2011 International Conference on Computer Science and Information Technology (ICCSIT 2011)
IPCSIT vol. 51 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V51.34

A Novel High Code Rate Runlength Limited Coding Method for Magnetic Recording Channel

Chun Liu [a,*], Changsheng Xie [a,b], Guangxi Zhu [a,b], Qingdong Wang [a,b]

[a]Dep. of computer science, HUST, Wuhan Hubei, P.R. China
[b]Dep. of optoelectronic storage, WNLO, Wuhan Hubei, P.R. China

Abstract. In this paper we present a novel high rate run length limited coding scheme for magnetic recording channel. The coding scheme splits the source data into sub-blocks, and constructs the encoding table of the sub-blocks as well as queuing rules of the indicating blocks, which can approach the maximum of RLL codes rate within the limits of special (d, k) conditions. Moreover, with the expansibility of the coding scheme, a higher rate RLL codes can be iteratively designed. As an illustration, a 32/33 (0, 6) RLL code is designed. Further an intersecting insert skill is used to design a 64/65 (0, 10) code based on the former encoder. Both the error propagation characteristic and the implementing method of the coding scheme are discussed.

Keywords: Magnetic Recording；RLL Code；High Rate

1. Introduction

The coding scheme based on RLL (Run Length Limited) sequence is the foundation of magnetic recording and optic recording. Run Length means the bit length between two adjacent transition channel bits. RLL sequence is characterized by 4 parameters (m, n, d, k). m means the number of input sequence bits, n means bit number of output code, d and k prescribe separately the minimum run length and maximum run length which maybe appear in sequence. Parameter d denotes that there are at least d continuous “0” between two logic “1” in the sequence, while k denotes that the length of continuous “0” are no more than k. So d control the highest transfer frequency and then it maybe impact the intersymbol interference when sequences transfer via bandwidth limited channel. k is used to provide appropriate transition frequency and to meet the requirement for timing self-synchronization.

Since Berkoff, Freiman, Wyner, Kautz, Gabor, Tang and Bahl presented the ideology of RLL coding, all kinds of recording coding based RLL coding popularized in storage devices. Such as EFM coding (code rate 8/17, d=2, k=10) used in CD system, and EFM+ coding (code rate 8/16, =2, k=10) in DVD system, and RLL coding (code rate 8/9, d=1, k=7) in former hard disk system.

But the common defective of all these former RLL coding is low code rate which wastes effective storage space. The code rate 2/3 means to transform 2 bits source data to 3 bits target data, the coding efficiency is only 66.7%. While 8/9 code rate improves the efficiency to 88.9% and then saves about 22% storage space. So code rate is quite important to weight the coding scheme, usually if the code rate is over 88.9%, we call it as high code rate scheme.

Now the capacity of current storage device is larger and larger, a single hard disk with over 1TB size is normal case. Hence the benefit of increasing code rate is obviously. Because just increasing the code rate, the

* Corresponding author.. Chun Liu
E-mail address: Chun_liu@tom.com.

storage size can increase outstandingly even without changing any media material or other recording technology.

2. The maximum code rate of RLL

2.1. Structure Key frame Extract and Pre-process

The possible maximum value of code rate R=m/n is called coding capacity, and denoted as C (d, k). According to Shannon's law:

$$C(d,k)=\lim_{n\to\infty}\frac{1}{n}\log_2 N_{dk}(n)\ ; \tag{1}$$

$N_{dk}(n)$ is the total number of (d, k) sequences with the same code length n, and can be derived from next equation set:

$$\begin{cases} N_{dk}(n)=n+1,\ 1\le n\le d+1\ ; & (2) \\ N_{dk}(n)=N_{dk}(n-1)+N_{dk}(n-d-1),\quad d+1\le n\le k\ ; & (3) \\ N_{dk}(n)=d+k+1-n+\sum_{i=d}^{k}N_{dk}(n-i-1),\ k<n\le d+k\ ; & (4) \\ N_{dk}(n)=\sum_{i=d}^{k}N_{dk}(n-i-1),\ n>d+k\ ; & (5) \end{cases}$$

