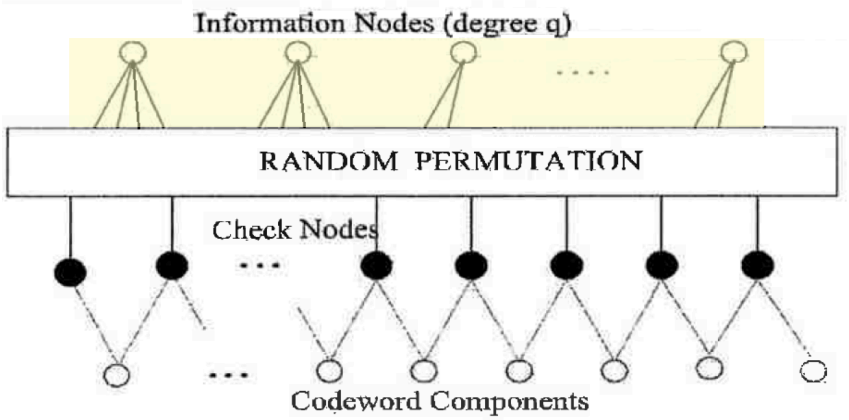
133. Compare the figure above with the below figure, excerpted from Frey:



Ex. 1002, p. 3 (annotations added)

The figure above is similar to the Tanner graph from the Khandekar thesis, but flipped vertically, so the information nodes and the connections leading from them to the permuter (highlighted in yellow) are at the bottom of the figure, not the top. Unlike the Tanner graph from the Khandekar thesis, some of the information nodes in Frey's Tanner graph are connected to four check nodes, while others are connected to two check nodes. This corresponds to the irregular repetition of Frey. In other words, it is what makes Frey's codes irregular.

134. Incorporating the irregularity of Frey into the RA codes of Divsalar would have required only the following change to Khandekar's Tanner graph above:



Tanner Graph of IRA code (Modified from Khandekar Thesis)

This Tanner graph is identical to the Tanner graph from the Khandekar thesis shown above, except that where the Khandekar Tanner graph had exactly three connections between the information bits and the "Random Permutation" block, the graph above has been made irregular, similar to the Frey graph, with some information nodes having four connections and other information nodes having two. Also, the total number of edges is the same as it was in Khandekar's original graph (4 + 4 + 2 + 2 = 3 + 3 + 3 + 3 = 12 edges), eliminating the need for any additional changes to the structure of the graph.

135. Accordingly, as shown by the figures above, modifying an RA code to incorporate the irregularity of Frey would have been a simple matter of redistributing a few edges. It would have been obvious to a person of ordinary skill in the art to make this minor modification, because of the proven benefits of irregularity demonstrated by Frey (and others, as discussed further below).

136. Additionally, one of ordinary skill in the art would have known that introducing irregularity into the RA codes of Divsalar would yield codes with higher performance. As explained above, Frey's conclusion was that introducing irregularity into turbo codes improved their performance. (*See, e.g.*, Ex. 1002, p. 6, explaining that "[t]he irregular turbocode clearly performs better than the regular turbocode for BER > 10^{-4}.") Divsalar teaches a class of "turbo-like" codes that it calls RA codes. Upon reading that the Frey authors had improved the performance of turbo codes by introducing irregularity, one of ordinary skill would have found it obvious to try improving RA codes by introducing irregularity.

137. One of ordinary skill would have been further motivated to incorporate the irregularity of Frey 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 *et al*. in 1998. Additional benefits of irregularity were documented by Richardson and Urbanke (*see* Ex. 1019, p. 38) and Frey. Based on these repeated demonstrations of the benefits of irregularity, one of ordinary skill would have had the motivation to incorporate Frey's irregular repetition into Divsalar's repeat-accumulate codes.

138. As explained in detail below, the similarity and combinability of Divsalar and Frey are further evidenced by the claim limitations they both teach.

2. *Claim 1*

139. As explained above, Frey teaches all the limitations recited by claim 1. Most of these limitations are also taught by Divsalar, except for the irregularity of the "first encoding" step, which Frey explicitly teaches.

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

140. To the extent that the preamble limits the claim, it is taught by Frey, as discussed above in the anticipation section, and also by Divsalar. Divsalar describes a "turbo-like" code called a repeat-accumulate code. The purpose of the disclosed repeat-accumulate code is to encode and decode signals. Specifically, Figure 3 of Divsalar, reproduced below, illustrates an "encoder" for a "repeat and accumulate code":

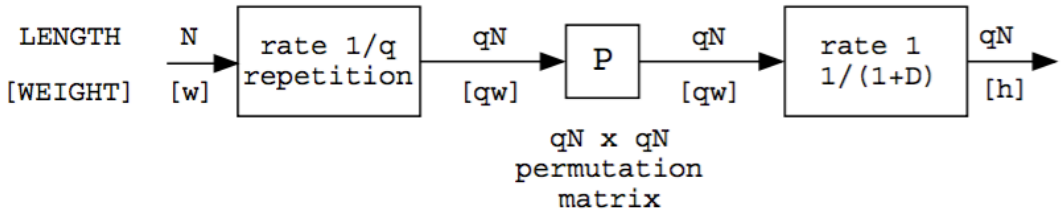


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

141. 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.) The "information block" consists of bits that form a portion of a "signal," as recited by the claim.

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

142. 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. 1003, p. 5.)

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

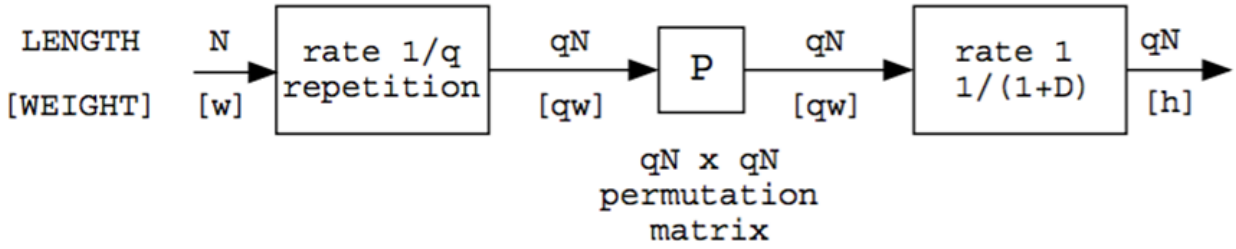


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

144. As Figure 3's caption explains, the numbers above and below the arrows connecting the components of the encoder indicate the length and weight of the corresponding "block." (*Id.*)

> 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"*

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

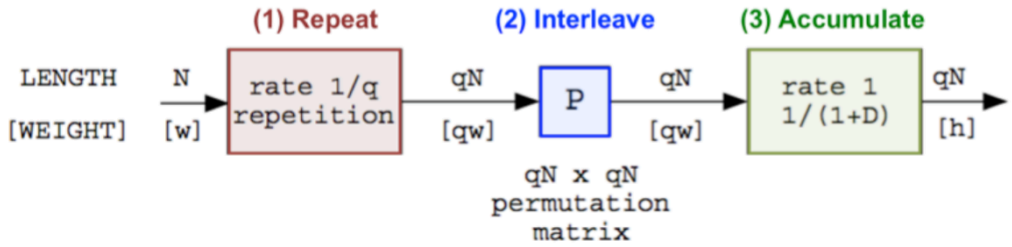
(1) repeat bits q times;

(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, reproduced below:



Ex. 1003, Figure 3 (annotations added)

146. 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.

147. Although 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" (meaning irregular repetition), Frey teaches these limitations, as explained above. Note specifically 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 (like that taught in Frey) will necessarily partition information bits into sub-blocks (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.) Applying different degrees of repetition to different bits is what *defines* the different sub-blocks of the partition.[3]

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

148. Divsalar teaches this limitation. Figure 3 of Divsalar, reproduced above, illustrates a "permutation matrix" (the box labeled "P.") After the repeater

[3]More formally, assigning degrees to each information bit groups the information bits into a set of *equivalence classes*, where two bits are in the same class if and only if they have the same degree. Every information bit is a member of one and only one equivalence class (because every information bit has exactly one degree) and so the set of equivalence classes satisfies the mathematical definition of partition.

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*." (Ex. 1003, 5.)

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

149. 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 $[x_1, \ldots, x_n]$ and output block $[y_1, \ldots, y_n]$." (Ex. 1003, p. 5.) An encoder that outputs n bits ("output block $[y_1, \ldots, y_n]$") for every *n* bits of input ("input block $[x_1, \ldots, x_n]$) has a rate of $n/n = 1$. Accordingly, 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.*) In fact, Divsalar's accumulator disclosure is nearly word-for-word identical to that of the '710 specification. (*Compare* Ex. 1003, p. 5 *with* '710 patent at 2:67-3:14.) Divsalar's accumulator therefore teaches the second encoding limitation of claim 1.

f) *Summary*

150. As explained above, the combination of Divsalar and Frey teaches all the limitations in independent claim 1. It would have been obvious to a person of ordinary skill in the art to apply the irregular repetition taught in Frey to Divsalar's repeat-accumulate coder.

3. *Claim 2*

151. Dependent claim 2 depends from independent claim 1. As explained above, Frey in combination with Divsalar teaches all the limitations of claim 1.

152. Claim 2 additionally requires that the second encoding be performed "via a rate 1 linear transformation." As explained below, the accumulator of Divsalar meets all these additional requirements.

153. Divsalar teaches that the second encoding is performed by a "rate 1 accumulator." (Ex. 1003, p. 5.) Every "rate 1 accumulator" is a "rate 1 linear encoder" that performs a "rate 1 linear transformation". Here the term "linear transformation" is used to refer to a mathematical operation that can be represented as a matrix.[4]

154. The "accumulation" operation described in Divsalar can be represented by a matrix $\mathbf{G}_{\text{ACCUM}}$ of the following form:

[4]More formally, a "linear" transformation is any transformation $\mathbf{G}$ such that:

$$\mathbf{G}(\alpha \boldsymbol{u_1} + \beta \boldsymbol{u_2}) = \alpha \mathbf{G} \boldsymbol{u_1} + \beta \mathbf{G} \boldsymbol{u_2}$$

All matrix transformations satisfy this criterion, so if an operation can be represented as a matrix transformation, it is "linear."

$$\mathbf{G}_{\text{ACCUM}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

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.

Accordingly, claim 2 is obvious over the combination of Divsalar in view of Frey.

4. *Claim 3*

155. Claim 3 depends from claim 1, and also 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." As explained above, claim 1 is obvious over the combination of Frey and Divsalar. As explained above, Frey teaches the additional limitation imposed by claim 3. Also, as was explained above for claim 2, Divsalar's second encoder has a rate exactly equal to one. In addition, 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 Frey's irregularity into Divsalar, thus making the rate

of the encoder variable. Accordingly, claim 3 is also obvious over the combination of Divsalar in view of Frey.

5. *Claim 4*

156. Claim 4 depends from claim 3, and also requires that "the second coder comprise[] an accumulator." As explained above, Divsalar teaches a second coder that is an accumulator. Claim 4 therefore is obvious over the combination of Divsalar in view of Frey.

6. *Claim 5*

157. Claim 5 depends from claim 4, and also requires that "the data elements comprise[] bits." Both Frey and Divsalar teach methods of encoding signals in which the "data elements" comprise bits. For example, Frey teaches codes in which "[e]ach codeword *bit* with degree *d* is repeated *d* times before being fed into the permuter." (Ex. 1002, p. 2, emphasis added; *see also id.* at p. 3, explaining that irregular turbo codes can be applied to "systematic *bits*," emphasis added.) Similarly, Divsalar teaches a "*binary* linear (*n, k*) block code." (Ex. 1003, p. 2, emphasis added.) One of ordinary skill in the art would understand Divsalar's "binary" block code to be a code in which the input data elements are "binary digits" or "bits."

7. *Claim 6*

158. Claim 6 depends from claim 5. As explained above, the combination of Divsalar and Frey teaches all the limitations of claim 5.

159. Claim 6 recites a limitation relating to irregular repetition, which would have been obvious over the combination of Frey and Divsalar, as explained above.

160. Specifically, Claim 6 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." As explained above, Frey teaches this additional limitation. Specifically, Frey teaches that "a *degree profile*, $f_d \in [0, 1]$, $d \in \{1, 2, \ldots, D\}$. f_d is the fraction of codeword bits that have degree d and D is the maximum degree. Each codeword bit with degree d is repeated d times before being fed into the permuter." (Ex. 1002, p. 2, emphasis added.) The "degree profile" described in the above passage from Frey determines what fraction of information bits are repeated d times, for all relevant values of d. (*See id.*)

161. This accords 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. 1001, 2:55-58, emphasis added.)

Accordingly, claim 6 is obvious over Divsalar in view of Frey.

8. *Claim 7*

162. Claim 7 depends from independent claim 4. As explained above, the combination of Divsalar and Frey teaches all the limitations in claim 4.

163. Claim 7 requires that the first coder comprise "a low-density generator matrix coder." Both Frey and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder.

164. As background, 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.

165. The generator matrix associated with a "repeat" encoder (whether regular, as taught by Divsalar, or irregular, as taught by Frey) is a low-density generator matrix. For example, the following is a generator matrix that can be used to repeat each information bit twice:

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

166. In this matrix, each column corresponds to bits input to the LDGM encoder: The first column corresponds to a first bit input to the encoder, the second column corresponds to a second input bit, the third to the third, and so forth. Because each row of this matrix contains only a single "1," each parity bit produced by this matrix will be a duplicate (or "repeat") of one of the input bits. The number of repeated bits generated by this encoder is defined by the number of "1s" appearing in each input bit's row. Using the generator matrix above, encoding a sequence of input bits "101010101" would result in the encoded sequence "110011001100110011."

167. Accordingly, a generator matrix corresponding to a repeat encoder has exactly one "1" per column. A $k \times n$ repeater matrix, with k rows and n columns, contains a total of n 1s, for a total density of $n/(kn) = 1/k$. One of ordinary skill in the art would know that a matrix having only a single 1 per row is a "low-density"

matrix. The first encoding step taught by both Frey and Divsalar is implemented using a repeater, and therefore both Frey and Divsalar teach first encoders that "comprise a low-density generator matrix coder" that implements a low-density generator matrix transformation. Also, as was explained above in the background section, a generator matrix for a repeat code has as low a density as any practical code. Therefore, both Frey and Divsalar teach the additional requirement imposed by claim 7.

9. *Claim 8*

168. Claim 8 depends from claim 1. As explained above, Frey alone teaches all the limitations of claim 1. Also, the combination of Divsalar and Frey teaches every claim limitation in claim 8.

169. Claim 8 also requires that "the second encoding use[] a transfer function of 1/(l+D.)" Divsalar teaches this additional limitation. As explained above, the second encoder of Divsalar is an accumulator. (*See* Ex. 1003, p. 5, "[a]n information block of length N is repeated *q* times, scrambled by an interleaver of size *qN*, *and then encoded by a rate 1 accumulator*," emphasis added.) Divsalar also 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.)

170. Accordingly, Divsalar teaches a second encoder that uses a transfer function of 1/(1+D), as required by claim 8.

10. *Claim 11*

171. The combination of Divsalar and Frey teaches all the limitations of independent claim 11.

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

172. To the extent the preamble is limiting, it is taught in both Frey and Divsalar, as explained above in the context of claim 1.

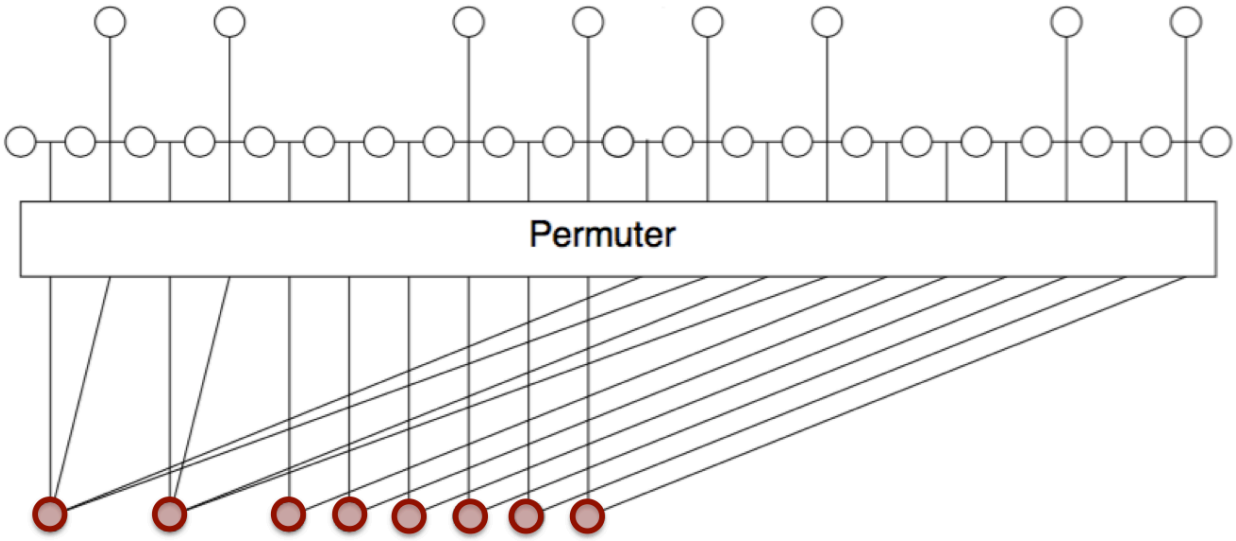
b) *"receiving a block of data in the signal to be encoded, the data block including a plurality of bits"*

173. Both Frey and Divsalar teach receiving a block of data to be encoded, as explained above in connection with claim 1. Frey teaches also that the data block includes "a plurality of bits." (*See, e.g.*, Ex. 1002, p. 4, explaining that "fraction f_d of the codeword <u>bits</u> are repeated d times, permuted and connected to a convolutional code," emphasis added.) Divsalar also teaches that the block of data to be encoded includes a plurality of bits. (Ex. 1003, p. 2, referring to "the signal-to-noise ratio <u>per bit</u>," emphasis added.)

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"*

174. As explained above with reference to claim 1, Frey teaches irregularly repeating each bit in the data block in order to form a first encoded data block.

Specifically, Figure 1 of Frey, a portion of which is reproduced below, illustrates that every information bit is repeated at least once:



Ex. 1002, Fig. 1 (excerpt, annotations added)

175. In the excerpt from Figure 1, above, the information bits in the data block are highlighted in red. Multiple lines are connected to each red circle, indicating that each of the information bits is repeated at least once. The two left-most information bits (red circles) are repeated four times, whereas the other information bits (red circles to the right) are repeated twice.

176. Also, as explained above, Frey teaches irregular repetition, in which some bits are repeated more than others. Accordingly, Frey teaches systems in which "two or more of said plurality of bits are repeated a different number of times," as the claim requires. As explained above, it would have been obvious to use Frey's irregularity in Divsalar.

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

177. As explained above in connection with claim 1, Divsalar teaches "second encoding the first encoded data block" using an accumulator. Accordingly, Divsalar in view of Frey teaches all the limitations of claim 11.

11. *Claim 12*

178. Dependent claim 12 depends from independent claim 11. As explained above, Divsalar in view of Frey teaches all the limitations of claim 11.

179. Claim 12 additionally requires that the second encoding be performed "via a rate 1 linear transformation." As explained above for claim 2, Divsalar teaches this limitation.

12. *Claim 13*

180. Claim 13 depends from independent claim 11. As explained above, the combination of Divsalar and Frey teaches all the limitations in claim 11.

181. Claim 13 requires that the first coding be "via a low-density generator matrix transformation." As explained above in the context of claim 7, both Frey and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder. Therefore, both Frey and Divsalar teach the additional requirement imposed by claim 13.

13. *Claim 14*

182. Claim 14 depends from independent claim 11, and also requires that "the signal to be encoded comprise[] a plurality of data blocks of fixed size." This additional limitation is taught by Divsalar. For example, Figure 3 of Divsalar, reproduced below, illustrates that each step of the encoding process is associated with a "block" length:

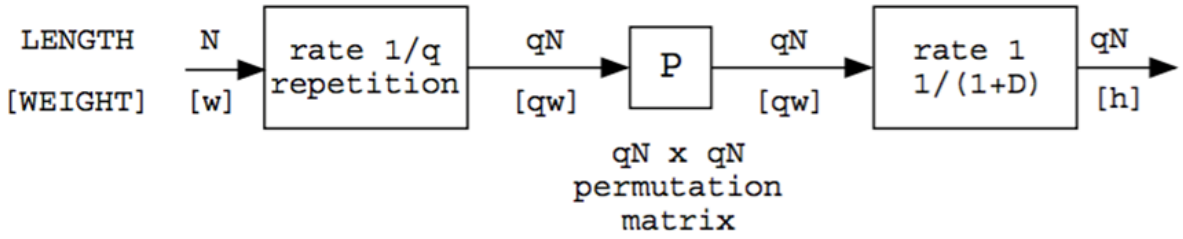


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

183. As this figure illustrates, 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. Accordingly, claim 14 is obvious over Divsalar in view of Frey.

C. Ground 3: Claims 15-17, 19-22, and 24-33 Are Obvious over Divsalar in View of Frey and also in view of Luby97

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

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

185. As explained above, one of ordinary skill in the art would have had the motivation to incorporate Frey's teaching of irregularity into Divsalar. One of ordinary skill in the art would also have had the motivation to combine Divsalar with Luby97, as both relate to error-correcting codes and the performance thereof. (*See generally* Exs. 1003, 1011.) Specifically, a person of ordinary skill in the art would have had the motivation 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.

