

Channel Coding  
Linda Doyle  
CTVR

Channel Coding

- We code the transmitted data to make it robust to errors - over the years many researcher have designed various different strategies for coding.
- There are two main kinds of coding - **block coding** and **convolution coding**.

coding

- In block coding as the name suggests the data is organised into blocks and encoded
- In convolutional coding the incoming stream of data is processed continuously

error detection & error correction

- on the receiver side the encoded data can be used to detect errors
- on the detection of errors the receiver can take action
- there are typically two broad strategies - ARQ and FEC approaches

ARQ approaches

- the main idea is that the receiver asks the transmitter to repeat the transmission



FEC approaches

- Forward error correction = FEC
- The main idea is that the transmitter corrects the errors





“Wait! Wait! Wait! ... Cancel that, I guess it says 'help.'”

so we are going to look at some codes

- we need to understand some basic principles about error detection and correction
- we are interested in binary codes
- we need modulo N arithmetic for many of our calculations

## some background

### modulo-N arithmetic

- Modulus N: the upper limit
- In modulo-N arithmetic, we use only the integers in the range 0 to N -1, inclusive.
- If N is 2, we use only 0 and 1
- No carry in the calculation (sum and subtraction)

### modulo-N arithmetic is used

$$0 \oplus 0 = 0 \qquad 1 \oplus 1 = 0$$

a. Two bits are the same, the result is 0.

$$0 \oplus 1 = 1 \qquad 1 \oplus 0 = 1$$

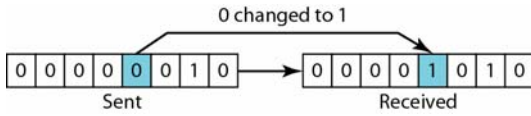
b. Two bits are different, the result is 1.

$$\begin{array}{r}
 1\ 0\ 1\ 1\ 0 \\
 \oplus\ 1\ 1\ 1\ 0\ 0 \\
 \hline
 0\ 1\ 0\ 1\ 0
 \end{array}$$

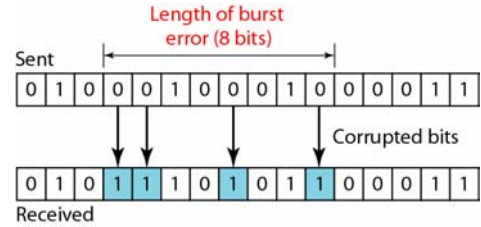
c. Result of XORing two patterns

### symbols

- in some books a simple + is used for the XORing operation
- in other books the symbol  $\oplus$  is used



In a single-bit error, only 1 bit in the data unit has changed.

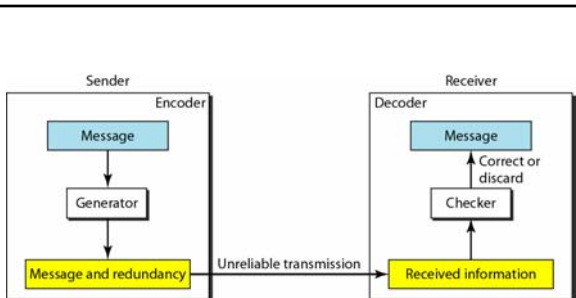


A burst error means that 2 or more bits in the data unit have changed.

we need codes for two purposes

- Error detection
  - Check if any error has occurred
  - Don't care the number of errors
  - Don't care the positions of errors
- Error correction
  - Need to know the number of errors
  - Need to know the positions of errors
  - More difficult

