



**Digital Video Broadcasting (DVB);
MPE-IFEC**

**DVB Document A131
November 2008**

MPE-IFEC v3.0.8

References	3
1 Introduction (informative)	4
2 Notation, abbreviations and definitions (normative)	7
2.1 Notations.....	7
2.2 Abbreviations	7
2.3 Definitions.....	7
2.3.1 Datagram burst	7
2.3.2 Encoding Period.....	8
2.3.3 Application Data Sub-Table	8
2.3.4 Encoding Matrix	9
2.3.4.1 Application Data Table	10
2.3.4.2 IFEC Data Table	10
2.3.5 MPE-IFEC section	10
2.3.6 IFEC burst	10
2.3.7 Time-slice burst	10
3 Sender operation (normative).....	11
3.1 Introduction	11
3.2 Parameters	12
3.3 Initialization	14
3.4 Reception of New Datagram Burst	14
3.5 Generation of IFEC-Burst	14
3.6 Time-Slice Burst Generation and Sending	15
3.7 Datagram Burst to ADT Mapping	16
3.8 Generation of iFDT.....	16
4 Carriage of MPE-IFEC sections (normative)	17
4.1 Generalities	17
4.2 Syntax and semantics.....	17
4.3 Real time parameters.....	18
5 Time Slice and FEC identifier descriptor (normative).....	20
5.1 Introduction (informative).....	20
5.2 Descriptor (normative)	20
5.3 Sliding Encoding with RS code (normative)	25
5.3.1 General.....	25
5.3.2 Parameter Definitions.....	25
5.3.3 Mapping Functions.....	25
5.3.4 Generation of iFDT	26
5.3.5 Memory considerations	26
5.4 Generalized Encoding with Raptor code (informative).....	27
5.4.1 General.....	27
5.4.2 Parameter Definitions.....	27
5.4.3 Mapping Functions.....	28
5.4.4 Generation of iFDT	28
5.4.5 Example Parameters	28
6 Receiver Operation (informative).....	30
6.1 Introduction	30
6.2 General Process.....	30
6.3 Parameters	31
6.4 Burst number detection.....	32

6.5	Section Reception.....	33
6.6	Padding in ADST mapping.....	34
6.7	Decoding.....	35
6.7.1	Input	35
6.7.2	RS decoding	35
6.7.3	Raptor decoding	35
6.7.4	Output.....	35

References

- [1] ETSI EN 301192 v1.4.1, Digital Video Broadcasting (DVB); DVB Specification for Data Broadcasting, November 2004.
- [2] ETSI TS 102472 v1.2.1, IP Datacast over DVB-H: Content Delivery Protocols (CDP), June 2006.
- [3] ISO/IEC 13818-6, Information technology -- Generic coding of moving pictures and associated audio information -- Part 6: Extensions for DSM-CC
- [4] ISO/IEC 13818-1: Information technology -- Generic coding of moving pictures and associated audio information: Systems
- [5] ETSI TS 102 584, Digital Video Broadcasting (DVB); DVB-SH Implementation Guidelines, 2008

1 Introduction (informative)

MPE-IFEC is introduced to support reception in situations of long duration erasure on the MPE section level spanning several consecutive time slice bursts. Such erasure situations may for example occur on satellite mobile channels (LMS: land mobile satellite) without any terrestrial repeaters in the vicinity: obstacles may hinder direct satellite reception and induce losses of several successive bursts. For example, with an MPE-IFEC protection where about 30% of TS data are allocated to parity overhead computed over 10 successive bursts, it is possible to compensate up to 3 successive complete burst losses whereas recovery of a complete burst loss with DVB-H MPE-FEC protection would not be possible. Such erasure situation may also occur in terrestrial networks so that MPE-IFEC may be useful in other channels than LMS.

The MPE-IFEC protection is computed over several successive datagram bursts, as opposed to MPE-FEC where the computation is performed on a single datagram burst. This multi-burst protection is enabled by an enlargement of the encoding matrix to sizes greater than one burst (an IFEC matrix is filled not by one burst as in MPE-FEC but by several successive bursts), by a parallelization of the encoding mechanism (instead of using only one matrix, the data are distributed to a number of parallel matrices equal to B) or by a combination of both principles. The datagrams themselves are sent in MPE sections without any modification compared to [1] section 9.6. The resulting parity may also be spread over several bursts instead of one single burst in the MPE-FEC case: each burst contains parity coming from S matrices.

An overview of the link layer operations, especially the MPE-IFEC, in the case of DVB-SH is presented in Figure 1. Datagram bursts of variable size (in terms of number of bytes and/or number of datagrams) are used as input and mapped by an ADST (Application Data Sub Table) function on to the ADTs (Application Data Tables) of up to M parallel encoding matrices. The IFEC burst collects all MPE-IFEC sections from iFDTs (IFEC Data Tables, the IFEC parity symbols for an ADT) of several encoding matrices. An MPE-IFEC section is comprised of a header, the data from multiple columns from the same iFDT, and a checksum. This IFEC burst is then merged with all MPE sections of an original datagram burst, including its MPE-FEC if present, to generate the time slice burst that is actually sent over the air. This merging is done with the datagram just received when the delay parameter D is set to $D=0$, or with a previously received datagram when $D>0$. Note that the original data in MPE sections fully complies with [1] section 9.6.

This multi-burst parity computation and spreading is achieved at the expense of some latency for generating and receiving parity data. Nevertheless, as datagram bursts are sent unmodified, many well-known receiver operations are still possible. For example, the received MPE sections may be immediately

forwarded to media decoders for fast channel-switching (zapping) support. Only when the parity is needed, the forwarding should be delayed to accumulate sufficient redundancy information.

