EITN21, PWC part 2
Lecture: Project overview and cyclic redundancy check (CRC) codes

Fredrik Rusek, Lund University

Project Overview

Goal of tasks 1 and 2:

Transfer a large file via speakers and mics from one computer to another

System should include:

- OFDM, minimum 64 carriers
- Packet based system
- ARQ, i.e., receiver should send ACK/NACK for each packet.
- Re-transmissions of the incorrectly received packets
- Cyclic redundancy check (CRC) code
- Minimum bit-rate during transmissions: 0.5 kbit/s
- Convolutional code is optional (you need to find out yourself if you need it or not)
- Minimum size of file: 20kbit
- Max packet length: 1kbit

Project Overview

Main goal is divided into two parts

- Task 1 – the basic link: Implement the OFDM part and send a data sequence from one computer to the other and decode.
- Deadline is Friday Dec 4 for task 1.

- Task 2 – the advanced link: Implement the packet based full duplex system with ARQ.
- Deadline is Friday Jan 15 (2016) for task 2.

A problem to be encountered in task 2

One particular problem with task 2 is that in Matlab, one have to record sound for a pre-defined amount of time.

Since this is a ”Matlab-problem” and not a ”communication-theory-problem”, you are **allowed to make use of the built in clock-function** in matlab. The internal clocks of the receiver and the transmitter are allowed to be synchronized with each other.

If you choose to use C/C++, this problem is completely alleviated since one can then record sound ”until something happens” – for example ”until there is no sond to record”. You can also start recording when ”there is something to record”

However, the overhead of using C/C++ is rather large if you are not experienced.

Joao will demonstrate how to handle this issue with the clock-function.

CRC

A CRC is used for ***error detection***, not for error correction.

Example: Single parity check bit

Suppose one wants to transmit the 5 bits

[0 0 1 0 1]

If one receives the bits

[0 0 **0** 0 1]

This error will pass by un-detected.

CRC

A CRC is used for ***error detection***, not for error correction.

Example: Single parity check bit

Suppose one wants to transmit the 5 bits

[0 0 1 0 1]

If one receives the bits

[0 0 **0** 0 1]

This error will pass by un-detected.

We can fix this by adding a single parity bit so that the total number of 1s is always even.

We then have

[0 0 1 0 1 **0**] ***Parity bit***

Total number of 1s = even

CRC

A CRC is used for ***error detection***, not for error correction.

Example: Single parity check bit

Suppose one wants to transmit the 5 bits

[0 0 1 0 1]

If one receives the bits

[0 0 **0** 0 1]

This error will pass by un-detected.

We can fix this by adding a single parity bit so that the total number of 1s is always even.

We then have

[0 0 1 0 1 **0**] ***Parity bit***

If we receive

[0 0 0 0 1 **0**] ***Total number of 1s = odd -> not correct***

We know that there has been 1 bit error on the channel, and we will ask for a re-transmission

CRC

The previous example was just meant as illustration, and in reality, much more advanced systems are used. **But, they are based upon the same principle!**

Suppose that we should send K bits, [u_0... u_{K-1}]. We denote these by the D-transform

$$\mathrm{u}(D) \stackrel{\text{def}}{=} u_{K-1}D^{K-1} + u_{K-2}D^{K-2} + \cdots + u_0$$

Hence, $1 + D + D^6 = [1\ 1\ 0\ 0\ 0\ 0\ 1]$ etc etc

The powers of the indeterminate D can be thought of as keeping track of which bit is which. The CRC is represented by another polynomial,

$$\mathrm{c}(D) \stackrel{\text{def}}{=} c_{L-1}D^{L-1} + c_{L-2}D^{L-2} + \cdots + c_0$$

The entire frame of data and CRC is then $x(D) = u(D)D^L + c(D)$, that is

$$\mathrm{x}(D) \stackrel{\text{def}}{=} u_{K-1}D^{L+K-1} + u_0D^L + c_{L-1}D^{L-1} \ldots + c_0$$

How to find c(D)

The check bits c(D) depend of course on u(D).

Question: How to find c(D) given a particular u(D) in a structured fashion?