As a special case, when d=0, $N_{dk}(n)$ can be simply described as:

$$N_k(n)=\sum_{i=1}^{k+1}N_k(n-i),\ n>k\ ; \tag{6}$$

Assume $N_k(n)=cZ_n$, and cZ_n is the homogenized solution of $N_{dk}(n)$, then the secular equation of equation (6) is:

$$Z^{k+1}-Z^k-Z^{k-1}-\cdots-Z-1=0; \tag{7}$$

Simplified to: $Z^{k+2}-2Z^{k+1}+1=0;$ (8)

Equation (8) is an equation of higher degree when k is assigned, but we can get a numerical solution with Matlab polynomial function toolbox. Assumed λ is the maximum positive real root of the equation, when n→∞,

$$N_k(n)=c_1\lambda^{n-1}+c_2\lambda^{n-2}+\cdots+c_{k+1}\lambda^{n-k-1}\propto c_1\lambda^{n-1}\ ; \tag{9}$$

In which the other items are the infinitesimal of higher order of λ^{n-1}, so the equation (1) can be simplified to:

$$C(0,k)=\lim_{n\to\infty}\frac{1}{n}\log_2(c_1\lambda^{n-1})=\log_2\lambda\ ; \tag{10}$$

Table 1 lists the maximum positive real roots and the coding capacity at different k. Because of the development of channel chipset technology, the high transfer speed is not the bottleneck any more, parameter d=0 was universal applied in modern Hard disk.

Table 1. the coding capacity of RLL

(d,k)	λ	C
(0,6)	1.992	0.99422
(0,8)	1.998	0.99856
(0,10)	1.9995	0.99964
(0,14)	1.9999	0.99993

On the other hand, along with the developing of partial response technology in magnetic storage channel, the channel response characteristic had transitioned from PR4、EPR4、E^2PR4 to ME^2PR4 (Modified Extend Enhanced Partial Response), and has a response polynome like $7+4D-4D^2-5D^3-2D^4$, which has a stronger capability to restrain the DC component, then it can tolerance a higher code rate in RLL than before[1]. From [2] we know that: only when d=0 and k=∞, the capacity C(d, k) is rational number, except that they are all irrational number. So the code length suit to computer architecture just as 32, 64, are impossible to reach the theoretic maximum value, just approach it only.

Fisher [3] [4] presented a coding method which can increased the code rate to 16/17 and 24/25. Fitzpatrick [3] gave a 16/17 coding method to meet the G/I limitation. Both of these method increased the code rate from 8/9 to 16/17 and even higher. In [5][6][7] the paper described several basic techniques such as partition limited coding, interleaving coding, inserting limited sub block coding, and presented the coding method including 16/17 (d=0, k=4) and 16/17 (d=0,k=6). But both these methods depended on designer's experience and techniques, it's hard to apply or extend to higher coding system. In this paper we present a RLL block encoder system which use a block search table method to meet the code rate limitation, and even can be extend used to design higher code rate like 64/65,128/129 etc on.

3. 32/33（0，6）RLL BLOCK CODING SCHEME

All tables should be numbered with Arabic numerals. Headings should be placed above tables, left justified. Leave one line space between the heading and the table. Only horizontal lines should be used within a table, to distinguish the column headings from the body of the table, and immediately above and below the table. Tables must be embedded into the text and not supplied separately. Below is an example which authors may find useful. The normal search table method is not easy applicable to long code length, such as the 32 bits codeword, there are totally 2^{32} entries of search table, so it's too complicated to implement. But if splitting the source codeword to sub blocks with same bit width, and encoding the sub blocks according to specific rule which used to fulfill the constrain condition, an entry table with much small size can be constructed easily[8][9].

Figure 1 describes the structure of RLL encoding system based sub block design. Assume the source code word has 2^m bits, 2^m =M, when a frame of data input to the encoder, a sub block splitting circuit separate them to 2^{m-k} sub blocks with each has 2^k bits, and mark them as n1, n2,……,n (2^{m-k}). A mode test module detects if all sub blocks fulfill the constrain condition: if all bits in the sub block are not 0, the sub block then can be seen as fulfilling the constrain and set the flag register to 0, while if all bits are 0, the sub block is defined as conflict block, and set the flag register to 1. For example if only sub block n1 has conflict, the flag register is equal to 00….001, if n1 and n3 have conflict, then the register is equal to 00….0101. For identifying all possible sub blocks, the bit width of register is set to 2^{m-k}. After finished all detection, the code word mapping process is followed.