186. Luby97 describes receiving data to be encoded in a *stream* of data symbols (which could be, for example, bits), where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1011, p. 150, emphasis added.) One of ordinary skill in the art would have known 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 (a "block" of

information bits), which are then encoded all at once, producing a codeword of fixed size.

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

2. *Claim 15*

188. The combination of Divsalar, Frey, and Luby97 teaches all the limitations in independent claim 15.

a) *"A coder comprising"*

189. To the extent the preamble is limiting, it is taught by both Frey and Divsalar. As explained above, Frey deals with the construction of irregular turbo codes, which are used for encoding and decoding signals. Specifically, Section 2 of the Frey paper describes how a turbo code is "a code that copies the systematic bits, permutes both sets of these bits, and then feeds them into a convolutional code." (Ex. 1002, p. 2.) Frey also explains that irregularity can be introduced into

this encoding process by "having some systematic bits replicated more than once." (*Id.* at 3.)

190. Frey also illustrates an encoder graphically in Figure 2, reproduced below:

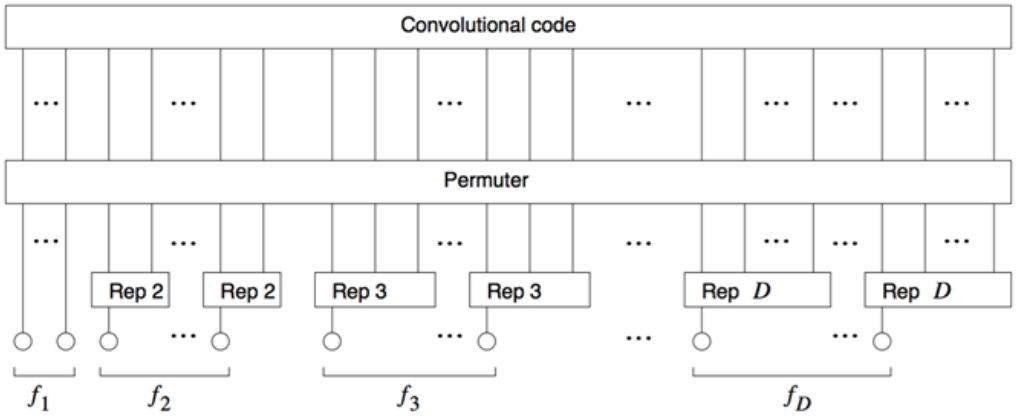


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

191. As explained in the caption, the figure illustrates that encoding is performed by repeating different groups of bits different numbers of times, permuting the repeated bits using a "Permuter," and encoding them using a convolutional encoder. (*See id.*)

192. The preamble is also 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. Specifically, Figure 3

of Divsalar, reproduced below, illustrates an "encoder" for a "repeat and accumulate code":

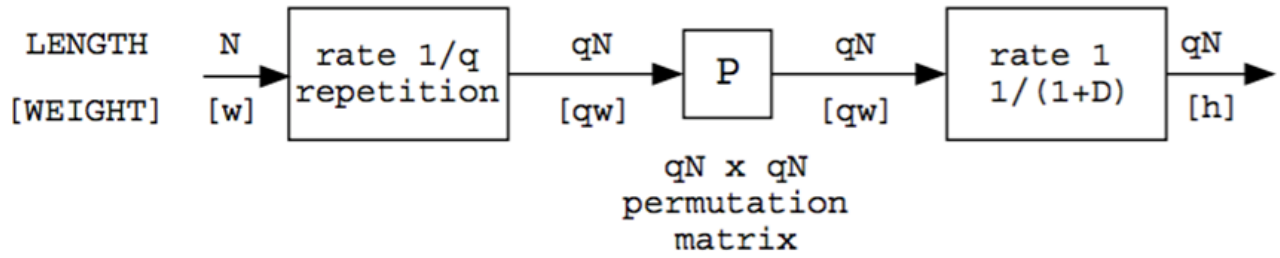


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

193. 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.)

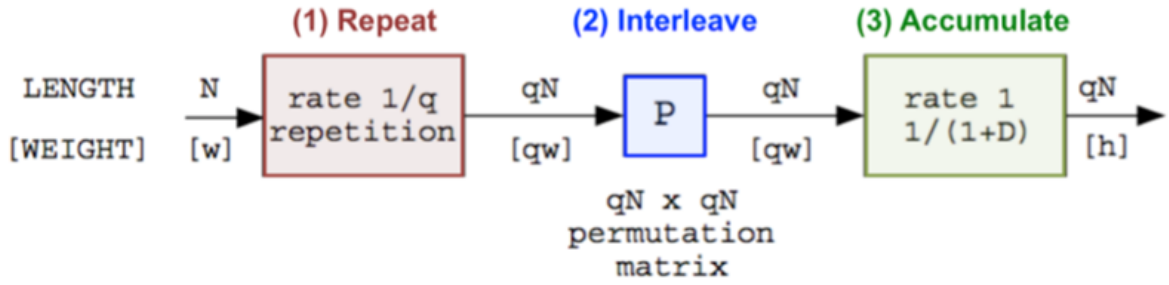
b) *"a first coder having an input configured to receive a stream of bits, said first coder operative to repeat said stream of bits irregularly and scramble the repeated bits"*

194. Luby97 teaches "a first coder having an input configured to receive a stream of bits." Luby97 teaches a coder configured to receive a "stream" of bits. (*See* Ex. 1011, p. 150). As explained above, one of ordinary skill in the art would

understand that the "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

195. Frey teaches a first coder operative to repeat bits irregularly and scramble the repeated bits. As explained above in the context of claim 1, Frey teaches a first coder that irregularly repeats bits. Frey also teaches that the irregularly repeated bits are passed as input to a permuter, which scrambles the repeated bits. Together, these two stages of the Frey encoder meet the above limitation of claim 15.

196. While Divsalar does not teach repeating bits "irregularly," it meets every other requirement of this claim limitation when modified with the "stream" teachings of Luby97, as explained above. Figure 3 of Divsalar, annotated below, shows an encoder with three stages:



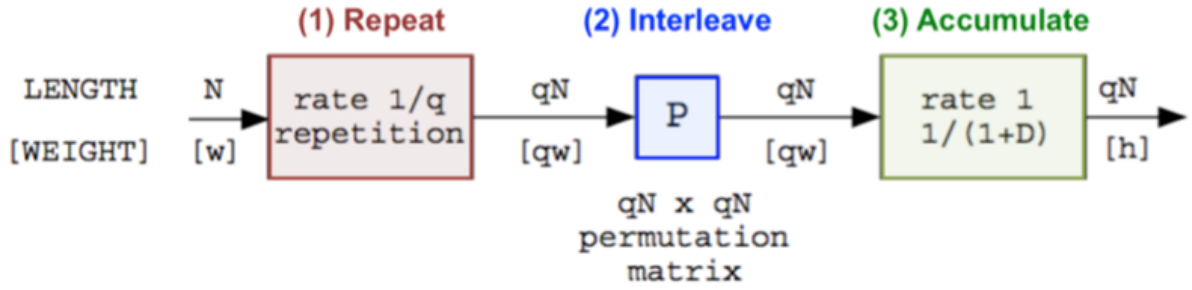
Ex. 1003, Figure 3 (annotations added)

197. Together, stages (1) and (2) of the above encoder – the "Repeat" and "Interleave" stages – modified to incorporate the "stream" teachings of Luby97 – constitute "a first coder having an input configured to receive a stream of bits, said

first coder operative to repeat said stream of bits and scramble the repeated bits." The "Repeat" stage of Divsalar does not repeat the bits *irregularly*, as claim 15 requires, but as explained above with reference to claim 1, it would have been obvious to replace the "Repeat" block above with the irregular repeater of Frey and the "stream" teachings of Luby97, resulting in a coder that meets every requirement of the limitation above.

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

198. As explained above in connection with claim 1, Divsalar teaches "a second coder operative to further encode bits output from the first coder at a rate within 10% of one." The second coder taught by Divsalar is an accumulator, as shown in Figure 3 of Divsalar, reproduced below:



Ex. 1003, Figure 3 (annotations added)

199. An accumulator, as shown in the figure above, produces qN bits of output for every qN bits of input, resulting in a coding rate of $qN/qN = 1$. Accordingly, the second encoder of Divsalar has a rate "within 10% of one" (in fact, its rate is 1.) (*See also id.*, p. 5, explaining that "[a]n information block of

length N is repeated q times, scrambled by an interleaver of size qN, and then encoded by a rate 1 accumulator.")

200. The second coder taught in Frey is a convolutional encoder, which accepts irregularly repeated and permuted bits as input and encodes these bits to produce parity bits, as shown in Figure 2, reproduced below:

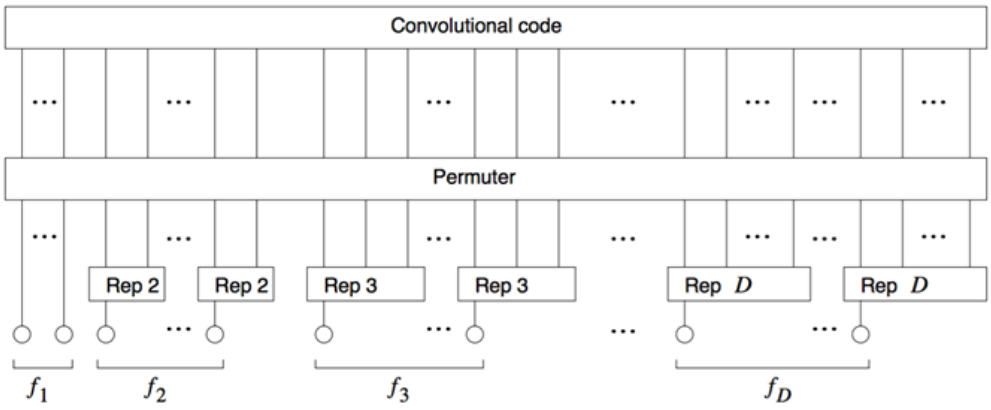


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

201. The convolutional coder of Frey is "a second coder operative to further encode bits output from the first coder," as required by claim 15, but it does not have the recited "rate within 10% of one."

202. However, as explained above, one of ordinary skill in the art would have had the motivation to substitute the convolutional coder of Frey with the rate-1 accumulator of Divsalar, yielding a second encoder that meets the above limitation of claim 11.

3. *Claim 16*

203. Claim 16 depends from claim 15, and also 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."

204. As explained above for claims 1 and 6, Divsalar in view of Frey teaches every limitation of claim 15 except for the possible exception of the "stream of bits includes a data block."

205. Also, as was 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. 1011, p. 150, emphasis added.) As explained above, one of ordinary skill in the art would have had the motivation to incorporate these "stream" teachings of Luby97 into the coders taught by Divsalar. Claim 16 is therefore obvious over Divsalar in view of Frey and also in view of Luby97.

206. Claim 17

207. Claim 17 depends from claim 16. As explained above, the combination of Divsalar, Frey, and Luby97 teaches every claim limitation in claim 16.

208. Claim 17 requires that "the second coder comprises a recursive convolutional encoder with a transfer function of 1/(l+D.)" Divsalar teaches this additional limitation. As explained above, the second encoder of Divsalar is an accumulator. (*See* Ex. 1003, p. 5, "[a]n information block of length N is repeated *q* times, scrambled by an interleaver of size *qN*, *and then encoded by a rate 1 accumulator*," emphasis added.) Divsalar also 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.)

209. Accordingly, Divsalar teaches a second encoder that that is a "recursive convolutional encoder with a transfer function of 1/(l+D)," as claim 17 requires.

4. *Claim 20*

210. Claim 20 depends from independent claim 15. As explained above, the combination of Divsalar, Frey, and Luby97 teaches all the limitations in claim 15.

211. Claim 20 requires that the first coder comprise "a low-density generator matrix coder." As explained above in the context of claim 7, both Frey and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder. Therefore, both Frey and Divsalar teach the additional requirement imposed by claim 20.

5. *Claim 21*

212. Dependent claim 21 depends from independent claim 15. As explained above, the combination of Divsalar, Frey, and Luby97 teaches all the limitations of claim 15.

213. Claim 21 also requires that the second encoder comprise "a rate 1 linear encoder." As explained above for claim 2, Divsalar teaches the limitation of claim 20.

214. Accordingly, claim 21 is obvious over Divsalar, Frey, and Luby97.

6. *Claim 22*

215. Dependent claim 22 depends from independent claim 21. As explained above, the combination of Divsalar, Frey, and Luby97 teaches all the limitations of claim 21.

216. Claim 22 additionally requires that "the second coder comprise[] an accumulator." As explained above for claim 4, Divsalar teaches the limitation of claim 22.

217. Accordingly, claim 22 is obvious over the combination of Divsalar, Frey, and Luby97.

7. *Claim 25*

218. The combination of Divsalar, Frey, and Luby97 teaches all the limitations of independent claim 25.

a) *"A coding system comprising"*

219. To the extent the preamble is limiting, it is taught in both Divsalar and Frey, as explained above in the context of claims 1, 11, and 15.

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"*

220. As explained above in the context of claim 15, this limitation would have been obvious over Divsalar, Frey, and Luby97.

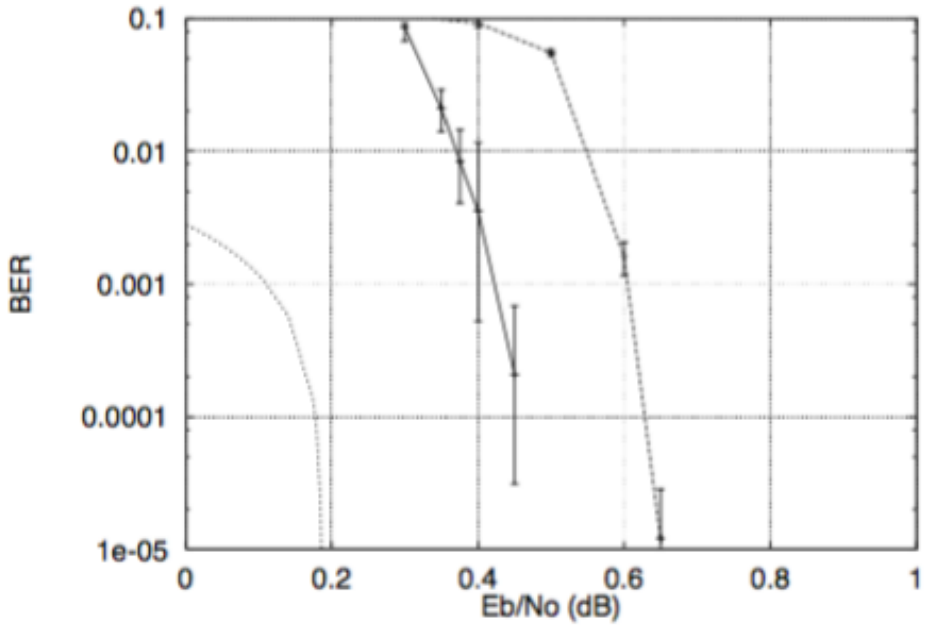
c) *"a second coder operative to further encode bits output from the first coder at a rate within 10% of one in order to form an encoded data stream"*

221. As explained above in the context of claim 15, Divsalar teaches this limitation.

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

222. Both Frey and Divsalar teach this limitation. Frey, for example, explains "[d]ecoding consists of the *iterative application* of the sum-product algorithm (a low complexity, more general form of turbodecoding) in this graph." (Ex. 1002, p. 3, emphasis added.) A person of ordinary skill in the art would have understood "iterative application of the sum-product algorithm" to be performed by "a decoder operative to receive the encoded data stream and decode the encoded data stream using an iterative decoding technique," as required by claim 25.

223. Also, in Figure 4, reproduced below, Frey presents experimental results obtained using a decoder that was built specifically to test and evaluate irregular turbo codes:



Ex. 1002, Fig. 4

As Frey explains, the above figure "shows the simulated BER-E_b/N_0 curves for the original block length N = 131,072 regular turbocode (dashed line) and its irregular cousin (solid line) …." (*Id.*, p. 6.) A person of ordinary skill in the art would know that the "simulated" curves were generated by encoding message bits using various turbo codes and decoding the resulting encoded bits using a decoder that implements an "iterative application of the sum-product algorithm" referenced above. The output of the decoder was then compared to the original message to determine how well each code is able to correct for transmission errors. This

decoder is therefore "a decoder operative to receive the encoded data stream and decode the encoded data stream using an iterative decoding technique," as claim 25 requires.

224. Divsalar also 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. 1003, p. 9, emphasis added; *see also id.*, Figure 5.) One of ordinary skill in the art would know 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 required by claim 25.

8. *Claim 26*

225. Claim 26 depends from claim 25. As explained above, the combination of Divsalar, Frey and Luby97 teaches all the limitations of claim 25.

226. Claim 26 also 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."

227. 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. 1011, p. 150, emphasis added.) As explained above, one of ordinary skill in the art would know that these "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

228. Also, as explained above in the context of claim 6, Frey teaches "repeat[ing] bits in the data block a different number of times according to a selected degree profile." Claim 26 is therefore obvious in view of Divsalar, Frey, and Luby97.

9. *Claims 19 and 27*

229. Claims 19 and 27 depend from claims 15 and 26, respectively, and both also require that the first coder comprise "an interleaver." Claim 19 also requires that the first coder comprise "a repeater having a variable rate." Divsalar and Frey both teach the claimed "interleaver" and Frey teaches the "repeater having a variable rate."

230. As explained above in the context of claim 3, Frey teaches a first coder having a variable rate. The first coder in Frey is also a repeater, as it repeats different codeword bits different numbers of times. (Ex. 1002, p. 2; *see also id.* Figure 2.) Accordingly, Frey teaches a first coder comprising "a repeater having a variable rate," as required by claim 19.

231. Frey also teaches a first coder that comprises an "interleaver." Figure 2 of Frey, reproduced below, illustrates an encoding method in which "[f]or d = 1,…D, fraction f_d of the code word bits are repeated d times, *permuted* and connected to a convolutional code." (Ex. 1002, p. 4, emphasis added):

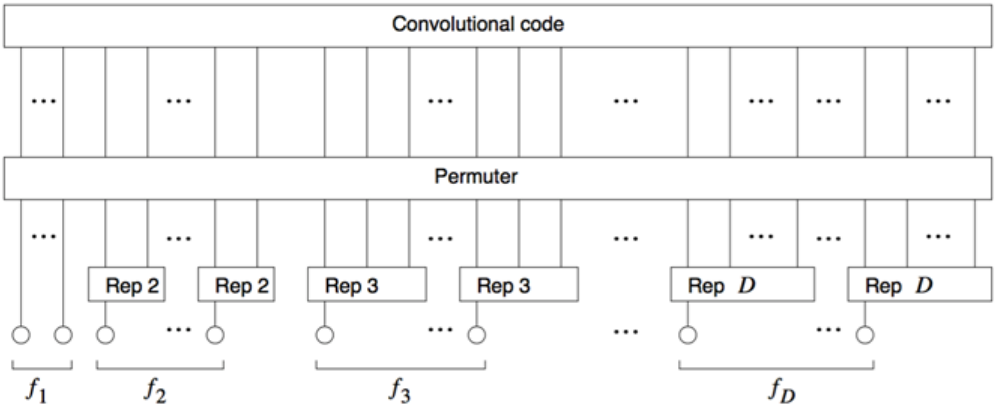


Figure 2: A general *irregular turbocode*. For $d = 1, \ldots, D$, fraction f_d of the codeword bits are repeated d times, permuted and connected to a convolutional code.

Ex. 1002, Fig. 2

232. As explained above, the "permuter" shown in Figure 2 of Frey is "an interleaver," as claims 19 and 27 require. In addition, as explained above in the context of claim 1, Divsalar also teaches a first coder comprising "an interleaver." The '710 patent discloses the "permuter" as the interleaver identified as the middle block (204) of the three-block encoder in Figure 2 of the '710 patent. (Ex. 1001 at 3:18-22, "The bits output from the outer coder 202 are scrambled before they are input to the inner coder 206. This scrambling may be performed by the interleaver 204, which performs a pseudo-random permutation of an input block v, yielding an

output block W having the same length as v.") In Divsalar, Figure 3 also has three-block encoder where the middle block (denoted with a "P") is also the interleaver. In Frey, Figure 2 similarly has a three-stage encoder where the middle stage (denoted as "Permuter") is the interleaver. To the extent that the repeater (outer coder 202) and the interleaver of the '710 patent collectively comprise the "first coder," the "first coders" of Divsalar and Frey also include both repetition and interleaving. Accordingly, Divsalar and Frey teach these additional limitations of claims 19 and 27.

233. As explained above, the combination of Divsalar, Frey, and Luby97 discloses each claim limitation in claims 15 and 26, from which claims 19 and 27 depend. Because Divsalar and Frey teach the additional limitations of claims 19 and 27, they are obvious over Divsalar, Frey, and Luby97.

10. *Claim 28*

234. Claim 28 depends from independent claim 25. As explained above, the combination of Divsalar, Frey, and Luby97 teaches all the limitations in claim 25.

235. Claim 28 requires that the first coder comprise "a low-density generator matrix coder." As explained above in the context of claim 7, both Frey and Divsalar teach a first encoder that is a low-density generator matrix (LDGM)

encoder. Claim 28 is accordingly obvious over Divsalar in view of Frey and also in view of Luby97.

11. *Claim 29*

236. Dependent claim 29 depends from independent claim 25. As explained above, the combination of Divsalar, Frey, and Luby97 teaches all the limitations of claim 25.

237. Claim 29 additionally requires that "the second coder comprise[] a rate 1 accumulator." As explained above for claims 2 and 4, Divsalar teaches the limitation added by claim 29.

