ECEN 5018

Storage Technology

Second Midterm Exam

4/24/2000

Reto Zingg

1 Head positioning in magnetic and optic drives

1.1 Head structures

As the magnetic and optic heads serve a different physical purpose, their structures are also very different. The magnetic head ought to pick up tiny changes in magnetic flux from the surface of the rotating platter whereas the optic head ‘looks’ at the surface of the rotating platter in the visible or near visible spectrum of electromagnetic waves.
Both structures should be as light as possible to reduce access time. The heavier the head, the more power is needed to accelerate and decelerate it.

1.1.1 Magnetic

As the magnetic head has to pick up the magnetic flux, which is very fast degrading as the head moves away from the surface, it has to be as close as possible to the surface. The head includes two different mechanisms to write and read data (MR head).
The head contains a coil to produce a magnetic field to write a data pattern on the disk. The coil has to produce a field that is strong enough to overcome the coercive field strength and saturate the medium (see Figure 1).

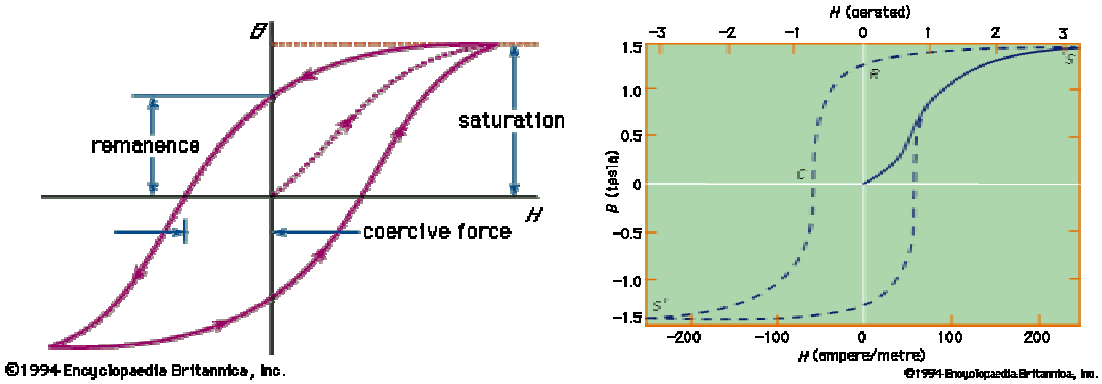


Figure 1 Hysteresis Loop [4]

To read data, the head contains a magneto resistive device. This device will change its electric resistance in presence of a magnetic field. This effect allows reading a stored data pattern from the disk.

1.1.2 Optic

The optic head, in contrast to the magnetic head, has theoretically no restriction on vertical positioning. Light (electromagnetic waves in the visible or near visible spectrum) can travel over ‘large’ distances and can be focused (not like magnetic flux). Therefore the optic can float at a large distance above the medium and need not to be controlled. The vertical control will be shifted to the focus mechanism.

1.2 Vertical head positioning

1.2.1 Magnetic

The vertical positioning of the head is achieved passively. The airflow and the pressure the head suspension provides control the distance of the head above the disk. It is passive, because no active control circuit is needed to keep the head in the correct position (vertically). The advantage is that the head can be kept small and simple. The disadvantage is that the distance of the head will depend on numerous parameters. It also can be disturbed easily by external shocks.

1.2.2 Optic

The optic head itself is not controlled in its vertical position above the disk, but the light beam needs to be focused on the data representing structure on the disk. This is much like focusing an image with your photo-camera. An active control circuitry is needed to keep the laser beam focused. As in all control circuits, an error signal is needed. There are different ways of measuring if the beam is off-focus. One way is to use a cylindrical lens and a four-quadrant sensor. If the laser beam is focused, a circular light pattern will be projected onto the four-quadrant sensor. Is the beam off-focus, the light pattern on the sensor will be stretched horizontally or vertically for long and short focuses respectively (see Figure 2).