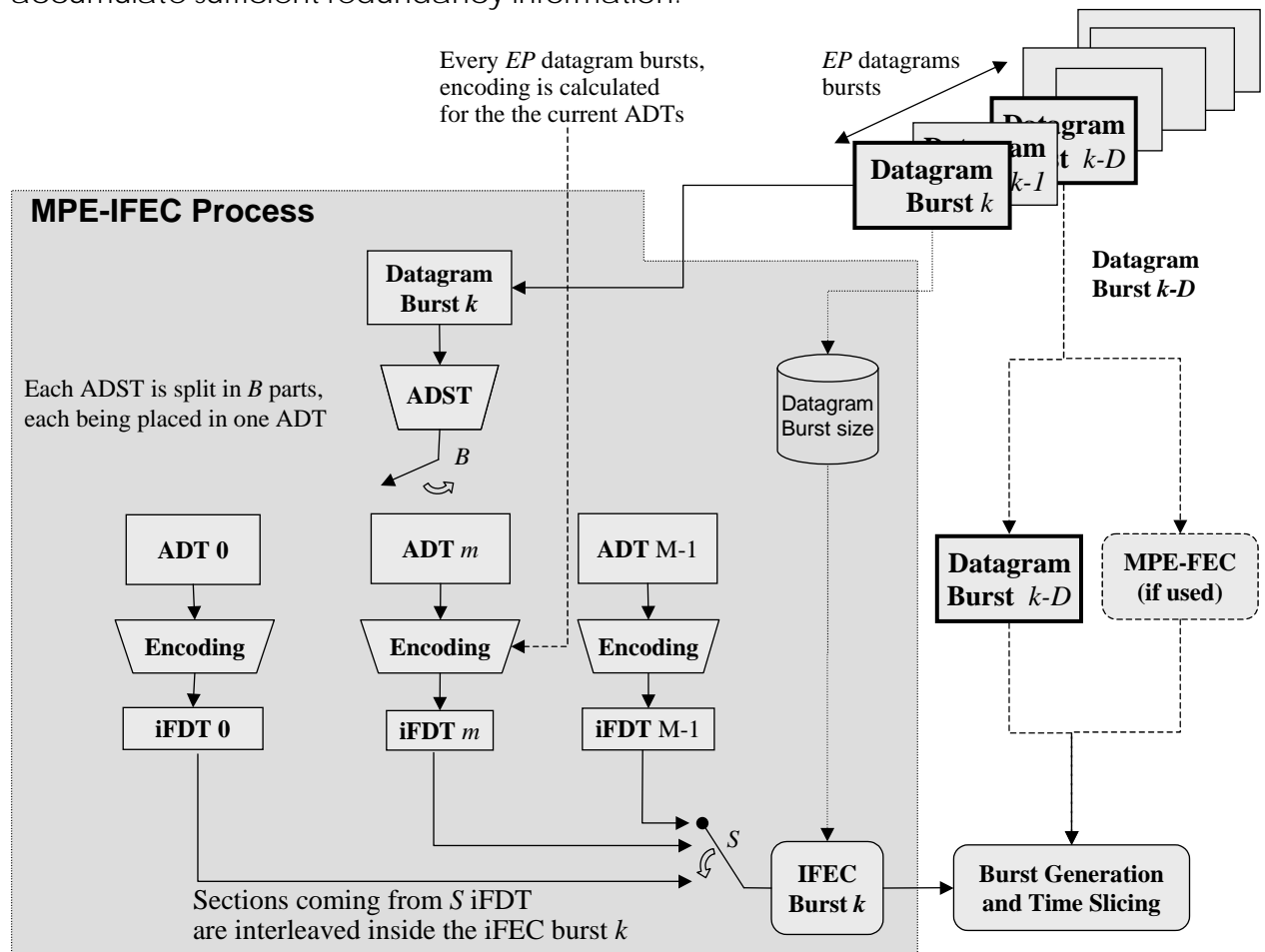


Figure 1: MPE-IFEC encoding process

The MPE-IFEC is introduced in such a way that MPE-IFEC ignorant (but MPE and MPE-FEC capable) DVB receivers will be able to extract the MPE stream in a fully backwards-compatible way. This backwards compatibility holds both when the MPE-IFEC is used with and without Time Slicing. The use of MPE-IFEC is not mandatory and is defined separately for each elementary stream in the transport stream. For each elementary stream it is possible to choose whether or not MPE-IFEC is used, and if it is used, to choose the trade-off between IFEC overhead, extra delay and performance. Time critical services, without MPE-IFEC and therefore minimal delay, could therefore be transmitted together with less time critical services using MPE-IFEC, on the same transport stream but on different elementary streams.

The MPE-IFEC is specified as a generic framework that presents enough flexibility for a variety of applications. For a usage in DVB-SH, its parameters are restricted to some specific values via the "framework mapping". Two of such "mappings" are presented in this document. One is based on MPE-FEC Reed Solomon code ([1], section 9.5.1) and is normative. The other mapping is based on Raptor code as specified in the Content Delivery Protocols (CDP) specification of IP Datacast over DVB-H [2] and is informative.

The document is intended to be ultimately added as an Annex to ETSI EN 301192 v1.4.1 but can be used independently as it is natively self-contained.

This document is structured as follows:

- Chapter 1 gives this (informative) introduction
- Chapter 2 introduces (normative) definitions and abbreviations
- Chapter 3 introduces the (normative) sender operation
- Chapter 4 describes the (normative) carriage of MPE-IFEC Frame
- Chapter 5 provides (normative) syntax of Time Slice and FEC identifier descriptor and the two mapping examples, sliding encoding with RS code (normative) and generalized encoding with Raptor code (informative).
- Chapter 6 introduces (informative) prototype IFEC decoding.

2 Notation, abbreviations and definitions (normative)

2.1 Notations

All symbols and parameters are in *italics*: e.g. k , *datagram_burst_size(k)*

$A [B]$ means A modulo B

All functions are defined according to this template:

output_unit function_name(*input_unit* parameter1, *input_unit* parameter2, ...) e.g. *ifdt_index(k)*

All fields that are transmitted are denoted with `courier new`, e.g. `burst_number`.

2.2 Abbreviations

IFEC: MPE-IFEC

ADT: Application Data Table

ADST: Application Data Sub Table

iFDT: IFEC Data Table

EM: Encoding Matrix

2.3 Definitions

The definitions are explicitly kept simple and not all parameters used by definitions are specified to avoid overloading explanations. However, all used parameters can be found, either in the generic sender operation description (section 3.2) or the specific mapping over the two parity computation codes (section 5.3 and 5.4).

2.3.1 Datagram burst

A datagram burst is a collection of one or several contiguous OSI layer 3 (Network layer) datagrams (e.g. IP datagrams). Datagram bursts are numbered with continuous sequence numbers $k=0, 1, \dots$

The bytes constituting one datagram burst are made of the succession of the bytes constituting the OSI layer 3 datagram sequence, starting with the first byte of the header of the first datagram of the sequence and ending with the last byte of the payload of the last datagram of the sequence. Each byte in the datagram burst is assigned a position address that indicates the number of bytes separating the start of the burst from this byte. Each datagram within each datagram burst is assigned an address pointing to the first byte of the

datagram. Therefore, each datagram is uniquely identified by its datagram burst number k and its *address*.

Furthermore, each datagram burst k is assigned a datagram burst size referred to by $datagram_burst_size(k)$ which is equal to the address of the last byte of the last datagram in the burst plus one byte.

