Department of Electrical & Computer Engineering

ELEC 691X/498X – Broadcast Signal Transmission Winter 2018

Instructor: DR. Reza Soleymani, Office: EV-5.125,
Telephone: 848-2424 ext.: 4103.

Office Hours: Wednesday, Thursday, 14:00 – 15:00

Lecture 7:
Error Control Coding

Department of Electrical & Computer Engineering

In this lecture we cover the following topics:

- Error Control Coding: FEC versus ARQ.
- Block and Convolutional Codes.
- Reed Solomon (RS) Codes.
- Concatenated Coding used in DVB-T.

Possibly in the future lectures:

- Low Density Parity Check (LDPC) Codes in DVB-S2.
- Raptor Codes.

Lecture 7:
Error Control Coding

Department of Electrical & Computer Engineering

In previous lectures, we have learnt about data compression, i.e., how to encode the video with the least possible bit rate given a desired level of quality. We have also learnt how to packetize, i.e., organizing the bits at the output of the encoder into stream of equal length packets: Transport Stream (MPEG-TS). These packets are then sent over the channel to the receiver so that they can be decoded and used by the client. Alternatively, the packetized stream (TS) can be saved on a storage medium such as a DVD or a Blu-ray disk for future viewing.

Data going through any medium encounter noise and other impairments resulting in errors in the transport stream. In the analog TV or in case we could encode the video pixel by pixel, the effect of noise would be limited to one pixel. However, in the case of MPEG encoding, an error may result in destroying several macroblocks or even Group of Pictures (GOP) resulting in destroying considerable part of a video. Even a single erroneous bit may cause effects that may persist until the arrival of a new Intra-frame (I-frame) or even longer. If error rate increases, the picture is

completely lost. While in the old analog system, the effect of noise was seen directly as snow flake like impairment in the picture and the effect of noise increased as a function of the quality of your receiver the noise power and your distance from the TV station (your received power) in a continuous (graceful) way, in the case of digital TV, you either have perfect picture or nothing.

Lecture 7:
Error Control Coding

4. A single-bit error introduced in the middle of an MPEG slice (row of blocks) can cause a huge ripple through the end of the row (a). Subsequent video fields continue to use the corrupt video because of the interframe coding and motion vectors (b). The next Intra-Frame (approximately 500 ms) will clean up the video by starting a new anchor frame (JPEG picture).

Lecture 7:
Error Control Coding

Department of Electrical & Computer Engineering

In Terrestrial TV (DVB-T/T2) or satellite TV (DVB-S/S2/S2X) transport stream is transmitted directly, we only need to concern ourselves with bit errors and use error correcting codes at the physical level. However, with the increase in the number of viewers watching TV over the Internet either IP-TV or OTT applications such as NetFlix, TS is more and more being transmitted over IP. That is, MPEG Transport Streams get encapsulated inside an IP stream. With the increase in the speed of the Internet whether it is the land line (cable and VDSL) or wireless such as LTE, real-time broadcasting of video over the Internet has become feasible. Going over an IP network, an MPEG-TS faces another type of data loss due to loss of the packets as a result of collision and/ buffer overflow. To overcome this type of transmission impairment, we need to use error or possibly erasure protection coding at the transport and higher layers.

Concordia University
Department of Electrical & Computer Engineering

Real-time transport protocol

IP networks were not designed for real-time delivery of data and can have unpredictable jitter and delay. To maintain QoS, the multimedia data that travels on the IP networks must arrive on time and in the same order it was sent. Real-time Transport Protocol (RTP) can be used to address the time-critical requirement of multimedia bit streams. RTP provides a timestamp and sequence number to IP packets containing media data. These can be used by the receiver to synchronize playback and manage buffers minimizing network jitter.