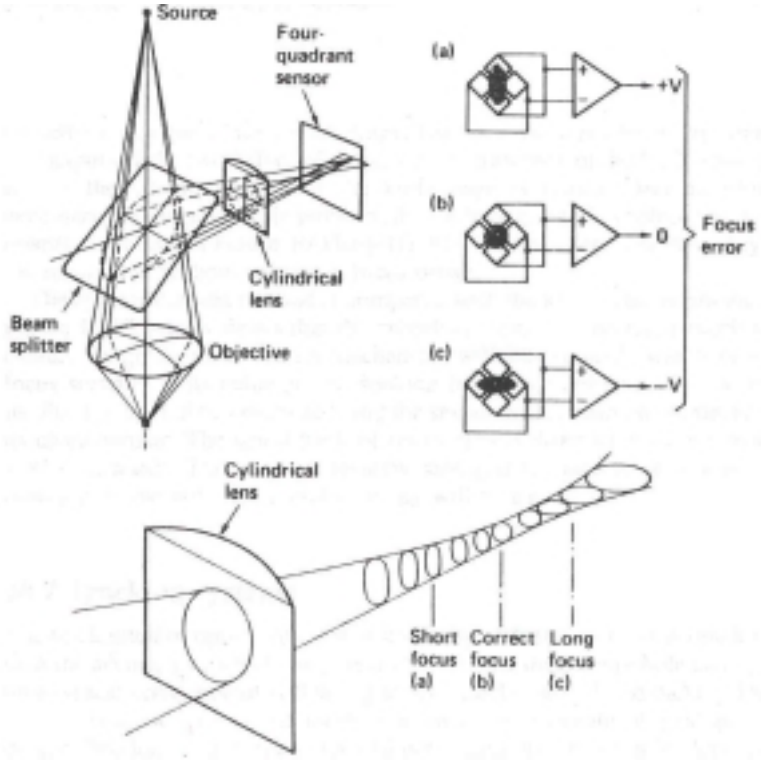


Figure 2 Four-Quadrant focus sensor [5]

An other method is to project the reflected beam over a ‘knife edge’. The ‘knife edge’ is positioned in such a manner, that if the beam is correctly focused, all the light can pass

over the edge. Is the beam off-focus, a part of it can't pass the edge and a sensor can determine if it is short or long focused (see Figure 3).

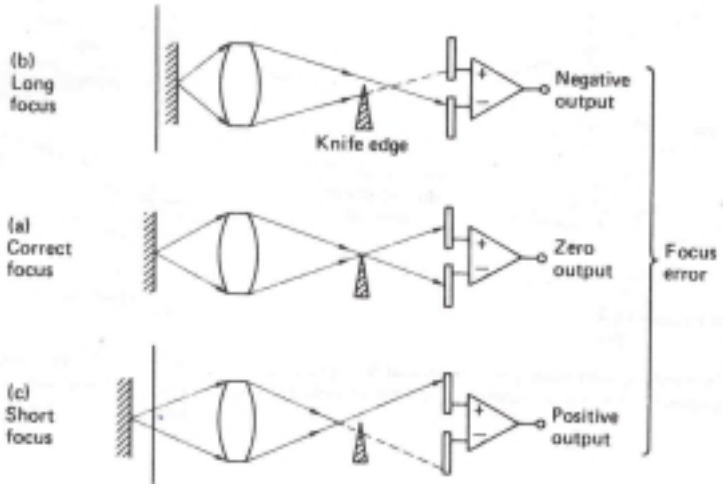


Figure 3 Knife-Edge focus sensing [5]

The advantage of the optic head is, that it can fly a 'large' distance above the disk end sense the data from this distance. The laser beam can be focused on the data through a large area of media surface. This allows overcoming dirt and scratches on the disk (see Figure 4).

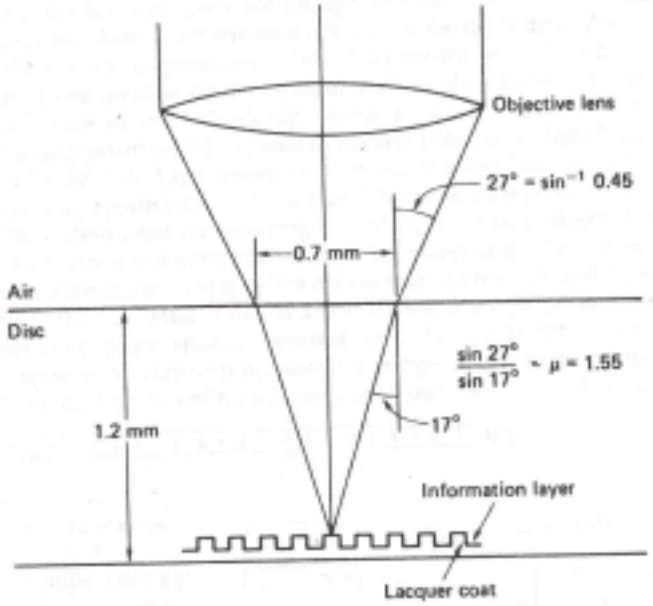


Figure 4 Sensing Laser beam through media [5]