238. Accordingly, claim 29 is obvious over the combination of Divsalar, Frey, and Luby97.

12. *Claim 30*

239. Claim 30 depends from claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream using a posterior [sic] decoding techniques." Frey discloses this additional limitation. Frey describes a process for decoding irregular turbo codes using "the BCJR algorithm." (Ex. 1002, pp. 3-4.) Frey also explains that "[t]he BCJR algorithm assumes the inputs are a priori log-probability ratios and uses the forward-backward algorithm to compute *a set of a posteriori log-probability ratios*. If codeword bit *i* has degree *d*, the algorithm produces *d a posteriori* log-probability ratios[.]" (*Id*. at 4.) Accordingly, the decoder in Frey decodes the encoded data using "a posterior[i] decoding techniques," as claim 30 requires.

240. As explained above, the combination of Divsalar, Frey, and Luby97 discloses all the limitations of claim 25, from which claim 30 depends. Claim 30 is therefore obvious over Divsalar, Frey, and Luby97.

1. *Claim 30*

241. 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." As described above, Luby teaches a "round"-based decoding algorithm, in which the correct values of the message bits are determined by iteratively passing messages among the information nodes and check nodes of a bipartite graph. A person of ordinary skill in the art would have understood that this decoding 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 Luby decodes the encoded data using "a posterior[i] decoding techniques," as claim 30 requires.

242. As explained above, the combination of Divsalar, Luby, and Luby97 discloses every limitation of claim 25, from which claim 30 depends. Claim 30 is therefore obvious over the Divsalar, the Frey Slides, and Luby97.

13. *Claim 31*

243. Claim 31 depends from claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream based on a Tanner graph representation."

244. Frey teaches this limitation. As Frey explains, "Fig. 2 can be interpreted as *the graphical model [6, 8-10]* for the irregular turbocode. Decoding consists of the iterative application of the sum-product algorithm (a low-complexity, more general form of turbodecoding) *in this graph*." (Ex. 1002, p. 3, emphasis added.)

245. A person of ordinary skill in the art would know that the "graphical model" Frey refers to in this passage is a Tanner graph. Therefore, Frey teaches decoding the encoded data stream "based on a Tanner graph representation," as claim 31 requires.

246. As explained above, the combination of Divsalar, Frey, and Luby97 discloses all the limitations of claim 25, from which claim 31 depends. Claim 31 is obvious over Divsalar, Frey, and Luby97.

14. *Claim 32*

247. Claim 32 depends from claim 25, and further requires that "the decoder [be] operative to decode the encoded data stream in linear time." Luby97 teaches this additional limitation. In particular, Luby97 discloses "randomized constructions of *linear-time* encodable and *decodable* codes." (Ex. 1011, 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.

248. 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. 1003, 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.

15. *Claims 24 and 33*

249. Claims 24 and 33 depend from independent claims 15 and 25, respectively, and also 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 for claims 2 and 4, Divsalar teaches a second coder (an

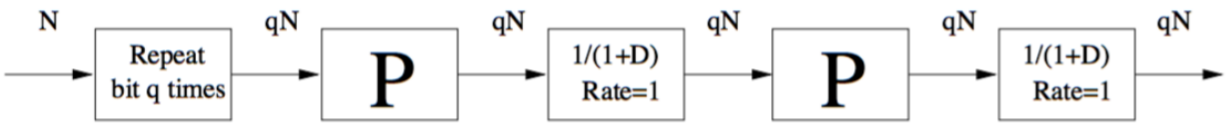
accumulator) that has a rate exactly equal to 1, and therefore has a rate that is "within 1% of one," as claims 24 and 33 require.

250. As explained above, the combination of Divsalar, Frey, and Luby97 teaches all the limitations of claims 15 and 25, from which claims 24 and 33 depend. Accordingly, for the reasons given above, claims 24 and 33 are obvious over Divsalar, Frey, and Luby97.

D. Ground 4: Claim 10 is Obvious in View of Divsalar and Frey, Also in View of Pfister

251. Claim 10 depends from claim 1. Claim 10 also requires that the second encoding "utilize[] two accumulators".

252. While Divsalar and Frey 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. 1005, p. 1 "Repeat-Accumulate-Accumulate (RAA) Codes.") In Pfister's RAA codes, the outer code comprises a repeater, which is followed by two accumulators with Rate 1. Specifically, page 20 of Pfister illustrates an RAA coder that uses two rate-1 accumulators, as shown below:



Ex. 1005, p. 20

253. 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"). Accordingly, 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 conclude that adding an extra accumulator improves the codes' performance. (*See id.*, p. 21.)

254. A person of ordinary skill in the art would have had the motivation to incorporate the two-accumulator design of Pfister into Divsalar. Both references are directed to the same field—the field of error-correcting codes. Both are directed to error-correcting codes containing serial concatenation of rate-1 codes. (*See, e.g.*, Ex. 1003, Figure 3; *see also* Ex. 1005, pp. 1-2.)

255. Additionally, incorporating a second accumulator into Divsalar's RA codes would have been obvious for one of ordinary skill in the art. As explained above, Pfister's RAA codes are simply RA codes with additional accumulators added to the end of the encoding chain in series. (*See* Ex. 1005, Figure 2.)

256. As explained above, the combination of Divsalar and the Frey Slides discloses all the limitations of claim 1, from which claim 10 depends. Claim 10 is therefore obvious over Divsalar and Frey, also in view of Pfister.

E. Ground 5: Claim 23 is Obvious in View of Divsalar, Frey, and Luby97, Also in View of Pfister

257. Claim 23 depends from claim 22. Claim 23 requires that the second coder comprise "a second accumulator." As described above in the context of claim 10, Pfister teaches this limitation.

258. As explained above, the combination of Divsalar, Frey, and Luby97 discloses all the limitations of claim 22, from which claim 23 depends. Claim 23 is therefore obvious over Divsalar, Frey, and Luby97, also in view of Pfister.

F. Ground 6: Claims 1 and 3 Are Anticipated by the Frey Slides

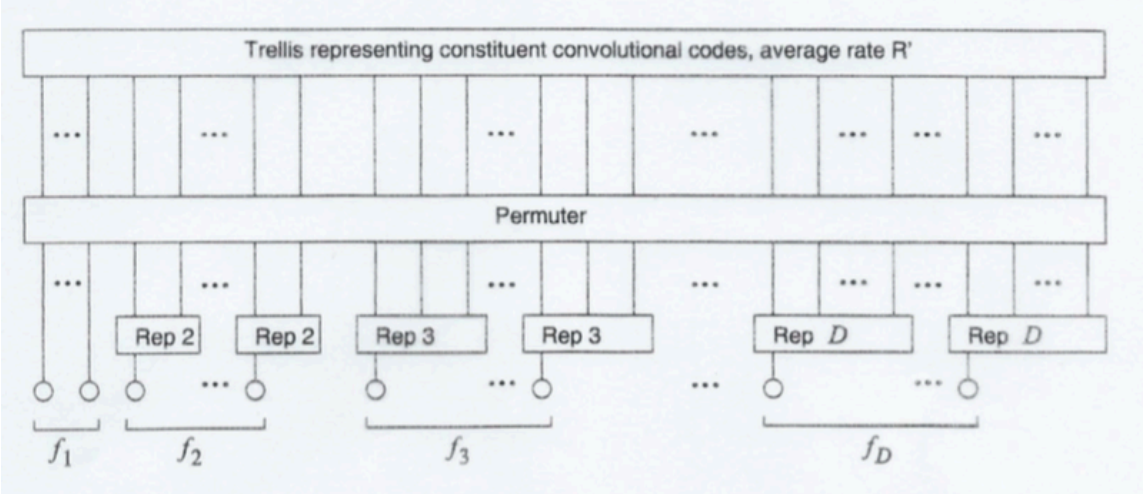
259. Claims 1 and 3 are anticipated by the Frey Slides. The Frey Slides were not considered by the Patent Office during prosecution of the '710 patent. Also, it is my understanding that grounds using the Frey Slides are not redundant of the grounds that use Frey. Accordingly, in the event the Board concludes that Frey does not qualify as prior art, the Frey Slides will nevertheless qualify as prior art to the '710 patent and invalidate the claims.

1. *Claim 1*

260. As described below, the Frey Slides teach all the limitations of claim 1 of the ’710 patent.

a) *“A method of encoding a signal”*

261. To the extent that the preamble limits the claim, it is taught by the Frey Slides. As explained above, the Frey Slides deal with the construction of irregular turbo codes, which are used for encoding and decoding signals. Specifically, the Frey Slides illustrate the encoding process graphically in a figure that is reproduced below:



Ex. 1013, p. 5

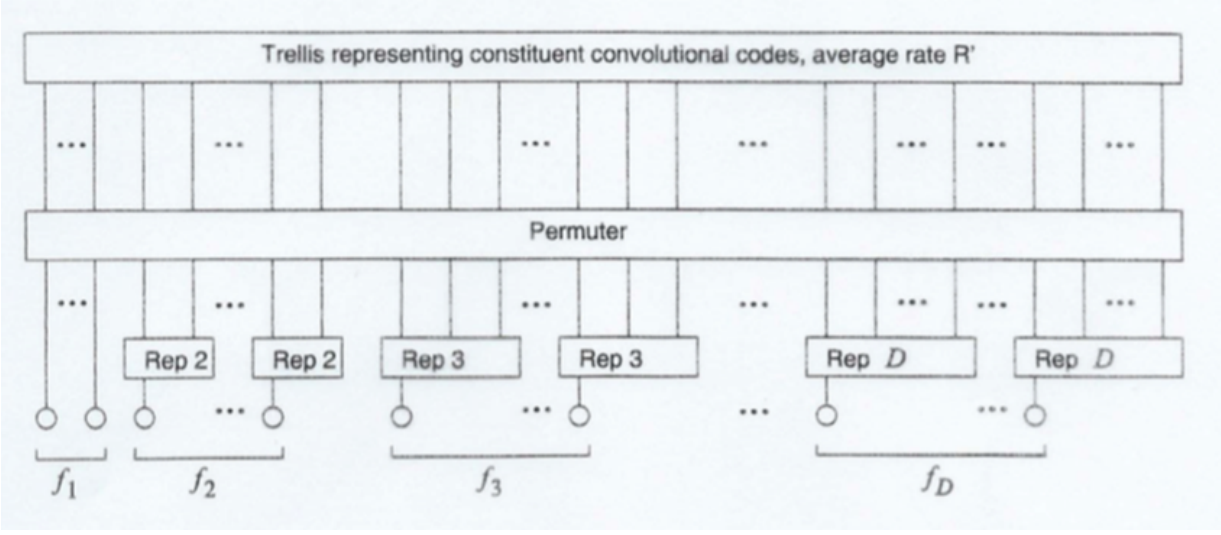
262. In the figure above, encoding is performed by repeating different groups of bits different numbers of times (the boxes labeled “Rep 2,”, “Rep 3”, etc.) permuting the repeated bits (in the box labeled “Permuter”), and encoding them using a convolutional encoder. (*See id.*)

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

263. The Frey Slides disclose obtaining data in blocks to be encoded. 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. 1013, p. 13, emphasis added.)

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

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

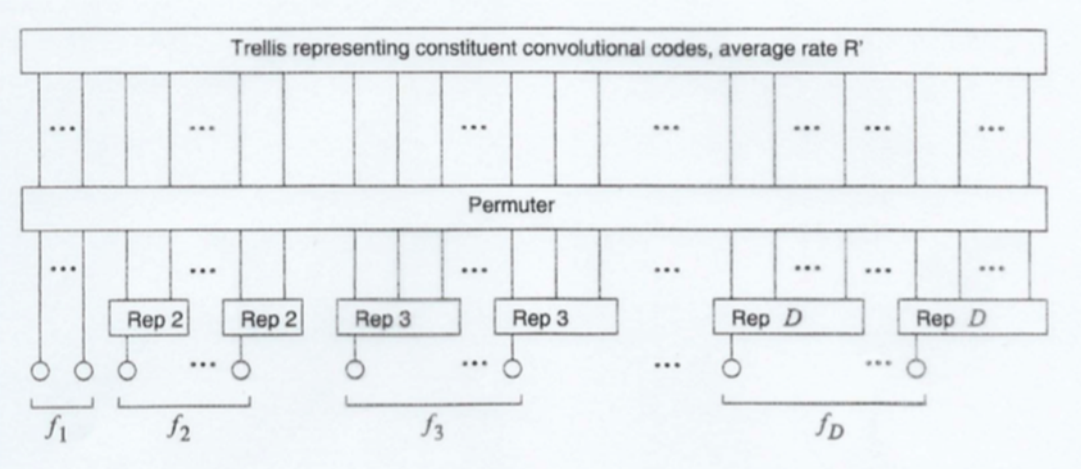


Ex. 1013, p. 5

265. 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 (meaning 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. Accordingly, 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 claim 1 of the '710 patent.

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"*

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



Ex. 1013, p. 5

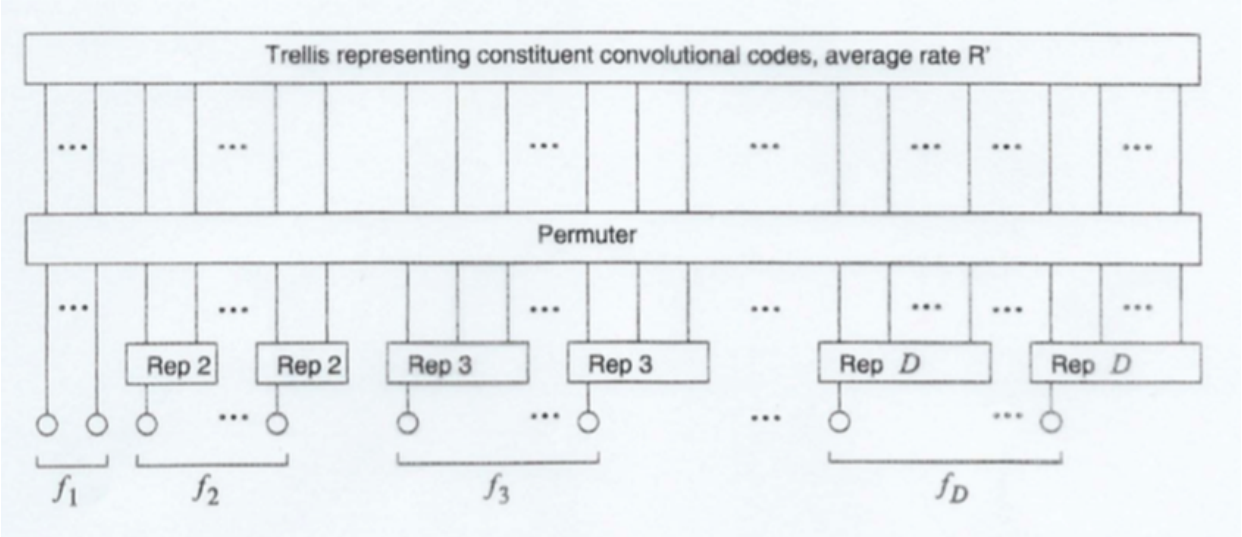
The figure reproduced above illustrates that the data elements in each sub-block are repeated a different number of times. The circles at the bottom represent information bits in the data block. The groups of information bits labeled $f_2, f_3, \ldots, f_D$ represent sub-blocks into which the data block is partitioned. The blocks labeled "Rep 2," "Rep 3," and "Rep *D*" represent the step of repetition. For example, 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. In Figure 2, the repeated bits are represented by the vertical lines connecting the "Rep *n*" boxes to the box labeled "Permuter." (*See id.*).

267. 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.

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

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



Ex. 1013, p. 5

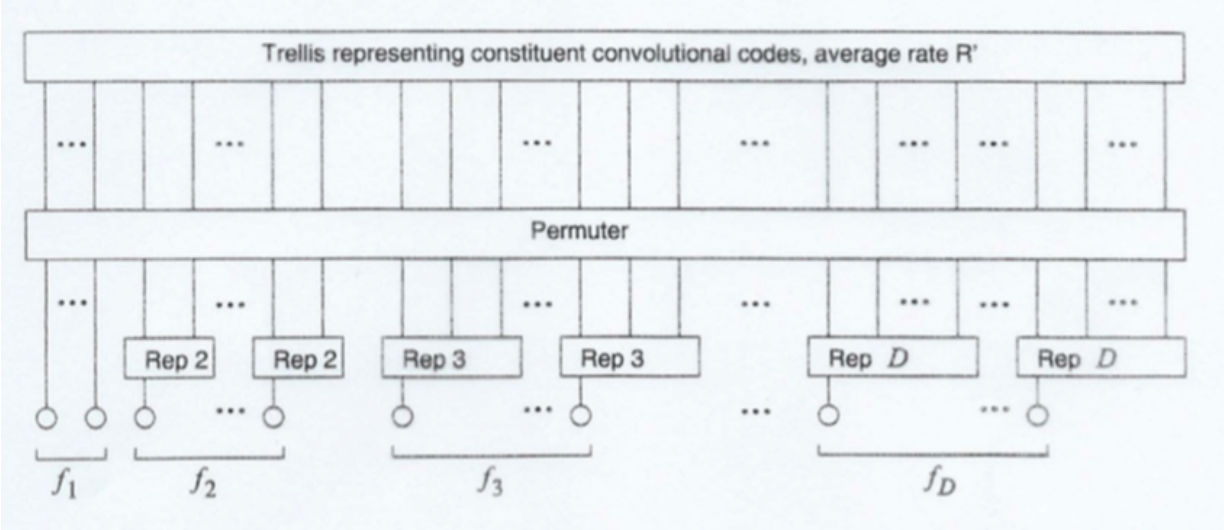
269. The "Permuter" shown above receives the repeated bits (produced by the blocks labeled "Rep 1," "Rep 2," …, and "Rep *D*") and permutes them.

270. One of ordinary skill in the art around the priority date of the '710 patent would have known 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. 1016, pp. 2-3, construing both "interleaver" and "permutation module" to mean "a module that changes the order

of data elements.") Divsalar also demonstrates that one of ordinary skill would have known that "permuting" means "interleaving." (Ex. 1003 at p. 2, "…and the ith encoder is preceded by an interleaver (permuter)…" (emphasis added)). Accordingly, the "Permuter" shown in the figure above satisfies the "interleaving" requirement of claim 1.

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

271. 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. 1013, p. 5

272. Additionally, 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. 1013, 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:

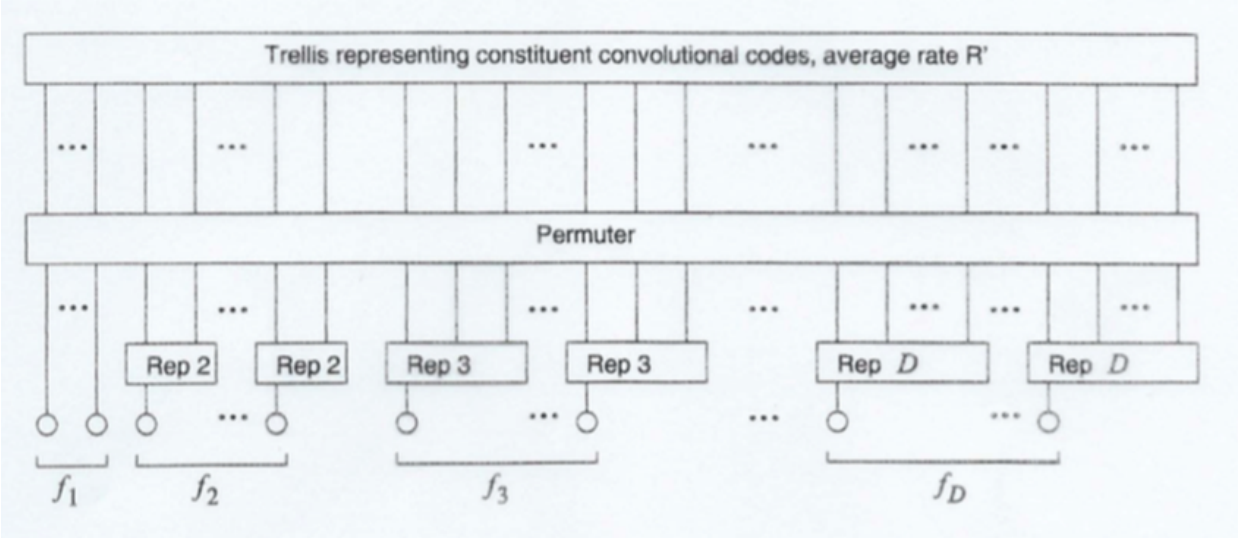
$$\bar{R} = 1 - \frac{1-R}{(1-R)+2(R-f_e)+d_e f_e} \approx \mathbf{0.72}$$

A rate of 0.72 is within 50% of one, and therefore, as explained above in the claim construction section, one of ordinary skill in the art would recognize that 0.72 is "close to one," as required by the claim.

2. *Claim 3*

273. Claim 3 of the '710 patent depends from claim 1, and also 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." The Frey Slides teach this additional limitation.

274. 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. 1013, p. 5

275. 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 therefore 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. Accordingly it is, by definition, an encoder with a "rate less than one," as required by claim 3.

276. 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."

277. The Frey Slides teach a convolutional coder with a rate of 0.72. A person of ordinary skill in the art 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.

G. Ground 7: Claims 1-8 and 11-14 are Obvious over Divsalar in View of the Frey Slides

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

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

279. The Frey Slides and Divsalar are both directed to the same field (the field of error-correcting codes), and they are both specifically directed to variations on turbo codes. Specifically, the Frey Slides are directed to introducing irregularity into turbo codes. (*See generally* Ex. 1013, 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 turbo codes, and teaches that all turbo codes 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. 1003 at Abstract; *see also id*., Title, p. 2, explaining that the paper "define[s] the class of '*turbo-like*' codes" and states "… a conjecture … about the ML decoder performance of *turbo-like* codes," emphasis added.)