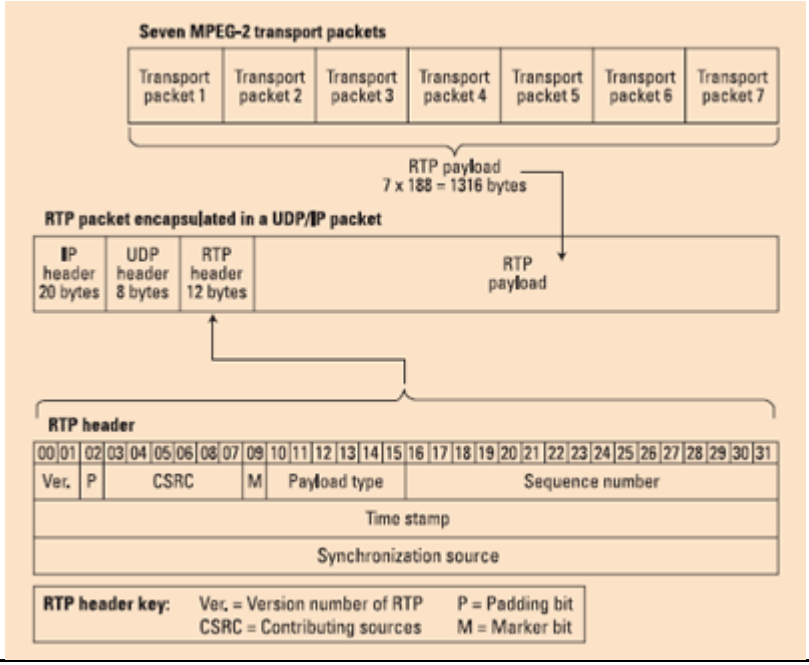
Encapsulating media data into IP packets

Delivering media bit streams over IP requires several layers of encapsulation. MPEG-2 transport streams, for example, consist of a series of 188-byte packets. These are grouped together and wrapped within an

Lecture 7:
Error Control Coding

Department of Electrical & Computer Engineering

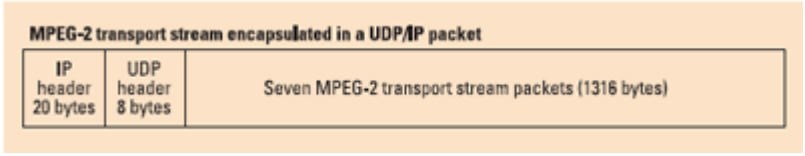
RTP packet. Finally, the RTP packet is encapsulated within a TCP or UDP datagram, forming an IP packet.
Figure below shows an RTP packet containing several MPEG-2 transport packets within its payload, all encapsulated using UDP in an IP packet. This diagram shows seven 188-byte transport packets that constitute the RTP payload. Each encapsulation adds additional header data and adds to the total required bandwidth.



Lecture 7:
Error Control Coding

Concordia University
Department of Electrical & Computer Engineering

If the network has sufficient QoS, it is possible to deliver media packets without the overhead of RTP. The packets are instead inserted directly into UDP packets. Next figure shows how MPEG-2 transport stream packets can be encapsulated within a UDP/IP packet.



Sending MPEG-2 transport stream packets over UDP is used extensively within the private networks of cable and telephone companies to deliver MPEG-2 transport streams throughout the system. For general delivery over the unmanaged Internet without QoS guarantees, streaming protocols such as RTP may be used, but even then, packets may be lost in delivery, resulting in artifacts in the media presentation.

TCP: Transmission Control Protocol .
UDP: User Datagram Protocol.
RTP: Real-time Transport Protocol.

Lecture 7:
Error Control Coding

In order to perform error detection/correction, we need to add overhead to the data to be transmitted in order to make a legitimate data stream different from a randomly corrupted stream of data. The overhead will be in the form of parities. As an example consider transmission of a number k of bits. The bits in a k bit long message represent values ranging from 0 to $2^k - 1$ for example if k=8, the numbers range from 0 to 255. Now if a bit is in error, we get another data stream which is in our list and we do not notice the error. Now, if we add on extra bit such that the number of 1's in the stream is odd (or even) then a single error will change the parity and we will get a pattern which we know could not have been transmitted. So, we detect a single error by adding a single parity bit. We may find the location of the error as well if we add several parity bits each making the parity of a subset of the bits.

Example: Consider that we have a 4 bit long message, x_1, x_2, x_3, x_4. If we add three parity bits, p_1, p_2, p_3 such that,

$$p_1 = x_1 \oplus x_2 \oplus x_3$$
$$p_2 = x_2 \oplus x_3 \oplus x_4$$
$$p_3 = x_3 \oplus x_4 \oplus x_1$$

Concordia University
Department of Electrical & Computer Engineering

Then an error in x_1 will result in parities p_1 and p_3 being violated and an error in in x_2 will result in parities p_1 and p_2 being changed and so on. This is a (7, 4) Hamming code that can correct any single error. Note that here we have increased the rate of transmission and, therefore, the required bandwidth by 7/4 in order to protect bits from being delivered erroneously.