How to find c(D)

The check bits c(D) depend of course on u(D).

Question: How to find c(D) given a particular u(D) in a structured fashion?

Define a generator polynomial g(D) of degree L

$$\boxed{g(D) \overset{\text{def}}{=} D^L + g_{L-1}D^{L-1} + \cdots + g_1 D + 1}$$

For a given g(D), the mapping from u(D) to the CRC c(D) is given by

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

Long Division

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

- This is just an ordinary long division of one polynomial with another.
- All operations are modulo 2. Thus (1+1) mod 2 = 0, and (0-1) mod 2 = 1.
- Subtraction using modulo 2 arithmetic is the same as addition

Example:

$$D^3 + D^2 + 1 \,\overline{\big)\, D^5 + \qquad D^3}$$

Find remainder of $\boldsymbol{D^5 + D^3}$ *divided with* $\boldsymbol{D^3 + D^2 + D}$

Long Division

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

- This is just an ordinary long division of one polynomial with another.
- All operations are modulo 2. Thus (1+1) mod 2 = 0, and (0-1) mod 2 = 1.
- Subtraction using modulo 2 arithmetic is the same as addition

Example:

$$\begin{array}{r|l} & D^2 \\ \hline D^3 + D^2 + 1 & D^5 + \qquad D^3 \end{array}$$

Long Division

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

- This is just an ordinary long division of one polynomial with another.
- All operations are modulo 2. Thus (1+1) mod 2 = 0, and (0-1) mod 2 = 1.
- Subtraction using modulo 2 arithmetic is the same as addition

Example:

$$\begin{array}{r|l} & D^2 \\ \hline D^3+D^2+1 & D^5 + \quad\quad D^3 \\ & +D^5 + D^4 + \quad\quad D^2 \\ \hline & \quad\quad D^4 + D^3 + D^2 \end{array}$$

Long Division

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

- This is just an ordinary long division of one polynomial with another.
- All operations are modulo 2. Thus (1+1) mod 2 = 0, and (0-1) mod 2 = 1.
- Subtraction using modulo 2 arithmetic is the same as addition

Example:

$$\begin{array}{r|l}
 & D^2 + D \\ \hline
D^3 + D^2 + 1 & D^5 + \qquad D^3 \\
 & + D^5 + D^4 + \qquad D^2 \\ \hline
 & \qquad D^4 + D^3 + D^2
\end{array}$$

Long Division

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

- This is just an ordinary long division of one polynomial with another.
- All operations are modulo 2. Thus (1+1) mod 2 = 0, and (0-1) mod 2 = 1.
- Subtraction using modulo 2 arithmetic is the same as addition

Example:

$$
\begin{array}{r|llllll}
 & D^2 + D & & & & & \\
\cline{2-5}
D^3 + D^2 + 1 & D^5 + & & D^3 & & & \\
 & + D^5 + & D^4 + & & D^2 & & \\
\cline{2-5}
 & & D^4 & + D^3 + & D^2 & & \\
 & & + D^4 & + D^3 + & & D & \\
\cline{3-6}
 & & & & D^2 & + D & = Remainder
\end{array}
$$

Long Division

$$c(D) = Remainder\left[\frac{u(D)D^L}{g(D)}\right]$$

- This is just an ordinary long division of one polynomial with another.
- All operations are modulo 2. Thus (1+1) mod 2 = 0, and (0-1) mod 2 = 1.
- Subtraction using modulo 2 arithmetic is the same as addition

Example:

quotient

$$\begin{array}{r|llllll}
 & D^2 + D & & & & & \\
\hline
D^3 + D^2 + 1 & D^5 + & & D^3 & & & \\
 & + D^5 + & D^4 + & & D^2 & & \\
\cline{2-6}
 & & D^4 & + D^3 + & D^2 & & \\
 & & + D^4 & + D^3 + & & D & \\
\cline{3-6}
 & & & & D^2 + D & = Remainder &
\end{array}$$

$$\boldsymbol{D^5 + D^3 = (D^2 + D)(D^3 + D^2 + 1) + (D^2 + D)}$$

CRC

Let z(D) denote the quotient. We then have:

$$u(D)D^L = g(D)z(D) + c(D)$$

Subtract c(D) from both sides and use ”+” = ”-” in modulo 2 arithmetic

$$x(D) = u(D)D^L + c(D) = g(D)z(D)$$

Thus, all valid code words x(D) are divisible by g(D)

Receiver operation

Assume x(D) is transmitted and that y(D) is received. Let the errors on the channel be e(D). Hence, y(D)=x(D)+e(D).