## block codes



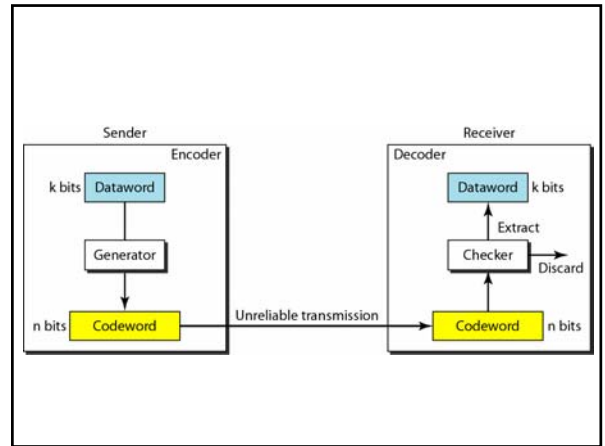
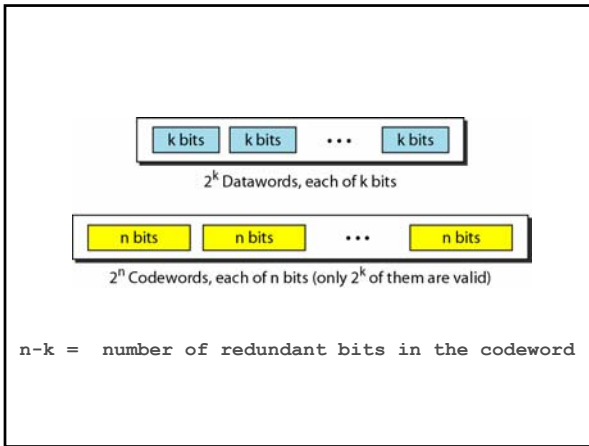
To detect or correct errors, we need to send extra (redundant) bits with data.

## block codes

In block coding, we divide our message into blocks, each of  $k$  bits, called **datawords**.

We add  $r$  redundant bits to each block to make the length  $n = k + r$ .

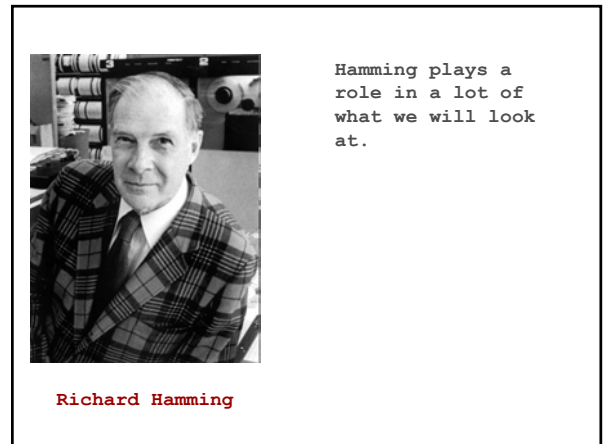
The resulting  $n$  bit blocks are called **codewords**.



Datawords	Codewords
00	000
01	011
10	101
11	110

Dataword	Codeword
00	00000
01	01011
10	10101
11	11110



**a small aside**

- At a seminar in the Bell Communications Research Colloquia Series, Dr. Richard W. Hamming, a Professor at the Naval Postgraduate School in Monterey, California and a retired Bell Labs scientist, gave a very interesting and stimulating talk, 'You and Your Research' to an overflow audience of some 200 Bellcore staff members and visitors at the Morris Research and Engineering Center on March 7, 1986.
- This talk centered on Hamming's observations and research on the question "Why do so few scientists make significant contributions and so many are forgotten in the long run?"

**some quotes**

"When you are famous it is hard to work on small problems. This is what did Shannon in. After information theory, what do you do for an encore? The great scientists often make this error. They fail to continue to plant the little acorns from which the mighty oak trees grow. They try to get the big thing right off. And that isn't the way things go."

"I worked for ten years with John Tukey at Bell Labs. He had tremendous drive. One day about three or four years after I joined, I discovered that John Tukey was slightly younger than I was. John was a genius and I clearly was not. Well I went storming into Bode's office and said, "How can anybody my age know as much as John Tukey does?" He leaned back in his chair, put his hands behind his head, grinned slightly, and said, "You would be surprised Hamming, how much you would know if you worked as hard as he did that many years." I simply slunk out of the office!"

"Now Alan Chynoweth mentioned that I used to eat at the physics table. I had been eating with the mathematicians and I found out that I already knew a fair amount of mathematics; in fact, I wasn't learning much. The physics table was, as he said, an exciting place, but I think he exaggerated on how much I contributed. It was very interesting to listen to Shockley, Brattain, Bardeen, J. B. Johnson, Ken McKay and other people, and I was learning a lot. But unfortunately a Nobel Prize came, and a promotion came, and what was left was the dregs. Nobody wanted what was left. Well, there was no use eating with them!"