Let's now access the performance of this simple code. Assume that we had an $\frac{E_b}{N_0}$ of 10 dB. That is $\frac{E_b}{N_0} = 10^{\frac{10}{10}} = 10$. Assume we are using QPSK modulation where $BER = Q\left(\sqrt{2\frac{E_b}{N_0}}\right) = Q(4.47) = 3.9 \times 10^{-6}$. If we use the above (7, 4) code, for each 4 bits we send 7 bits. So, in order to keep $\frac{E_b}{N_0}$ constant, we need to reduce the energy per each coded bit by 4/7. This means that we transmit at an $\frac{E_c}{N_0}$ of $10 \times \frac{4}{7} = 5.71$. The bit error rate before error correction will be $Q\left(\sqrt{2 \times 5.71}\right) = Q(3.38) = 3.62 \times 10^{-4}$. But the probability of error after decoding is the probability that we have more than one error at the output of the demodulator (before the decoding).

Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

The probability of an error is then,

$$P_e = \sum_{i=2}^{7} P(e = i) = 1 - P(e = 0) - P(e = 1)$$

or,

$$P_e = 1 - (1 - p)^7 - 7p(1 - p)^6$$

where p=3.62×10^{-4} and, therefore, $P_e = 2.75 \times 10^{-6}$.
Assuming that when there is an error in a 7 bit codeword on the average half the bits are in error, the bit error rate of the coded system is 1.37×10^{-6} which is roughly the same as the un-coded system.
The reason for the poor performance of the code is its short length. Hamming codes may be designed of any length $n = 2^m - 1$and information length $k = 2^m - 1 - m$ for any integer $m > 2$. The number of parity bits is m and they can all correct one erroneous bit. We may, therefore, have Hamming codes (15, 11) or (31, 26) and so on.

Concordia University
Department of Electrical & Computer Engineering

Now let's go back to the previous example of QPSK working at $\frac{E_b}{N_0} = 10\ dB$. But this time use a (31, 26) Hamming code.
The uncoded $BER = 3.9 \times 10^{-6}$ as before. Now, however, the channel symbol energy over noise density is $\frac{E_c}{N_0} = 10 \times \frac{26}{31} = 8.39$. The bit error rate before error correction will be $Q\left(\sqrt{2 \times 8.39}\right) = Q(4.1) = 2.066 \times 10^{-5}$.

$$P_e = \sum_{i=2}^{31} P(e = i) = 1 - P(e = 0) - P(e = 1)$$

or,

$$P_e = 1 - (1 - p)^{31} - 31p(1 - p)^{30}$$

where $p = 2.066 \times 10^{-5}$ and, therefore, $P_e = 1.9 \times 10^{-7}$ and $BER = 9.5 \times 10^{-8}$. This is more than an order of magnitude better than the uncoded case.

Lecture 7:
Error Control Coding

Discussing the simple example of using Hamming code or even only a single parity bit, hopefully, has clarified for you the concept of using redundancy for the purpose of error detection and error correction. It is clear that detecting the existence of errors is both simpler (in terms of decoder complexity) and less costly (requiring less number of parity bits and, therefore, requiring less bandwidth). Cyclic Redundancy Codes (CRC) are used in order to detect errors. After detecting errors, the receiver has two options: 1) to discard the erroneous data and 2) ask the transmitter to retransmit the corrupted data again. The latter is called Automatic Repeat Request (ARQ). ARQ has the advantaged of being adaptive to channel condition. When the channel is good, there are not many erroneous packets and therefore, not much retransmission. So, the overhead is mostly the few parity bits used for detecting the errors. The problem with ARQ is, however, the fact that it is not suitable for delay sensitive real-time traffic such as voice and real-time video. This is particularly true for links with high transmission delay. Take a satellite link where the end to end delay is close to 500 milliseconds. Such delay is not tolerable in the case of voice and real-time video.
For real-time traffic, enough redundancy is added so that the errors can be

Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

Schemes allowing enough redundancy so that the receiver can correct the erroneously received packets are called Forward Error Correction (FEC) schemes. Unlike ARQ where the scheme adapts itself to the channel condition, for FEC the system should be designed with an as accurate as possible assessment of the channel. If the code chosen correct less than the number of errors created by the channel, the errors remain uncorrected and even extra errors may be added. On the other hand if the code used is capable of correcting more errors than what the channel causes, the extra error correcting capability is wasted, i.e., we have wasted bandwidth without gaining anything.