The receiver knows that a valid y(D) should leave no remainder if divided by g(D).
So, the receiver declares:

ACK if $\boldsymbol{Remainder\,\frac{y(D)}{g(D)} = 0}$

NACK if $\boldsymbol{Remainder\,\frac{y(D)}{g(D)} \neq 0}$

When does it fail?

Since we have shown that x(D) is divisible by g(D),

$$Remainder\,\frac{y(D)}{g(D)} = Remainder\left[\frac{x(D)+e(D)}{g(D)}\right] = Remainder\;\frac{e(D)}{g(D)}$$

If no errors occur, i.e., e(D)=0, then this remainder is zero, and the receiver declares a successful transmission.

If e(D) is not zero, the receiver fails to detect the error only if Rem[e(D)/g(D)]=0. This is the same as saying that e(D) is a valid code word, i.e.,

$$e(D) = g(z)z(D)$$

For some non-zero polynomial z(D)

When does it fail?

Suppose that a single error occurs, i.e., $e(D) = D^i$, for some integer *i.*

We have an un-detected error if and only if

$$e(D) = g(D)z(D) \quad \text{for some z(D)}$$

But since g(D) have at least two non-zero terms (1 and D^L), so must g(D)z(D) have.

When does it fail?

Suppose that a single error occurs, i.e., $e(D) = D^i$, for some integer *i.*

We have an un-detected error if and only if

$$e(D) = g(D)z(D) \quad \text{for some z(D)}$$

But since g(D) have at least two non-zero terms (1 and D^L), so must g(D)z(D) have.
Hence, g(D)z(D) cannot possibly equal D^i and we can conclude

All single event errors are detectable

Burst errors

We next consider bursts of errors **e**=[...000 $\underbrace{\mathbf{1}10\ 10110\ \ldots0110\mathbf{1}}_{P}$ 000....]

We know that it will pass un-detected if and only if e(D) = g(D) z(D) for some z(D)

But,

$$(D^L + \cdots + 1)(D^j + \cdots + D^i) = D^{L+j} + \cdots + D^i$$

g(D) z(D)

Hence, g(D)z(D) will consist of a burst of at least length L.

Burst errors

We next consider bursts of errors **e**=[...000 $\underbrace{\mathbf{1}10\ 10110\ \ldots 0110\mathbf{1}}_{P}$ 000....]

We know that it will pass un-detected if and only if e(D) = g(D) z(D) for some z(D)

But,

$$(D^{L}+\cdots+1)(D^{j}+\cdots+D^{i}) = D^{L+j}+\cdots+D^{i}$$

g(D) z(D)

Hence, g(D)z(D) will consist of a burst of at least length L.

If P<L, e(D)=g(D)z(D) is not possible!

All error bursts of length L and less are detectable

What about double errors?

What about e(D) of the type $D^j + D^i$?

We already know that if j-i<L+1, then it is detectable.

For j-i>L, more advanced theory must be used (theory of finite fields – Galois theory)

When the smoke clears, the result is

if g(D) is *primitive*, all double errors are detectable (if $K < 2^L - 1$)

Some known results

With a primitive g(D) times (1+D), that is g(D) =(1+D)g_p(D)

- All single and double errors are detectable
- Burst-detecting capability of at least L
- Probability of detecting a completely random e(D): 2^{-L}

Standard g(D)s with L=16.

- $D^{16} + D^{15} + D^2 + 1$ CRC-16
- $D^{16} + D^{12} + D^5 + 1$ CRC-CCITT

CRC in Matlab

Luckily, there is Matlab

Play around with the CRC-class (just type ***help crc*** in Matlab)