

# Chapter 11

## Turbo Codes

### 1. Concatenated Coding System

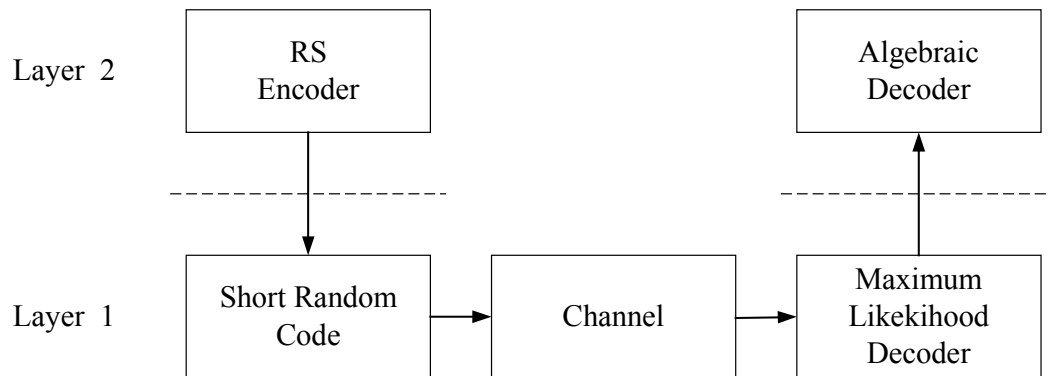


Fig. 12.1

- Concatenated coding has the multilevel coding structure, illustrated in Fig.12.1

A relatively short random “inner code” can be used with MLD to achieve a modest error probability, say a bit-error rate of  $P_b=10^{-2}$ , at a code-rate which is near channel capacity.

Then a long high-rate algebraic Reed-Solomon “outer code” can be used along with a powerful algebraic error-correction algorithm to drive down the error probability to a level as low as desired with only a small code-rate loss.

- For convolutional codes with Viterbi decoding, soft decisions are incorporated easily into the Viterbi decoding algorithms in a very nature way, providing an increase in coding gain of over 2.0 dB with respect to the comparable hard-decision decoder over AWGN channel. However, the convolutional codes cannot be implemented easily at high coding rates. They also have an unfortunate tendency to generate burst errors at the decoder output as the noise level at the input is increased.
- In the concatenated coding system, the convolutional code (with soft-decision Viterbi decoding) is used to “clean up” the channel for Reed-Solomon code, which in turn corrects the burst errors emerging from the Viterbi decoder.

- By the proper choice of codes the probability of error can be made to decrease exponentially with overall code length at all rates less than capacity.
- Generally, the “outer” code is more specialized in preventing errors generated by the “inner” code when it makes a mistake.

The “inner” code can also be a binary block code other than a binary convolutional code. For a hard-limited channel, trellis codes often are selected as “inner” codes.

## 2. The Shannon Limit on Performance

- The capacity of an AWGN channel

$$C = B \log_2 \left( 1 + \frac{S}{N_0} \right) \quad \text{bits / sec} \quad \dots\dots \quad (1)$$

where B is the bandwidth of the channel in Hz and S is the average power of the signal. N is the noise power.

- Shannon bound

For BPSK-modulated system, let  $\eta$  be the spectral efficiency of the modulation format. It is the average number of information bits transmitted per two-dimensional signaling interval of duration T.

If T is normalized and related in terms of bits per second per Hz (b/s/Hz). It follows that  $S/N = \eta E_b/N_0$ , where  $E_b$  is the average energy per information bit.  $N_0$  : noise power spectral density.

Substituting into equation (1) and rearranging, we obtain

$$\frac{E_b}{N_0} > \frac{2^n - 1}{\eta} \quad \dots(2)$$

For a given spectral efficiency, this expression sets the limit on SNR above which there exist coding schemes that will provide arbitrarily low bit error rate.