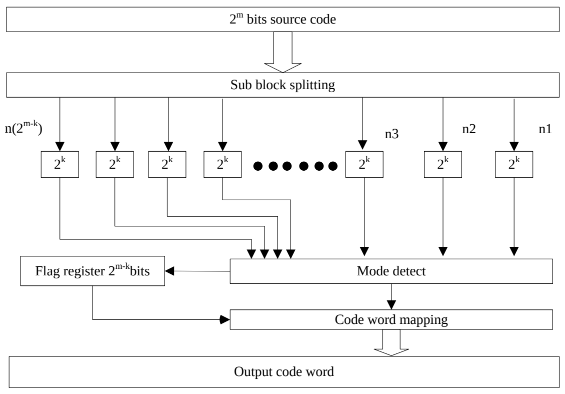

Figure 1. the structure of encoding system

To keep the availability and compatibility, we describe the detail high code rate scheme with RLL (d=0,k=2^{k+1}-2) as following:

- A symmetry structure which can simplify the design of encoder is introduced to output code word, the middle bit is used to flag bit.
- Detect and judge if the flag bit is equal to 0 which means all sub block have no conflict, while 1 means at least one sub block has conflict. Accumulating each bit in flag register can judge the conflict value, but a sum weight method can further locate the conflict point. Assume the bit width is 8 bits, the calculation method with sum weight is:

$$Sum = S_7 \times 2^7 + S_6 \times 2^6 + S_5 \times 2^5 + S_4 \times 2^4 + S_3 \times 2^3 + S_2 \times 2^2 + S_1 \times 2^1 + S_0 \times 2^0 \tag{11}$$

  If no conflict, the Sum=0, then the sub blocks can be mapped directly to the corresponding location from left to right side: n1，n2，……，n(2^{m-k} /2),flag bit, n(2^{m-k} /2＋1)，n(2^{m-k} /2＋2)，……， n(2^{m-k}) , then set the flag bit to 1 and combine a output code word. For example if the 32 bits input data has no conflict, the output code is {n8,n7,n6,n5,1,n4,n3,n2,n1}.

- The rest may be deduced by analogy, if the sum weight of flag register is equal to i (1≤i≤2^{m-k}）， means that there are i sub blocks have conflict, and the sum value can locate the serial number of conflict sub block and its conflict bit. The mapping rule of conflict code word is: set the middle flag bit to 0, the sub block n(2^{m-k} /2＋1) identify the all 0 conflict block with the maximum serial number; the sub block n(2^{m-k} /2) identify the all 0 conflict block with the secondary maximum serial number; the queue order of following identifying sub blocks is given by the queue rule described later. Other no conflict sub blocks map directly to the correspondent location, the proper data in block n(2^{m-k} /2 ＋1) and n(2^{m-k} /2) are metastatic mapped to the location in output codeword and the metastatic mapping rule is given in later content. To distinguish the cases of only 1 ~ (i-1) sub blocks, the LSB (Least Significant Bit) of block n(2^{m-k} /2＋1) and n(2^{m-k} /2) are cleared to 0, while the LSB of sub block i is set to 1 (when i=2, the MSB (Most Significant Bit) set to 1). For 32 bits input data, if there is only a conflict at sub block n3, then after transforming, the output code word is: {n8,n7,n6, '00…0101', 0 ,n4,n5,n2,n1} .
- The queue rule is: the most proximate sub block in the left side of flag bit acts as the first identify sub block, while the most proximate sub block in the right side of flag bit acts as the second identify sub block. Then all succedent identify sub blocks distribute with an ordinal and symmetrical rule of firstly left side then right side. All identify blocks organize to a queue: n(2^{m-k-1}+1)- n(2^{m-k}-1)- n(2^{m-k-1}+2)- n(2^{m-k-1}-1)……-n(2^{m-k})-n1. The former block is used to identify the conflict block with the high serial number, the tail block identifies the conflict block with small serial number. Their own data in the identify sub blocks need metastatic mapped to the unoccupied positions. According to their serial number, the block with the smallest number is filled to the lowest location, the block with high number is filled to the high sequence location. If the identify itself is also a conflict block, it need not transferring map any more.
- The special cases: if there are 3 groups conflict sub block, the parameter k may not held constrain condition. Let’s look at the 32/33 coding: if n1,n2,n5 are conflict blocks, according to the mapping rule, the identify sub blocks n6,n5, n4 are equal to '0001', '1000', '0001', plug in the middle flag bit, and assembly a sequence {0001100000001}. So it appears totally 7 successive '0' which violate the constrain condition of k=6. To solve the problem, the map rule have to be modified based on the special cases: when there are 3 groups conflict sub block, the binary value of second identify block which locate at the right side neighbor the flag bit need to add 1. Correspondingly the value need subtract 1 firstly when decoding, and then transfer the value to identify serial number. Because of the help of queue rule of identify blocks, we can assure that after add or subtract 1 the value of second identify block will not beyond its range. All other cases of over 3 groups conflict block don’t exist special cases.

