Chapter 3

RECOVERING DATA FROM FAILING FLOPPY DISKS

Frederick Cohen and Charles Preston

Abstract As floppy disks and other similar media age, they may lose data due to a reduction in the retention of electromagnetic fields over time, mainly due to environmental factors. However, the coding techniques used to write data can be exploited along with the fault mechanisms themselves to successfully read data from failing floppy disks. This paper discusses the problem of recovering data from failing floppy disks and describes a practical example involving a case of substantial legal value.

Keywords: Floppy disks, field density loss, weak bits, data recovery

1. Introduction

This paper discusses a method for recovering data from floppy disks that are failing due to "weak bits." It describes a repetitive read technique that has successfully recovered data in forensic cases and discusses the analysis of the results of repetitive reads in terms of yielding forensically-sound data. This technique is not new; however, neither the technique nor the analysis necessary to support its use in legal matters have been published.

The case discussed in this paper involved a fifteen-year-old floppy disk, which contained the only copy of the binary version of a software program that was subject to intellectual property claims of sufficient value to warrant recovery beyond the means normally used by commercial recovery firms. After attempts to read the disk by these firms had failed, the disk was given to the authors to use more rigorous and possibly destructive data recovery methods, subject to court approval.

Several techniques for recovering data from hard-to-read floppy disks are in common use, including reading only relevant sectors from a disk where other sectors fail to read properly, and altering the drive alignment to better align the heads with the tracks as originally written. In the case of interest, important data was contained in hard-to-read sectors of the disk, and custom head alignment only marginally altered the recovery characteristics of the disk.

A floppy disk can also be modified to read analog signals and allow the detection thresholds to be altered. Additionally, signals from the read heads can be amplified, rates of rotation can be increased to boost induced currents, and other similar methods can be attempted. But they introduce various problems, including increased time requirements and cost. Furthermore, it is difficult to prove that the methods recover valid data instead of merely turning noise into data.

Other exotic techniques involve analog reads using digital storage scopes, the use of epoxies with suspended fine ferrous material that attach to the media and are visible under a microscope, and the use of magnetic force scanning tunneling microscopy. Some of these techniques are destructive; all are expensive and may result in data loss.

2. Data Recovery Methodology

The obvious data recovery method is to attempt repeated sector-by-sector reads of a disk; failed sectors are repeated until valid reads are completed. Data from the sectors is then assembled to create a complete image of the disk. This technique has several advantages: (i) it only uses the designed features of the floppy disk drive and, thus, requires very little in the way of explanation or analysis to be considered credible; (ii) it is relatively low cost and takes relatively little time to perform; and (iii) it uses the built-in coding analysis methods and phased lock loops of the floppy drive to decode changes resulting from orientations of charges in areas on the disk. This eliminates the problems involved in explaining coding errors, side band signals, additional introduced errors and other issues associated with building special-purpose hardware.

The specific program used in the case was executed from a bootable White Glove Linux CD, which was kept with the evidence after processing to ensure that the process could be repeated if necessary. The following shell script code was executed:

```
for i in ‘count 0 1439’; do
     dd conv$=$noerror bs$=$512 count=1 skip=\$i if=/dev/fd0$>$
     noerr/\$i.out
done
```

The `count` command counts from the first value (0) to the second value (1,439) in increments of one. For each count value, the `noerr` command is executed with the conversion option that, in the event of errors, retries are to be attempted an unlimited number of times. The block size is set to 512 (normal block size for such a floppy disk) and a count of one block per execution is used. This is done after skipping `count` number of blocks from the beginning of the media (in this case the floppy disk `/dev/fd0`). The output is stored in a file whose name includes the block number (`noerr/[count].out`, where `[count]` is the block number and `noerr` is the directory used to store all the blocks). On each read attempt, a file is created, but unless the file read succeeds with a valid checksum, the file is overwritten on the next attempt.

Reading one sector at a time is beneficial because a single error in a read produces a failure for the entire read. If a single sector takes twenty attempts on average to succeed, reading two sectors would require an average of 400 attempts. Since reading less than one sector does not involve any special execution, this approach minimizes the number of reads and reduces unnecessary wear and tear on the disk while reading it repeatedly until the CRC code and the data match.