Datagram burst size is limited to $2^{18}-1 = 262143$ bytes due to signalling range restrictions of the address field in the real-time parameters in MPE header. However, the maximum size of a datagram burst MAY be restricted by other constraints. It is assumed that each datagram burst does not exceed a certain maximum datagram burst size $max_datagram_burst_size$, whereby an upper limit for $max_datagram_burst_size$ is $2^{18}-1$ due to the above-mentioned signalling reasons. Furthermore, the number of datagrams in each burst as well as the datagram burst size MAY vary for each datagram burst. The generation of datagram bursts from the sequence of OSI layer 3 (Network layer) datagrams is not further discussed, as it MAY depend on many different aspects.

Note that a datagram burst corresponds to a time-slice burst as defined in [1].

2.3.2 Encoding Period

The Encoding Period, or EP , determines the frequency with which FEC is computed: an EP of 1 means that the encoding process occurs at every datagram burst whereas an EP greater than 1 means that the encoding process occurs every EP bursts. For $EP > 1$, the encoding matrix capacity (ADT) is EP times greater than with $EP=1$ so that it takes EP times longer to fill it with data. EP is expressed in datagram burst units.

This parameter normalizes several other parameters:

- EP normalizes the encoding process depth B , which is the number of ADTs, expressed in units of EPs, over which the datagram bursts are interleaved.
- EP also normalizes the spreading process parameter S . The IFEC sections in one IFEC burst are interleaved from S iFDTs; here again S is expressed in EP units.

Finally, EP increases the encoding matrix size: since each ADT has $K=C*EP$ columns, when EP is increased, the encoding matrix size is therefore increased. This allows IFEC to use very large encoding matrices where appropriate.

2.3.3 Application Data Sub-Table

The Application Data Sub-Table (ADST) is a function that allows mapping of a datagram burst to an ADT: thereby $ADST(j)$ refers to the j^{th} column of the result of the mapping of the datagram burst on a matrix of C columns and T rows whereby $j=0,1,\dots, C-1$.

The mapping of a datagram burst with Layer 3 datagrams on such a $C*T$ matrix is shown in Figure 2: the leftmost columns of the matrix host all datagrams of a datagram burst. The remaining columns are filled with

possible padding. The first datagram in the datagram burst starts with its first byte in the upper left corner of the matrix and goes downwards to the first column. The length of the datagrams MAY vary arbitrarily from datagram to datagram. Immediately after the end of one datagram the following datagram starts. If a datagram does not end precisely at the end of a column, it continues at the top of the following column. When all datagrams have entered the matrix, any unfilled byte positions are padded with zero bytes, making all columns completely filled. After the mapping, each position in the matrix hosts an information byte or padding byte.

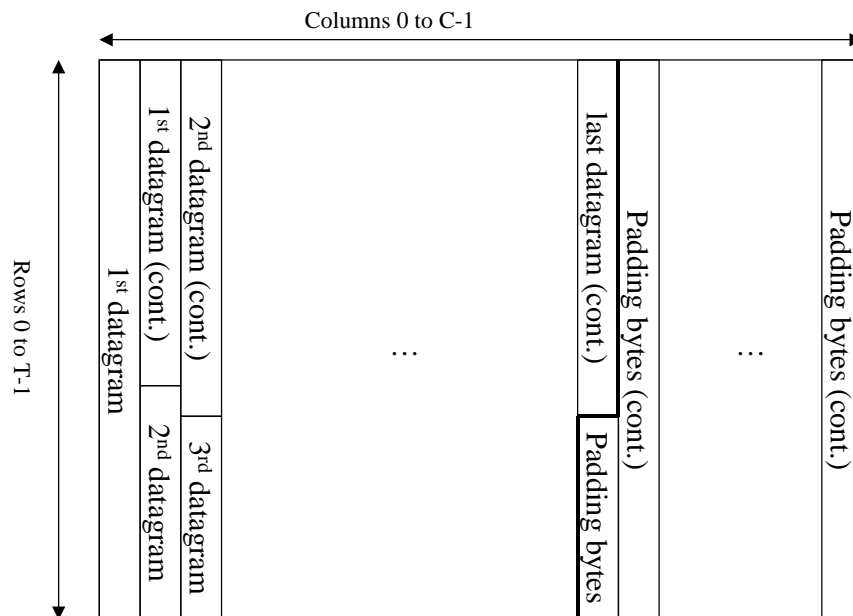


Figure 2: Datagram burst to ADST mapping

Note that maximum size of each datagram burst, $max_datagram_burst_size$, is restricted to $C \cdot T$.

2.3.4 Encoding Matrix

The IFEC encoding process hosts M encoding matrices. Encoding matrices are sequentially numbered by $m= 0, \dots, M-1$. The actual number M depends on several parameters and is specified in chapter 5. Each encoding matrix m contains exactly one Application Data Table (ADT), referred to by $ADT(m)$ and one IFEC Data Table (iFDT), referred to by $iFDT(m)$.



Figure 3: encoding matrix

2.3.4.1 Application Data Table

Each ADT (Application Data Table) is arranged as a matrix of $K=C*EP$ columns and T rows. Each column in each ADT is uniquely identified by its encoding matrix number m and its column number $n=0, \dots, K-1$ by $ADT(m,n)$.

2.3.4.2 IFEC Data Table

Each iFDT (IFEC Data Table) is arranged as a matrix of N columns and T rows. Each column in each iFDT is uniquely identified by its encoding matrix number m and its column number $n=0, \dots, N-1$ by $iFDT(m,n)$. The iFDT hosts the parity symbols generated for the corresponding ADT of the same encoding matrix.

2.3.5 MPE-IFEC section

An IFEC Section is comprised of a header, the data (parity symbols) from multiple iFDT columns in sequence, and a checksum. Structure of an MPE-IFEC section is specified in section 5 and generation of the IFEC section is covered in section 3.5.

2.3.6 IFEC burst

An IFEC burst is the collection of IFEC sections sent in one time-slice burst. The maximum size of an IFEC burst is R , where R is a number of IFEC sections. A new IFEC burst is generated with the reception of each datagram burst k . Each IFEC section of an IFEC burst is uniquely defined by its burst number $k' = k[k_{\max}]$ and its IFEC section index $j=0, \dots, R-1$. Each such IFEC section is therefore referred to as $IFEC(k', j)$. Note that an IFEC burst MAY contain less than R IFEC sections and that the section indices MAY be not consecutive.

2.3.7 Time-slice burst

The time-slice burst is *what is actually sent over the air*. A time-slice burst corresponding to datagram burst k consists of a collection of:

- MPE sections generated from datagram burst $k-D$.
- MPE-FEC sections generated from datagram burst $k-D$ (if MPE-FEC is also used).
- IFEC burst k containing MPE-IFEC sections generated when datagram burst k was received.

The transmission order of MPE sections, MPE-FEC sections, and MPE-IFEC sections is specified in section 3.6. Note that each of the sections requires settings of relevant real-time parameters, including time slicing. Time-slicing information can only be set after the definition of the transmission order.