Table 2. the mapping rule of 32/33 （0，6）RLL encoder

	G8(29～32b)	G7(25～28b)	G6(21～24b)	G5(17～20b)	F(16b)	G4(12～15b)	G3(8～11b)	G2(4～7b)	G1(0～3b)
no conflict	n8	n7	n6	n5	1	n4	n3	n2	n1
n1 conflict	n8	n7	n6	0001	0	n4	n3	n2	n5
……									
n8 conflict	n5	n7	n6	1111	0	n4	n3	n2	n1
n1n2 conflict	n8	n7	n6	0010	0	1000	n3	n5	n4
……									
n2n3 conflict	n8	n5	n6	0100	0	1001	n3	n2	n1
……									
n7n8conflict	n5	n4	n6	1110	0	1111	n3	n2	n1
n1n2n3 conflict	n8	n7	0001	0100	0	**0001+1**	n6	n5	n4
……									
n6n7n8conflict	n5	n4	1011	1110	0	**0110+1**	n3	n2	n1
……									

- The coding mapping table of 32/33 (0, 6) RLL using the method mentioned above is shown in Table 2, the code rate is 96.97%. The number “i” in the paper is not a special given value, so the method can be extended to the encoder of 128/129.

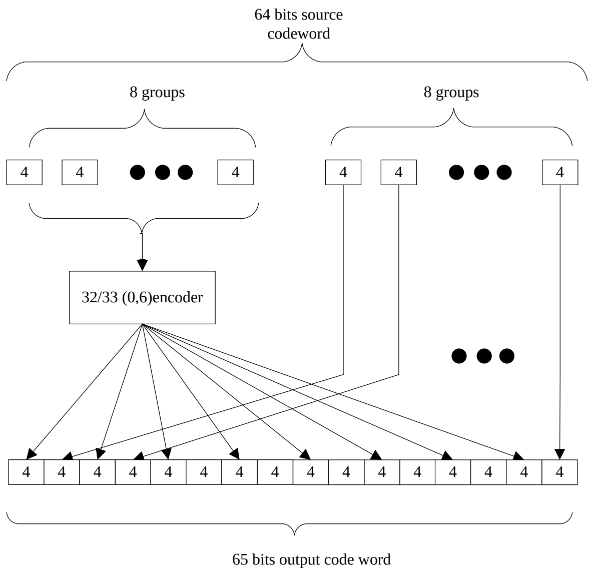

Figure 2. the structure of 64/65 （0, 10）RLL encoder

From the process of encoding, the rules of mapping and queue assure the requirement of parameter d and k together. The consecutive “1”s are permitted in the input sequence because there is no limitation to number of “!” in the coded blocks, in another word, d=0. After mapping, in the output codeword each sub block can’t be all “0”, at least include one “1”, and between the first and the last “1” in the sequence packaged by arbitrary 2 sub blocks there are at most “0” of $2\times 2^k-2=2^{k+1}-2$, means that it meets the RLL design parameter of d=0, $k=2^{k+1}-2$.

With a technique of sub block intersecting insert, the coding method can be used to construct higher code rate n/(n+1) RLL coding design. The technique is splitting the source data to two parts, and one part is encoded with foregone encoder, while another part split again to sub blocks then insert them between the first part coded blocks. Figure 2 illustrates how to construct 64/65 (0, 10) RLL encoder based on 32/33(0, 6) encoder.