When applied to the evidence disk, this process produced different numbers of retry cycles on different sectors. There were no retry cycles on sectors that could be consistently read without errors. For the previously unreadable sectors, the number of retry cycles required ranged from one to more than 70, most of them in the 20 to 30 range. Each sector was stored individually in a file of 512 bytes on a hard disk as it was read, and stored with a filename associated with the sector number as described above. The total number of blocks was 1,440 with 512 bytes each (737,260 bytes of data), corresponding to the entire readable contents of the 720K floppy disk.

The individual files representing the blocks on the evidence disk are independently examinable. Alternatively, they may be assembled in a single file and mounted using a loopback mounting interface or written to a fresh floppy, which can then be read as if it were the original evidence disk. For the case being discussed, the assembly was done using the following program:

```
for i in ‘count 0 1439’; do
     dd seek=\$i if=noerr/\$i.out of=noerrdd.out
done
```

The blocks were written to the file at the appropriate locations in the same way as they were read from the evidence disk. Multiple copies were made of the recovered disk for use by all parties. Having read the disk and created forensic duplicates, it is necessary to show that the method is forensically sound.

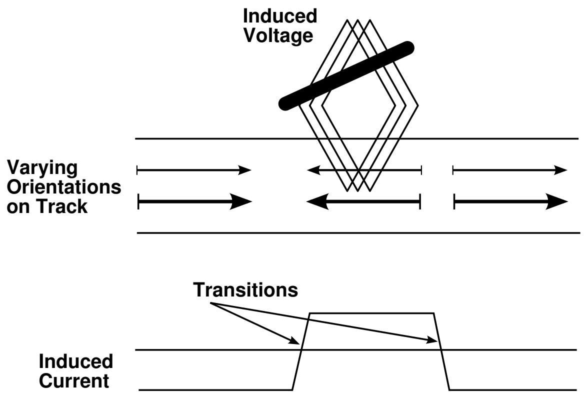

Figure 1. Digital signal encoding on floppy disks.

3. Weak Bits and Floppy Disk Failure Modes

Floppy disks tend to degrade over time and under various environmental conditions such as temperature and humidity. This sometimes results in the presence of so-called "weak bits." Weak bits are caused by degraded electromagnetic orientation alignments or charge densities that reduce voltage and current swings to levels that are too low to reliably trigger transitions in the hardware detectors. Weak bits may also be caused by the physical shifting of magnetic materials under temperature changes, by the growth of organisms on the media, or by friction that abrades portions of the coatings used to retain charges.

Floppy disks typically use the modified frequency modulation (MFM) hardware-level coding [5], in which timed flux density transitions are used to encode bits. Figure 1 illustrates how floppy disks store data. The write head causes magnetic particles to align in one of two orientations along cylinders (concentric circles) at different distances from the center of the platter. Because the circles have different radii, the timings of transitions from one orientation to the other vary with radius. Consequently, lead-in transitions are required to set up an oscillator to synchronize this change detection.

Figure 2 (adapted from [3]) illustrates the mechanisms used to read data from floppy disks. These include a read head, amplifier, pulse generator, phased lock loop, demodulator and additional hardware needed to produce a controller that is usable by a computer at the bus level.

Figure 3 (adapted from [3]) shows the signals that appear at different locations in Figure 2 [3]; it helps clarify the effects of reduced signal levels in the media. As the analog signal (A) degrades, peak pulses