- For BPSK,  $\eta = 1$ , the lower limit on SNR is 1 or 0 dB.
- The maximum spectral efficiency of an AWGN channel

$$\eta_{\max} = \log_2 \left( 1 + \frac{S}{N} \right) \quad \text{bits / s / Hz}$$

- In the limit as the signal is allowed to occupy an infinity amount of bandwidth, that is, as  $\eta_{\max} \rightarrow 0$ , one obtains

$$\frac{E_b}{N_0} \geq \lim_{\eta_{\max} \rightarrow 0} \frac{2^{\eta_{\max}} - 1}{\eta_{\max}} = \ln(2) = -1.59 \text{ dB}$$

- Shannon's 1948 paper entitled "A Mathematical Theory of Communication" launched the twin fields of Information Theory and Error Control Coding. In this paper, Shannon defined the concept of channel capacity. He then show that, as long as the rate at which information is transmitted is less than the channel capacity, there exist error control codes that can provide arbitrarily high levels of reliability at the receiver output.

The prof of this, the "Noisy Channel Coding Theorem" was existential and not constructive. We were left knowing that nice decoding schemes existed, but had no idea how to construct them.

The subsequent fifty years of error control coding have been an effort to achieve this goal.

- The 1993 paper by C. Berrou, A. Glavieux, and P. Thitimajshima entitled "Near Shannon Limit Error Correcting Coding and Decoding : Turbo Codes" has brought us within a hair's

breadth of achieving Shannon's promise. This paper was presented at 1993 IEEE International Conference on Communications in Geneva.

- In their presentation, Berrou et al claimed that a combination of parallel concatenation and iterative decoding can provide reliable communication at a SNR that is within a few tenths of a dB of the Shannon limit. They show by simulation that turbo codes are capable of achieving a bit-error rate (BER) of  $10^{-5}$  at an  $E_b/N_0$  ratio of just 0.7dB. It appears that the 40-year effort to reach channel capacity was achieved by this latest discovery. However, in order to achieve this level of performance, long block sizes of 65,532 data bits are required. Because of the probability long latency involved with such long block sizes, the original turbo codes are not applicable to most real-time two-way communication systems.

## Key Design Innovations of Turbo Coding:

(1) parallel concatenated encoding.

(2) iterative decoding.

- Parallel concatenated encoders (PCEs) consist of two or more component encoder (or constituent encoders) for block or convolutional encoders.

- Iterative decoding :

Although, in principle, it is possible to derive the optimal decoder for any given PCE, the result would be an extremely expensive and computationally inefficient system.

Iterative decoding is a suboptimal alternative that provides extremely good performance which requiring only a modest level of complexity.

The key to the iterative decoder is its exploitation of the component code structure of the PCE.

The iterative decoder has a soft-input/soft-output (SISO) component decoder for each of the component decoder in the PCE.

The decoders take turns operating on the received data, forming and exchanging estimates of the message block.

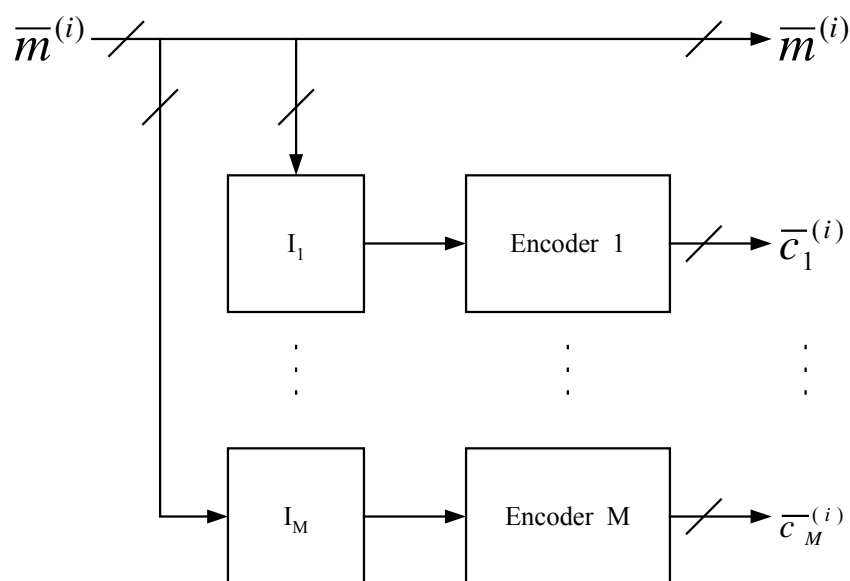
Since these component decoders operate on each other's "incompletely decoded" outputs, they are reminiscent of a turbo charged engine, and hence the name "turbo codes".