The disadvantage is that the laser-beam needs to be focused with an active control circuit and a heavy focusing mechanism is needed on the optical head. The depth of focus is typically ±1μm and is dependent on the wavelength.

1.3 Radial head positioning

The radial positioning of the heads is the same for the magnetic and optic heads. Either a voice coil motor with a rotational arm or some 'linear' mechanism (only radial motion) is used. The difference between magnetic and optic systems is how the error signal is acquired.

1.3.1 Magnetic

To 'know' where about the head is on the disk, so-called servo spikes are pre-recorded on the disk. These spikes contain information used to center the head on the track and to determine the track and sector number (see Figure 5).

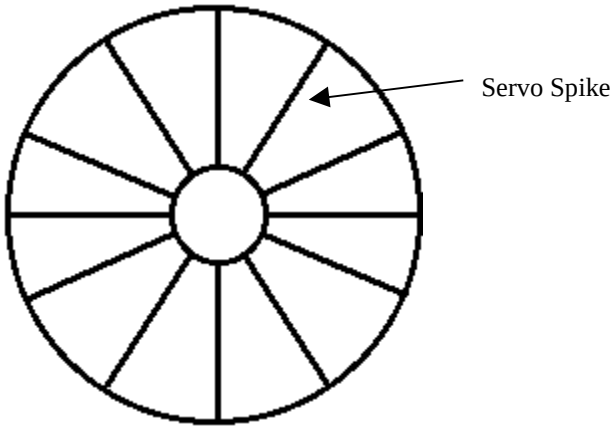


Figure 5 Servo Spikes on magnetic disk

The first part of the servo spikes contain four packets (A, B, C and D) of peak bursts. Only packet C is recorded on the center of the track. The other three packets are recorded so, that a Position Error Signal (PES) can be derived from them (see Figure 6). The PES can be acquired every time the head is flying over a servo spike. This is not a continuous signal and the head is assumed to stay on track (within a certain limit) until it passes the next servo spike.

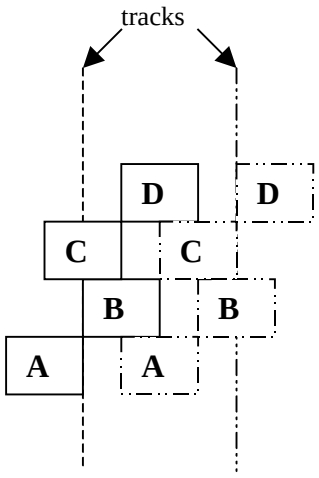


Figure 6 Servo position sensing

The second part of the servo spike contains the number of the track and sector, which is used to find the correct data once the head is positioned correctly over the track.
Let's assume a disk with 15,000 tracks per inch. Track spacing is therefore 1.7 µm. Track miss-registration should be less than twenty percent. This comes to 340 nm. Assuming a drive with 7500 RPM we get a frequency (from disk run-out) of 7500 / 60 = 125 Hz. The servomechanism sure has to be able to follow the track with this frequency. Other disturbances are periodic but with a shorter period than the disk rotation. This can be caused by disk flutter (vertical resonant movement of disk) or bearing defects. Other sources of disturbance are not periodic at all and resemble more a broad spectrum of noise. Most of these disturbances origin from head deflection. This can be due to windage, external shock, mechanical forces (and resonance) from other components and control noise. These disturbances need to be compensated by the control system. Many of these sources can only be modeled based on statistical data and have to be considered being random. Control circuits of modern disk drives have typically a bandwidth of about 800 Hz to 1 kHz. One should not forget that the PES can only be acquired at a sample rate of

$$\frac{RPM \cdot Spikes}{60} = \frac{7500 \cdot 32}{60} = 4kHz$$

Where in this example Spikes, the number of servo spikes is 32 and the disk rotates at a rate of 7500 rotations per minute. We get a sample rate of 4 kHz. From the Niquist Theorem we know that the highest frequency component we can get out of a signal sampled at 4 kHz is 2 kHz. Therefore this figure gives the upper limit in the frequency spectrum.Optic

There are several strategies to acquire an error signal from the head. In contrast to the magnetic disk, the optic disk (CD) uses the data itself to generate an error signal. The three basic methods are:

- Use three laser beams. The main laser beam reads the data, whereas two servo laser-beams, to the right and left side of the main beam, sense if the main beam is centered on the data track (see Figure 7).