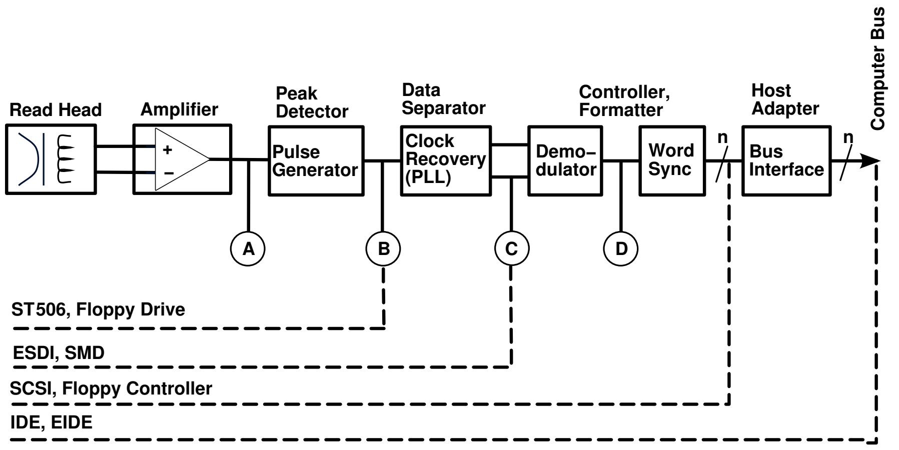

Figure 2. Floppy disk components.

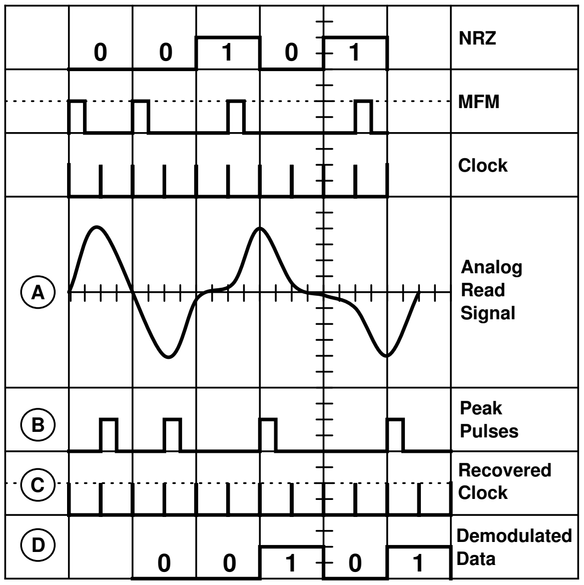

Figure 3. Floppy disk controller signals.

(B) disappear, causing the loss of MFM transitions, which result in demodulated data (D) and phased lock loop desynchronization.

When a floppy disk is being read, changes in field density produce induced currents in the head, which, regardless of the field direction, is seen as a transition (T); the lack of a change at a timing signal produces “no transition” (N). The MFM coding uses a “no transition–transition”

Table 1. Code space changes from flux density reductions.

Data	Original	Possible	Result
[11]	NTNT	NNNT	1[01]
	NTNT	NTNN	[10]
	NTNT	NNNN	invalid
0[00]	TNTN	NNTN	1[00]
	TNTN	TNNN	invalid
	TNTN	NNNN	invalid
1[00]	NNTN	NNNN	invalid
[10]	NTNN	NNNN	invalid
0[01]	TNNT	NNNT	1[01]
	TNNT	TNNN	invalid
1[01]	NNNT	NNNN	invalid

(NT) sequence to indicate a 1, a "transition–no-transition" (TN) to indicate a 0 preceded by a 0, and a "no transition–no-transition" (NN) to indicate a 0 preceded by a 1. If a transition is not detected because of a loss in electromagnetic flux density, an NT can turn into an NN or a TN can turn into an NN, but an NN cannot turn into an NT or a TN.

Pairs of bits always involve a transition. In particular, a 11 will produce NTNT, a 00 will produce either TNTN (if a 0 preceded it) or NNTN (if a 1 preceded it), a 10 will always produce NTNN, and a 01 will produce either TNNT (if a 0 preceded it) or NNNT (if a 1 preceded it). If no transitions are detected, the controller normally indicates an error condition and the CRC code at the end of every 512-bit block of data is irrelevant. Thus, weak bits produce controller errors due to the inability to observe transitions, or weak transitions change a T to an N. They cannot turn the lack of a transition into a transition. As a result, seven out of eleven possible field reductions turn into invalid codings that should be detected by the drive controller as invalid data. Of the remaining four errors that could produce valid data, three require that the previous bit be a 1 or they too produce invalid data in the controller.