\* When decoded by an iterative process, turbo codes offer an optimum performance. The way to achieve this decoding is using the principle of Maximum A Posteriori (MAP)

★ Note: The iterative decoding scheme is based on alternately decoding the two constituent codes nearing a soft-input soft-output (SISO) algorithm and transferring the soft information (extrinsic information) to the next decoding stage. The extrinsic information generated in the previous decoding stage will be used as the priori information for the next decoding stage. This iterative manner is similar to a turbo engine.

### 3.Parallel Concatenated Coding

- Turbo coding is accomplished by the parallel concatenation of two or more systematic codes.
- Fig. 12.27 shows a generalized turbo encoder.



$\bar{m}^{(i)}$  : k-bit message vector at the I-th time unit.

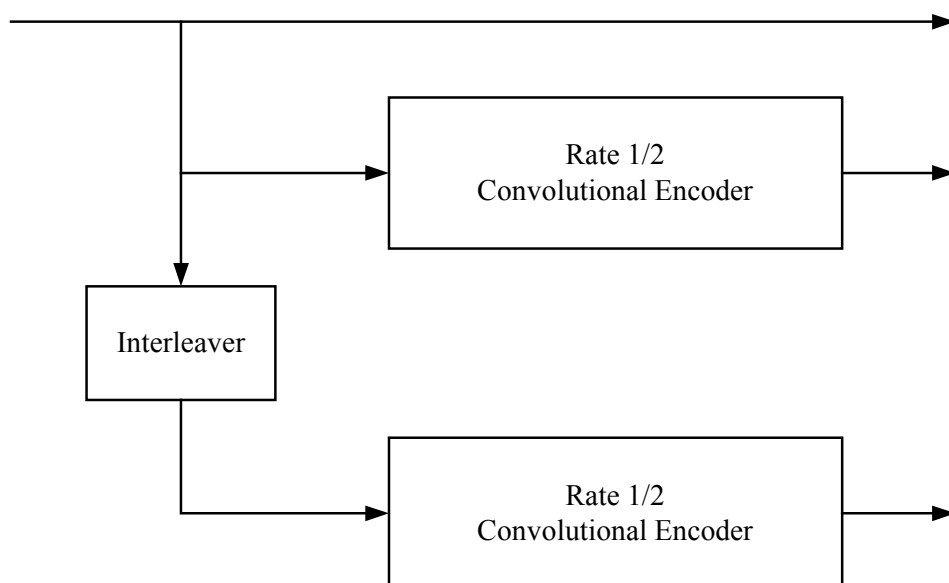
$I_1, \dots, I_M$ : interleaver

$\bar{c}_j^{(i)}$  : (n-k)-bit output vectors generated by the j-th constituent encoder.

- The input vector  $\bar{m}^{(i)}$  together with the M other output vector  $\bar{c}_j^{(i)}$  are concatenated to form the codeword.
- The overall code rate of such an encoder is

$$r = \frac{k}{(n - k)M + k}$$

- This encoder structure is called a parallel concatenation because all of the constituent encoders operate on the same set of input bits.
- For most of applications, only two such encoders are used and the input to the 1<sup>st</sup> encoder is not interleaved.
- These Constituent encoders themselves are identical and are usually recursive systematic convolutional encoders (RSC codes).
- Extra tail bits are added in order to ensure that the constituent convolutional encoders return to the all-zero state at the end of such block.
- Example : Rate 1/3 turbo encoder



## 4. The constituent Encoder

- In the turbo encoder, the most common used constituent encoder is the recursive systematic convolutional encoder.

This makes it possible for the decoder to utilize a modified version of the Viterbi algorithm.