280. The Frey Slides teach that making a turbo code irregular can lead to codes ("irregular turbocodes") with better performance than classical turbo codes. (*See, e.g.*, Ex. 1013, p. 11, including a graph showing that irregular codes have lower BERs than regular codes.) To a person of ordinary skill in the art, this would have supplied a significant motivation to apply irregularity to Divsalar's "turbo-like" RA codes by, for example, 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.

281. The motivation to combine the teachings of the Frey Slides and Divsalar would have been particularly 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. 1013, 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. 1003, p. 5, emphasis added.) Accordingly, one of ordinary skill in the art would have understood Frey and Divsalar to disclose coding methods and systems with components (for example, convolutional encoders) that could be substituted for one another.

Indeed, Frey *himself* suggested this combination to Dariush Divsalar in an email sent months before the priority date of the patent at issue. Specifically, Dr. Frey suggested applying the irregularity disclosed in his publications relating to irregular turbo codes to the RA codes discussed in Dr. Divsalar's "Coding Theorems for 'Turbo-Like' Codes" paper. (*See* Ex. 1017, 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.)

282. Incorporating the irregular repetition of the Frey Slides into the RA codes of Divsalar would have required only a minor 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 in the art to make to an existing RA coder.

283. Additionally, one of ordinary skill in the art would have known 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 turbo codes improved their performance. (*See, e.g.*, Ex. 1013, 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 turbo codes by introducing irregularity, one of ordinary skill would have found it obvious to try improving RA codes by introducing irregularity.

284. One of ordinary skill would have also had the motivation 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. 1004, p. 257; Ex. 1011.) Additional benefits of

irregularity were documented by Richardson and Urbanke. (Ex. 1019, p. 38.) Based on these repeated demonstrations of the benefits of irregularity, one of ordinary skill would have had the motivation to incorporate Frey's irregular repetition into Divsalar's repeat-accumulate codes.

285. As explained in detail below, the similarity and combinability of the Frey Slides and Divsalar are also evidenced by the claim limitations they both teach.

2. *Claim 1*

286. As explained above, the Frey Slides teach all the limitations 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.

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

287. To the extent that the preamble limits the claim, it 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. Specifically, Figure 3 of Divsalar, reproduced below, illustrates an "encoder" for a "repeat and accumulate code":

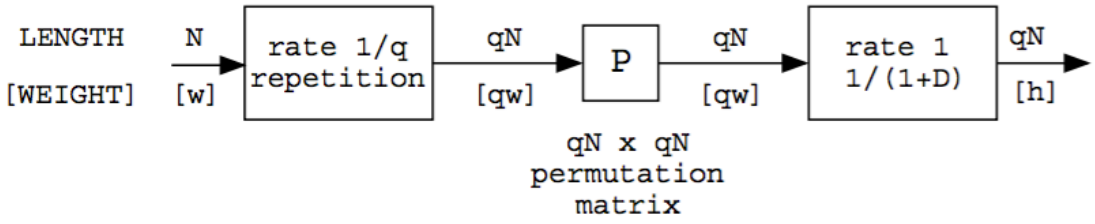


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

288. 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.)

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

289. 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. 1003, p. 5.)

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

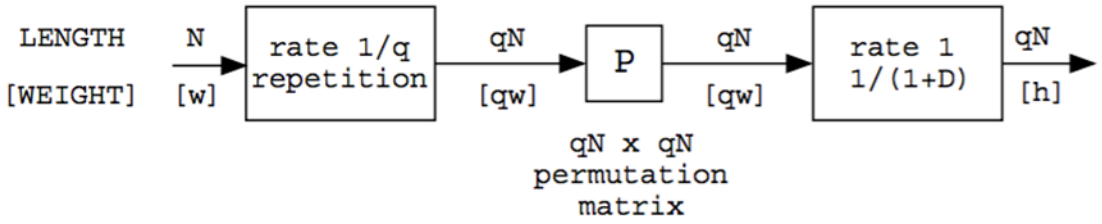


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

291. 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*..)

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"*

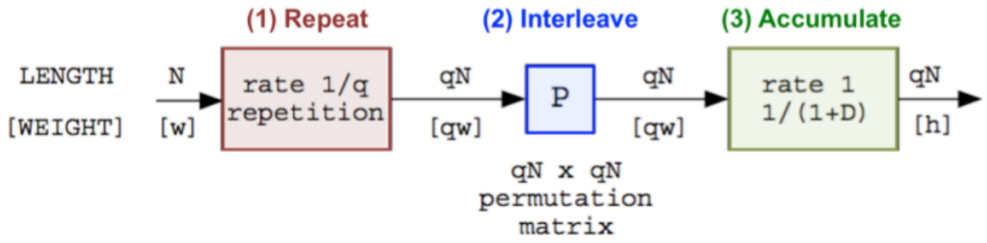
292. As explained above, Divsalar teaches an RA code that uses three steps:

(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, reproduced below:



Ex. 1003, Figure 3 (annotations added)

293. 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.

294. Although 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" (irregular repetition), the Frey Slides teach these limitations, as explained above. Note specifically 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 (like that taught in the Frey Slides) will necessarily partition information bits into sub-blocks (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.) Applying different degrees of repetition to different bits is what *defines* the different sub-blocks of the partition.[5]

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

295. Divsalar teaches this limitation. Figure 3 of Divsalar, reproduced above, illustrates a "permutation matrix" (the box labeled "P.") 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." (Ex. 1003, 5.)

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

296. 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. 1003, p. 5.) An encoder that outputs n bits ("output block [y_1, …, y_n]") for every n bits of input ("input block [x_1, …, x_n]) has a rate of $n/n = 1$. Accordingly, the "second encoder" taught by Divsalar has a rate of 1 (and

[5]More formally, assigning degrees to each information bit groups the information bits into a set of *equivalence classes*, where two bits are in the same class if and only if they have the same degree. Every information bit is a member of one and only one equivalence class (because every information bit has exactly one degree) and so the set of equivalence classes satisfies the mathematical definition of partition.

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.

f) *Summary*

297. As explained above, the combination of Divsalar in view of the Frey Slides teaches all the limitations in independent claim 1. It would have been obvious to a person of ordinary skill in the art to apply the irregular repetition taught in the Frey Slides to Divsalar's repeat-accumulate coder.

3. *Claim 2*

298. Dependent claim 2 depends from independent claim 1. As explained above, Divsalar in view of the Frey Slides teaches all the limitations of claim 1.

299. Claim 2 additionally requires that the second encoding be performed "via a rate 1 linear transformation." As explained below, the accumulator of Divsalar meets this additional requirement.

300. Divsalar teaches that the second encoding is performed by a "rate 1 accumulator." (Ex. 1003, p. 5.) Every "rate 1 accumulator" is a "rate 1 linear encoder" that performs a "rate 1 linear transformation". Here the term "linear transformation" is used to refer to a mathematical operation that can be represented as a matrix.[6]

[6]More formally, a "linear" transformation is any transformation $\mathbf{G}$ such that:

$$\mathbf{G}(\alpha \boldsymbol{u_1} + \beta \boldsymbol{u_2}) = \alpha \mathbf{G} \boldsymbol{u_1} + \beta \mathbf{G} \boldsymbol{u_2}$$

301. The "accumulation" operation described in Divsalar can be represented by a matrix $\mathbf{G}_{\text{ACCUM}}$ of the following form:

$$\mathbf{G}_{\text{ACCUM}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \cdots \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

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.

302. Accordingly, claim 2 is obvious over Divsalar in view of the Frey Slides.

All matrix transformations satisfy this criterion, so if an operation can be represented as a matrix transformation, it is "linear."

4. *Claim 3*

303. Claim 3 depends from claim 1, and also 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." As explained above, claim 1 is obvious over Divsalar in view of the Frey Slides. As also 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 exactly equal to one. Further, 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. Accordingly, claim 3 is also obvious over Divsalar in view of the Frey Slides.

5. *Claim 4*

304. Claim 4 depends from claim 3, and also requires that "the second coder comprise[] an accumulator." As explained above, Divsalar teaches a second coder that is an accumulator. Claim 4 therefore is obvious over Divsalar in view of the Frey Slides.

6. *Claim 5*

305. Claim 5 depends from claim 4, and also requires that "the data elements comprise[] bits." Both the Frey Slides and Divsalar teach methods of

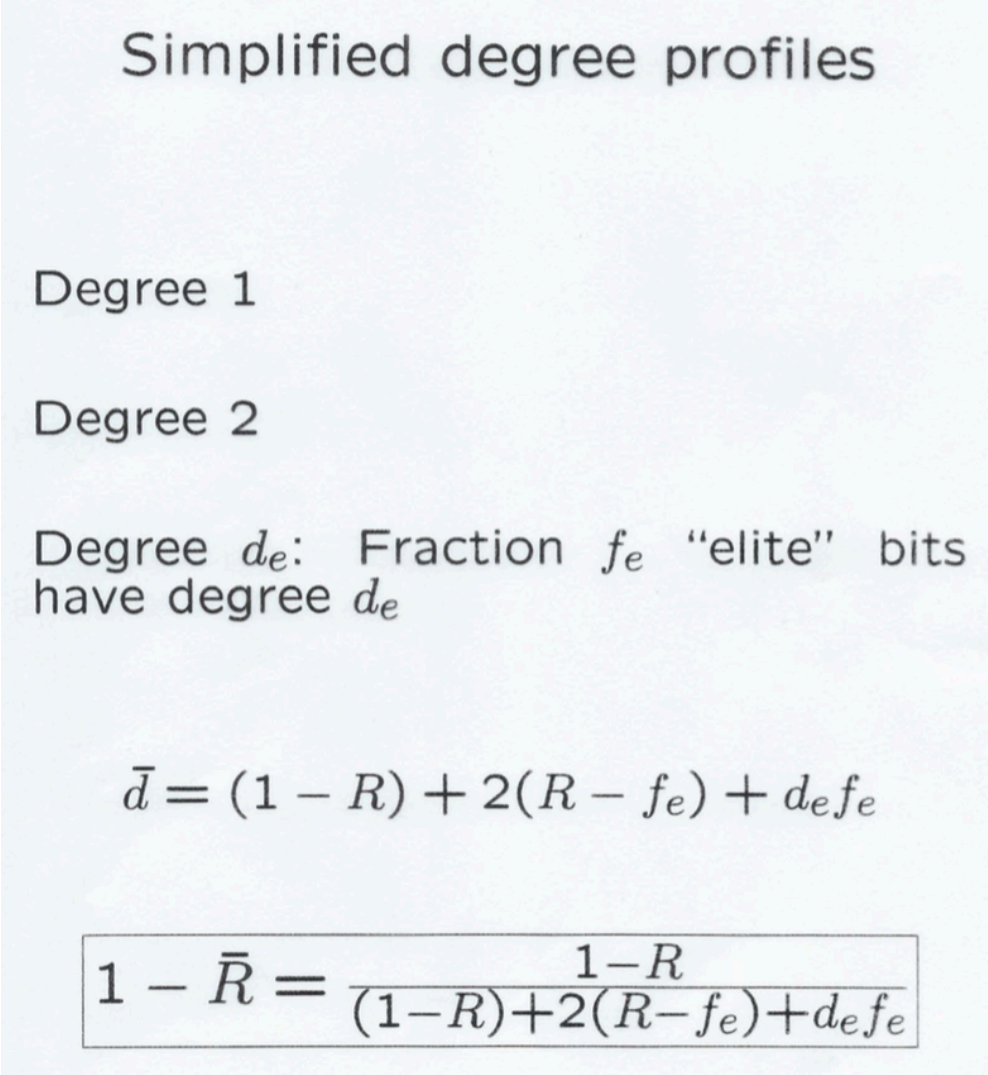
encoding signals in which the "data elements" comprise bits. For example, the Frey Slides refer to the "*bits* in a transmitted codeword" (Ex. 1013, p. 2, emphasis added; *see also id.*, p. 4, referring to "systematic bits," which a person of ordinary skill in the art would understand refers to the information bits to be encoded.) Similarly, Divsalar teaches a "*binary* linear (*n, k*) block code." (Ex. 1003, p. 2, emphasis added.) One of ordinary skill in the art would understand Divsalar's "binary" block code to be a code in which the input data elements are "binary digits" or "bits."

7. *Claim 6*

306. Claim 6 depends from claim 5. As explained above, claim 5 is obvious over Divsalar in view of the Frey Slides.

307. Claim 6 recites limitations relating to irregular repetition, which would have been obvious over Divsalar in view of the Frey Slides.

308. Specifically, Claim 6 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." The Frey Slides teach the use of a degree profile that defines how many times each sub-block is repeated, as shown below:



Ex. 1013, p. 6

309. 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.*).

310. This is supported by 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. 1001, 2:55-58, emphasis added.) Accordingly, claim 6 is obvious over Divsalar in view of the Frey Slides.

8. *Claim 7*

311. Claim 7 depends from claim 4. As explained above, the combination of Divsalar and the Frey Slides teaches all the limitations in claim 4.

312. Claim 7 requires that the first coder comprise "a low-density generator matrix coder." Both the Frey Slides and Divsalar teach a first encoder that is a low-density generator matrix (LDGM) encoder.

313. 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.

314. As explained above, 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. For example, the following is a generator matrix that can be used to repeat each information bit twice:

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

315. In this matrix, each column corresponds to bits input to the LDGM encoder: The first column corresponds to a first bit input to the encoder, the second column corresponds to a second input bit, the third to the third, and so on. Because each row of this matrix contains only a single "1," each parity bit produced by this matrix will be a duplicate (or "repeat") of one of the input bits. The number of repeated bits generated by this encoder is defined by the number of "1s" appearing in each input bit's row. Using the generator matrix above, encoding a sequence of input bits "**1**0**1**0**1**0**1**" would result in the encoded sequence "**11**00**11**00**11**00**11**0**11**."

316. Accordingly, a generator matrix corresponding to a repeat encoder has exactly one "1" per column. A $k \times n$ repeater matrix, with k rows and n columns, contains a total of n 1s, for a total density of $n/(kn) = 1/k$. One of ordinary skill in the art would know that a matrix having only a single 1 per column is a "low-density" matrix. Also, as was explained above in the background section, a

generator matrix for a repeat code has as low a density as any practical code. 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. Therefore, both the Frey Slides and Divsalar teach the additional requirement imposed by claim 7.

9. *Claim 8*

317. Claim 8 depends from claim 1. As explained above, the Frey Slides alone teach all the limitations of claim 1. Also, the combination of Divsalar and the Frey Slides teaches all the claim limitations in claim 1.

318. Claim 8 also requires that "the second encoding use[] a transfer function of 1/(l+D.)" Divsalar teaches this additional limitation. As explained above, the second encoder of Divsalar is an accumulator. (*See* Ex. 1003, p. 5, "[a]n information block of length N is repeated *q* times, scrambled by an interleaver of size *qN*, *and then encoded by a rate 1 accumulator*," emphasis added.) Divsalar also 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.)

319. Accordingly, Divsalar teaches a second encoder that uses a transfer function of 1/(1+D), as required by claim 8.

10. *Claim 11*

320. The combination of Divsalar and the Frey Slides teaches all the limitations of independent claim 11.

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

321. To the extent the preamble is limiting, it is taught in both the Frey Slides and Divsalar, as explained above in the context of claim 1.

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

322. 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 also that the data block includes "a plurality of bits." (*See, e.g.*, Ex. 1013, 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. 1003, p. 2, referring to "the signal-to-noise ratio *per bit*," emphasis added.)

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"*

323. 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. Accordingly, 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.

324. Also, as explained above with reference to claim 1, Divsalar teaches codes in which each bit in the data block is repeated. (*See* Ex. 1003, 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.

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

325. As explained above in connection with claim 1, Divsalar teaches "second encoding the first encoded data block" using an accumulator. Divsalar in view of the Frey Slides thus teaches every limitation of claim 11.

11. *Claim 12*

326. Dependent claim 12 depends from independent claim 11. As explained above, Divsalar in view of the Frey Slides teaches all the limitations of claim 11.

327. Accordingly, claim 12 is obvious over Divsalar in view of the Frey Slides.

12. *Claim 13*

328. Claim 13 depends from independent claim 11. As explained above, the combination of Divsalar and the Frey Slides teaches all the limitations in claim 11.

329. Claim 13 requires that the first coding be "via a low-density generator matrix transformation." 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. Therefore, both the Frey Slides and Divsalar teach the additional requirement imposed by claim 13.

13. *Claim 14*

330. Claim 14 depends from independent claim 11, and also requires that "the signal to be encoded comprise[] a plurality of data blocks of fixed size." This additional limitation is taught by Divsalar. For example, Figure 3 of Divsalar, reproduced below, illustrates that each step of the encoding process is associated with a "block" length:

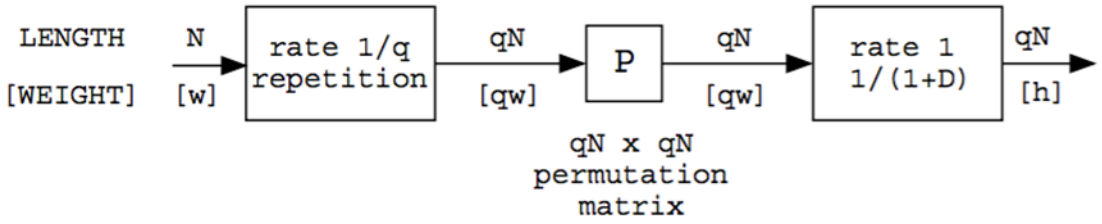


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

331. As this figure illustrates, 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. Accordingly, claim 14 is obvious over Divsalar in view of the Frey Slides.

H. Ground 8: Claims 15-17, 19-22, and 24-33 Are Obvious over Divsalar in View of the Frey Slides and also in view of Luby97

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

332. As explained above, one of ordinary skill in the art would have had the motivation to incorporate the irregularity of the Frey Slides into Divsalar. One of ordinary skill in the art would also have had the motivation to combine Divsalar with Luby97, as both relate to error-correcting codes and the performance thereof. (*See generally* Exs. 1003, 1011.) Specifically, a person of ordinary skill in the art would have had the motivation 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.

333. As explained above, Luby97 describes receiving data to be encoded in a *stream* of data symbols (which could be, for example, bits), where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1011, p. 150, emphasis added.) One of ordinary skill in the art would have known 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 (a "block" of information bits), which are then encoded all at once, producing a codeword of fixed size.

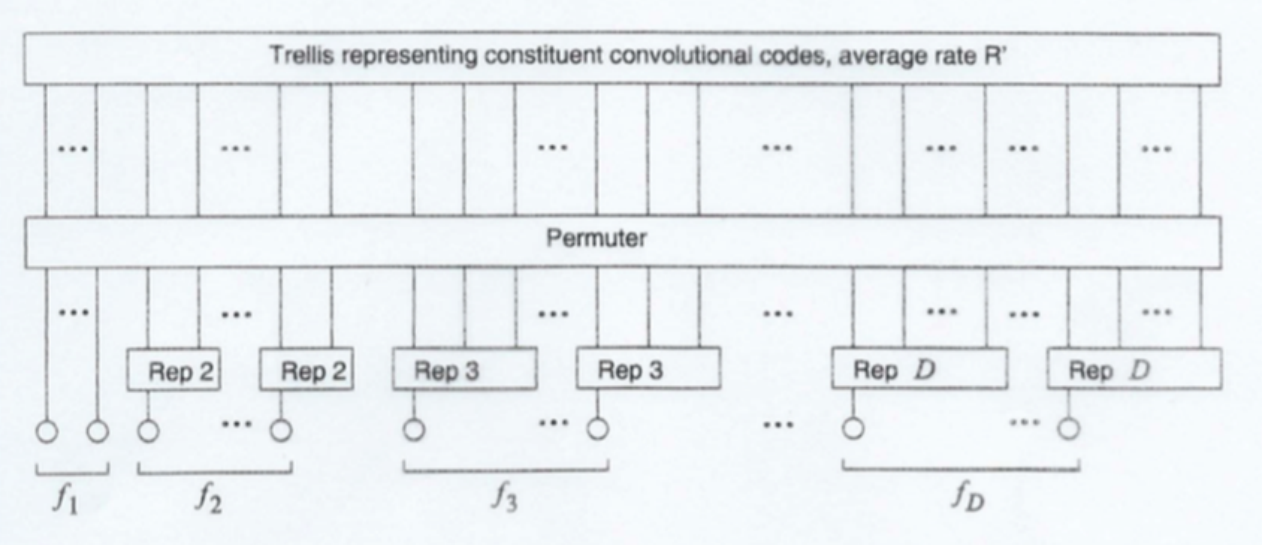
334. Coders that receive "streams" of bits that comprise one or more "blocks" of data were common, and would have been familiar to a person of ordinary skill. To the extent that "streams" were not obvious in view of Divsalar and/or the Frey Slides alone, it would have been obvious to use Luby97's teaching 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. Accordingly, it would have been obvious to a person of ordinary skill to incorporate the "stream" teachings of Luby97 above into the encoder of Divsalar.