Table 1 shows all the possible changes. In the table, data values represented by T and NT sequences are enclosed in brackets (e.g., [11]) and the required preceding bits are indicated prior to the bracketed pairs (e.g., 1[00]).

None of these errors can produce a transition of the coded data from a 0 to a 1. Thus, a weak bit error can never turn a 0 into a 1; it can only turn a 1 into a 0 or produce an invalid code space output. Additional consistency checks could potentially detect errors such as the transition of 0[00] to 1[00], but the previous 1 bit could not be the result of a

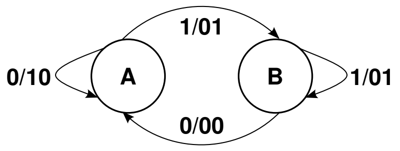

Figure 4. Finite state machine for floppy disk reads.

weak bit (or its coding would be a 0 to 1 transition that a weak bit cannot produce in that position). Therefore, this eliminates the other possible errors that turn 0[01] into 1[01] and 0[00] into 1[00], leaving only the transitions of 1[11] to 1[01] and [11] to [10] due to reduced electromagnetic flux density. If the previous bit was not a 1[NT] or the reduction in flux density reduced the T to an N, then the 1[01] error is also impossible.

Unfortunately, depending on their design, floppy controllers do not always produce error outputs for non-existent transitions. Figure 4 shows the finite state machine for producing output bits based on the current state [5]. Note the lack of state transitions for the 0/11 and 0/01 cases and the 1/10 and 1/11 cases. They are typically designated as "Don't Care" (DC) values, which leaves the designer free to optimize the electronics by ignoring outputs that in theory cannot happen. In practice, a weak transition could produce a change from 0/10 to 0/11; however, the controller would be in State A and this is only identified as a transition for State B. The incomplete specification of error states produces arbitrary behavior depending on the design choice. Fortunately, the CRC code used in floppy disks can compensate for most errors.

Our analysis is based on the assumption that a weakened field density in the locality of a bit cannot trigger a transition; this is worth discussing further. Normally, for a transition to be detected by a floppy disk controller, the electromagnetic field density in one region has to be oriented in one direction while that in the adjacent region has to be oriented in the opposite direction. Which direction is 01 and which direction is 10 coupled with the direction of movement of the disk in the drive dictate whether the drive head gets a positive or negative impulse; but these are not differentiated by the controller – both are considered to be transitions. If a transition from the maximum field density to a zero field density were to trigger a transition, floppy disks would be very unreliable because regions near the tracks are commonly not used and any minor movement in the head could cause such a transition. In addition, the devices are designed so that positive and negative field densities can ensure sound triggering. A half-level density change should not trigger a transition on most floppy disk drives. For this reason, even a maximum field density area adjacent to a zero field density area should not trigger a transition; thus, the weakening of the electromagnetic field strength on the disk should not create transitions where none existed. Of course, the physical phenomena associated with weak bits are analog in nature at this level of granularity. The size of a region of storage on a 720K floppy disk is on the order of 1/8000" in circumference. Because of this relatively high density, most common physical phenomena are unlikely to reduce the field density of one region to near zero while preserving the density of the area next to it at full strength. A scratch could cause this to happen, but then the damage would be permanent and would likely produce the same level of transition on each use.

An electromagnetic field such as that produced by a magnet passing near the disk, a temperature condition or a biological phenomenon is highly unlikely to produce such a dramatic edge condition. There is a strong tendency for these phenomena to produce regions with decreasing effects as a function of distance. This produces a slow transition in field density resulting in a change in field strength with distance that will not normally produce a transition in the floppy disk controller. As a result, it reasonable to assume that no transitions will be created by reductions in electromagnetic field density associated with weak bits, and only the loss of transitions is likely to occur from these physical phenomena.