Note that a time-slice burst does not correspond strictly to a time-slice burst as defined in [1] but to a "burst" instead.

3 Sender operation (normative)

3.1 Introduction

According to Figure 1, the sender operation takes as its input the datagram bursts and generates at its output the time-slice bursts. The flow diagram of the sender operation is shown in Figure 4. The initialization process before receiving the first datagram burst is described in section 3.3. Then following procedure is applied for each newly received datagram burst.

- 1) **Reception of new datagram burst:** the burst number k is incremented by one. The actual datagram burst k is stored such that it can be delivered after delay D . Its *datagram_burst_size(k)* is stored.
- 2) **Generation of IFEC burst:** The corresponding IFEC burst is generated taking into account the current datagram burst k , the data in iFDTs, as well as the *datagram_burst_size* of previous datagram bursts. The details of this process are described in section 3.5.
- 3) **Time-slice burst generation and sending:** The time-slice burst is generated from the MPE sections obtained from datagram burst $k-D$, possibly from the corresponding MPE-FEC sections included in this burst, and the IFEC sections of the corresponding IFEC burst. The details of this process are described in section 3.6.
- 4) **Datagram burst to ADTs mapping:** The actual datagram burst k is mapped by the use of the ADST to the ADTs. The ADST mapping includes the necessary padding. The details of this process are described in section 3.7.
- 5) **Generation of iFDT:** If the next burst number $k+1$ is a multiple of the encoding period EP it is necessary to generate the parity symbols in $\text{iFDT}(m)$ from the data symbols in $\text{ADT}(m)$, with $m = \text{floor}(k/EP) [M]$. The details of this process are described in section 3.8.
- 6) **Return to Receive the Next Datagram Burst:** The sender process continues in step 1 once a new datagram burst is received.

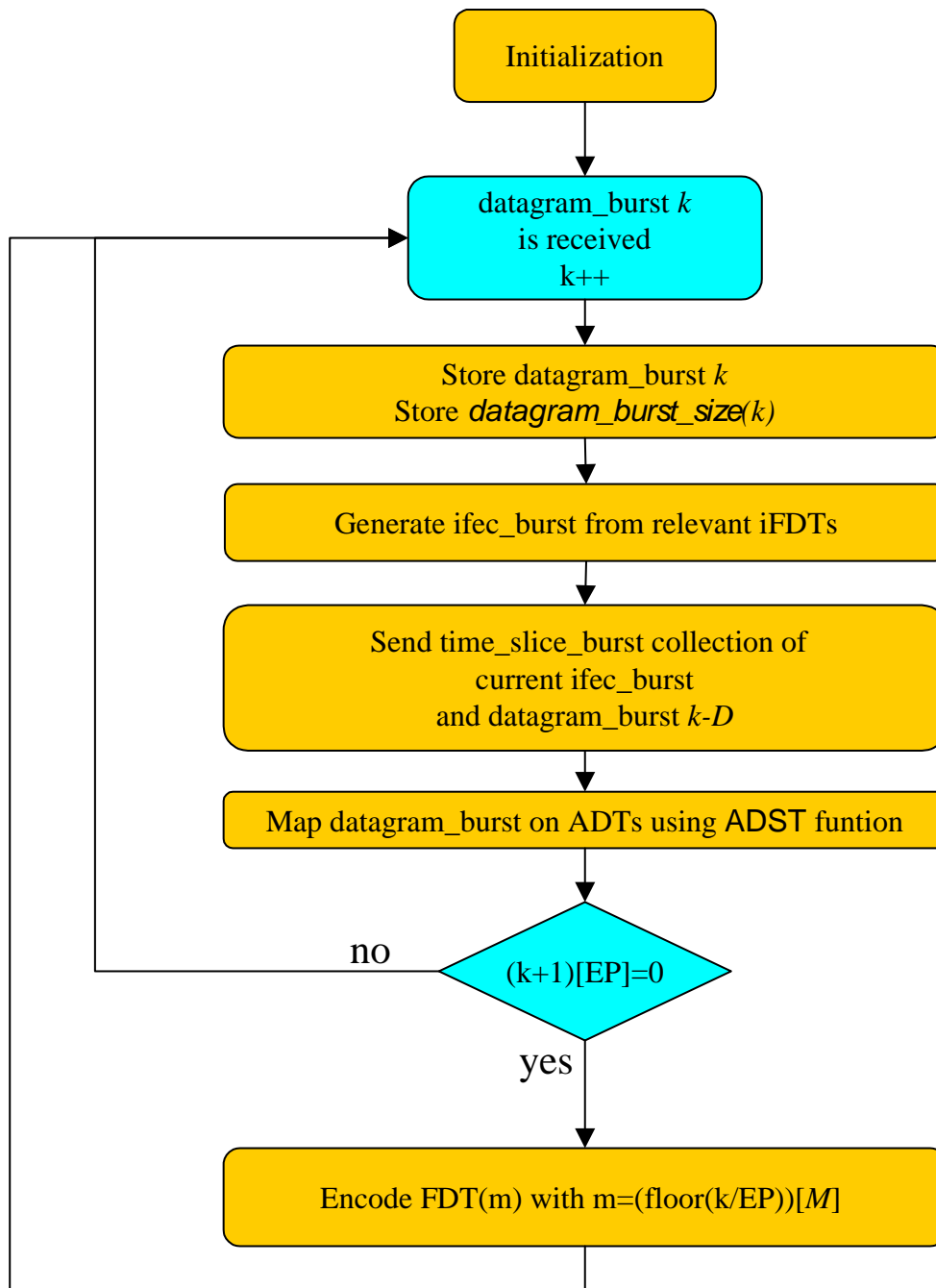


Figure 4: sender operation input/output

The description of the sender operation is the common framework. Functions and variables of this description are either part of the common framework and defined in this section or are specialized according to the code and defined in chapter 5.

3.2 Parameters

The following parameters are assumed to be available to the sender process and either by direct signalling or by indirect computation. The carriage or derivation of the parameters is detailed in chapter 5. Only those parameters required in chapter 3 are presented; as a consequence those parameters

that are used only by the mapping functions `adt_index` and `ifdt_index` are actually given in chapter 5. This is in particular the case for B & S parameters that are defined in the chapter 5.