2. *Claim 15*

335. The combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations in independent claim 15.

a) *"A coder comprising"*

336. To the extent the preamble is limiting, it is taught by both the Frey Slides and Divsalar, As explained above, the Frey Slides illustrate an encoder graphically, as shown below:



Ex. 1013, p. 5

337. 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. (*See id.*)

338. The preamble is also 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. Specifically, Figure 3

of Divsalar, reproduced below, illustrates an "encoder" for a "repeat and accumulate code":

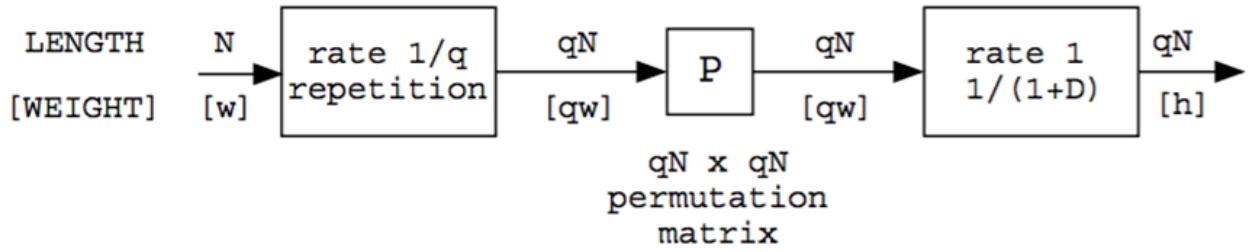


Figure 3. Encoder for a (qN, N) repeat and accumulate code. The numbers above the input-output lines indicate the length of the corresponding block, and those below the lines indicate the weight of the block.

Ex. 1003, Figure 3

339. 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.)

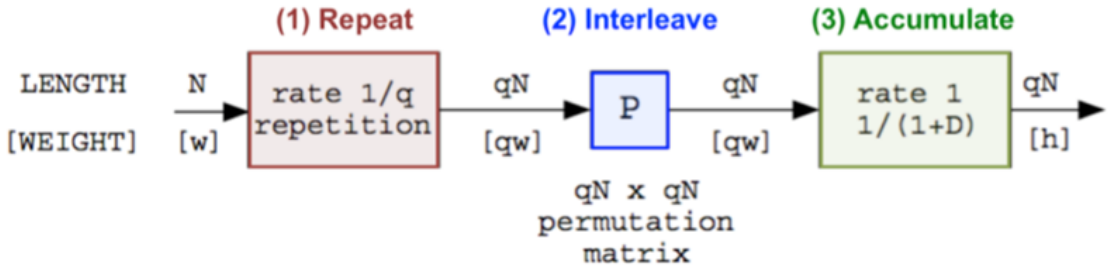
b) *"a first coder having an input configured to receive a stream of bits, said first coder operative to repeat said stream of bits irregularly and scramble the repeated bits"*

340. Luby97 teaches "a first coder having an input configured to receive a stream of bits." Luby97 teaches a coder configured to receive a "stream" of bits. (*See* Ex. 1011, p. 150). As explained above, one of ordinary skill in the art would

know that the "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

341. The Frey Slides teach a first coder operative to repeat bits irregularly and scramble the repeated bits. As explained above in the context of claim 1, the Frey Slides teach a first coder that irregularly repeats bits. The Frey Slides also teach that the irregularly repeated bits are passed as input to a permuter, which scrambles the repeated bits. Together, these two stages of the Frey encoder meet the above limitation of claim 15.

342. Although Divsalar does not teach repeating bits "irregularly," it meets all the other requirements of this claim limitation when modified with the "stream" teachings of Luby97, as explained above. Figure 3 of Divsalar, annotated below, illustrates an encoder with three stages:



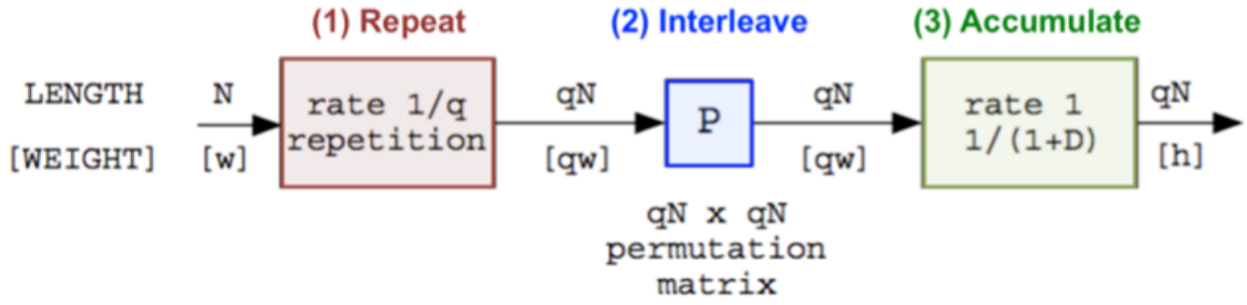
Ex. 1003, Figure 3 (annotations added)

343. Together, stages (1) and (2) of the above encoder – the "Repeat" and "Interleave" stages – modified to incorporate the "stream" teachings of Luby97 – constitute "a first coder having an input configured to receive a stream of bits, said

first coder operative to repeat said stream of bits and scramble the repeated bits." The "Repeat" stage of Divsalar does not repeat the bits *irregularly*, as claim 15 requires, but as explained above with reference to claim 1, it would have been obvious to replace the "Repeat" block above with the irregular repeater of the Frey Slides and the "stream" teachings of Luby97, resulting in a coder that meets all the requirements of the limitation above.

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

344. As explained above in connection with claim 1, Divsalar teaches "a second coder operative to further encode bits output from the first coder at a rate within 10% of one." The second coder taught by Divsalar is an accumulator, as shown in Figure 3 of Divsalar, reproduced below:

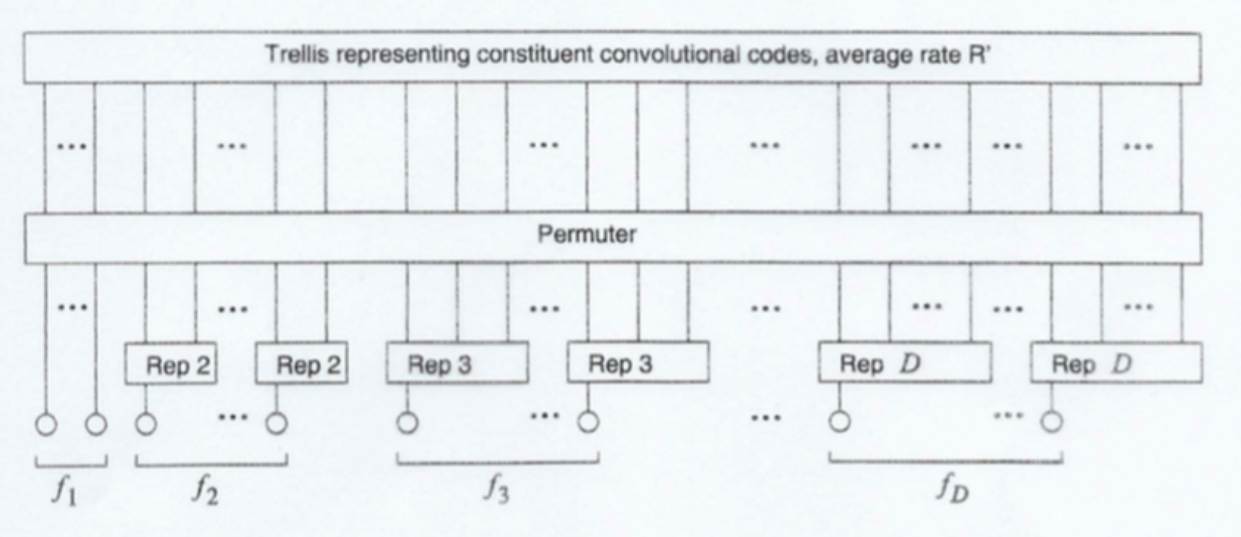


Ex. 1003, Figure 3 (annotations added)

345. An accumulator, as shown in the figure above, produces qN bits of output for every qN bits of input, resulting in a coding rate of $qN/qN = 1$. Accordingly, the second encoder of Divsalar has a rate "within 10% of one" (in fact, its rate is 1.) (*See also id.*, p. 5, explaining that "[a]n information block of

length N is repeated q times, scrambled by an interleaver of size qN, and then encoded by a rate 1 accumulator.")

346. The second coder taught in 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 reproduced below:



Ex. 1013, p. 5

347. The convolutional coder of the Frey Slides is "a second coder operative to further encode bits output from the first coder," as required by claim 15, but it does not have the recited "rate within 10% of one."

348. However, as explained above, one of ordinary skill in the art would have had the motivation to substitute the convolutional coder of the Frey Slides with the rate-1 accumulator of Divsalar, yielding a second encoder that meets the above limitation of claim 11.

3. *Claim 16*

349. Claim 16 depends from independent claim 15, and it also 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."

350. 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." 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. 1011, p. 150, emphasis added.) As explained above, one of ordinary skill in the art would have had the motivation to incorporate these "stream" teachings of Luby97 into the coders taught by Divsalar.

351. Accordingly, claim 16 is obvious in view of Divsalar, the Frey Slides, and Luby97.

4. *Claim 17*

352. Claim 17 depends from claim 16. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches all the claim limitations in claim 16.

353. Claim 17 requires that "the second coder comprises a recursive convolutional encoder with a transfer function of 1/(l+D.)" Divsalar teaches this additional limitation. As explained above, the second encoder of Divsalar is an accumulator. (*See* Ex. 1003, p. 5, "[a]n information block of length N is repeated *q* times, scrambled by an interleaver of size *qN*, *and then encoded by a rate 1 accumulator*," emphasis added.) Divsalar also 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.)

354. Accordingly, Divsalar teaches a second encoder that is a "recursive convolutional encoder with a transfer function of 1/(l+D)," as claim 17 requires.

5. *Claim 20*

355. Claim 20 depends from independent claim 15. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations in claim 15.

356. Claim 20 requires that the first coder comprise "a low-density generator matrix coder." 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. Therefore, both the Frey Slides and Divsalar teach the additional requirement of claim 20.

6. *Claim 21*

357. Dependent claim 21 depends from independent claim 15. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations of claim 15.

358. Claim 21 also requires that the second encoder comprise "a rate 1 linear encoder." As explained above for claim 2, Divsalar teaches the limitation of claim 20.

359. Accordingly, claim 21 is obvious over Divsalar, the Frey Slides, and Luby97.

7. *Claim 22*

360. Dependent claim 22 depends from claim 21. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations of claim 21.

361. Claim 22 additionally requires that "the second coder comprise[] an accumulator." As explained above for claim 4, Divsalar teaches this limitation.

362. Accordingly, claim 22 is obvious over Divsalar, the Frey Slides, and Luby97.

8. *Claims 24 and 33*

363. Claims 24 and 33 depend from independent claims 15 and 25, respectively, and also 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 (an accumulator) that has a rate exactly equal to 1 and therefore has a rate that is "within 1% of one," as claims 24 and 33 require.

364. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations of claims 15 and 25, from which claims 24 and 33 depend. Accordingly, for the reasons given above, claims 24 and 33 are obvious over Divsalar, the Frey Slides, and Luby97.

9. *Claim 25*

365. The combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations of independent claim 25.

a) *"A coding system comprising"*

366. To the extent the preamble is limiting, it is taught in both the Frey Slides and Divsalar, as explained above in the context of claims 1, 11, and 15.

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"*

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

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"*

368. As explained above in the context of claim 15, Divsalar teaches this limitation.

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

369. 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. 1003, p. 9, emphasis added; *see also id.*, Figure 5.) One of ordinary skill in the art would know 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.

10. *Claim 26*

370. Claim 26 depends from claim 25, and also 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."

371. 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. 1011, p. 150, emphasis added.) As explained above, one of ordinary skill in the art would know that these "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

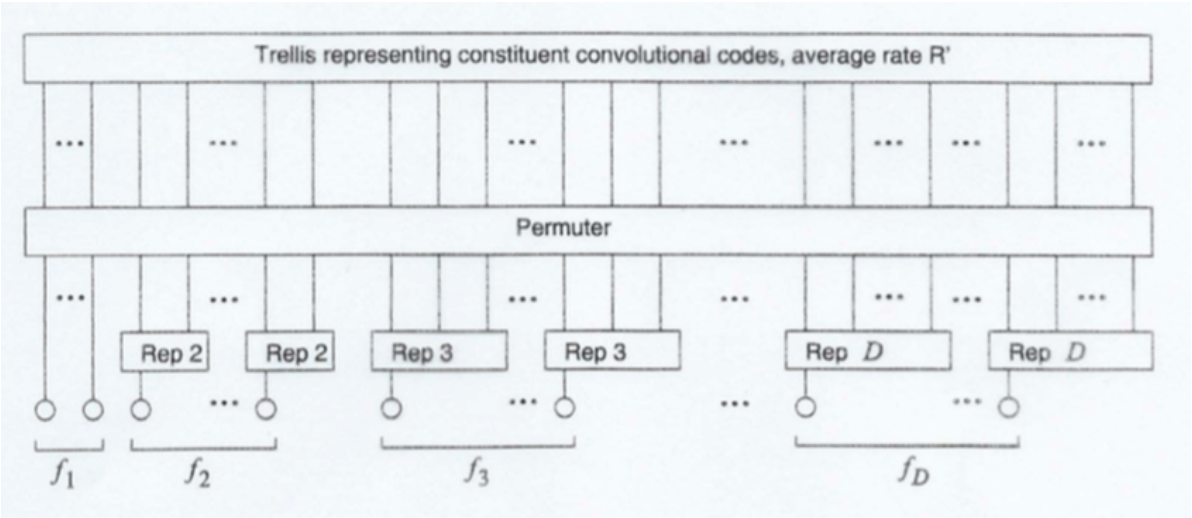
372. Additionally, as explained above in the context of claim 6, the Frey Slides teach "repeat[ing] bits in the data block a different number of times according to a selected degree profile." Claim 26 is therefore obvious in view of Divsalar, the Frey Slides, and Luby97.

11. *Claims 19 and 27*

373. Claims 19 and 27 depend from claims 15 and 26, respectively, and both also require that the first coder comprise "an interleaver." Claim 19 also requires that the first coder comprise "a repeater having a variable rate." Divsalar and the Frey Slides both teach the claimed "interleaver" and the Frey Slides teach the "repeater having a variable rate."

374. 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, as it repeats different codeword bits different number of times. (Ex. 1013, p. 5). Accordingly, the Frey Slides teach a first coder comprising "a repeater having a variable rate," as claim 19 requires.

375. The Frey Slides also teach a first coder that comprises an "interleaver":



Ex. 1013, p. 5

376. As explained above with reference to claim 1, the "permuter" shown in the above figure is "an interleaver," as claims 19 and 27 require. In addition, 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 additional limitations of claims 19 and 27.

377. 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, the Frey Slides, and Luby97.

12. *Claim 28*

378. Claim 28 depends from independent claim 25. As explained above, the combination of Divsalar and the Frey Slides teaches all the limitations in claim 25.

379. Claim 28 requires that the first coder comprise "a low-density generator matrix coder." 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. Therefore, Divsalar and the Frey Slides teach the additional requirement imposed by claim 28.

13. *Claim 29*

380. Dependent claim 29 depends from independent claim 25. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 teaches all the limitations of claim 25.

381. Claim 29 requires also that "the second coder comprise[] a rate 1 accumulator." As explained above in the context of claims 2, 4 and 22, the accumulator of Divsalar meets all these additional requirements. Accordingly, claim 29 is obvious over Divsalar, the Frey Slides, and Luby97.

14. *Claim 30*

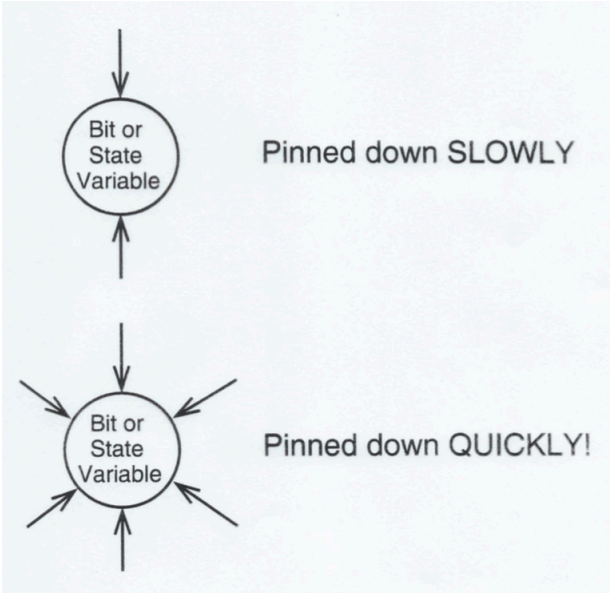
382. Claim 30 depends from claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream using a posterior [sic] decoding techniques." The Frey Slides disclose this additional limitation. The Frey Slides describe a process for decoding irregular turbo codes using "density evolution." (Ex. 1013, p. 13.) A person of ordinary skill in the art would know 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. Accordingly, the decoder in the Frey Slides decodes the encoded data using "a posterior[i] decoding techniques," as required by claim 30.

383. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 discloses all the limitations of claim 25, from which claim 30 depends. Claim 30 is therefore obvious over the Divsalar, the Frey Slides, and Luby97.

15. *Claim 31*

384. Claim 31 depends from claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream based on a Tanner graph representation."

385. 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. 1013, p. 3

386. A person of ordinary skill in the art would know 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 also know 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.

387. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 discloses all the limitations of claim 25, from which claim 31 depends. Claim 31 is obvious over Divsalar, the Frey Slides, and Luby97.

16. *Claim 32*

388. Claim 32 depends from claim 25, and further requires that "the decoder [be] operative to decode the encoded data stream in linear time." Luby97 teaches this additional limitation. In particular, Luby97 discloses "randomized constructions of *linear-time* encodable and *decodable* codes." (Ex. 1011, 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 the Frey slides, as decoding in linear time makes the decoding process faster and more efficient.

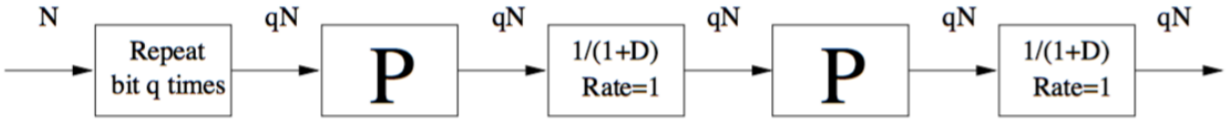
389. Divsalar also teaches this limitation. As Divsalar points out, "the complexity of [maximum likelihood] decoding of RA codes, like that of all turbo-like codes, is prohibitively large. But an important feature of turbo-like codes is the

availability of a simple iterative, message passing decoding algorithm that approximates ML decoding." (Ex. 1003, 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.

I. Ground 9: Claim 10 is Obvious in View of Divsalar and the Frey Slides, Also in View of Pfister

390. Claim 10 depends from claim 1. Claim 10 also requires that the second encoding "utilize[] two accumulators".

391. 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. 1005, p. 1 "Repeat-Accumulate-Accumulate (RAA) Codes.") In Pfister's RAA codes, the outer code comprises a repeater, which is followed by two accumulators with Rate 1. Specifically, page 20 of Pfister illustrates an RAA coder that uses two rate-1 accumulators, as shown below:



Ex. 1005, p. 20

392. 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"). Accordingly, 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 conclude that adding an extra accumulator improves the codes' performance. (*See id.*, p. 21.)

393. A person of ordinary skill in the art would have had the motivation to incorporate the two-accumulator design of Pfister into Divsalar. Both references are directed to the same field—the field of error-correcting codes. Both are directed to error-correcting codes containing serial concatenation of rate-1 codes. (*See, e.g.*, Ex. 1003, Figure 3; *see also* Ex. 1005, pp. 1-2.)

394. Also, incorporating a second accumulator into Divsalar's RA codes would have been obvious for one of ordinary skill in the art. As explained above, Pfister's RAA codes are simply RA codes with additional accumulators added to the end of the encoding chain in series. (*See* Ex. 1005, Figure 2.)

395. As explained above, the combination of Divsalar and the Frey Slides discloses all the limitations of claim 1, from which claim 10 depends. Claim 10 is therefore obvious over Divsalar and the Frey Slides, also in view of Pfister.

J. Ground 10: Claim 23 is Obvious in View of Divsalar, the Frey Slides, and Luby97, also in View of Pfister

396. Claim 23 depends from claim 22. Claim 23 requires that the second coder comprise "a second accumulator". As described above in the context of claim 10, Pfister teaches this limitation.

397. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 discloses all the limitations of claim 22, from which claim 23 depends. Claim 23 is therefore obvious over Divsalar, the Frey Slides, and Luby97, also in view of Pfister.

K. Ground 11: Claims 1-8 and 11-14 are Obvious over Divsalar in view of Luby

398. Claims 1-8 and 11-14 are obvious over the combination of Divsalar in view of Luby (while Divsalar was before the Patent Office during prosecution of the '710 patent, Luby was not).

1. *Motivation to Combine Divsalar and Luby*

399. One of ordinary skill in the art would have had the motivation to incorporate the irregularity taught by Luby into the RA codes of Divsalar.

a) *Luby's Impact in the Field of Coding Theory*

400. As explained above, Luby represented a significant advance in the design of error-correcting codes. Not only was Luby the first to describe irregularity, a novel technique for constructing codes, but it concluded that irregularity could be used to design codes that are "*far superior to any regular code*." (Ex. 1004, p. 257, emphasis added.) Luby even asserts that irregular codes provide "better error-correcting capabilities than *possible* with regular codes" (Ex. 1004, 249, emphasis added), which implies that irregular codes are not only better than any regular code that exists, but are also better than any regular codes that *could* exist. Based on these claims, those of ordinary skill in the art would have had the motivation to incorporate Luby's teachings into a wide range of known coding methods and systems, including the RA codes of Divsalar.