4. Code Analysis and Error Rates

In addition to the MFM coding, floppy disks also use a CRC code at the end of each sector after it is written. This is highly likely to be inconsistent when certain classes of errors occur in portions of the sector. It is easy to detect single bit flips, multiple bit flips in close proximity and several other combinations of bit flips. According to Freeman [1]:

> "Any bit error term E(x) which is an exact multiple of P(x) will not be detected. This is the case for the two-bit error 10000001, where the two bad bits are 7 bits apart. Note that 10000001 = (1011) (1101)(11). The allowable separation between two bad bits is related to the choice of P(x). In general, bit errors and bursts up to N bits long will be detected for a prime P(x) of order N. For arbitrary bit errors longer than N bits, the odds are 1 in 2N than a totally false bit pattern will nonetheless lead to a zero remainder. In essence, 100% detection is assured for all errors E(x) not an exact multiple of P(x). For a 16-bit CRC, this means:
>
> > 100% detection of single-bit errors
> >
> > 100% detection of all adjacent double-bit errors
> >
> > 100% detection of any errors spanning up to 16 bits
> >
> > 100% detection of all two-bit errors not separated by exactly 216–1 bits (this means all two-bit errors in practice!)

> For arbitrary multiple errors spanning more than 16 bits, at worst 1 in 216 failures, which is nonetheless over the 99.995% detection rate."

If we assume that the CRC is intact, the available error modes from weak bits are such that the degradation mechanism would have to produce reduced flux densities exactly 32 transition distances from each other for the CRC code to fail to detect pairs of errors. Reductions in flux density producing lost transitions in adjacent bits or other sequences of less than 32 transition areas (representing 16 bits of data) are detected by CRC codes with 100% accuracy unless they range over large areas, in which case they would produce invalid codes in the MFM decoding mechanism. Thus, the physical phenomena that produce weak bits are very unlikely to create conditions under which data from a sector correctly matches the CRC code and no MFM coding error is produced, but an alteration from the loss of a transition occurs.

This implies that if weak bits cause errors and a successful read of the data with matching CRC code is completed, it is highly likely that the data recovered accurately reflects the data written to that sector. While it is difficult to calculate the probability, it is certainly less than the probability of errors associated with MFM or CRC alone. In other words, there is no known synergistic effect that can cause one of them to correct an error produced by the other.

The method used tends to support the contention that disk failures are caused by weak bits. Specifically, if another mechanism was in play (e.g., alignment errors or mechanical defects in the original writer), then the realignment process would have yielded better or worse data instead of nearly identical error behavior. If bits were not written at all or if a typical contemporaneous weak bit writing mechanism were used, the levels would be unlikely to vary across such a wide range of re-reads. The fact that different numbers of re-reads are needed at different locations on the disk indicates that the failure mechanism produces errors distributed over a range of electromagnetic field losses, e.g., as a result of overheating due to improper storage, contamination by fungi or the loss of data with age, all of which take place over time rather than instantaneously. These are precisely the sorts of errors that the CRC codes were designed to detect.

One issue that must be addressed is the potential that repeated reads could eventually lead to a valid CRC code and no MFM errors, which would result in false sector data being accepted as legitimate. This particular scenario, because it involves weak bits, is less complicated to analyze than a scenario in which random changes are made. Specifically, the changes associated with weak bits tend to be all in one direction, which eliminates transitions and, thus, changes of 1's to 0's. The probability of lost transitions causing detections is at least 17/22 for each transition based on Table 1 (number of invalid transitions versus number of rows). Because of the nature of the CRC coding, errors that go undetected must be in quantities larger than 16 bits and distributed across the sector data area, or as combinations of the sector data area and the CRC area with a probability no higher than 1 in 216. Since the CRC and MFM methods are not correlated in any way as far as we are aware, a reasonable assumption is that the probability of both methods failing to detect a change due to reduced electromagnetic density is no greater than 1 in $2^{16} \times (5/22)^{16}$, which is less than 1 in 1,015. The probability of encountering an erroneous data recovery is low enough that even for hundreds of retries, there is almost no chance that false recovery would occur.