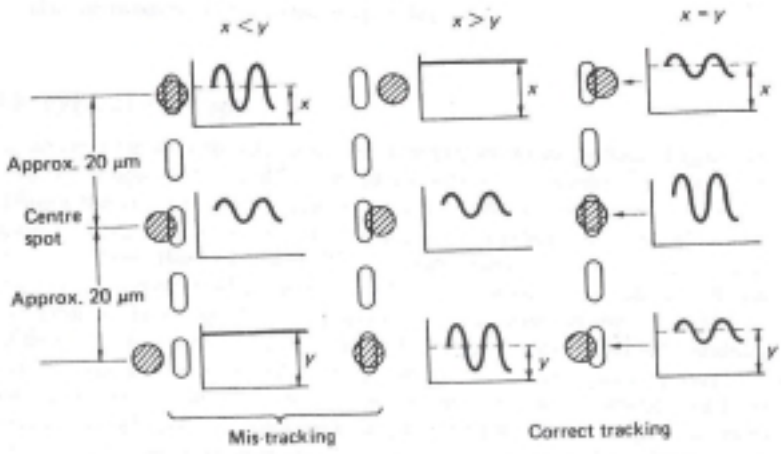


Figure 7 Three Laser PES sensing [5]

- Use a split sensor (two-zone) to determine if the laser is centered on the track. If the laser moves off-track, one side of the laser beam will be modulated less than the other side. The split sensor senses this difference and an error signal can be derived to keep the laser on track (see Figure 8).

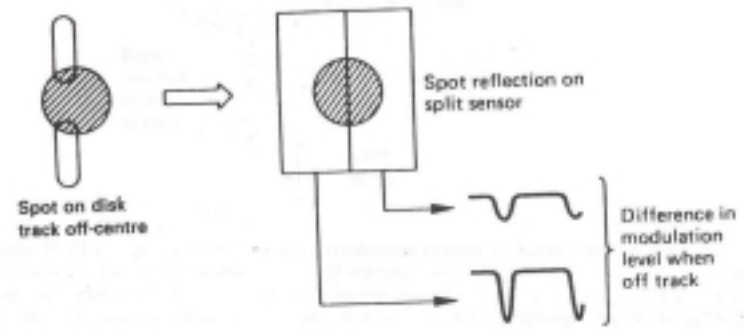


Figure 8 Split Sensor PES sensing [5]

- Wobble one main laser beam, which also is reading the data, about the center of the track to always find the radial position with the best data signal which occurs on the center of the track (see Figure 9).