4. Error Propagation Characteristic and Realization Method

Table 3 illustrates the probability of conflict sub blocks when 0 and 1 in the input source sequence accompany with the same probability. Actually before the source sequence input to the magnetic writing channel, they have already scrambled by the scrambler, whose effect is approximately equal-probability distribution of 0 and 1. So we can consider that the result of Table 2 is accepted in the magnetic channel. In the table 2 the probability sum of no conflict and one group conflict is over 90%, which means only less than 10% sequences with splitting sub block method need increasing complexity of coding computation to deal with over 2 groups conflict sub blocks, while more than 90% mapping action just need few computation. From the structure of magnetic channel we know that when an error appears at Viterbi decoder or post-processor the error will propagate backward to the post decoder, furthermore after the post decoder perform $1+D^2$ transforming, one error bit will evolve to 2 error bits. When this error event entries to RLL encoder more error bits will be introduced and form error propagation. The mentioned coding method in this paper has a degree of capability of restraining error propagation-its characteristic of error propagation depends on the error probability of flag identify bit. When no conflict and the middle flag identify bit is correctly decoded, internal error in each sub block can only impact at most the block itself or just crossing the boundary of two sub block and will not propagate any more. If there are conflict blocks, but every identify bit is correctly decoded, the error will be limited in its internal body. Advanced if the middle flag identify bit is fault, the worst case is that errors distribute at all sub blocks and need send again. But with the ECC/RLL permute technology which is most commonly used in modern magnetic channel and has powerful ability to constrain error propagation, the worst case usually can be found and corrected as early as possible.

It’s convenient to implement the encoder with Verilog language. The source code shows below realize a 32/33 (0, 6) RLL encoder, which can be finished via 632 logic cell after analyzed and synthesized with Quartus II. Because of the symmetry structure, the decoder can be easily to realize by reversing the coding table.

Table 3. the probability of conflict sub blocks

the number of conflict blocks in 32 bits data	Probabilty
0	0.597
1	0.318
2	0.074
3	9.90E-03
4	8.25E-04
5	4.40E-05
6	1.50E-06
7	2.80E-08
8	2.30E-10

5. Conclusion

Along with the increasing of capacity of HDD, it needs high code rate encoding method to decrease redundant storage space, in this paper a scheme of splitting sub blocks, mapping and then queue the identify blocks is present, which can meet the constrain condition of (0, k) and approach the limitation value of coding capacity, moreover it has the expansibility which can be used to design much higher code rate RLL coding. The method can constrain the error propagation to a degree of range and an implementation with Verilog VHDL is easily.

6. Acknowledgements

T This work is financially supported by National 973 Program (contract No. 2011CB302303) and National 863 Project (contract No. 2009AA01A402).

7. References

[1] Roy D. Cideciyan, Jonathan D. Coker, Evangelos Eleftheriou, and Richard L.G, “Noise Predictive Maximum Likelihood Detection Combined with Parity-Based Post-Processing”, IEEE Transactions on Magnetics, vol. 37, No.2, pp. 714-720, Mar 2001.

[2] K.S.Immink, Xu Duan-yi, Lei Zhi-jun, “Codes for Mass Data Storage Systems”, Beijing: Science Press, 2004

[3] Fisher, “Bit-Interleaved Rate 16/17 Modulation Code with Three-Way Byte-Interleaved EC”, U.S. Pat. No. 5,757,822, May 26, 1998.

[4] Fitzpatrick, “Rate 16/17 (D=0, G=6/I=7) Modulation Code for a Magnetic Recording Channel”, U.S. Pat. No. 5,635,933, June 3, 1997

[5] Fisher, “Rate 24/25 Modulation Code for PRML Recording Channels”, U.S. Pat. No. 5,737,594, April 7, 1998

[6] Wijinggaarden, Immink. “Combinatorial Construction of High Rate Runlength-limited Codes”, IEEE Proceedings of Globecom 1996, pp. 343-347

[7] Roy D.Ciddiyan, “High-rate RLL encoding”, US Patent No. 7679535B2, Mar.16, 2010

[8] Jamieson, C., Fair. I, “Construction of Constrained Codes for State-Independent Decoding”, Selected Areas in Communications, IEEE Journal on February 2010

[9] Louidor, E., “The tradeoff function for a class of RLL(d, k) constraints”, Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on June 2010