The above analysis ignores retries during actual recovery. Many read errors were corrected after a relatively small number of re-reads, ranging from 1 to 15 retries, with a few samples having more retries. Several sectors could be read only after about 80 retries; none took significantly more than 80 retries. Since the floppy drive does three retries per reported retry, the actual number of attempts was about 240. Exact figures are unavailable because of court orders and the examination cannot be repeated because there is no way to create another equivalent disk. It is somewhat disturbing that many sectors had on the order of 80 retries and the individuals who received the disk indicated that certain portions of the recovered blocks were corrupted. Future research should attempt to understand this problem.

The well-known birthday paradox [2] appears to be relevant to the case at hand. According to the paradox, if a group of 23 people has randomly distributed birthdays, the probability is about $\frac{1}{2}$ that two of them have the same birthday. Furthermore, the 50% probability of matching birthdays occurs when the number of samples is approximately 1.1 times the square root of the sample size (for large sample sizes). For a 16-bit CRC (65,536 possible values), the value $1.1 \times \sqrt{65536}$ is 281.6. Therefore, as the number of reads approaches 282, the probability of a collision is about 50%. However, the CRC situation is slightly different from the birthday paradox because the CRC values are not selected without replacement in the sample. Furthermore, plots of the birthday paradox, which has no known closed-form solution, show that the probability changes more or less linearly around the square root; thus, it would be unexpected to have a peak near the square root.

Some other mechanism is possibly at work, but we do not know what it is. The birthday paradox does not explain the uneven distribution of recoveries. Moreover, the CRC results are not necessarily generalizable to weak bit failures that produce less than random results. If a floppy drive is unable to detect coding errors at the level of transitions and the "Don't Care" (DC) states of the finite state machine that decodes the content do not produce errors, other sources of error likely exist, which is a potential weakness of the technique.

5. Correcting Errors

As discussed above, some blocks that are read successfully after about 80 retries are suspect. However, the errors produced by weak bits are still limited, which is very helpful.

Two approaches for error correction may be considered. One is to perform the re-read process repeatedly and match the results from multiple runs to determine if there is consistency in some portion of the bits decoded across multiple runs. The other is to determine which bits could have been altered. Unfortunately, repeated reads cause a floppy disk to degrade further because of mechanical wear. This is especially problematic when only one evidence sample exists.

At this time, we have not analyzed the errors produced by weak bits with consistent CRC codes. However, we have investigated the reconstitution of the original content, albeit to a limited extent.

Note that only 1-0 transitions can occur and only in particular locations within bit sequences. In particular, a 1-0 transition can only occur when a 11 turns into a 10 or 01, which is denoted as 11-[10/01]. Moreover, patterns appearing on decoded disk content cannot all result from lost transitions. Therefore, the candidates for lost transition changes are very limited and specific bits can be definitively determined not to have resulted from a flux density loss.

One approach for revealing the bits that could and could not have been altered by such faults is to examine all possible 11-[10/01] transitions in each re-read block and identify those that form valid parts of the code space both before and after transitions are lost. An observed 11 or 00 cannot come from such a change, so all pairs of 1's and 0's can be eliminated from the analysis, reducing the number of possible faults on a random content block by 50%.

Substantial improvements are possible when the language is known and the language has redundancy. The typical content of English, for example, is on the order of 2.3 bits per byte [4]. This means that if four bits per byte are potentially corrupted and each of the two remaining pairs could only have been produced by one of two codings, all of the original text should be recoverable. For example if the original text is

$$\begin{aligned}
\text{This}_8 &= 124\ 150\ 151\ 163 \text{ (Initial 0 bit stops intra-word effects)}\\
124_8 &= 01\ 010\ 100_2 \text{ (No valid weak bit errors)}\\
150_8 &= 01\ 101\ 000_2 \rightarrow 00\ 101\ 000_2 = \ \ 50_8 = \text{`('} = \text{T(is}\\
151_8 &= 01\ 101\ 001_2 \text{ (No valid weak bit errors)}\\
163_8 &= 01\ 110\ 011_2 \rightarrow 00\ 110\ 011_2 = \ \ 63_8 = \text{`3'} = \text{Thi3}\\
163_8 &= 01\ 110\ 011_2 \rightarrow 01\ 010\ 011_2 = 123_8 = \text{`S'} = \text{ThiS}\\
163_8 &= 01\ 110\ 011_2 \rightarrow 01\ 100\ 011_2 = 143_8 = \text{`c'} = \text{Thic}\\
163_8 &= 01\ 110\ 011_2 \rightarrow 01\ 110\ 010_2 = 162_8 = \text{`r'} = \text{Thir}\\
163_8 &= 01\ 110\ 011_2 \rightarrow 01\ 110\ 001_2 = 161_8 = \text{`q'} = \text{Thiq}
\end{aligned}$$