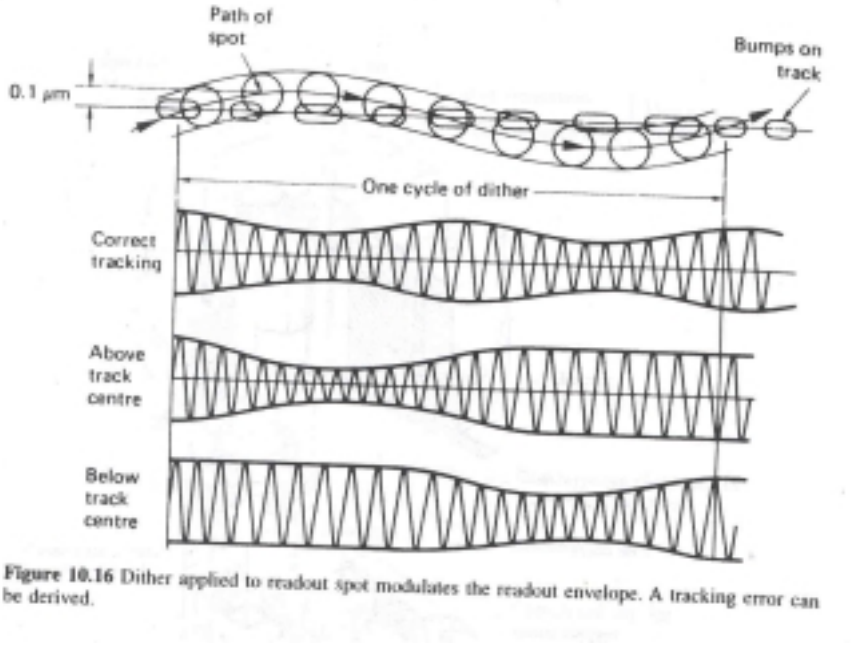


Figure 10.16 Dither applied to readout spot modulates the readout envelope. A tracking error can be derived.

Figure 9 Wobble PES sensing [5]

The track pitch on an optical disk is typically in the order of 1 μm (CD: 1.6 μm, one continuous track, 22188 times crossing a radius of the CD). The spot size of a sensing laser beam on the layer of data has a diameter of about 1.2 μm (dependent on wavelength of laser and aperture size of the lens).
Like with the magnetic disk drive the TMS noise has different sources which can be divided into periodic and non-periodic ones. The periodic noise includes disk slip, disk flutter and bearing defects. Disk flutter in an optical drive with removable media can be very large, as the medium has to be centered with each loading action. Non-periodic, wide spectrum noise includes control noise, external shock and disturbance from other mechanical parts of the drive assembly.
Taking a CD as example we can calculate the frequency of disk rotation related noise. We get the average RPM of a CD with:

$$\frac{Tracks}{Playtime} = \frac{22188}{72 \cdot 60} = 5.136Hz$$

This frequency is much lower than the one of a modern magnetic disk drive. But computers of course don't use single speed CDs anymore. CD drives with a maximum speed of 50 times the one of an audio CD can be bought off the shelve. These drives will have a basic noise frequency of 50*5.136=256.8 Hz. Noise from disk flutter will be a multiple of this frequency (depending on the mode). Such a CD drive will need a control circuit with a bandwidth in the order of 1 kHz to 2 kHz.
In contrast to the magnetic drive the PES is continuous and it can be sampled at an arbitrary rate. This is because the PES is derived from the 'continuous' data pattern.

2 Channel coding and error correction

2.1 Why we need channel coding

Many data channels can not carry a DC signal. Therefore the user of this channel needs to ensure that the data signal is DC free. This is not very convenient, as we want to send certain data and don't want to be restricted on data that's DC free. The solution is the so-called channel coding (see Figure 10). The channel-coding algorithm takes the user data and transforms it in such a manner, that the data channel can carry it. The other reason is that we want to transmit data without sending a separate clock signal. The channel data stream should be 'self clocking'. This requires a transition in the channel data within a specified maximum time (or speaking in the frequency domain it requires a minimum signal frequency).

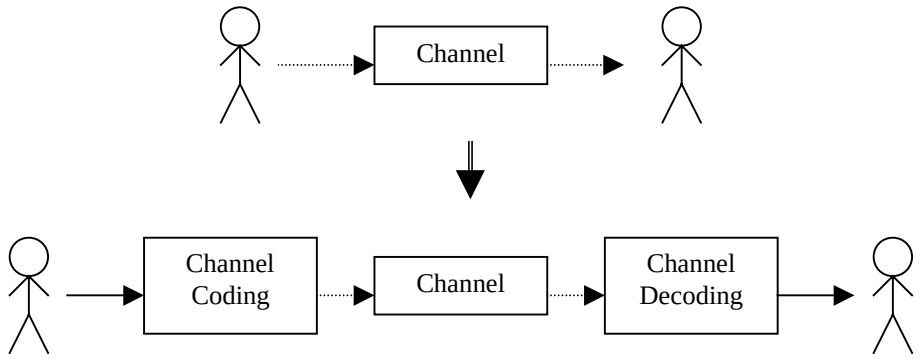


Figure 10 Adding Channel Coding to simplify use of Channel

In order to give the (user) data stream the properties that are needed for the channel some data (or bandwidth) needs to be added. Therefore the user data rate will be lower than the channel data rate.