401. Frey is direct evidence of this phenomenon. Frey, appropriately titled "Irregular Turbocodes," incorporated irregularity into a class of known codes (in this case, turbo codes). (Ex. 1002) In fact, the first sentence of Frey credits the Luby paper for providing the motivation to study irregular turbo codes: "Recent work on irregular Gallager codes (low density parity check codes) has shown that by making the codeword bits participate in varying numbers of parity check equations, significant coding gains can be achieved [1, 3]" (Ex. 1002, p. 1) (Reference [1] mentioned in this passage is the Luby paper (*Id.*, p. 7).) Frey

demonstrates that researchers were specifically motivated to incorporate the teachings of Luby into known turbo-like codes, like the RA codes of Divsalar.

402. Luby describes irregularity in the context of various different types of error-correcting codes, including Gallager codes (LDPC codes) and turbo codes. (*See* Ex. 1004, p. 250.) Luby's statement that irregular codes are "far superior to *any* regular code" invited incorporating irregularity into a very broad range of existing codes, including the "turbo-like" RA codes of Divsalar.

403. After Frey and MacKay found that irregularity improved turbo codes, it would have become even more obvious for a person of ordinary skill to incorporate the irregularity of Luby into the codes of Divsalar. By then, irregularity had been shown to dramatically improve the performance of two distinct groups of codes: LDPC codes (by Luby) and turbo codes (by Frey). Following the success of Frey, a person of ordinary skill would have known that incorporating irregularity into RA codes would be even more likely to produce favorable results.

404. In fact, Aamod Khandekar, one of the inventors named on the '710 patent, wrote in his Ph.D. thesis that "Luby et al. also introduced the concept of irregularity, which seems to provide hope of operating arbitrarily close to channel capacity in a practical manner, on a wide class of channel models" (Ex. 1018, p. 2.) Khandekar hails "the introduction of irregular LDPC codes by Luby et al." as a "major breakthrough" and admits that IRA codes were simply an application of

Luby's "concept of irregularity to the ensemble of RA codes". (*Id.*, pp. 46-47; *see also id.*, p. 51.)

b) *Combining Luby with Divsalar for Research*

405. Separately, one of ordinary skill would have known that incorporating irregularity into RA codes would produce codes that could be used to study irregularity, and ultimately to know why it improved decoding performance. As explained above, Divsalar's original motivation for studying RA codes was the fact that they "are *simple enough* so that we can prove the IGE conjecture." (Ex. 1003, p. 5, emphasis added.) One of ordinary skill would have known that the simplicity of RA codes, which motivated Divsalar to study them in the first place, could also be leveraged to help study irregularity.

c) *Ease of Implementation*

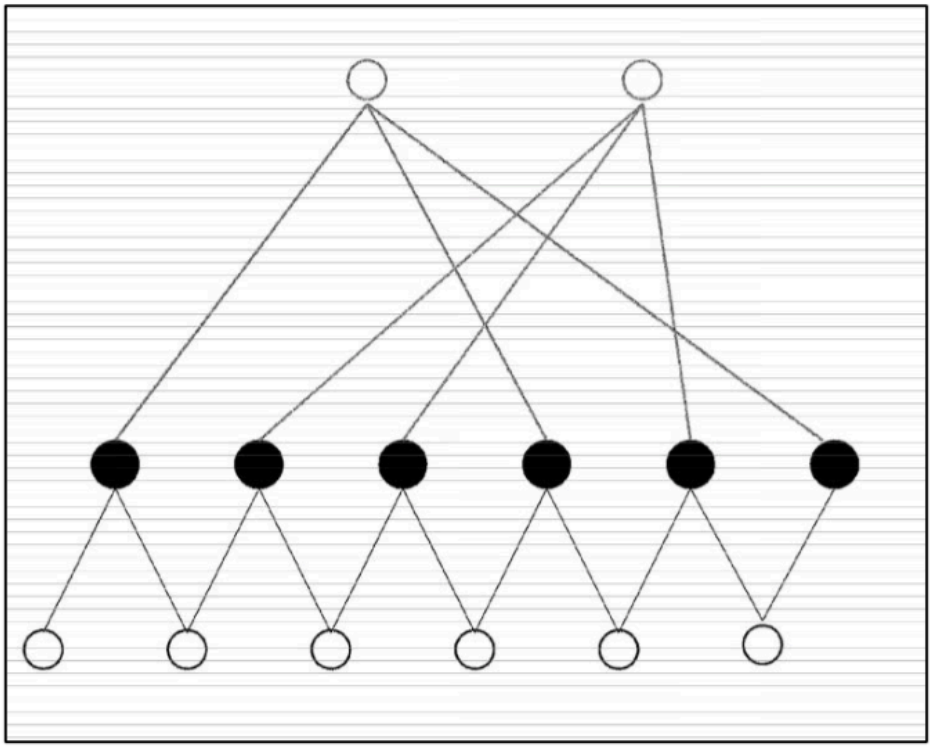
406. Additionally, incorporating irregularity into Divsalar's RA codes would have been a simple matter. This could be accomplished in a number of ways, but the simplest would have been to modify the repeater taught by Divsalar to make it repeat some bits more than other bits, resulting in an *irregular* repeater. This modification would not require any change to the other components of the encoder taught by Divsalar (the interleaver and the accumulator).

407. The modification to the repeater would not have presented any significant technical challenges to a person of ordinary skill. Repeaters – whether

regular or irregular – are conventional components that have been used for decades in a wide variety of digital electronics. Modifying an existing encoder to repeat bits irregularly, or replacing a regular repeater with an irregular one, would have been a routine matter for one of ordinary skill in the art.

408. Also, incorporating irregularity into RA codes would also have required very little modification to the decoder. The algorithms for decoding RA codes would have been suitable for irregular RA codes as well. Modifying the message passing decoder would have been a simple matter for one of ordinary skill, requiring only reprogramming/reconfiguring the decoder to use the Tanner graph of the updated code. This would involve modifying the decoder algorithm to reflect the irregularity of the Tanner graph edges, to ensure that variable nodes communicate with the right check nodes during the decoding process

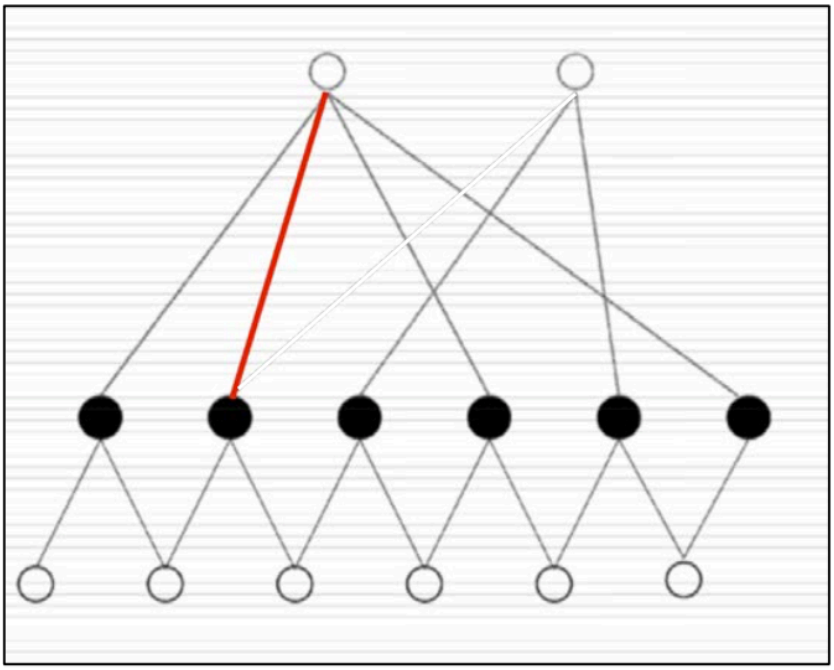
409. Even this reconfiguration process would have been simple for one of ordinary skill, because irregular codes are so similar to RA codes that the Tanner graph representing any RA code can be modified to represent an irregular code by the addition (or removal) of a single edge. For example, the figure below, taken from a presentation given by Aamod Khandekar, one of the named inventors on the patent at issue, corresponds to an RA code that encodes two information bits at a time:



Tanner Graph of an RA Code (Ex. 1017, p. 65)

410. In the code corresponding to the graph above, both information bits (represented by the hollow circles at the top) are repeated three times, and the repeated bits are accumulated to generate the parity bits (represented by the hollow circles at the bottom.) (*Cf.* Ex. 1001, Fig. 3.) This graph shows each bit as repeated three times by the three lines (edges) between each bit and the check nodes (represented by the solid circles in the middle).

This RA code can be made irregular by removing one edge from the Tanner graph above and adding another:



Tanner Graph of an IRA Code

The modified Tanner graph above, with one edge removed and one edge added (shown in red), corresponds to an IRA code. This code is identical to the RA code of the previous figure, except that instead of both information bits being repeated three times, one of the information bits is repeated *four* times, while the other information bit is repeated *twice*, yielding an irregular code.

411. The similarity between the two graphs above, one representing an RA code, the other representing an IRA code, illustrates how simple it would have been for one of ordinary skill in the art to incorporate the irregularity of Luby into the RA codes of Divsalar.

412. Luby also uses Tanner graphs to describe its codes. Repeat codes represented by a Tanner graph can be implemented in a number of ways. For

example, these Tanner graphs may be implemented by repeating, or duplicating, the message bits for each check node in which they are used. An alternative way to implement the code represented by a Tanner graph would be to *reuse* the information bits different numbers of times in computing the parity bits, which would not involve repetition. Because Divsalar explicitly teaches the use of repetition, it would have been obvious to a person of ordinary skill in the art to implement the Tanner graphs of Luby using repetition as well.

413. Finally, as explained in the remainder of this section, the similarity and combinability of Divsalar and Luby is evidenced by the number of claim limitations they both teach.

2. *Claim 1*

414. As explained above, Divsalar teaches all the limitations recited by claim 1, except for the irregularity of the "first encoding" step. As also explained above, incorporating irregularity, as taught by Luby, into Divsalar would have been obvious.

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

415. To the extent that the preamble limits the claim, it is taught by Divsalar, as explained above.

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

416. Both Divsalar and Luby teach this limitation. As explained above, 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. (*See* Ex. 1003, p. 5, Fig. 3.)

417. Similarly, Luby teaches codes having varying "block lengths" (Ex. 1004, p. 256.) Specifically, Luby describes "experiments on codes of rate 1/2 with *16,000 message bits*" and concludes that "for *block lengths of length 16,000* the codes appear to perform extremely well …." (*Id.*, emphasis added.) In this passage, the term "block length" refers to the number of bits in each block of data that is obtained by the encoder.

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"*

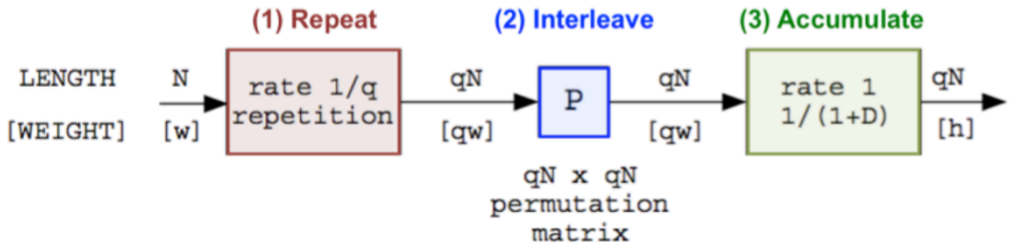
418. As explained above, Divsalar teaches an RA code that uses three steps:

(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, reproduced below:



Ex. 1003, Figure 3 (annotations added)

419. 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.

420. 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" (irregular repetition), Luby teaches these limitations, as explained below.

421. As explained above, the RA encoder of Divsalar, shown in Fig. 3, begins by repeating each data bit *q* times, interleaving the repeated bits, and accumulating the interleaved bits to generate parity bits. Repeat codes, like the RA codes of Divsalar, are represented by Tanner graphs in which the degree of a message node is equal to the number of times the corresponding message bit is repeated. In other words, if a bit is repeated *q* times, the corresponding message node will have degree *q*. (*See,* Ex. 1001, 3:39-43, "in the Tanner graph 300, each of the f_2 information nodes are connected to two check nodes, corresponding to a

repeat of q=2, and each of the f_3 information nodes are connected to three check nodes, corresponding to q=3.") Because the message bits in Divsalar are *all* repeated q times, *every* message node has degree q, yielding a regular code.

422. As explained above, Luby describes how irregularity can be introduced into a code by "varying the degrees of the message nodes" in the Tanner graph. (Ex. 1004, pp. 253-54.) A person of ordinary skill would have known that this could easily be accomplished by varying the number of times each message bit is repeated. For example, if some message bits are repeated q times, while others are repeated p times, the message nodes of the Tanner graph would have varying degrees – some would have degree q, and others would have degree p – yielding an irregular code. Accordingly, a person of ordinary skill in the art would have known that the irregularity of Luby could readily be incorporated into the RA codes of Divsalar by modifying the number of times each message bit is repeated, so that some message bits are repeated more times than others (for example, some q times and others p times).

423. As explained above, this modification would not have been technically challenging to a person of ordinary skill in the art. Also, for the reasons explained above, a person of ordinary skill would have been highly motivated to combine Divsalar with Luby in this way.

424. The RA codes of Divsalar modified to include an irregular repeater would also meet the "partitioning" requirement of claim 1. As explained above, any coding scheme that repeats different information bits different numbers of times will necessarily partition information bits into sub-blocks, 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. Applying different degrees of repetition to different bits is what *defines* the different sub-blocks of the partition.

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

425. Divsalar teaches this limitation, as explained above.

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

426. Divsalar teaches this limitation, as explained above.

f) *Summary*

427. As explained above, the combination of Divsalar and Luby teaches all the limitations of claim 1. Also, for the reasons given above, a person of ordinary skill in the art would have had the motivation to incorporate Luby's irregularity into Divsalar. Accordingly, claim 1 is obvious over the combination of Divsalar in view of Luby.

3. *Claim 2*

428. Dependent claim 2 depends from independent claim 1. Claim 2 additionally requires that the second encoding be performed "via a rate 1 linear transformation."

429. As explained above, the accumulator of Divsalar meets this additional requirement. Also, as explained above, the combination of Divsalar and Luby teaches all the limitations of claim 1, from which claim 2 depends. Therefore, claim 2 is obvious over the combination of Divsalar and Luby.

4. *Claim 3*

430. Claim 3 depends from claim 1, and also 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."

431. The combination of Divsalar and Luby also teaches "first coder with a variable rate less than one." As explained above, modifying the RA codes of Divsalar to incorporate irregularity of Luby would result in a first coder that irregularly repeats the information bits (that repeats some information bits more than others.) As explained above, an irregular repeater has a "variable rate" that is less than one. Also, Divsalar's regular repeater is a first coder with a "rate less than one."

432. Finally, as explained above, the accumulator of Divsalar is a "second coder with a rate substantially close to one."

433. As explained above, claim 1 is obvious over the combination of Divsalar and Luby. Accordingly, claim 3 is also obvious over Divsalar in view of Luby.

5. *Claim 4*

434. Claim 4 depends from claim 3, and also requires that "the second coder comprise[] an accumulator." As explained above, Divsalar teaches a second coder that is an accumulator.

6. *Claim 5*

435. Claim 5 depends from claim 4, and also requires that "the data elements comprise[] bits." As explained above, both Divsalar and Luby teach methods of coding signals in which the "data elements" comprise bits.

7. *Claim 6*

436. Claim 6 depends from claim 5. As explained above, the combination of Divsalar and Luby teaches all the limitations of claim 5.

437. Claim 6 recites limitations relating to irregular repetition, which would have been obvious over the combination of Luby and Divsalar, as explained above.

438. Specifically, Claim 6 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."

439. The '710 patent specification 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. 1001, 2:55-58, emphasis added.)

440. Luby also teaches designing irregular codes according to a "degree profile." Luby specifies codes using "sequences $(\lambda_1, \lambda_2, \ldots, \lambda_{dl})$ and $(\rho_1, \rho_1, \ldots, \rho_{dr})$, where λ_i (ρ_i) is the fraction of edges with left (right) degree i." (Ex. 1004, p. 254.) Like the "degree profile" described in the '710 patent, these sequences represent fractions of the information bits that have various degrees. Accordingly, claim 6 is obvious over Divsalar in view of Luby.

8. *Claim 7*

441. Claim 7 depends from independent claim 4. As explained above, the combination of Divsalar and Luby teaches all the limitations in claim 4.

442. Claim 7 requires that the first coder comprise "a low-density generator matrix coder". As explained above, Divsalar teaches this limitation.

9. *Claim 8*

443. Claim 8 depends from claim 1. As explained above, Luby in combination with Divsalar teaches all the limitations of claim 1.

444. Claim 8 also requires that "the second encoding uses a transfer function of 1/(l+D)." As explained, Divsalar teaches this limitation.

10. *Claim 11*

445. As described in detail below, the combination of Divsalar and Luby teaches all the limitations of claim 11.

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

446. To the extent the preamble is limiting, it is taught by Divsalar, as explained above in the context of claim 1.

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

447. Both Divsalar and Luby teach receiving a block of data to be encoded, as explained above in connection with claim 1.

448. Luby teaches also that the data block includes "a plurality of bits." (*See, e.g.*, Ex. 1004, p. 256, "We first describe experiments on codes of rate 1/2 with 16,000 *message bits* ….) Divsalar also teaches this limitation, as explained above in the context of claim 1.

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"*

449. As explained above in connection with claim 1, the combination of Divsalar and Luby teaches repeating some bits more than other bits. Accordingly, this combination teaches repeating at least two bits in a data block a different number of times from one another.

450. Divsalar and Luby also both teach that "each bit in the data block is repeated." Figure 3 of Divsalar, reproduced above, illustrates that each bit is repeated *q* times. Luby teaches irregular codes in which the "minimum degree" of the message nodes is "at least five." (Ex. 1004, p. 255.)

451. As explained above, these correspond to codes in which each message bit is repeated at least five times. Luby describes codes using bipartite graphs called Tanner graphs. The code represented by a Tanner graph can be implemented in a number of ways. For example, these Tanner graphs may be implemented by repeating, or duplicating, the message bits for each check node in which they are used. An alternative way to implement the code represented by a Tanner graph would be to *reuse* the information bits different numbers of times in computing the parity bits, which would not involve repetition. Because Divsalar explicitly teaches the use of repetition, it would have been obvious to a person of ordinary skill in the art to implement the Tanner graphs of Luby using repetition as well.

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

452. As explained above with reference to claims 1 and 4, Divsalar teaches "second encoding the first encoded data block" using an accumulator.

11. *Claim 12*

453. Dependent claim 12 depends from independent claim 11. Claim 12 additionally requires that the second encoding be performed "via a rate 1 linear transformation."

454. As explained above for claim 2, the accumulator of Divsalar meets this additional requirement. Also, as explained above, the combination of Divsalar and Luby teaches all the limitations of claim 11, from which claim 12 depends. Therefore, claim 12 is obvious over the combination of Divsalar and Luby.

12. *Claim 13*

455. Claim 13 depends from independent claim 11. As explained above, the combination of Divsalar and Luby teaches all the limitations in claim 11. Claim 13 requires that the first coding be "via a low-density generator matrix transformation". As explained above for claim 7, Divsalar teaches this limitation.

13. *Claim 14*

456. Claim 14 depends from independent claim 11, and also requires that "the signal to be encoded comprise[] a plurality of data blocks of fixed size." As explained above, both Luby and Divsalar teach this limitation.

L. Ground 12: Claims 15-17, 19-22, and 24-33 Are Obvious in view of Divsalar and Luby, also in view of Luby97

457. Claims 15-17, 19-22, and 24-33 are obvious over the combination of Divsalar and Luby, also in view of Luby97 (while Divsalar was before the Patent Office during prosecution of the '710 patent, Luby and Luby97 were not).

1. *Motivations to combine Divsalar and Luby with Luby97*

458. As explained above, one of ordinary skill in the art would have had the motivation to incorporate the irregularity of Luby into Divsalar. One of ordinary skill in the art would also have had the motivation to combine Divsalar with Luby97, as both relate to error-correcting codes and the performance thereof. (*See generally* Exs. 1003, 1011.) Specifically, a person of ordinary skill in the art would have had the motivation 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.

459. As explained above, Luby97 describes receiving data to be encoded in a *stream* of data symbols (which could be, for example, bits), where the "*stream* of data symbols [] is partitioned and transmitted in logical units of *blocks*." (Ex. 1011, p. 150, emphasis added.) One of ordinary skill in the art would have known 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 (a "block" of

information bits), which are then encoded all at once, producing a codeword of fixed size.

460. 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 Luby alone, it would have been obvious to use Luby97's teaching of "stream" in Divsalar and/or Divsalar in view of Luby, to make the encoder capable of receiving and processing "streams" as opposed to just blocks. Accordingly, it would have been obvious to a person of ordinary skill to incorporate the "stream" teachings of Luby97 above into the encoder of Divsalar.

2. *Claim 15*

461. As described in detail below, the combination of Divsalar in view of Luby and further in view of Luby97 teaches all the limitations of claim 15.

a) *"A coder comprising"*

462. To the extent the preamble is limiting, it is taught by Divsalar, as explained above with reference to claim 1.

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"*

463. Luby97 teaches "a first coder having an input configured to receive a stream of bits." Luby97 teaches a coder configured to receive a "stream" of bits.

(*See* Ex. 1011, p. 150). As explained above, one of ordinary skill in the art would know that the "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

464. Additionally, as explained above in the context of claim 1, the combination of Divsalar and Luby teaches a first coder operative to repeat bits "irregularly" and to "scramble" (meaning permute) the repeated bits.