Figure 5. Procedure for inverting faults.

"This" in ASCII, only a few outputs can arise from missing transitions. Note that all intra-byte pairings include a 0 because ASCII is a seven-bit code and, thus, the initial 0 bit stops any 1-0 transitions from crossing the byte boundaries.

Double bit errors can be produced by weak bits in this situation, and they produce new valid codes, resulting in additional codes for "s" in "This" only. There are also other valid codes that can produce these same values from different lost transactions. For example, 161_8 (01 110 0012) can be produced by 11 110 001 and a wide range of other values that involve turning 0's into 1's. The procedure for inverting these faults involves generating the set of all possible source bytes and eliminating those that do not make sense in the language.

The procedure for inverting faults is illustrated in Figure 5. For example, several different characters can replace the "q" in "Thiq," but the only valid ones in English would be "n" and "s," corresponding to "Thin" and "This," respectively. The code for "n" is 110_8 or 01 001 0002, which cannot produce 161_8 through any combination of missed transitions. Similarly, "Thir," "Thic," "ThiS," and "Thi3" cannot be generated from "Thin," but can be generated from "This" with only 1-0 transitions. Extending this to the word as a whole, "T" and "i" cannot be altered by 1-0 failures from missed transitions, and other sources of "(" (50_8) that fit in the English word "T?i?," where the second "?" must be transformable into any one of the identified values are again limited.

6. Conclusions

The multiple read technique is effective at recovering data from failing floppy disks. It produces accurate results with a high probability in a reasonable amount of time with relatively low damage to the original evidence. Because the technique relies on normal floppy disk reads using standard unmodified equipment, it is easier to implement than exotic methods and less likely to be challenged in court.

The principal disadvantage of the technique is that repeated reads cause wear and tear. Another disadvantage is that the technique does not reveal the specific mechanism of failure even if it produces reasonable results. Also, large numbers of reads may not produce valid results for a sector, requiring the technique to be terminated manually and restarted at the next sector. Furthermore, the possibility exists that repeated reads could produce invalid data that matches the CRC codes without creating invalid MFM codes in the controller. Fortunately, in cases where the numbers of re-reads are on the order of hundreds or where reads can be completed with questioned data, the limits on 11-[10/01] transitions and language redundancy can be used to correct errors.

Avenues for future work involve automating the decoding and analysis processes and conducting a detailed investigation of multiple errors in CRC codes. While the data recovery technique is applicable to all MFM-coded media, it does not apply directly to other storage media (e.g., hard drives and CD-ROMs). Our future research will attempt to develop reliable data recovery techniques for modern storage media.

References

[1] R. Freeman, *Practical Data Communications*, John Wiley and Sons, New York, 1995.

[2] D. Knuth, *The Art of Computer Programming – Sorting and Searching*, Addison-Wesley, Reading, Massachusetts, 1973.

[3] H. Leimkuller, Computer evidence analysis and recovery of magnetic storage media data, *Proceedings of the Twenty-Ninth IEEE International Carnahan Conference on Security Technology*, pp. 147–153, 1995.

[4] C. Shannon, A mathematical theory of communications, *Bell Systems Technical Journal*, vol. 27, pp. 379–423; vol. 27, 623–656, 1948.

[5] P. Siegel, Recording codes for digital magnetic storage, *IEEE Transactions on Magnetics*, vol. 21(5), pp. 1344–1349, 1985.