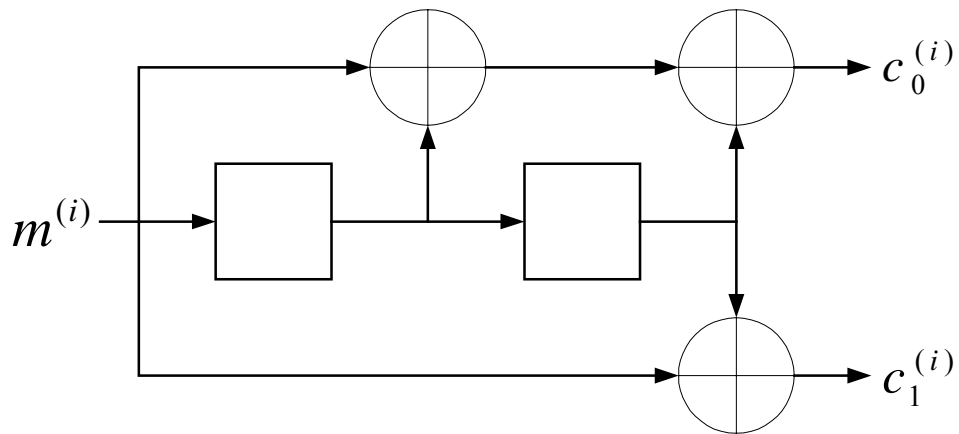
Use of a non-recursive systematic convolutional encoder is unacceptable because of the poor distance properties of the resulting code.

- A non-recursive non-systematic convolutional encoder, as shown in Fig. 12.29, cannot be used as the constituent encoder for a turbo code simply because it is not systematic.

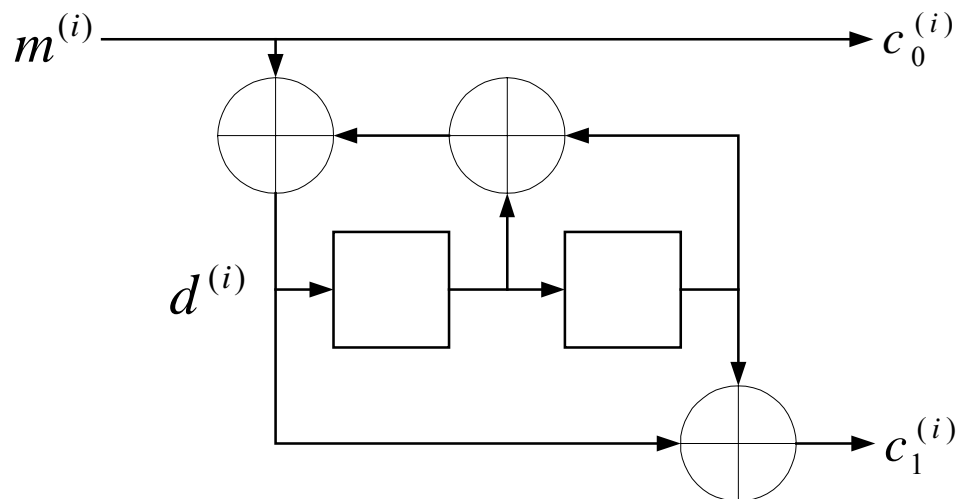
- The encoder shown in Fig.12.29 can be modified in such a way that it becomes systematic and retains its distance properties.

This modification yields a feedback systematic encoder of the form as shown in Fig.12.30.

This encoder is recursive since the state of the internal shift register depends on past outputs.



**Figure 12.29** Example of a non-recursive nonsystematic convolutional encoder



**Figure 12.30** Example recursive systematic convolutional encoder

- A convolutional encoder can be used to generate a block code if the internal state of the encoder is known at the beginning and end of the codeword. The usual convention is to initialize the encoder to the all-zero state and then to encode the data. After the  $k$  data bits are encoded, a number of tail bits are added to the encoded word in order to force the encoder block to the all-zero state. The number of tail bits required is on the order of the memory  $m$  of the convolutional encoder.
- A tail of  $m$  zeros brings a nonrecursive convolutional encoder back to the all zeros state.
- Due to the presence of feedback, a tail of zeros does not necessarily bring a recursive convolutional encoder back to the all-zeros state.

- The tail required to bring a recursive convolutional encoder to the all-zero state is found by solving a state-variable equation at the feedback element.