465. Therefore, this limitation is taught by the combination of Divsalar, Luby, and Luby97.

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

466. As explained above for claims 1 and 2, Divsalar teaches this limitation.

3. *Claim 16*

467. Claim 16 depends from claim 15, and also 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."

468. 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. 1011, p. 150, emphasis

added.) As explained above, one of ordinary skill in the art would know that, to the extent they are not already obvious in view of Divsalar and Luby, these "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

469. Also, as explained above in the context of claim 1, the combination of Divsalar and Luby teaches that the first coder partitions (or "apportions") the data block into a plurality of sub-blocks, where bits in different sub-blocks are repeated a different number of times. Accordingly, claim 16 is obvious over Divsalar in view of Luby, and further in view of Luby97.

4. *Claim 17*

470. Claim 17 depends from claim 16, and also requires "the second coder comprises a recursive convolutional encoder with a transfer function of 1/(l+D)." As explained for claims 1, 4 and 8, Divsalar teaches this limitation. Also, as explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations of claim 16, from which claim 17 depends. Therefore, claim 17 is obvious over the combination of Divsalar, Luby, and Luby97.

5. *Claim 20*

471. Claim 20 depends from independent claim 15, and also requires that the first coder comprise "a low-density generator matrix coder". As explained above for claim 20, Divsalar teaches this limitation.

472. Additionally, as explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations of claim 15, from which claim 20 depends. Therefore, claim 20 is obvious over the combination of Divsalar, Luby, and Luby97.

6. *Claim 21*

473. Dependent claim 21 depends from independent claim 15, and also requires that the second encoder comprise "a rate 1 linear encoder."

As explained above for claims 1 and 2, the accumulator of Divsalar meets this additional requirement. Also, as explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations of claim 15, from which claim 21 depends. Therefore, claim 21 is obvious over the combination of Divsalar, Luby, and Luby97.

7. *Claim 22*

474. Dependent claim 22 depends from claim 21, and also requires that "the second coder comprise[] an accumulator."

475. As explained above for claim 4, the accumulator of Divsalar meets this additional requirement. Also, as explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations of claim 21, from which claim 22 depends. Therefore, claim 22 is obvious over the combination of Divsalar, Luby, and Luby97.

8. *Claim 25*

476. As described in detail below, the combination of Divsalar in view of Luby, and further in view of Luby97 teaches all the limitations of claim 25.

a) *"A coding system comprising"*

477. To the extent the preamble is limiting, it is taught by Divsalar, as explained above in the context of claim 15.

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"*

478. As explained above in the context of claim 15, this limitation would have been obvious over Divsalar in view of Luby and Luby97.

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"*

479. As explained above in the context of claim 15, Divsalar teaches this limitation.

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

480. Divsalar teaches this limitation, as explained above.

481. Luby also teaches a decoder operative to decode the encoded data stream "using an iterative decoding technique." Specifically, Luby teaches that "[t]he decoding process proceeds in *rounds*, where in each round first the message nodes send each incident check node a single bit and then the check nodes send each incident message node a single bit." (Ex. 1004, p. 250, emphasis in original.)

The process "continues until the proposed value for the message nodes satisfy all the check nodes …" (*Id.*, pp. 251-52.)

482. A person of ordinary skill would have known that a decoding algorithm that repeats a mathematical or computational process multiple times (for example, the "rounds" of the Luby algorithm) is an "iterative" decoding technique. One of ordinary skill would have also known that Luby's algorithm is an example of "belief propagation," a technique that was well known in the relevant field. In fact, Luby explicitly references "belief propagation," and includes a reference to Frey & Kschischang, "Probability Propagation and *Iterative Decoding*," for additional information. (*Id.*, pp. 250, 257, emphasis added.)

9. *Claim 26*

483. Claim 26 depends from claim 25, and also 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."

484. 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. 1011, p. 150, emphasis added.) As explained above, one of ordinary skill in the art would know that, to the

extent they are not already obvious in view of Divsalar, these "stream" teachings of Luby97 can be incorporated into the coders taught by Divsalar.

485. Also, as explained above in the context of claim 6, Luby teaches "repeat[ing] bits in the data block a different number of times according to a selected degree profile." Claim 26 is therefore obvious in view of Divsalar, Luby, and Luby97.

10. *Claims 19 and 27*

486. Claims 19 and 27 depend from claims 15 and 26, respectively, and both also require that the first coder comprise "an interleaver." Claim 19 also requires that the first coder comprise "a repeater having a variable rate."

487. As explained above, the combination of Divsalar and Luby meets all these additional limitations. Specifically, the RA codes of Divsalar, modified according to the teachings of Luby to include an irregular repeater, include a "repeater having a variable rate," as described above for claims 1 and 3. Divsalar also teaches a first coder comprising "an interleaver," as explained above for claim 1.

488. Accordingly, Divsalar and Luby teach these additional limitations of claims 19 and 27. As explained above, the combination of Divsalar, and Luby 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, Luby, and Luby97.

11. *Claim 28*

489. Claim 28 depends from independent claim 25. As explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations in claim 25. Claim 28 requires that the first coder comprise "a low-density generator matrix coder". As explained above for claim 13, Divsalar teaches this limitation.

12. *Claim 29*

490. Dependent claim 29 depends from independent claim 25, and also requires that "the second coder comprise[] a rate 1 accumulator."

491. As explained above for claims 2, 4 and 21, the accumulator of Divsalar meets this additional requirement. Also, as explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations of claim 25, from which claim 29 depends. Therefore, claim 29 is obvious over this combination as well.

13. *Claim 30*

492. Claim 30 depends from claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream using a posterior [sic] decoding techniques." As described above, Luby teaches a "round"-based decoding algorithm, in which the correct values of the message bits are determined by iteratively passing messages among the information nodes and check nodes of a bipartite graph. A person of ordinary skill in the art would have known that this

decoding 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. Accordingly, the decoder in Luby decodes the encoded data using "a posterior[i] decoding techniques," as claim 30 requires.

493. As explained above, the combination of Divsalar, the Frey Slides, and Luby97 discloses all the limitations of claim 25, from which claim 30 depends. Claim 30 is therefore obvious over the Divsalar, the Frey Slides, and Luby97.

14. *Claim 31*

494. Claim 31 depends from independent claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream based on a Tanner graph representation."

495. Luby teaches this additional limitation. Specifically, Luby discloses a decoding technique that is based on "bipartite graphs." (*See, e.g.*, Ex. 1004, Abstract ("[W]e consider error-correcting codes based on random irregular *bipartite graphs*,"emphasis added.) Luby describes "bipartite graphs" as follows:

In the following we refer to the nodes on the left and right sides of a bipartite graph as its *message* nodes and *check* nodes respectively. A bipartite graph with n nodes on the left and r nodes on the right gives rise to a linear code of dimension $k \geq n - r$ and block length n in the following way: the bits of a codeword are indexed by the message nodes. A binary vector $\mathbf{x} = (x_1, \ldots, x_n)$ is a codeword if and only if $H\mathbf{x} = 0$, where H is the $r \times n$ incidence matrix of the graph whose rows are indexed by the check nodes and whose columns are indexed by the message nodes. In other words, $(x_1, \ldots, x_n)$ is a codeword if and only if for each check node the exclusive-or of its incident message nodes is zero.

Ex. 1004, p. 250

496. A person of ordinary skill in the art would know that the "bipartite graphs" of Luby are "Tanner graphs," as that term is used in the '710 patent. Both are bipartite. (*See* Ex. 1001, 3:27-28, "The set of parity checks may be represented in a *bipartite graph*, called the Tanner graph," emphasis added.) Both contain message nodes and check nodes. (*See* Ex. 1001, 3:31-32, "The Tanner graph includes two kinds of nodes: variable nodes (open circles) and check nodes (filled circles).") And both use check nodes to represent parity constraints (*See* Ex. 1001, 3:65-4:1 "The value of a parity bit is determined uniquely by the condition that the mod-2 sum of the values of the variable nodes connected to each of the check nodes 304 is zero.")

497. Similarly, a person of ordinary skill in the art would know that the decoding procedure described by Luby is "based on a Tanner graph." The '710

patent describes decoding "based on a Tanner graph" as an iterative process in which, "in each iteration, every variable/check node receives messages from its neighbors, and sends back updated messages. Decoding is terminated after a fixed number of iterations or detecting that all the constraints are satisfied." (Ex. 1001, 5:40-45.)

498. Luby describes the same general decoding algorithm, consisting of a series of iterations (or "rounds") in which each variable node sends messages to the check nodes to which it is connected and receives reply messages in return (*See* Ex. 1004, p. 250, "The decoding process proceeds in *rounds*, where in each round first the message nodes send each incident check node a single bit and then the check nodes send each incident message node a single bit.") Also, like the algorithm described in the '710 patent, the algorithm of Luby continues to iterate until the parity-check constraints are satisfied. (*See id.*, pp. 250-251, "The algorithm continues until the proposed values for the message satisfy all the check nodes, at which point the algorithm terminates with the belief that it has successfully decoded the message.")

499. Therefore, Luby teaches decoding the encoded data stream "based on a Tanner graph representation," as claim 31 requires. As explained above, the combination of Divsalar, Luby, and Luby97 discloses each limitation of claim 25,

from which claim 31 depends. Therefore, claim 31 is obvious over Divsalar, Luby, and Luby97.

15. *Claim 32*

500. Claim 32 depends from claim 25, and also requires that "the decoder [be] operative to decode the encoded data stream in linear time." Luby teaches this additional limitation. It discloses a "sequential algorithm" for decoding, where "[t]he overall decoding time is linear." (Ex. 1004, p. 255; *see also id.*, p. 249 ("We developed tools … for designing *linear time irregular codes* with better error-correcting capabilities than possible with regular codes.")

501. 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. 1003, 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 Luby's decoding techniques into Divsalar, which are explicitly taught to be linear time.

16. *Claims 24 and 33*

502. Claims 24 and 33 depend from independent claims 15 and 25, respectively, and also require that the second coder comprises a coder "operative to further encode bits output from the first coder at a rate within 1% of one." As explained above for claims 2, 4, 12, and 21, Divsalar teaches a second coder that is an accumulator, with a rate equal to 1 and therefore is "within 1% of one," as claims 24 and 33 require.

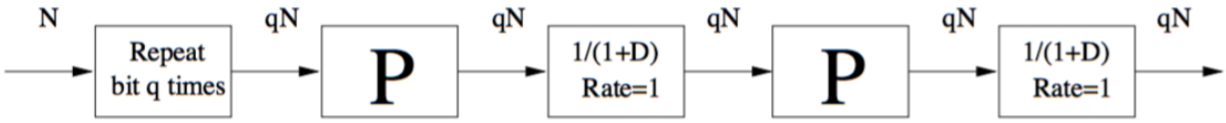
503. Additionally, as explained above, the combination of Divsalar, Luby and Luby97 teaches all the limitations of claim 15 and 25, from which claims 24 and 33 depend. Therefore, claims 24 and 33 are obvious over the combination of Divsalar, Luby, and Luby97.

M. Ground 13: Claim 10 is Obvious Over Divsalar and in View of Luby, and Also in View of Pfister

504. Claim 10 depends from claim 1. Claim 10 also requires that the second encoding "utilize[] two accumulators."

505. While Divsalar and Luby 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. 1005, p. 1 "Repeat-Accumulate-Accumulate (RAA) Codes.") In Pfister's RAA codes, the outer code comprises a repeater, which is followed by two accumulators

with Rate 1. Specifically, page 20 of Pfister illustrates an RAA coder that uses two rate-1 accumulators, as shown below:



Ex. 1005, p. 20

506. 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"). Accordingly, 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 conclude that adding an extra accumulator improves the codes' performance. (*See id.*, p. 21.)

507. A person of ordinary skill in the art would have had the motivation to incorporate the two-accumulator design of Pfister into Divsalar. Both references are directed to the same field—the field of error-correcting codes. Both are directed to error-correcting codes containing serial concatenation of rate-1 codes. (*See, e.g.*, Ex. 1003, Figure 3; *see also* Ex. 1005, pp. 1-2.)

508. Also, incorporating a second accumulator into Divsalar's RA codes would have been obvious for one of ordinary skill in the art. As explained above, Pfister's RAA codes are simply RA codes with additional accumulators added to

the end of the encoding chain in series, and Pfister showed that adding the extra accumulator improves performance. (*See* Ex. 1005, p. 21, Figure 2.)

509. As explained above, the combination of Divsalar and the Frey Slides discloses all the limitations of claim 1, from which claim 10 depends. Claim 10 is therefore obvious over Divsalar and Luby, also in view of Pfister.

N. Ground 14: Claim 23 is Obvious Over Divsalar, Luby, and Luby97, also in View of Pfister

510. Claim 23 depends from claim 22. Claim 23 requires that the second coder comprise "a second accumulator." As described above in the context of claim 10, Pfister teaches this limitation.

511. As explained above, the combination of Divsalar, Luby, and Luby97 discloses all the limitations of claim 22, from which claim 23 depends. Claim 23 is therefore obvious over Divsalar, Luby, and Luby97, also in view of Pfister.

X. PUBLICATION STATUS OF LUBY97

512. Luby97 is cited by a number of academic publications that were published before the claimed priority date of the '710 patent, including Mitzenmacher, M. "Studying balanced allocations with differential equations," *Combinatorics, Probability & Computing* 8.5, pp. 473-482, September 1999 (Ex. 1022, p. 482), Barg *et al.*, "Linear-time Binary Codes Correcting Localized Structures," *IEEE Transactions on Information Theory* 45.7, pp. 2545-2550, November 1999 (Ex. 1023, p. 2552), and Shokrollahi, "New Sequences of Linear Time Erasure Codes Approaching the Channel Capacity," *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes Proceedings*, pp. 65-76, 1999 (Ex. 1024, p. 76).

513. Further, as mentioned above, Luby97 was cited in the Ph.D. thesis of Aamod Khandekar, one of the inventors named on the face of the '710 patent.

Khandekar wrote in his Ph.D. thesis that "Luby et al. also introduced the concept of irregularity, which seems to provide hope of operating arbitrarily close to channel capacity in a practical manner, on a wide class of channel models" (Ex. 1018, p. 2.) Khandekar hails "the introduction of irregular LDPC codes by Luby et al." as a "major breakthrough" and admits that IRA codes were simply an application of Luby's "concept of irregularity to the ensemble of RA codes". (*Id.*, pp. 46-47; *see also id.*, p. 51.)

514. I am an academic and work with academic publications and have done so for decades. The fact that multiple different authors cited Luby97 before the priority date of the '710 patent convinces me that Luby97 was published, and generally available to the community of researchers studying error-correcting codes before the priority date of the '710 patent. The fact that Luby97 was identified as a "1997" publication by the inventor, Khandekar, further convinces me that Luby97 is prior art.

XI. AVAILABILITY FOR CROSS-EXAMINATION

515. In signing this declaration, I recognize that the declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for

cross-examination within the United States during the time allotted for cross-examination.

XII. RIGHT TO SUPPLEMENT

516. I reserve the right to supplement my opinions in the future to respond to any arguments that the Patent Owner raises and to take into account new information as it becomes available to me.

XIII. JURAT

517. I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the full knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States code.

Dated: 11/14/16

James A. Davis, Ph.D.
Richmond, Virginia

APPENDIX A

Curriculum Vita

James A. Davis 2214 Grainmill Court Richmond, VA 23233 (804) 289-8094 (W) (804) 747-5858 (H) jdavis@richmond.edu

EDUCATION:

Ph.D. Mathematics, University of Virginia, August, 1987. Dissertation: "Difference sets in abelian 2-groups"; Advisor: Dr. Harold Ward.

B.S. Mathematics with Honors, Magna cum laude, Lafayette College, May, 1983.

EMPLOYMENT:

Professor, University of Richmond, 2001-Present.

Associate Professor, University of Richmond, 1994-2001.

Assistant Professor, University of Richmond, 1988-1994.

Visiting Assistant Professor, Lafayette College, August 1987-1988.

Teaching Assistant, University of Virginia, August 1983-1987.

Summer Actuary Intern, Provident Mutual Life Insurance Company, Summer of 1982.

DESCRIPTION OF RESEARCH:

Use of algebraic and number theoretic tools to construct combinatorial structures in Design Theory and Coding Theory.

UNDERGRADUATE HONORS THESES DIRECTED:

Cameron-Libeler line classes and partial difference sets, Uthaipon Tantipongpipat, 2016.

Partitioning groups with difference sets, Becca Funke, 2016.

Nonexistence of non quadratic Kerdock sets in 6 variables, John Clikeman, 2016.

Difference sets in nonabelian groups of order 256, Taylor Applebaum, 2013.

Boolean functions with high second order nonlinearity, Cornel Bodea, 2010.

Crossover property of the nonperiodic autocorrelation function of quaternary sequences, Michael Pohl, 2007.

Relative difference sets in 2-groups: a group cohomological viewpoint, Brian Wyman, 2005.

Difference sets, symmetric designs, and coding theory: a look at the incidence matrices for several difference sets, Kate Nieswand, 2000.

On some new constructions of difference sets, Sarah Spence, 1997.

Generalizations of the mod4 construction of the Nordstrom-Robinson Code, Tim Frey, 1995.

Topics in cyclotomic and quadratic fields, Pam Mellinger, 1993.

A new view of McFarland difference sets, John Polhill, 1993.

SUMMER RESEARCH PROJECTS DIRECTED: Almost difference sets, David Clayton, University of Richmond Undergraduate Research Committee, Summer 2016.

Kerdock sets and constructions of bent functions via covering EBSs, John Clikeman and David Clayton, Summer 2016.

Covering EBS bent functions in 8 variables, Uthaipon Tantipongpipat, John Clikeman, Becca Funke, Tedi Aliaj, Sam Bell, David Clayton, Sami Malik, Alexandru Pana, University of Richmond Undergraduate Research Committee, Spider Research Fund, and Virginia Foundation for Independent Colleges as sponsors, Summer 2015.

Bent functions in 6 and 8 variables, Uthaipon Tantipongpipat, John Clikeman, Tedi Aliaj, Sam Bell, Hayu Gelaw, University of Richmond Undergraduate Research Committee, Spider Research Fund, and Virginia Foundation for Independent Colleges as sponsors, Summer 2014.

Bent functions and difference sets, Gavin McGrew, Uthaipon Tantipongpitat, Becca Funke, John Clikeman, Kevin Erb, Erin Geoghan, Richuan Hu, Ningxi Wei, Thomas Meyer, Tahseen Rabbani, Daniel Habibi, Apollo Jain, University of Richmond Undergraduate Research Committee and HHMI as sponsors, Summer 2013.

Boolean functions with high second order nonlinearity, Gavin McGrew, Dayton Steele, Katherine Marsten, Xi He, Shalleetta Hicks, Taryn Smith, Zach Davis, Timmy Albright, Irina Kiseeva, Kirstin Ladas, Alex Martin, Kate Pitchford, Summer 2011. Sponsored by the National Science Foundation, the Grainger Foundation, and the University of Richmond Undergraduate Research Committee.

Boolean functions with high second order nonlinearity, Gavin McGrew, Alex Martin, Kate Pitchford, Katherine Utz, Francesco Spadaro, Josh Wilson, Taryn Smith, summer 2010. Sponsored by the National Science Foundation and the University of Richmond Undergraduate Research Committee.

Sequences with the shared autocorrelation property, Corneliu Bodea, Calina Copos, Matt Der, David Oneal, Summer 2008. Sponsored by the National Science Foundation.

Sequences with the shared autocorrelation property, Calina Copos, Matt Der, David Oneal, and Lauren Vanson, Summer 2007. Sponsored by the National Science Foundation.

Reversible difference sets in 2-groups, Ryan Daut, Summer 2005. Sponsored by the National Security Agency.

A search for new partial difference sets, Peter Magliaro and Adam Weaver, Summer 2003. Sponsored by the National Security Agency.

Partial difference sets in new groups, Tiffany Broadbent and Ed Kenney, Summer 2002. Sponsored by the University of Richmond Undergraduate Research Committee, Collaborative Research Grant.

Difference sets in 2-groups and their associated codes,Mohammed Abouzaid and Jamie Bigelow, Summer, 2000. Sponsored by Hewlett-Packard.

Construction of New Asymptotic Classes of Binary Sequences Based on Existing Asymptotic Classes, Anthony Kirilusha and Ganesh Narayanaswamy, summer 1999. Sponsored by Hewlett-Packard.

Intersections of Golay codes with higher order Kerdock codes, Mohammed Abouzaid and Nick Gurski, summer 1999. Sponsored by Hewlett-Packard.

An examination of codewords with optimal merit factor, Mike Cammarano and Anthony Kirilusha, summer, 1998. Sponsored by Hewlett-Packard.

Octary codewords with peak power 3.2^m, Katie Nieswand and Kara Wagner, summer, 1998. Sponsored by Hewlett-Packard.

Using the Quantum Computer to break and Elliptic Curve Cryptosystem, Jodie Eicher and Yaw Opoku, summer, 1997. This won the Student Research Symposium Natural Science Award at the University of Richmond in spring, 1998. Sponsored by Hewlett-Packard.

Power control in Golay cosets, Mike Cammarano and Meredith Walker, summer, 1997. Sponsored by Hewlett-Packard.

Coding theory over Galois Rings, Sarah Spence and Brian McKeever, Summer, 1995. This won a poster presentation at the annual Joint Math meetings of the American Mathematical Society and the Mathematics Association of America in January, 1996.

PH.D. DISSERTATION DIRECTED: Constructing partial difference sets using Galois Rings, John Polhill, University of Virginia, 1999.

GRANTS:

National Security Agency, research award, $55,000, 2015-2017.

National Science Foundation, LURE, 2008-2011, $100,000

National Science Foundation, LURE, 2007-2011, $1,500,000.

National Security Agency Grant, salary, travel expenses, and support for undergraduate research, January 2003-December 2005.