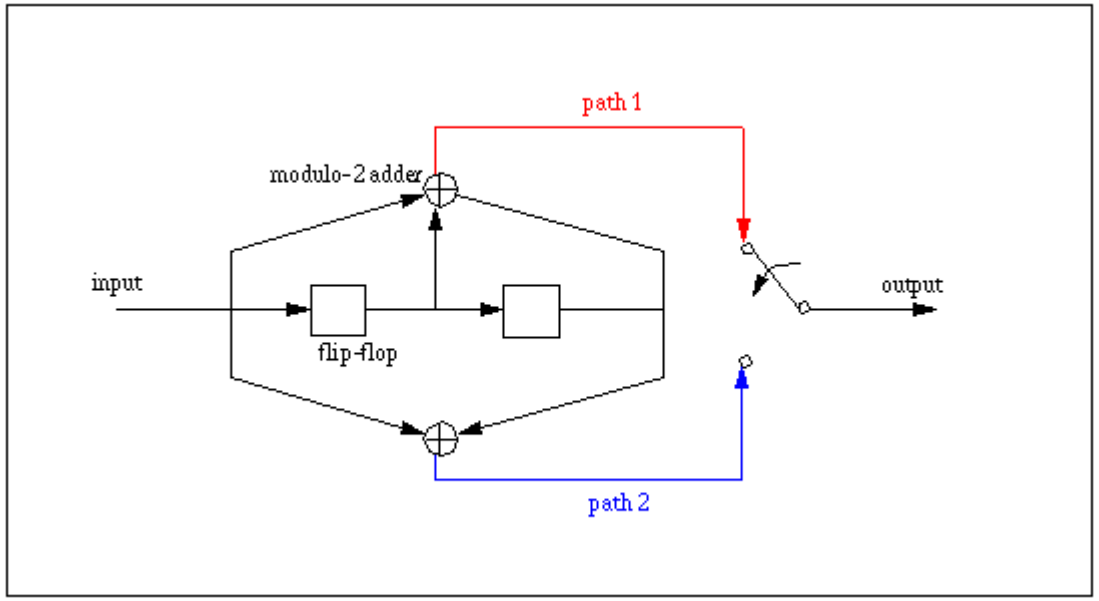
Block Codes and Convolutional Codes:

The error correcting codes are broadly divided into two categories: block codes and convolutional codes. For a block code, blocks of k information bits enter the encoder and the encoder outputs $n \geq k$ bits. In a systematic encoder, these n bits include the k information bits plus $n - k$ parity bits. A block code encoder, after encoding a block of k bits starts working on the next k bits and forgets all about the previous block, i.e., a block does not have any memory beyond the block on which it is working at a given time.

Lecture 7:
FEC: Convolutional Codes

Department of Electrical & Computer Engineering

The output of a convolutional encoder, on the other hand, does not only depend on the current input, but also on the previous inputs and/or outputs.

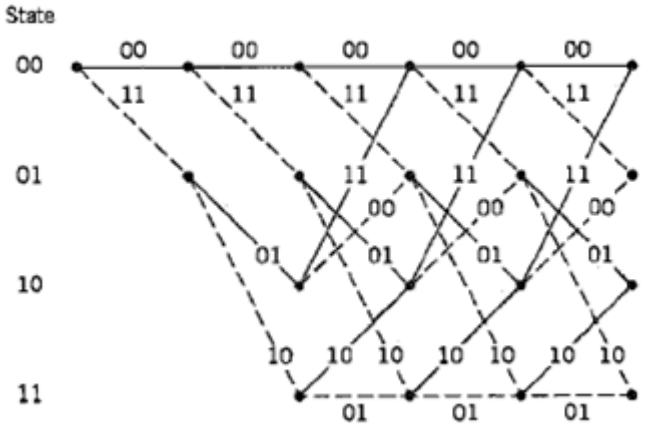


Lecture 7:
FEC: Convolutional Codes

Department of Electrical & Computer Engineering

The above figure shows simple convolutional code for two bits of memory, hence four states. The two outputs are formed by XORing the input and some of the past inputs. This is a rate ½ code.