For example, in Fig.12.31, the state equation at the feedback element is given by

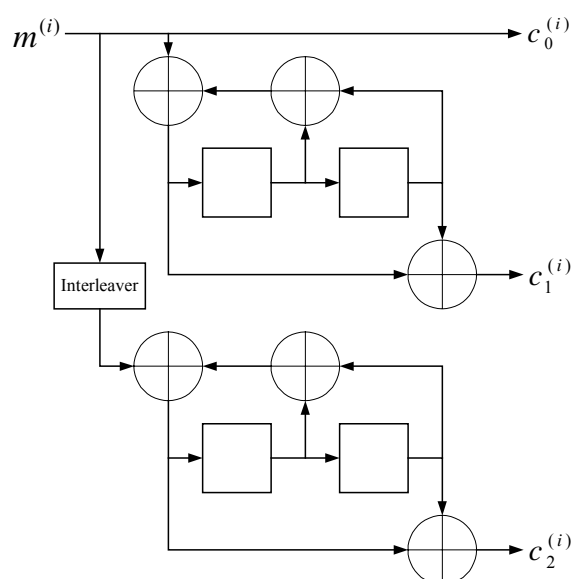
$$d^{(i)} = m^{(i)} \oplus d^{(i-1)} \oplus d^{(i-2)}$$

Solving this equation for  $m^{(i)}$  yields:

$$m^{(i)} = d^{(i)} \oplus d^{(i-1)} \oplus d^{(i-2)}$$

Since one wants to bring the encoder back to the all-zero state,  $d^{(i)}$  is set to zero so that one obtains

$$m^{(i)} = d^{(i-1)} \oplus d^{(i-2)}$$



## 5. Interleaver Design

- One key to the effectiveness of turbo coding systems is the interleaver.

The interleaver allows the constituent decoders to generate separate estimates of the a posteriori probability (APP's) for a given information symbol based on data sources that are not highly corrected.

- The interleaver also ensure that the set of code sequences generated by the PCE has nice “weight” properties, which reduces the probability that the decoder will mistake one codeword for another.
- Example : (Fig.13.31)

A rate 1/3 turbo encoder, which use the example RSC encoder in Fig.12.31 is shown in Fig.12.31.

- The pseudorandom interleaver in this circuit is used to permute the input bits in such a manner that the two encoders operate

on the same set of input bits, but with different input sequence to the encoders.

In [20] and [21], it is shown that by the proper design of the pseudorandom interleaver, it is possible to force both of the constituent encoders back to all-zero state with a single  $m$ -bit tail.

- The condition on the interleaver is based on the fact that the impulse response of each RSC constituent encoder is periodic with some period given by

$$P \leq 2^m - 1$$

If the feedback polynomial is primitive of degree  $m$ , then the impulse response is a maximal length sequence, and equality holds in the above equation. The feedback polynomial for the example (Fig.12.31) encoder is given by

$$g(X) = 1 + X + X^2$$

which is primitive. Thus, one concludes

that the impulse response of this example constituent RSC encoder is periodic with period  $p=2^2-1=3$ .

- (From Chapter 9)

An interleaver maps an integer  $0 < l < n+1$  onto another integer  $l_0$  for  $0 < l_0 < n+1$ . A bit in position  $l$  at the input to the interleaver is placed in position  $l_0$  at the output of the interleaver.

If this mapping function is call  $a$ , then the mapping can be expressed by

$$l_0 = a(l)$$

A condition for the interleaver [ref.21] that allows both of the RSC encoders to be terminated with the same tail, is the following :

$$l \bmod p = a(l) \bmod p$$

for all integer  $l$ .

- The design condition is best illustrated with example.

If we use the example turbo encoder in Fig.12.31 and encode data words of length  $k = 32$ , then the interleaver must be capable of encoding blocks of length  $n = k + m = 34$ .

An interleaver that meets the condition stated above is shown in Table 12.5.

## 6. Puncturing the output

- In Fig.12.31, the output of the turbo encoder consists of the information sequence and two parity sequences, thus it has a code rate of  $1/3$ . For many applications, it is common practice to puncture the output of the encoder in order to increase the code rate to, say for example,  $1/2$ .
- By alternatively puncturing (deleting) bits from the two parity sequence  $\bar{c}_1$  and  $\bar{c}_2$  in Fig.12.31 produces a code rate of  $1/2$  (as shown in Fig.12.32). Other code rate can be achieved by using additional parity generators and/or different puncturing patterns.