2.2 Manchester coding as an example of channel coding

The rules to form a Manchester code are as follows (see also Figure 11):

- A logic '1' is represented by a signal level sequence of '+1 -1'. This defines the channel symbol for a logic user data bit '1'.
- A logic '0' is represented by a signal level sequence of '-1 +1' This defines the channel symbol for a logic user data bit '0'.

This results in a channel data stream that is DC-free (average signal level is zero) as each user data bit in itself results in a DC free channel symbol. Every channel symbol contains a level transition. This transition can be used to recreate the bit-clock. The disadvantage is that the bit rate is doubled.
Figure 11 shows the signal levels for the Manchester code and the doubling of the data rate.

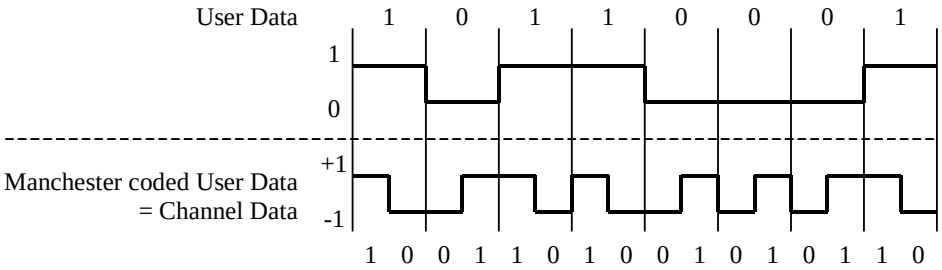


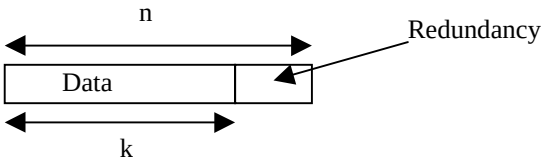
Figure 11 Manchester coding

A side effect of Manchester coding is that we added 100% redundancy. For each user data bit we send two bits over the channel. On the receiver side we can use the added redundancy. A channel symbol of two bit length represents one bit of data. The channel symbols '+1 -1' and '-1 +1' represent our data. The two other unused possible channel symbols '+1 +1' and '-1 -1' we declare as illegal. If we receive an illegal symbol we know that a bit error occurred. This leads us to the next section dealing with error correcting codes.

2.3 Introduction to error correcting codes

Error correcting codes add redundancy to data. The receiver uses the redundancy to detect and correct errors. We can define:

k:	data block size
n:	channel-symbol size
$M=2^k$	number of used channel-symbols (for binary information)
2^n	number of possible channel-symbols (for binary information)



Because of the added redundancy the number of possible channel symbols is larger than the number of channel symbols we need to represent the data. Out of the set of 2^n symbols, we choose 2^k symbols to represent the data. These 2^k symbols out of 2^n symbols we choose so that they have a maximum 'distance' to each other (they should be as different to each other as possible, so they are better distinguishable).

Example: To overcome the sometimes poor quality of radio transmission the phonetic alphabet was introduced. This allows communication with little misunderstandings (error rate) even over a 'bad' link.
In this example, the data block size k is one character. Therefore, the number of used channel symbols is 26 (number of characters we want to represent). The channel symbol size n is one (short) word. The number of possible channel symbols is very large (does anybody know how many words there are?).
The words are chosen so that they are a large 'distance' apart, in other words, they are easily distinguishable. No good choice of channel symbols would be

'Cheese' for the character C and 'Please' for the character P. These two words are very 'close' together (they are not easily distinguishable).

Table 1 Ponetic Alphabet [3]

Alpha	Juliet	Sierra
Bravo	Kilo	Tango
Charlie	Lima	Uniform
Delta	Mike	Victor
Echo	November	Whiskey
Foxtrot	Oscar	Xray
Golf	Papa	Yankee
Hotel	Quebec	Zulu
India	Romeo	

What the additional redundant information of an error correction code is doing is illustrated in the figure below:

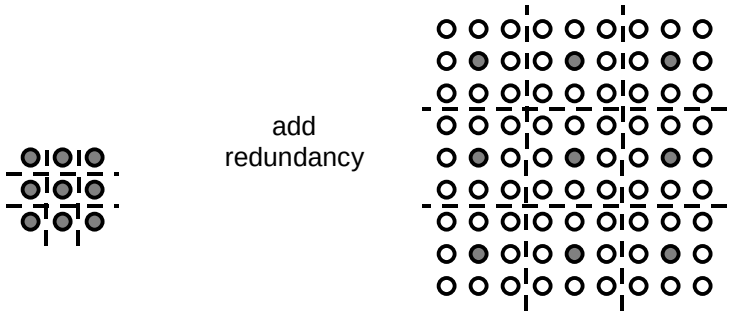


Figure 12 Code words evenly distributed over symbol space

The redundant information 'spreads out' the data representing words over a larger word space. This allows the receiver of the information in the case of disturbance to guess the correct data. The added redundancy of course needs to be very special in order to achieve the above properties. The redundant information should be easily calculated and on the receiver side the 'guessing' of the correct answer should be in form of an algorithm. Very different such codes and algorithms have been proposed and implemented. In the next section we will have a look at the Hamming (7,4,3) code.

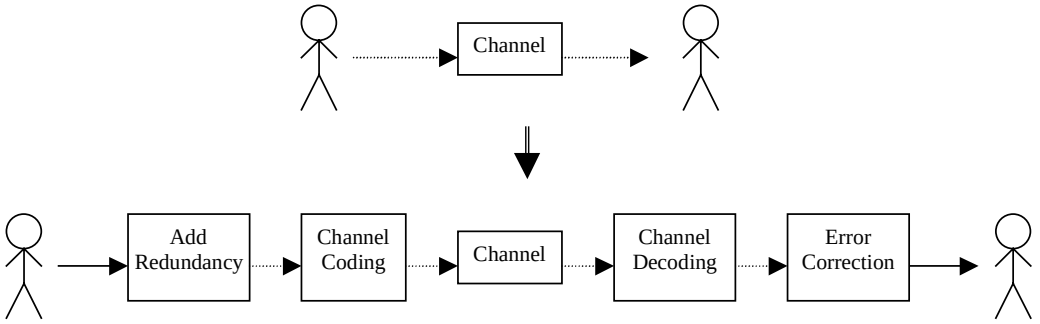


Figure 13 Error correction on the system level

2.4 Hamming Distance

The so-called Hamming distance is the distance between any two code words. In the above example of the phonetic alphabet the Hamming distance would define how different one word to the other is.

Example: The Hamming distance between 'Charlie' and 'Papa' is much larger that the Hamming distance between 'Cheese' and 'Please'.

In a binary code we can define the Hamming distance as the number of bits that are different from one code word to another code word.

Example: The Hamming distance between the code words '100101' and '110100' is two, as bits are different.

We can illustrate the Hamming distance of two code words as shown in Figure 14.

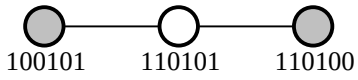


Figure 14 Hamming distance of two

Of interest is the minimum Hamming distance of a code.

Example: Assuming a code with a minimum Hamming distance of two we get properties as follows: Each code word has at least a distance of two (see Figure 14) to each other code word in that code. If we receive a symbol that is between two code words (e.g. '110101' in Figure 14) we know that a bit error occurred. Therefore this code can detect a single bit error.

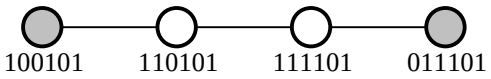


Figure 15 Hamming distance of three

Example: Assuming a code with a minimum Hamming distance of three we get properties as follows: Each code word has at least a distance of three (see Figure 15) to each other code word in that code. If we receive a symbol that is between two code words (e.g. '110101' in Figure 15) we know that a bit error occurred. We also can guess that '100101' was the correct data as it is the least distance from the received symbol. If two bit errors occurred, we would 'correct' in the wrong direction. Therefore this code can detect and correct a single bit error.

In analogy of the two above examples the Hamming distance can be extended and more errors can be detected and / or corrected. Of course we need more redundancy to provide the 'filling' symbols between the accepted code words.

The aim is to find $M=2^k$ code words out of 2^n symbols that are evenly as far apart as possible. In other words, we want to distribute the code words as evenly as possible over the space of possible symbols (see Figure 12).