The operation of a convolutional encoder can be described by a state diagram, a tree or a trellis. Using trellis is more common. Following is trellis representation of a rate ½ encoder with memory two:



Lecture 7:
FEC: Convolutional Codes

Department of Electrical & Computer Engineering

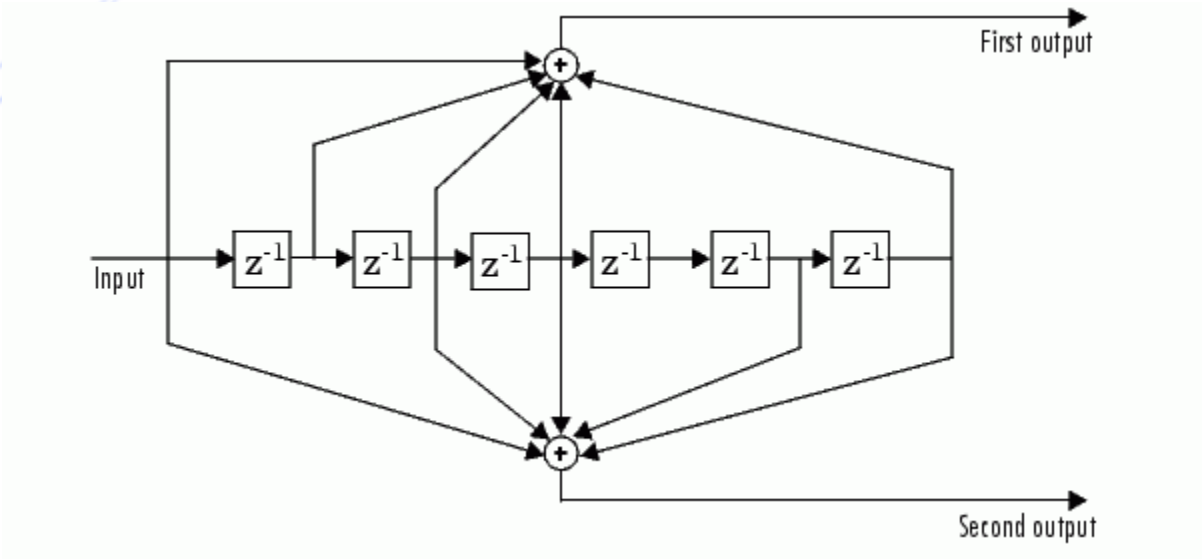
The decoder of a convolutional code consists of any mechanism that can go through the trellis and finds the one path that is closest to the received sequence. The most common decoding technique is the Viterbi decoder. Viterbi decoder can either be used after demodulation,. In such a case it looks for the path that differs in least number of bits with the output of the demodulator. The number of places in which two binary sequences are different is called the Hamming distance. So, in this case, the decoder tries to find the path whose labels have the minimum Hamming distance with the demodulated sequence. This is called hard Viterbi decoding.

It is also possible that the decoder finds the Euclidean distance between the output of the receiver filter and the sequence of labels of the paths and finds the path whose branch labels have the minimum Hamming distance from the received values. This is called soft decision Viterbi decoding. Soft decision decoding has usually about 3 dB advantage over decision.

The convolutional code used in DVB and many industrial applications is a rate ½ 64 state encoder, i.e., one with 6 memory length (6 Flip-flops saving the past inputs).

Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

The encoder for this code is,



One can have other rates by puncturing the output of the rate ½ encoder. For example, one can feed two bits at the input and get a total of 4 outputs (2 at each output point). Deleting one of 4 outputs one will have a rate 2/3 code (2 bits in, 3 bits out).

Université Concordia University
Department of Electrical & Computer Engineering

This table shows the puncturing pattern for other rates:

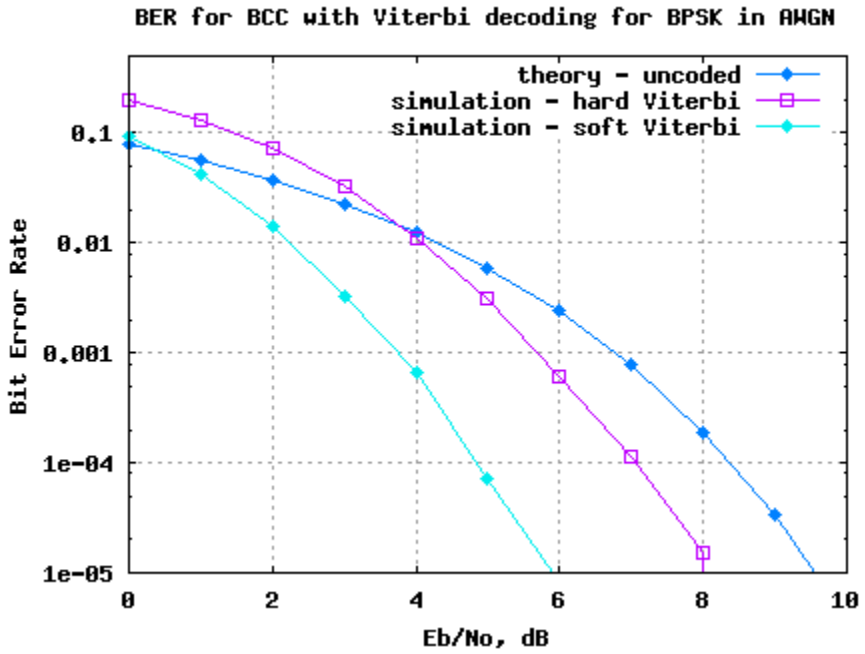
Code Rate	Puncturing Pattern
1/2	X: 1 Y: 1
2/3	X: 1 0 Y: 1 1
3/4	X: 1 0 1 Y: 1 1 0
5/6	X: 1 0 1 0 1 Y: 1 1 0 1 0
7/8	X: 1 0 0 0 1 0 1 Y: 1 1 1 1 0 1 0

1 and 0 denote transmitting and puncturing of coded bits, respectively.

Lecture 7:
FEC: Convolutional Codes

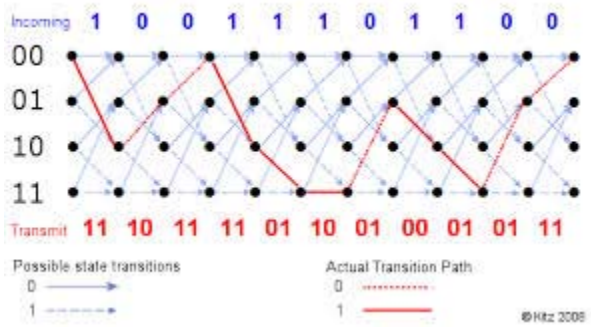
Department of Electrical & Computer Engineering

Following is a comparison of the performance of the convolutional code with soft and hard Viterbi decoding with the uncoded case:



Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

The memory of the convolutional codes causes the errors to occur in burst. To see this assume that a bit is decoded in error, i.e., a wrong path has been chosen on a trellis. It takes the deocdeer at least K stages to get back back to the right track where K is the constraint length of the code which is the number of memory bits plus one (to count for the present bit as well as K-1 past input bits).



This means that the convolutional encoding turns a memoryless AWGN channel into a burst error channel. In order to combat this Reed Solomon codes are used in a concatenation arrangement. The information is first encoded using an RS encoder (called the outer code) and then the output of the RS encoder is encoded using a convolutional encoder (the inner code). At the receiver first the Viterbi decoder is applied to recover the input of the convolutional encoder and then the

Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

RS decoding is performed.

Reed Solomon (RS) codes are non-binary codes. For each integer m one can define an RS code with $n = 2^m - 1$ symbols, each symbol consisting of m bits. $k \leq n$ of the symbols can be the information and $n - k$ symbols will be the parity symbols. An (n, k) RS code can correct up to $t = \left\lfloor \frac{n-k}{2} \right\rfloor$ errors, where $\lfloor x \rfloor$ represents the floor of x, i.e., the largest integer smaller than x.

Example: Take $m = 4$. Then $n = 2^4 - 1 = 15$. So each code word consists of 15 symbols each being 4 bits long. So each codeword will be 60 bits long. Taking, for example k=11, we will have a (15, 11) RS code that encodes 11 symbols (4x11=44 bits) into 15 symbols (60 bits). It can correct up to 2 erroneous 4 bit symbols.

Example: The RS code with m=8 is of particular interest as each of the symbols represent a byte. For m=8, the code length n=255. That is each codeword consists of 255 8-bit symbols. Taking k=239, we will have (255, 239) code. This code encodes 239 bytes (1912 bits) into 255 bytes (2040 bits) by appending 16 bytes of parity. It can correct up to 8 erroneous bytes. So, it can correct any error burst of length 7x8=56 bits.