<http://www.cs.virginia.edu/~robins/YouAndYourResearch.html>

### Hamming Distance

- The Hamming distance between two words is the number of differences between corresponding bits.
- The minimum Hamming distance is the smallest Hamming distance between all possible pairs in a set of words.

### examples

- The Hamming distance  $d(000, 011)$  is 2 because

$$000 \oplus 011 \text{ is } 011 \text{ (two 1s)}$$

- The Hamming distance  $d(10101, 11110)$  is 3 because

$$10101 \oplus 11110 \text{ is } 01011 \text{ (three 1s)}$$

We count the number of 1s in the Xoring of two words

Find the minimum Hamming distance of the coding scheme in Table 10.1.

Datawords	Codewords
00	000
01	011
10	101
11	110

Solution

We first find all Hamming distances.

$$\begin{array}{llll} d(000, 011) = 2 & d(000, 101) = 2 & d(000, 110) = 2 & d(011, 101) = 2 \\ d(011, 110) = 2 & d(101, 110) = 2 & & \end{array}$$

The  $d_{\min}$  in this case is 2.

Find the minimum Hamming distance of the coding scheme in Table 10.2.

Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

$d(00000, 01011) = 3$     $d(00000, 10101) = 3$     $d(00000, 11110) = 4$   
 $d(01011, 10101) = 4$     $d(01011, 11110) = 3$     $d(10101, 11110) = 3$

The  $d_{\min}$  in this case is 3.

So What?

- Why do we care about the Hamming distance?
- To guarantee the detection of up to  $s$  errors in all cases, the minimum Hamming distance in a block code must be  $d_{\min} = s + 1$ .
- Why?

Datawords	Codewords
00	000
01	011
10	101
11	110

The minimum Hamming distance for our first code scheme is 2. This code guarantees detection of only a single error.

For example, if the third codeword (101) is sent and one error occurs, the received codeword does not match any valid codeword.

If two errors occur, however, the received codeword may match a valid codeword and the errors are not detected.

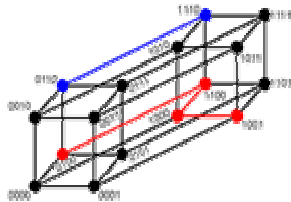
Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

Here  $d_{\min} = 3$ . This code can detect up to two errors.

When any of the valid codewords is sent, two errors create a codeword which is not in the table of valid codewords.

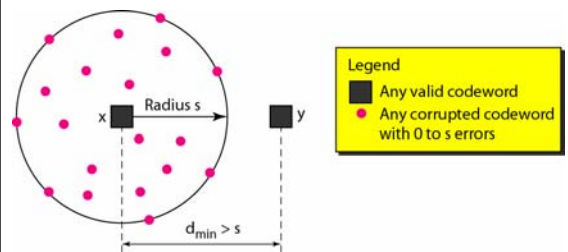
The receiver cannot be fooled.

What if there are three error occurrences?



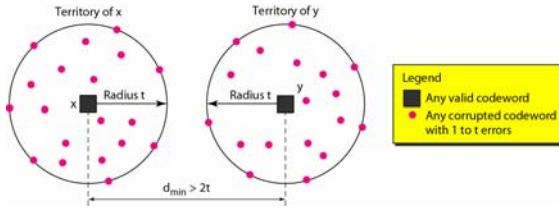
Two example distances: 0100→1001 has distance 3 (red path); 0110→1110 has distance 1 (blue path)

we can think of the error detection situation graphically



Legend  
 ■ Any valid codeword  
 ● Any corrupted codeword with 0 to  $s$  errors

we can think of the error correction situation graphically



To guarantee correction of up to  $t$  errors in all cases, the minimum Hamming distance in a block code must be  $d_{\min} = 2t + 1$ .

A code scheme has a Hamming distance  $d_{\min} = 4$ . What is the error detection and correction capability of this scheme?

Solution

This code guarantees the detection of up to three errors ( $s = 3$ ), but it can correct up to one error.

In other words, if this code is used for error correction, part of its capability is wasted.

**Error correction codes need to have an odd minimum distance (3, 5, 7, . . .).**