2.5 The Binary Hamming (7,4,3) Code

The binary Hamming (7,4,3) code takes a data block of length 4 and adds 3 bit of redundancy forming a 7-bit channel symbol. This code results in a Hamming distance of 3 (therefore the name (7,4,3)).

2.5.1 Coding

The three added bits (C_1, C_2, C_3) each represents a parity bit of selected three of the four data bits. These parity bits then are appended to the four data bits resulting in the seven-bit code word (see Figure 16).

Data:	D_1	D_2	D_3	D_4	
Parity bit 1:	$D_1 \oplus$	$D_2 \oplus$	D_3	D_4	$= C_1$
Parity bit 2:	$D_1 \oplus$	$D_2 \oplus$	D_3	D_4	$= C_2$
Parity bit 3:	$D_1 \oplus$	D_2	$D_3 \oplus$	D_4	$= C_3$
Code word:	D_1	D_2	D_3	D_4	C_1 C_2 C_3

Figure 16 Generation of the Hamming (7,4,3) code

The ⊕ symbol represents here a modulo two addition (i.e. 0⊕0=0, 0⊕1=1, 1⊕1=0). This is equivalent to the exor function.
The operation shown in Figure 16 can be written as a matrix operation. We define the data vector u=(D_1, D_2, D_3, D_4) and the generator matrix G

$$u = \begin{bmatrix} D_1 & D_2 & D_3 & D_4 \end{bmatrix}$$

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

To obtain the code word v we only need to multiply the data vector u with the generator matrix G:

$$v = u \cdot G$$

Example: We calculate v of the data '1010'

$$v = \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

because:
(1*1 ⊕ 0*0 ⊕ 1*0 ⊕ 0*0) = 1
(1*0 ⊕ 0*1 ⊕ 1*0 ⊕ 0*0) = 0
(1*0 ⊕ 0*0 ⊕ 1*1 ⊕ 0*0) = 1
(1*0 ⊕ 0*0 ⊕ 1*0 ⊕ 0*1) = 0
(1*1 ⊕ 0*1 ⊕ 1*1 ⊕ 0*0) = 0
(1*1 ⊕ 0*1 ⊕ 1*0 ⊕ 0*1) = 1
(1*1 ⊕ 0*0 ⊕ 1*1 ⊕ 0*1) = 0

We see that the first four columns of the generator matrix pick the four data bits. The last three columns define over which three data bits the checksum should be taken.

2.5.2 Decoding

Now the receiver of the code word v could take only the first four bits of v, which represent u. This of course wouldn't make use of the error correcting ability of the added redundancy. So let's use the three redundant bits to check the received code word. To make it more interesting we put an error in our code word v:

Code word:	1	0	1	0	0	1	0
Error:	0	1	0	0	0	0	0
Received code word:	1	1	1	0	0	1	0

Doing again the parity check over the data bits we get:

$C_1 = (1*1 \oplus 1*1 \oplus 1*1 \oplus 0*0) = 1$

$C_2 = (1*1 \oplus 1*1 \oplus 1*0 \oplus 0*1) = 0$

$C_3 = (1*1 \oplus 1*0 \oplus 1*1 \oplus 0*1) = 0$

Now we see that we have a mismatch of the received check bits and the calculated check bits C_1 and C_2 whereas C_3 is correct. From this we can conclude that bit 2 is wrong, as C_1 and C_2 include bit 2 but C_3 does not.

As with coding we can use a matrix to analyze the received data. We define the check matrix H:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

By multiplying the check matrix H with the code word (column) vector v we get the so-called syndrome vector s:

$$s = H \cdot v = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

If the received code word v had been correct, the syndrome would have been zero. A non-zero syndrome indicates therefore an error. Each syndrome (001…111) denotes a different syndrome (a different bit is incorrect). In fact the check bits C_1, C_2 and C_3 can be inserted between the four data bits in such a manner that the resulting syndrome will give the binary number of the erroneous bit.

3 Reference and Sources

[1] Digital Communications Fundamentals and Applications
Bernard Sklar, Prentice Hall
ISBN0-13-211939-0
[2] http://www.mindspring.com/~gwil/phon.menu.html
[3] http://schof.colorado.edu/~ecen5682/
[4] http://www.britannica.com
[5] Handouts from ECEN-5018
[6] Data Communication for Engineers
Michael Duck, Peter Bishop, Richard Read
ISBN0-201-42788-5