Table 1: MPE-IFEC generic parameters list

Parameter	Unit	Category	Description	Signalling	Scoping
EP	Datagram burst	Taxonomy	IFEC Encoding Period	Direct via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
D	Datagram burst	Taxonomy	Datagram burst sending delay	Direct via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
T	rows	Table sizing	Number of ADST, ADT, iFDT rows: $T = \text{MPE-IFEC Frame rows} / G$	Indirect via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
C	columns	Table sizing	Number of ADST columns	Direct via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
R	sections	Table sizing	Maximum number of MPE IFEC sections per Time-Slice Burst	Direct via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
K	columns	Table sizing	Number of ADT columns = $EP * C$	Indirect via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
N	columns	Table sizing	Number of iFDT columns = $EP * R * G$	Indirect via <code>Time_slice_fec_identifier</code>	<code>Time_slice_fec_identifier</code>
G	columns	Table sizing	Maximum number of iFDT columns per IFEC section	Direct	<code>Time_slice_fec_identifier</code>
M	ADT	Protocol sizing	Number of concurrent encoding matrices M	Indirect (formula dependent on <code>T_code</code> and given in the parameter definition of chapter 5)	<code>Time_slice_fec_identifier</code>
k_{max}	N/A	Protocol sizing	Modulo operator for IFEC burst counter	Indirect (formula dependent on <code>T_code</code> and given in the parameter definition of chapter 5)	<code>Time_slice_fec_identifier</code>
j_{max}	N/A	Protocol sizing	Maximum backward pointing for datagram burst size used in <code>PREV_BURST_SIZE</code> parameter in §3.5	Indirect (formula dependent on <code>T_code</code> and given in the parameter definition of chapter 5)	<code>Time_slice_fec_identifier</code>
k	datagramburst	Index	continuous burst counter internal to sender	N/A	Loop
k'	IFEC burst	field	Burst number	N/A	IFEC section

3.3 Initialization

Before the first datagram burst is received, the following actions are required.

- The internal burst number k is set to any suitable value, e.g. $k_{\max}-1$
- M concurrent encoding matrices are allocated, each with one ADT of size K times T and one iFDT with size N times T .
- Each ADT and iFDT is filled entirely with 0 bytes.
- The datagram burst size of virtual previous bursts $k=0, \dots, k_{\max}-1$ is set to 0, i.e. for all $k=0, \dots, k_{\max}-1$ $datagram_burst_size(k)=0$.

3.4 Reception of New Datagram Burst

With the reception of a new datagram burst, the following steps are carried out

- Assign datagram burst number k by incrementing counter k of one unit.
- Store the corresponding datagram burst size $datagram_burst_size(k)$
- Store the datagram burst k such that it can be delivered after delay D .

3.5 Generation of IFEC-Burst

An IFEC burst is generated taking into account the current burst number k , data in iFDTs, as well as the datagram burst size of previous datagram bursts. In total, at most R IFEC sections are generated for each IFEC burst. Each IFEC section in this burst gets assigned a unique IFEC burst number $k'=k \llbracket k_{\max} \rrbracket$. In addition, each IFEC section gets assigned a unique section index j with $j=0, 1, \dots, R-1$.

For each MPE-IFEC section $IFEC(k', j)$ with IFEC burst number k' and section index j included in the IFEC burst, the following information is generated:

- `burst_number` = k' ; this value is used to identify to which IFEC burst the current section belongs. This value MUST be lower or equal to 255.
- `IFEC_burst_size`: this value defines the total number of IFEC sections included in this IFEC burst and is coded over 8 bits as the number of IFEC sections present in the IFEC burst minus one. `IFEC_burst_size` value of '0' means only one IFEC section is present in the IFEC burst. `IFEC_burst_size` maximum value MUST be lower or equal to `last_section_number` and $R-1$.
- `section_number` = j ; j varies between '0' (first section) and `last_section_number`. j MAY be larger than `IFEC_burst_size`. The iFDT index is obtained as $m = ifdt_index(k', j)$; equivalently m is the encoding matrix number as defined in section 2.3.4.
- `last_section_number`: the largest value that is encoded in the `section_number` field for the same table-id, version fields and `burst_number`. A value of 255 SHALL be used for `last_section_number` if this largest value is not known.
- `section_length` = g^*T+13 .
- `prev_burst_size` = $datagram_burst_size((k-j \llbracket j_{\max}-1 \rrbracket -1 + k_{\max}) \llbracket k_{\max} \rrbracket)$
- The remaining real-time parameters are set according to section 3.6.
- `IFEC_data_bytes` are obtained from $ifdt(m)$ and correspond to the sequence of the iFDT columns $ifdt(m, i)$, $ifdt(m, i+1)$, ..., $ifdt(m, i+g-$

1) with $i = \text{ifdt_column}(k', j)$ and g any integer between 1 and G . Each column $\text{iFDT}(m)$ contributes databytes in order from row 1 to row T .

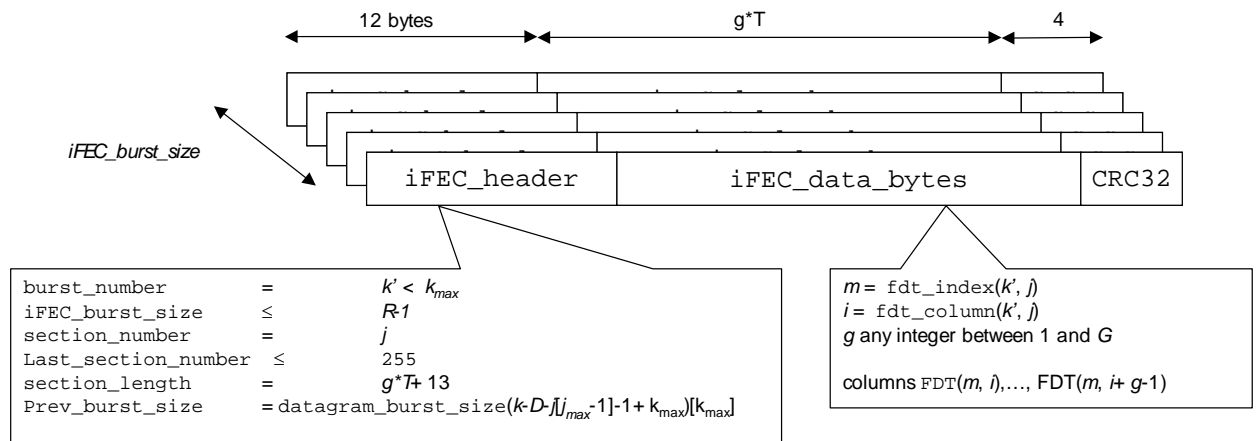


Figure 5: generation of iFEC burst

The exact mapping of this header and payload information on iFEC sections is provided in section 4.2. The functions $\text{ifdt_index}(k', j)$ and $\text{ifdt_column}(k', j)$ are specified in section 5 and depend on T_code signalled in `id_selector_bytes` of `time_slicing_fec_descriptor` and some additional parameters in the `id_selector_bytes` and `time_slicing_fec_descriptor`.