University of Richmond Undergraduate Research Commmittee Collaborative Grant, salary and travel for me and two students, summer 2002.

Hewlett-Packard Grant for Leave of Absence, academic year 2000-2001, paying salary, moving expenses, cost of living expenses, and benefits.

Hewlett-Packard Grant for undergraduate summer research institute, May 1997-August 2000, to fund 4 students full time working on applied problems.

Hewlett-Packard sabbatical Grant, August 1995-August 1996, funded move to England, half salary, other expenses for the academic year.

National Security Agency Young Investigator Grant, June 1995-June 1997, summer salary and travel.

University of Richmond Summer Research Grant, Summer 1994, salary.

National Security Agency Young Investigator Grant, June 1992-June 1994, summer salary and travel.

Hewlett-Packard equipment grant (50 graphing calculators and supporting equipment), summer 1993.

University of Richmond Summer Research Grant, Summer 1989, salary.

CONSULTING:

Consultant for Hughes Corp, May-June 2014, patent dispute vs. Cal Tech.

Testifying Expert for Apple in Patent dispute, October 2011-June 2012. Deposed on April 10, 2012, testified at trial on June 12, 2012. Wrote Markman report (claims constructions), Invalidity report, Rebuttal non-infringement report.

Consultant for Hewlett-Packard, August 1996-August 2000. Worked on patent preparation, standards body submissions, basic research.

GRANTED PATENTS:

K.K. Smith, J. Jedwab, J.A. Davis, D. Banks and S.R. Wyatt, System for error correction coding and decoding, U.S. Patent No. 7418644, 26 Aug 2008.

D. Banks, J. Davis, and J. Jedwab, Identifying uncorrectable codewords in a Reed-Solomon decoder for errors and erasures, US Patent No. 7278086, 2 October 2007.

J. Davis, K. Eldredge, J. Jedwab, G. Seroussi, and K. Smith, MRAM with controller, US Patent No. 7266732, 4 September 2007.

S.M. Brandenberger, T. Munden, J. Jedwab, J. Davis, and D. Banks, System and method for configuring a solid-state storage device with error correction coding, U.S. Patent No. 7,210,077, 24 Apr 2007.

J. Jedwab, J.A. Davis and G. Seroussi, Method for error correction decoding in a magnetoresistive Solid-state storage device, U.S. Patent No. 7,149,949, 12 Dec 2006.

J.A. Davis, J. Jedwab, S. Morley, K.G. Paterson, F.A. Perner, K.K. Smith and S.R. Wyatt, Manufacturing Test for a fault tolerant magnetoresistive solid-state storage device, U.S. Patent No. 7,149,948, 12 Dec 2006.

J. Jedwab, J.A. Davis, K.G. Paterson and G. Seroussi, Manufacturing test for a fault tolerant magnetoresistive solid-state storage device, U.S. Patent No. 7107508, 12 Sep 2006.

J.A. Davis, J. Jedwab, S. Morley, and K.G. Paterson, Magnetoresistive solid-state storage device and Data storage methods for use therewith, U.S. Patent No. 7,107,507, 12 Sep 2006.

J.A. Davis, J. Jedwab, D. McIntyre, K.G. Paterson, F. Perner, G. Seroussi, K.K. Smith, and S. Wyatt,

Error correction coding and decoding in a solid-state storage device, U.S. Patent No. 7036068, 25 April 2006.

F.A. Lerner, J. Jedwab, J.A. Davis, D. McIntyre, D. Banks, S. Wyatt, and K.K. Smith, Magnetic memory including a sense result category between logic states, U.S. Patent No. 6,999,366, 14 Feb 2006.

J.A. Davis, J. Jedwab, K.G. Paterson, and G. Seroussi, Method for error correction decoding in an MRAM device (historical erasures), U.S. Patent No. 6,990,622, 24 Jan 2006.

J.A. Davis, J. Jedwab, K.G. Paterson, G. Seroussi, and K.K. Smith, Data storage method for use in a magnetoresistive solid-state storage device, U.S. Patent No. 6,981,196, 27 Dec 2005.

J.A. Davis and J. Jedwab, Allocation of sparing resources in a magnetoresistive solid-state storage device, U.S. Patent No. 6,973,604, 6 Dec 2005.

J. Jedwab and J.A. Davis, Methods and apparatus for encoding and decoding data, European patent application No. 97 3 10252.8-2209; US patent 6,373,859, granted 16 April 2002.

J. Jedwab, J.A. Davis, and K.G. Paterson, Method and apparatus for decoding data, US patent 6,487,258, granted 26 November 2002.

M. Mowbray, J.A. Davis, K.G. Paterson, and S.E. Crouch, System and Method for Transmitting Data, US patent 6,119, 263, granted 12 September 2000.

HONORS AND AWARDS: University of Richmond Outstanding Mentor Award, 2004.

University of Richmond Distinguished Educator, 2000.

Roger Francis and Mary Saunders Richardson Chair of Mathematics, 1998-2010.

President's Fellowship for graduate study, University of Virginia, l983-86.

Phi Beta Kappa, l983.

Francis Shunks Downs Award for leadership, awarded by the President of Lafayette College, 1983.

PUBLICATIONS: Near-complete External Difference Families, with S. Huczynska and G. Mullen, *Designs, Codes, and Cryptography*, accepted June 2016.

Bi-Cayley normal uniform multiplicative designs, with L. Martinez and M.J. Sodupe, *Discrete Mathematics*, accepted April 2016.

A comparison of Carlet's nonlinearity bounds, with G. McGrew, S. Mesnager, D. Steele, and K. Marsden, *International Journal of Computer Mathematics*, accepted December 2015.

Linking systems in nonelementary abelian groups, with J. Polhill and W. Martin, *J. Comb. Th. (A),* Vol. 123 Issue 1, 92-103, April 2014.

A new product constructions for partial difference sets, with J. Polhill and K. Smith, *Designs, Codes, and Cryptography*, Vol. 68. 155-161, July 2013.

Shared Autocorrelation property of sequences, with C. Bodea, C. Copos, M. Der, and D. O'Neal, *IEEE Transactions on Information Theory*, Vol. 57 issue 6, 3805-3809, June 2011.

On the nonexistence of a projective (75,4,12,5) set in PG(3,7), with A. Chan and J. Jedwab, *Journal of Geometry*, vol. 97 pp. 29-44, 2010.

Novel Classes of Minimal Delay and Low PAPR Rate 1/2 Complex Orthogonal Designs, with S. Adams, N. Karst, M. K. Murugan, B. Lee, *IEEE Trans. on Inf. Th.*, Vol. 57 issue 4, 2254-2262, April 2011.

Difference set constructions of DRADs and association schemes, with J. Polhill, *J. Comb. Th. (A)*, Vol. 117 Issue 5, 598-605, July 2010.

G-perfect Nonlinear Functions, with L. Poinsot, *Designs, Codes and Cryptography* 46:1 January 2008

The design of the IEEE 802.12 coding scheme, with S. Crouch and J. Jedwab, *IEEE Transactions on Communications*, October 2007

Proof of the Barker array conjecture, with J. Jedwab and K. Smith, *Proceedings of the AMS*, Vol. 135, 2011-2018, 2007.

New amorphic association schemes, with Q. Xiang, *Finite Fields and their Applications*, Vol. 12, 595-612, 2006.

Negative Latin square type partial difference sets in nonelementary abelian 2-groups, with Q. Xiang, *J. London Math. Soc.*, Vol. 70 issue 1, 125-141, 2004.

Exponential numbers of inequivalent difference sets in Z_{2^{s+1}}^2, with D. Smeltzer, *Journal of Combinatorial Designs*, Vol. 11 number 4, 249-259, 2003.

A Family of partial difference sets with Denniston parameters in nonelementary abelian 2-groups, with Q.Xiang, *The European Journal of Combinatorics*, Vol. 00, 1-8, 2000.

Constructions of low rank relative difference sets in 2-groups using Galois rings, with Q.Xiang, *Finite Fields and their Applications 6*, 130-145, 2000.

A unified approach to difference sets with gcd(v,n)>1, with J. Jedwab, in T.Helleseth et al.,eds., NATO Science Series C, Difference Sets, Sequences and Their Correlation Properties, Vol. 542, Kluwer Academic Publishers, editor A. Pott and others, 85-112, Dordrecht.

Codes, correlations and power control in OFDM, with J.Jedwab and K.Paterson, in T.Helleseth et al., eds., NATO Science Series C, Difference Sets, Sequences and Their Correlation Properties, Vol. 542, Kluwer Academic Publishers, editor A. Pott and others, 113-132, Dordrecht.

A new family of relative difference sets in 2-groups, with J. Jedwab, *Designs, Codes, and Cryptography* Memorial Volume for Ed Assmus, Vol. 17, 305-312, 1999.

Peak-to-mean power control in OFDM, Golay complementary sequences and Reed-Muller codes, with

J.Jedwab, *IEEE Transactions on Information Theory*, Vol. 45, 2397-2417, 1999.

Some recent developments in difference sets, with J.Jedwab, in K. Quinn, B. Webb, F. Holroyd and C. Rowley, eds., Pitman Research Notes in Mathematics,Combinatorial Designs and their Applications, Addison Wesley Longman, 83-102, 1999.

A unifying construction for difference sets, with J.Jedwab, *Journal of Combinatorial Theory Series (A)*, Vol. 80 No.1, 13-78, Oct. 1997.

Hadamard difference sets in nonabelian 2-groups, with J.Iiams, *Journal of Algebra*, 199, 62-87, 1998.

Nested Hadamard difference sets, with J. Jedwab, *Journal of Statistical Planning and Inference*, vol. 62, 13-20, 1997.

Finding cyclic redundancy check polynomials for multilevel systems, with M.Mowbray and S.Crouch, *IEEE Transactions on Communications*, October, 1998, Vol. 46, Number 10, 1250-1253.

Peak-to-Mean power control and error correction for OFDM transmission using Golay sequences and Reed-Muller codes, with J.Jedwab, *Electronics Letters*, Vol. 33 No.4, 267-268, 13 February 1997.

New families of semi-regular relative difference sets, with J.Jedwab and M.Mowbray, *Designs, Codes, and Cryptography* 13, 131-146, 1998.

New semi-regular divisible difference sets, *Discrete Mathematics*, 188, 99-109, 1998.

Construction of relative difference sets in 2-groups using the simplex code, with S.Sehgal, *Designs, Codes, and Cryptography*, 11, 1-11, 1997.

Recursive construction for families of difference sets, with J. Jedwab, *Bulletin of the ICA*, vol. 13, p. 128, 1995, Research announcement.

Partial difference sets in p-groups, *Archiv Mathematik*, 63, 103-110, 1994.

A Survey of Hadamard difference sets, with J. Jedwab, In K.T. Arasu et.al., editors, *Groups, Difference Sets, and the Monster*, 145-156, de Gruyter, Berlin-New York, 1996.

Exponent bounds for a family of abelian difference sets, with K.T.Arasu, J.Jedwab, S.L. Ma, and R. McFarland, In K.T. Arasu et.al., editors, *Groups, Difference Sets, and the Monster*, 145-156, de Gruyter, Berlin-New York, 1996.

A nonexistence result for abelian Menon difference sets using perfect binary arrays, with K.T.Arasu and J.Jedwab, *Combinatorica*, 15 (3), 311-317, 1995.

A construction of difference sets in high exponent 2-groups using representation theory, with K. Smith, *Journal of Algebraic Combinatorics*, Vol. 3, 137-151, 1994.

A note on new semiregular divisible difference sets, with J. Jedwab, *Designs, Codes, and Cryptography*, 3, 379-381, 1993.

Almost difference sets and reversible divisible difference sets, *Archiv Mathematik*, Vol. 59, 595-602, 1992.

Reply to `Comment on ``Nonexistence of certain perfect binary arrays'' and ``Nonexistence of perfect binary arrays,'' ', with J.Jedwab, *Electronics Letters*, vol. 29 p. 1002, 1993.

A summary of Menon difference sets, with J. Jedwab, *Congressus Numerantium*, vol. 93, 203-207, 1993.

Nonexistence of certain perfect binary arrays, with J.Jedwab, *Electronics letters*, 29 No.1, 99-100, Jan.1992.

New constructions on Menon difference sets, with K.T. Arasu, J. Jedwab, and S.K. Sehgal, *Journal of Combinatorial Theory Series A*, Vol. 64 No.2, 329-336, Nov. 1993.

New constructions of divisible designs, *Discrete Mathematics*, Vol. 120, 261-268, 1993.

An exponent bound for relative difference sets in p-groups, *Ars Combinatoria*, 34, 318-320, 1992.

Construction of relative difference sets in p-groups, *Discrete Mathematics*, 103, 7-15, 1992.

A note on products of relative difference sets, *Designs, Codes, and Cryptography*, 1, 117-119, 1991.

A note on intersection numbers of difference sets, with K.T. Arasu, D.Jungnickel, and A.Pott, *European Journal of Combinatorics*, 11, 95-98,1990.

Some nonexistence results on divisible difference sets, with K.T.Arasu, D.Jungnickel, and A.Pott, *Combinatorica*, 11, 1-8, 1991.

A generalization of Kraemer's result on difference sets, *Journal of Combinatorial Theory Series A*, March, 1992, 187-192.

A result on Dillon's conjecture, *Journal of Combinatorial Theory Series A*, July 1991, 238-242.

A note on nonabelian (64,28,12) difference sets, *Ars Combinatoria*, 32, 311-314, 1991.

Difference sets in nonabelian 2-groups, *Coding Theory and Design Theory*, Part II, IMA, v.21, 65-69, Springer-Verlag.

Difference sets in abelian 2-groups, *Journal of Combinatorial Theory Series A*, 57, Issue 2, July, 1991, 262-286.

SUBMISSIONS TO INTERNATIONAL NETWORKING STANDARDS: Low complexity decoding for power controlled OFDM codes, with J. Jedwab and K.G. Paterson, ETSI BRAN

Working Group 3 (Hiperlan Type 2) Temporary Document 3hpl081a, March-April 1998.

Options for variable rate coding in OFDM using binary, quaternary and octary modulation, with J. Jedwab, A. Jones, K. Paterson, and T. Wilkinson, ETSI BRAN Working Group 3 (Hiperlan Type 2) Temporary Document 50, Sept 1997.

EXPOSITORY ARTICLES: The most exciting thing ever in mathematics, New Book of Popular Science, Grolier, 1994.

How do we know that anything is true?, New Book of Popular Science, Grolier, 1993.

From schoolgirls to CDs to Outer Space, New Book of Popular Science, Grolier, 1992.

PRESENTATIONS:

Apple vs. Samsung: a mathematical battle, Public Lecture at the Institute for Mathematical Sciences, Singapore, May 2016.

Construction of bent functions via covering EBSs, Workshop on Combinatorics, Institute for Mathematical Sciences, Singapore, May 2016.

Apple vs. Samsung, a mathematical battle, VCU Discrete Mathematics Seminar, 27 October 2015.

Near-complete external difference families, Finite Fields conference, Saratoga Springs, NY, 14 July 2015.

Near-complete external difference families, seminar, University of Delaware, May, 2015.

Apple-Samsung and Coding Theory, Sequences, Codes, and Designs, Banff, Canada, January 2015.

The final four: difference sets in groups of order 256, Finite Geometries, Irsee, Germany, September 2014.

Linking systems in non elementary abelian groups, Combinatorics 2014 conference, Gaeta, Italy, June 9, 2014.

Apple-Samsung: the codes in 3G communication, Finite Fields conference, Magdeburg, Germany, July 25 2013.

Epic failures in math, University of Richmond, April 2013.

Apple-Samsung: a personal story, University of Richmond, August 2012.

What do those engineers do? Penn State Harrisburg, April 2012.

Difference sets, Virginia Commonwealth University, November 2011.

Difference sets and Association Schemes, AMS sectional in Lincoln Nebraska, October 2011.

Difference sets and Association Schemes, Finite Geometries in IRSEE, Germany, June 2011.

McFarland difference sets and Association schemes, Vancouver, Canada, June 2010.

Difference sets and DRADs, Finite Fields and their Applications, Dublin, Ireland, June 2009.

Linear algebra is the answer to everything: an industrial research mathematics problem, Bloomsburg University colloquim, April 2005.

Relative difference sets and cohomology, a preliminary study, University of Delaware discrete mathematics seminar, October 2004.

Bluffhead and other games, Cool Math seminar, University of Richmond, September 2004.

The mathematics of 100 base vg, Central Michigan University Colloquium, February 2004.

The mathematics of 100 base vg, Simon Fraser University Colloquium, January 2004.

The mathematics of 100 base vg, National meeting of the American Mathematical Society, Phoenix, January 2004.

Negative Latin Square type partial difference sets in non-elementary abelian groups, The British Combinatorial Conference, July, 2003.

Finite Geometry meets Galois Rings: the search for Denniston Partial Difference Sets, University of Richmond Colloquium, March, 2002.

Inequivalent difference sets, University of Delaware Combinatorics Seminar, October, 2001.

Finite Geometry meets Galois Rings: Construction of Denniston Partial Difference Sets, The National University of Ireland, Maynooth, October, 2000.

The XTR Cryptosystem, University of Bristol/Hewlett-Packard Cryptography Seminar, February, 2001.

Denniston partial difference sets, The National University of Singapore, June, 2000.

Denniston partial difference sets, The Ohio State-Denison conference, May, 2000.

The game of SETS and coding theory, Greater Richmond Council of Teachers of Mathematics, March, 2000.

Relative difference sets and Galois Rings, conference in Augsburg, Germany, August, 1999.

Relative difference sets in groups whose order is not a prime power, seminar at the University of

Delaware, March, 1999.

Golay complementary sequences and Reed-Muller codes, conference in Bad Windsheim, Germany, August 6, 1998.

Unifying construction for difference sets, Part II, conference in Bad Windsheim, Germany, August 5, 1998.

Unifying construction for difference sets, Part I, conference in Bad Windsheim, Germany, August 4, 1998.

Beauty and the mathematician, Lunchtime forum, University of Richmond, December, 1997.

Construction of relative difference sets in groups of non-prime power order, British Combinatorial Conference, July, 1997.

The Kirkman schoolgirl problem to error correcting codes, Randolph-Macon College colloquium, November, 1996.

The Kirkman schoolgirl problem to error correcting codes, University of Richmond colloquium, September, 1996.

From the Kirkman schoolgirl problem to error correcting codes: different views of difference sets, The University of Wales at Aberystwyth colloquium, May, 1996.

A unifying construction of difference sets, The Basic Research in Mathematical Sciences (BRIMS) colloquium, Bristol, England, April, 1996.

A unifying construction of difference sets, Institut fur Mathematik der Universitat Augsburg, Augsburg, Germany, March, 1996.

A recursive construction of Relative difference sets, Royal Holloway University, London, England, February, 1996.

A recursive construction of Relative difference sets, Bose Memorial Conference, Ft. Collins CO, June 1995.

A new family of difference sets, National Security Agency, May 1995.

K-matrices revisited, Special Session on Codes and Designs, University of Richmond, Nov, 1994.

Partial difference sets in p-groups, The Ohio State University/Denison University conference, March 1994.

If at first you don't succeed... a search for difference sets, Virginia Commonwealth University colloquium, November, 1993.

Partial results on partial difference sets, the British Combinatorial Conference, July, 1993.

Nonexistence results on difference sets, the conference on Difference Sets at Ohio State University, May, 1993 (one hour talk).

A survey of Menon difference sets-- the nonabelian case, Southeastern Combinatorics conference, Feb. 1993.

Difference sets and perfect binary arrays, William and Mary colloquium, Jan. 1993.

If at first you don't succeed... a search for difference sets, Central Michigan University colloquium, May, 1992.

If at first you don't succeed... a search for difference sets, talk to Ohio State combinatorics seminar, May, 1992.

New constructions of divisible designs, the conference at Denison University, May, 1992.

Almost difference sets and divisible difference sets with multiplier -1, special session on Designs and Codes, Joint meeting of the MAA and AMS, Baltimore, January 1992.

Error Correcting Codes, Sigma Xi lecture, University of Richmond, December, 1990.

Relative Difference Sets in p-Groups, Oberwolfach, April, 1990.

Coding Theory, Hampden-Sydney colloquium, October, l988.

Character Theory Applied to Difference Sets, Design Theory Workshop, IMA, June, l988.

Difference Sets in Abelian 2-Groups, Joint National Meeting of the AMS-MAA, January, l988.

Difference Sets in Abelian 2-Groups, Wright State University colloquium, January, l988.

Difference Sets in Abelian 2-Groups, National Security Agency seminar, January, l987.

CLASSES TAUGHT: 100 level: Introduction to Statistics, Statistics for the Social Sciences, Calculus with precalculus.

200 level: Discrete math, Calculus 1, Calculus 2, Multivariate Calculus, Linear Algebra.

300 level: Introduction to Abstract Reasoning, Abstract Algebra 1, Abstract Algebra 2, Coding Theory, Cryptography.

SERVICE: Board of Trustees Academic and Enrollment Management Committee, 2013-2016.

Presidential Hiring Committee, 2014-2015.

Statistics Hiring Committee, 2014-2015.

Chair, hiring committee, 2001-2002.

Chair, General Education Committee, 2001-2002.

Chair of Math and Computer Science department, 1997-2000, spring 2005.

Science Scholars Committee, 1994-present.

Chair, hiring committee, 1994-95.

Curriculum committee, 1993-95.

Colloquium chair, 1991-94.

Freshman advisor, 1991-95, 2001-Present.

Calculus committee, 1990-94.

PROFESSIONAL AFFILIATIONS: Mathematical Association of America

American Mathematical Society.

Fellow, Institute of Combinatorics and its Applications