Lecture 7:
FEC: RS Codes

As we discussed earlier MPEG-TS consists of packets of length 188 bytes. If we wanted to use an RS code of length 255, we needed to add 67 bytes of parity which is too much and results in unnecessary wastage of bandwidth. In order to avoid this, we can shorten the RS code. We can append the 188 data bytes with 51 bytes of zero (408 zeros). Since the RS code is systematic, i.e., the data and parity parts are separate and distinguishable, we can omit these zeros prior to transmission (in fact these zeros are just conceptual). The result will be a (204, 188) shortened RS code.

Performance Evaluation of the RS Codes:

We will try to find an approximate value for the probability of error using Reed Solomon codes. As it was stated above, an (n, k) RS code can correct up to $t = \left\lfloor \frac{n-k}{2} \right\rfloor$ erroneous m-bit symbols. IF the number of errors is more than t, then some or all of the errors remain un-corrected. It is even possible that the decoder add extra errors by trying to correct symbols that are not in error. We take the worst case and assume that if there are $i > t$ errors, the decoder not only does not correct any of the i errors, but also add t errors. So, the probability of error given that we there are $i > t$ errors is $\frac{i+t}{n}$. The probability that a received codeword

Concordia UNIVERSITÉ UNIVERSITY
Department of Electrical & Computer Engineering

has i errors is ${p_c}^i \ (1-p_c)^{n-i}$ where p_c is the probability that a symbol is in error. Let the probability of encoded bits before decoding be p. Then,

$$p_c = 1-(1-p)^m \approx mp$$

The bit error rate p is determined based on the modulation scheme used for example, for QPSK it is given by,

$$p = Q\left(\sqrt{2\frac{E_c}{N_0}}\right) = Q\left(\sqrt{2\frac{k}{n}\frac{E_b}{N_0}}\right)$$

There are $\binom{n}{i}$ sequence with i errors among the possible n symbol sequences where,

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}.$$

So, the symbol error rate will, approximately, be:

$$P_e \approx \sum_{i=t+1}^{n} \frac{i+t}{n}\binom{n}{i}{p_c}^i \ (1-p_c)^{n-i}$$

Concordia UNIVERSITY
Department of Electrical & Computer Engineering

Assuming that when a symbol error occurs, on the average, half the bits are in error, we get,

$$P_B \approx \frac{1}{2} P_e \approx \frac{1}{2n} \sum_{i=t+1}^{n} (i+t) \binom{n}{i} {p_c}^i \ (1-p_c)^{n-i}$$

DVB Example: In DVB standard a (204, 188) RS code is used, so, t=8. That is, the decoder can recover all the 204 byte long TS packet that have less than 9 erroneous bytes. So,

$$P_B \approx \frac{1}{2 \times 204} \sum_{i=9}^{204} (i+8) \binom{204}{i} {p_c}^i \ (1-p_c)^{204-i}$$

Assume that we are using BPSK modulation at $\frac{E_b}{N_0} = 8\ dB$.

The un-coded BER is, $P_{B,u} = Q\left(\sqrt{2 \times 10^{0.8}}\right) = Q(3.55) \approx 1.92 \times 10^{-4}$.

Now let's see that RS code can do for us. The BER of the coded stream after the demodulator and before decoding is,

$$p = Q\left(\sqrt{2 \frac{k}{n} \frac{E_b}{N_0}}\right) = Q\left(\sqrt{2 \frac{188}{204} \times 10^{0.8}}\right) \approx 3.25 \times 10^{-4}$$

Concordia University
Department of Electrical & Computer Engineering

The symbol error rate before decoding is

$$p_c = 1 - (1-p)^8 \approx 8p = 0.0026.$$

So,

$$P_B \approx \frac{1}{2 \times 204} \sum_{i=9}^{204} (i+8) \binom{204}{i} (0.0026)^i (1-0.0026)^{204-i}$$

Since the terms are decreasing rapidly, we can use the first term as a lower bound on the probability of error. If you calculate the term for $i = 10$ you see that it is about 0.05 of the term for $i = 9$. So,

$$P_B \approx \frac{17}{2 \times 204} \binom{204}{9} (0.0026)^i (1-0.0026)^{204-i} = 4.6 \times 10^{-9}$$