3.6 Time-Slice Burst Generation and Sending

The time-slice burst is generated from the MPE sections obtained from datagram burst $k-D$, possibly from the corresponding MPE-FEC sections included in this burst $k-D$, and the iFEC sections of the corresponding iFEC burst generated when datagram burst k was received. The rules for generating this time-slice burst are listed below:

- A time-slice burst MUST contain all datagrams in the same order coming from `datagram_burst(k-D)`:
 - o Datagram encapsulation is identical to the one specified in [1], section 9.6.
 - o MPE sections within one time-slice burst SHALL be sent with increasing address.
- A time-slice burst MAY contain MPE-FEC sections related to this datagram burst $k-D$, if present. The generation and encapsulation of such MPE-FEC sections SHALL follow the procedure described in [1] section 9.
- A time-slice burst MAY contain iFEC sections of the corresponding iFEC burst. The encapsulation of the iFEC burst into iFEC sections is described in section 3.5 and 4.2.

The following rules apply to MPE, MPE-FEC, and iFEC sections sending order within time-slice burst corresponding to the received datagram burst k :

- iFEC sections and MPE sections MAY be freely interleaved if their respective orders are respected.

- Since the only MPE-IFEC section conveys burst numbering information, at least one MPE-IFEC section SHOULD be positioned close to the beginning of the time-slice burst.
- Last IFEC section MAY be sent before last MPE section.

After having specified the sending order, for each of the MPE sections transmitted, the real-time parameters SHALL be set accordingly. For IFEC sections, the real-time parameters `delta_t`, `MPE_boundary`, and `frame_boundary` SHALL be set as specified in section 4.3.

3.7 Datagram Burst to ADT Mapping

The datagram burst k is mapped by `ADST` function to the ADT, including the necessary padding as specified in section 2.3.

The mapping of the datagram burst to the ADST is specified in section 2.3. Then, for $j = \text{all } 0, \dots, C-1$ the ADST columns `ADST(j)` are shifted into `ADT(m)`, whereby:

- $k' = k[k_{\max}]$ and $m = \text{adt_index}(k', j)$
- Shifting means that column `ADST(j)` is inserted at the right of `ADT(m)` moving all other columns to the left
 - o $\text{ADT}(m, 0) = \text{ADT}(m, 1)$
 - o $\text{ADT}(m, 1) = \text{ADT}(m, 2)$
 - o ...
 - o $\text{ADT}(m, K-2) = \text{ADT}(m, K-1)$
 - o $\text{ADT}(m, K-1) = \text{ADST}(j)$

$m = \text{adt_index}(k', j)$ is specified in section 5. Note that the insertion of columns in ADTs via `ADST` function is independent of whether `ADST(j)` is a data or a padding column: they are treated identically.

3.8 Generation of iFDT

If the next burst number $k+1$ is a multiple of the encoding period EP it is necessary to generate the N parity symbols in `iFDT(m)` from the K data symbols in `ADT(m)`, with $m = \text{floor}(k/EP) [M]$.

The details of iFDT generation are presented in section 5 and depend on `T_code` signalled in `id_selector_bytes` of `time_slicing_fec_descriptor`.

4 Carriage of MPE-IFEC sections (normative)

4.1 Generalities

When `time_slice_fec_identifier_descriptor` specifies that MPE-IFEC is used on an elementary stream, the MPE-IFEC parity of each Time-Slice burst SHALL be delivered in MPE-IFEC sections described in Table 2. MPE-IFEC sections are carried in the same elementary stream as the corresponding MPE data sections.

4.2 Syntax and semantics

MPE-IFEC sections are compliant to the DSMCC_section Type "User private" in [3]. The mapping of the section into MPEG-2 Transport Stream packets is defined in [4]. The syntax and semantics of the MPE-IFEC section are defined in Table 2.

Table 2: MPE-IFEC section

Syntax	Number of bits	Identifier
MPE-IFEC_section () {		
table_id	8	uimsbf
section_syntax_indicator	1	Bslbf
Private_indicator	1	Bslbf
Reserved	2	Bslbf
section_length	12	uimsbf
Burst_number	8	uimsbf
IFEC_burst_size	8	uimsbf
reserved	2	Bslbf
version	5	Uimsbf
Current_next_indicator	1	Bslbf
section_number	8	uimsbf
last_section_number	8	
real_time_parameters()	32	uimsbf
for(i=0; i<Nmax; i++) {		
IFEC_data_byte	8	uimsbf
}		
CRC_32	32	rpchof
}		

The semantics for the MPE-IFEC section:

table_id: Shall be set to value of 0x7A.

section_syntax_indicator: This field SHALL be set to 1 and be interpreted as defined by [3] §9.2.2.1.

private_indicator: This field SHALL be set to 0 and be interpreted as defined by [3] §9.2.2.1.

reserved: Shall be set to '11'.

section_length: Specifies the number of remaining bytes in the section immediately following this field up to the end of the section, including the CRC.

burst_number: This carries a burst continuity counter that SHALL vary between 0 and $k_{max}-1$, where k_{max} is the largest multiple of EP below 256. See section 3.5.

IFEC_burst_size: this 8-bit field gives total number of IFEC sections transmitted in the current time_slice_burst as defined in section 3.5.

reserved: These two bits SHALL be set to '11'.

version: this 5-bits field gives the version of the MPE-IFEC protocol signalled by this section. It MUST be set to '00000'.

current_next_indicator: Shall be set to a value of '1' and interpreted as defined by [4] § 2.4.4.11 ("current section is immediately applicable").

section_number: this 8-bit field gives the index of the section as defined in section 3.5.

last_section_number: the largest value that is encoded in the section-number field for the same table-id, version fields and burst_number fields. See section 3.5.

Real time parameters: see chapter 4.3.

IFEC_data_byte: Contains the IFEC data delivered as specified in section 3.5. N_{max} is equal to section_length-9.

CRC_32: This field SHALL be set as defined by [3]. It is calculated over the entire MPE-IFEC_section.

4.3 Real time parameters

Each MPE-IFEC section SHALL carry real time parameters described in Table 3.

Table 3: Time Slicing and MPE-IFEC real time parameters

Syntax	Number of bits	Identifier
real_time_parameters () {		
delta_t	12	uimsbf
MPE_boundary	1	bslbf
frame_boundary	1	bslbf
prev_burst_size	18	uimsbf
}		

delta_t: Usage of this 12-bit field depends on whether Time Slicing is used on the elementary stream.