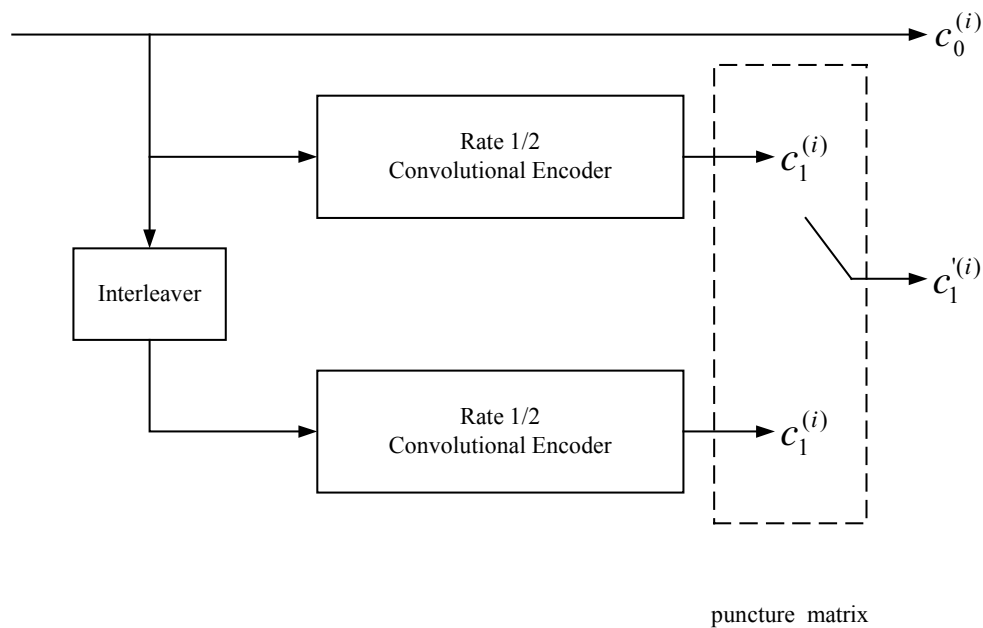


Fig. 12.32

- When puncturing is used, the joint effect of the interleaver and multiplexer on the code's "distance properties" must be taken into account.

It is suggested in [24] that when puncturing is used, the interleaver should be designed in such a manner that even bits are mapped to the even indexed bits and the odd indexed bits are mapped to odd indexed bits.

## 7. Decoding of Turbo codes

- The iterative decoder uses a soft-in/soft-out maximum a posteriori probability (MAP) decoding algorithm.

This algorithm was first applied to convolutional codes by BCJR [28] algorithm.

- In the original paper on the turbo codes, Berrou et al proposed an iterative decoding scheme based on BCJR algorithm.
- The BCJR algorithm differs from the Viterbi algorithm in the sense that it produces soft outputs

While the VA outputs either a 0 or 1 for each estimated bit, the BCJR algorithm, outputs a continuous values that weights the confidence or log-likelihood of each bit estimate.

- The BCJR algorithm attempts to minimize the bit-error-rate by estimating the

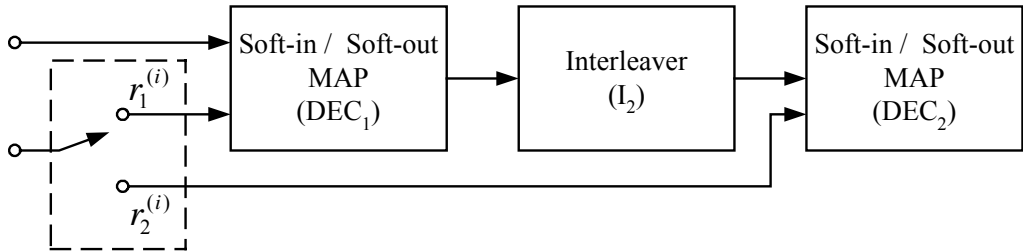
a posteriori probability (APP) of the individual bits of the codeword, rather than the ML estimate of the transmitted codeword.

- For encoder given in Fig.12-32, there are two elementary decoders that are interconnected as Fig.12-33 and Fig.12-34.