The following applies when Time Slicing is used:

- The field indicates the time (delta-t) to the next Time-slice burst within the elementary stream. The time information is in each MPE-IFEC sections within a burst and the value MAY differ section by section. The resolution of the delta-t is 10 ms. Value 0x00 is reserved to indicate that no more bursts will be transmitted within the elementary stream (e.g. end of service). In such a case, all MPE-IFEC sections within the burst SHALL have the same value in this field as the MPE section.
- Delta-t information is the time from the start of the transport packet carrying the first byte of the current IFEC section to the start of the transport packet carrying the first byte of next burst. Therefore the delta-t information MAY differ between IFEC sections within a burst.
- The time indicated by delta-t SHALL be beyond the end of the maximum burst duration of the actual elementary stream. This ensures a decoder

can always reliably distinguish two sequential bursts within an elementary stream.

The following applies when Time Slicing is not used:

- The field supports a cyclic ADST index within the elementary stream. The value of the field increases by one for each subsequent ADST. After value '111111111111', the field restarts from '000000000000'.
- In case of large portions of lost data this parameter makes it possible to identify to which ADST the actual received section belongs.

MPE_boundary: This 1-bit flag, when set to '1', indicates that there are not any more MPE sections after this IFEC section in the time slice burst.

frame_boundary: This 1-bit flag, when set to '1', indicates that the current section is the last IFEC section of the time slice burst.

prev_burst_size: The field carries a description of the previous datagram burst size as specified in section 3.5.

5 Time Slice and FEC identifier descriptor (normative)

5.1 Introduction (informative)

This descriptor identifies whether MPE-IFEC is used on an elementary stream, in addition to the signalling of MPE-FEC. So the current specification comes in addition to the one found in [1] section 9.5 without contradicting it. In particular absolutely no change has been done on fields used by MPE-FEC.

The differences are:

- 1) the use of the `id_selector_bytes` (text in `id_selector_byte` table)
- 2) the `time_slice_fec_id` that can be set to 0x0 and 0x1 values (a new value MUST be inserted wherever a reference to `time_slice_fec_id` is made).

For clarity purpose, the text before the `id_selector_bytes` has been reproduced.

5.2 Descriptor (normative)

Table 4: Time Slice and FEC identifier descriptor

Syntax	Number of bits	Identifier
<code>time_slice_fec_identifier_descriptor () {</code>		
<code>descriptor_tag</code>	8	uimsbf
<code>descriptor_length</code>	8	uimsbf
<code>time_slicing</code>	1	bslbf
<code>mpe_fec</code>	2	uimsbf
<code>reserved_for_future_use</code>	2	bslbf
<code>frame_size</code>	3	uimsbf
<code>max_burst_duration</code>	8	uimsbf
<code>max_average_rate</code>	4	uimsbf
<code>time_slice_fec_id</code>	4	uimsbf
<code>for(i=0; I<id_selector_length; i++)</code>		
{		
<code>id_selector_byte</code>	8	bslbf
}		
}		

Semantics for Time Slice and FEC identifier descriptor

descriptor_tag: Shall be set to value of 0x77.

descriptor_length: This 8-bit field specifies the number of bytes of the descriptor immediately following this field.

time_slicing: This 1-bit field indicates, whether the referenced elementary stream is Time Sliced. The value "1" indicates Time Slicing being used, and the value "0" indicates that Time Slicing is not used.

mpe_fec: This 2-bit field indicates whether the referenced elementary stream uses MPE-FEC, and which algorithm is used. Coding is according to Table 5.

Table 5: Syntax and semantics for `mpe_fec`

value	MPE-FEC	algorithm
00	MPE-FEC not used	n/a
01	MPE-FEC used	Reed-Solomon(255, 191, 64)
10	Reserved for future use	
11	Reserved for future use	

reserved_for_future_use: This 2-bit field SHALL be set, when not used, to "11".

frame_size: This 3-bit field is used to give information that a decoder MAY use to adapt its buffering usage:

- In case Time Slicing is used (i.e. `time_slicing` is set to "1"), this field indicates the maximum number of bits on section payloads allowed within a Time-slice burst on the elementary stream, excluding MPE-IFEC sections. For MPE sections, bits are counted over `ip_datagram_data_bytes` or `LLC_SNAP` field (whichever is supported), excluding any possible `stuffing_bytes`. For MPE-FEC sections, bits are counted over `rs_data_bytes`.
- When MPE-FEC is used (i.e. `mpe_fec` is set to 0x1), this field indicates the exact number of rows on each MPE-FEC Frame on the elementary stream.
- If both Time Slicing and MPE-FEC are used on an elementary stream, both constraints (i.e. the maximum burst size and the number of rows) apply.
- When MPE-IFEC is used (`time_slice_fec_id` is set to 0x1) this field is computed as in the case both time slicing and MPE-FEC are used.
- Coding of the `frame_size` is according to Table 6.

Table 6: Syntax and semantics for `frame_size`

Size	Max Burst Size	MPE-FEC Frame rows
0x00	512 kbits = 524 288 bits	256
0x01	1 024 kbits	512
0x02	1 536 kbits	768
0x03	2 048 kbits	1 024
0x04 to 0x07	Reserved for future use	reserved for future use

max_burst_duration: This 8-bit field is used to indicate the maximum burst duration in the elementary stream when time slicing is used. A burst SHALL not start before T1 and SHALL end not later than at T2, where T1 is the time indicated by `delta-t` in the previous burst, and T2 is T1 + maximum burst duration. If the `time_slice_fec_id` is set to 0x0 or 0x1, the indicated value for maximum burst duration SHALL be from 20 ms to 5,12 s, the resolution is 20 ms, and the field is decoded according to the following formula:

$$\text{Maximum burst duration} = (\text{max_burst_duration} + 1) \times 20 \text{ ms}$$

If the `time_slice_fec_id` is set to any other value than 0x0 or 0x1, the coding of the `max_burst_duration` is currently not defined. When `time_slicing` is set to '0' (i.e. Time Slicing not used), this field is reserved for future use and SHALL be set to 0xFF when not used.

max_average_rate: This 4-bit field is used to define the maximum average bit rate in MPE section payload level over one time slicing cycle or MPE-FEC frame or ADST cycle and it is given by:

$$C_b = \frac{B_s}{T_c},$$

where B_s is the size of the current Time Slicing burst or MPE-FEC Frame or ADST cycle in MPE section payload bits and T_c is the time from the transport packet carrying the first byte of the first MPE section in the current burst/frame to the transport packet carrying the first byte of the first MPE section in the next burst/frame within the same elementary stream. Note that, when MPE-FEC is used, the RS data is not included in B_s nor is the IFEC parity data when MPE-IFEC is used. If `time_slice_fec_id` is set to 0x0 or 0x1, the coding of the `max_average_rate` is according to Table 7. If `time_slice_fec_id` is set to any other value, coding of the `max_average_rate` is currently not defined.