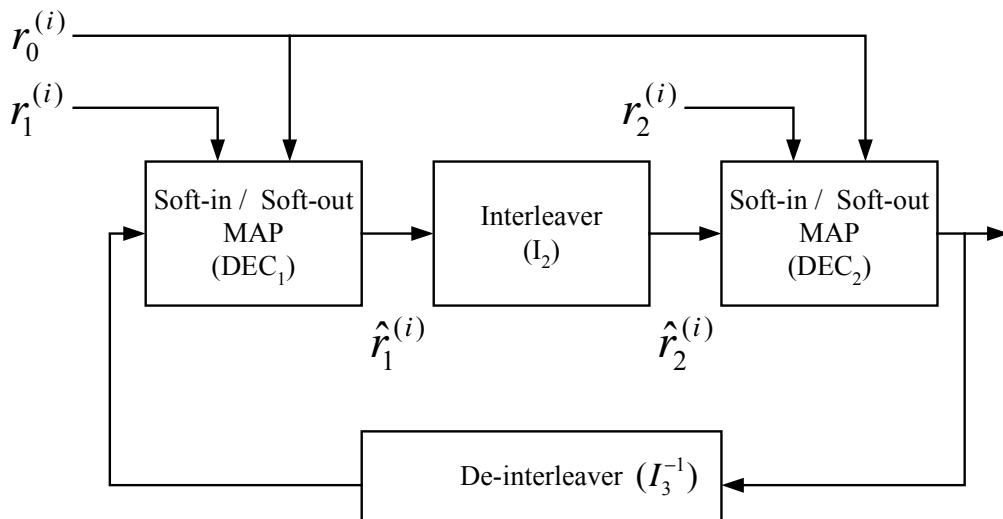
In Figs.12.33 and 12.34, it is assumed that the information stream  $C_0^{(i)}$  in Fig.12.32 is received as  $r_0^{(i)}$ , while the parity stream  $C_1'^{(i)}$  is received as  $r_1'^{(i)}$ .

The received information stream  $r_0$  is demultiplexed into the received parity-check bits  $r_1^{(i)}$  and  $r_2^{(i)}$ , corresponding to  $C_1^{(i)}$  and  $C_2^{(i)}$ , respectively.

- For the encoder given in Fig.12.32, there are two elementary decoders that are interconnected as shown in Fig.12.33 and Fig. 12.34.



**Fig. 12.33** Initialization stage



**Fig. 12.34** Interaction stage

In Fig12.33. (Initialize stage)

- The information stream  $C_0^{(i)}$  is received as  $r_0^{(i)}$ , while the parity stream  $C_1'^{(i)}$  is received as  $r_1'^{(i)}$ .
- $r_1'$  is demultiplexed into  $r_1^{(i)}$  and  $r_1^{(i)}$ , corresponding to  $C_1^{(i)}$  and  $C_1^{(i)}$ , respectively.
- The parity bits  $r_1^{(i)}$  is sent to  $DEC_1$  and  $r_1^{(i)}$  are sent to  $DEC_2$ .
- $DEC_1$  produces a soft decision which is based on its received parity bits  $r_1'^{(i)}$  along with the received information stream  $r_0^{(i)}$ .
- $DEC_2$  uses the received parity bits  $r_2^{(i)}$  along with the interleaver soft decision from  $DEC_1$  to produce a soft decision of its own.

### Fig 12.34 (Iteration stage)

- In order to find an estimate that approaches the MAP estimate, the information produced by the two elementary decoders must be shared with one another.
- In Fig12.34.  
The soft-decision of DEC<sub>2</sub> is de-interleaved by  $I_2^{-1}$  and re-introduced as the input to DEC<sub>1</sub>.  
DEC<sub>1</sub> once again produces a soft estimate that interleaved by  $I_2$  and feedback into DEC<sub>2</sub>.
- As the number of these iterations approaches infinity, the estimates  $\hat{r}_1^{(i)}$  and  $\hat{r}_2^{(i)}$  at the output of DEC<sub>1</sub> and DEC<sub>2</sub>, respectively, approach the MAP solution.
- In practice, the number of iteration needs only to be about 18, and in many cases, as few as 6 iterations provide satisfactory performance.

Note: The elementary decoders in Fig.12.34 used the modified BCJR algorithm.

- The modified BCJR algorithm suffers from a complexity that is significantly higher than that of the Viterbi algorithm
- Efforts have been underway to find reduced-complexity elementary decoders.
- Modification of the Viterbi algorithm, such as the soft-output Viterbi Algorithm (SOVA) seems most promising.