Table 7: Syntax and semantics for `max_average_rate`

Code	Bitrate
0000	16 kbps
0001	32 kbps
0010	64 kbps
0011	128 kbps
0100	256 kbps
0101	512 kbps
0110	1 024 kbps
0111	2 048 kbps
1000 to 1111	reserved for future use

time_slice_fec_id: This 4-bit field identifies the usage of following `id_selector_byte(s)`. Note that this field affects on coding of `frame_size`, `max_burst_duration` and `max_average_rate` fields on the actual descriptor, and the address field of real-time parameters on the referred elementary stream. If this field is set to value 0x0 `id_selector_byte(s)` SHALL not be present. If this field is set to value 0x1 then the id bytes are the following. Other values are not defined.

id_selector_length:

gives the lengths of the `id_selector_byte` field counted in bytes as given by Table 8.

Table 8: Syntax and semantics for `id_selector_length`

Time_slice_fec_id	Id_selector_length in bytes
0x0	0
0x1	9
other	undefined

id_selector_byte:

if `time_slice_fec_id` is set to 0x1 the bytes have the following definition:

Table 9: semantics for `time_slice_fec_id`

Syntax	Number of bits	Identifier
<code>Time_slice_fec_id_0x1()</code> {		
T_code	2	Uimsbf
G_code	3	uimsbf

Reserved for future use	3	bslbf
R	8	uimsbf
C	13	uimsbf
Reserved for future use	3	bslbf
B	8	uimsbf
S	8	uimsbf
D	8	uimsbf
EP	8	uimsbf
Max_rate_averaged_over_B	8	uimsbf
}		

T_code: This 2-bit field indicates the type of IFEC code used. Currently only value 0 is used and corresponds to MPE-FEC Reed Solomon (255,191). Section 5.4 adds an informative section introducing the use of Raptor codes as specified in [2], Annex C.4 in the framework.

Table 10: Syntax and semantics for T_code

Bit rate	Description
00	Reed Solomon code ([1], 9.5.1)
01	Raptor Codes ([2], Annex C.4)
01 to 11	reserved for future use

G_code: This 3-bit field indicates value of the G parameter as $G=2^{G_code}$ (number of symbols).

Table 11: Syntax and semantics for G_code parameter

Bit rate	G
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

reserved_for_future_use: This 3-bit field SHALL be set, when not used, to "111".

R: This 8-bit field indicates the maximum number of MPE-IFEC sections transmitted in an IFEC burst.

C: This 13-bit field indicates the number of columns for the ADST.

reserved_for_future_use: This 3-bit field SHALL be set, when not used, to "111".

B: This 8-bit field indicates value of the encoding process depth parameter normalized by EP . B is the number of ADTs over which the datagram bursts are interleaved.

s: This 8-bit field indicates value of the spreading process parameter normalized by EP : the IFEC sections in one IFEC burst are interleaved from S FDT.

D: This 8-bit field indicates value of the delaying process parameter. D is the number of datagram bursts by which the current transmission lags the current datagram burst.

EP: This 8-bit field indicates value of the encoding period parameter.

Max_rate_averaged_over_B: This 8-bit field is used to define the maximum bit rate in MPE section payload averaged over B time slicing cycle or MPE-FEC frame or ADST cycle and it is given by:

$$C_B = \max \left(\frac{\sum_{i=n-n[EP]}^{i=(n-n[EP]+(EP-1))+B-1} B_i}{T_n} \right)_{\forall n \geq 0},$$

where

- B_i is the size of i^{th} Time Slicing burst or MPE-FEC Frame or ADST cycle in MPE section payload bits
- T_n is the time from the transport packet carrying the first byte of the first MPE section in burst/frame $n-n[EP]$ to the transport packet carrying the first byte of the first MPE section in burst/frame $(n-n[EP]+(EP-1)+B)$ within the same elementary stream.

When MPE-FEC is used, the RS data is not included in B_i nor is the IFEC parity data when MPE-IFEC is used.

If `time_slice_fec_id` is set to 0x1, the coding of the `max_rate_averaged_over_B` is equal to $\text{floor}(\text{max_rate_averaged_over_B}/8)$ where `max_rate_averaged_over_B` is expressed in kbps. If `time_slice_fec_id` is set to any other value, coding of the `max_rate_averaged_over_B` is currently not defined.

Note that `max_rate_averaged_over_B` is always \leq `max_average_rate`

5.3 Sliding Encoding with RS code (normative)

5.3.1 General

This encoding scheme is an extension of the MPE-FEC. It uses the same encoding parity code as the MPE-FEC, benefiting from its existing hardware support.

5.3.2 Parameter Definitions

The following parameter restrictions apply:

- The τ_{code} value SHALL be set to 00.
- The encoding period MUST be set to 1, i.e. EP is set to 1.
- The IFEC data spread, B , is signalled in the $id_selector_byte \rightarrow B$, [0;255]
- The IFEC spread, S , is signalled in the $id_selector_byte \rightarrow S$, [0;255]
- The data burst delay, D , is signalled in the $id_selector_byte \rightarrow D$, [0;255]
- The number of data columns in the ADST, C , is signalled in the $id_selector_byte \rightarrow C$, [0;191].
- The maximum number of sections in one IFEC burst, R , is signalled in $id_selector_byte \rightarrow R$, [0;64]
- The number of rows in the ADST, ADT, and iFDT T is signalled in $frame_size \rightarrow mpe_fec_frame_rows \in \{256; 512; 768; 1024\}$
- The maximum number of iFDT columns in one MPE-IFEC section, G , MUST be set to 1, i.e. G_code MUST be set to 000.
- The number of encoding matrices, M , is given as $M=B+\max(0, S-D)+\max(0, D-B)$
- Furthermore, with $EP=1$, it is obvious that the number of columns in each ADT is given as $K=C$ and, in addition with $G=1$ the number of columns in each iFDT is given as $N=R$.
- The modulo burst counter is set to $k_{max}=256-256[M]$.
- The modulo counter for the previous burst signalling is set to $j_{max}=M$

For the special case $D=0$, this encoding scheme results in the case that each Datagram Burst is interleaved in a sliding window of B ADTs out of the total of $M=B+S$ ADTs. For each datagram burst only one iFDT is computed, that will be sent in the IFEC Bursts of the S following time-slice bursts.

5.3.3 Mapping Functions

The following parameter mapping functions SHALL be used.

The ADT index for a given modulo datagram burst number k' and a given ADST column number j is given as:

$$adt_index(k', j) = (k' + \lfloor j/B \rfloor)[M]$$

The iFDT index for a given modulo datagram burst number k' and a given section index number j is given as:

$$\text{ifdt_index}(k', j) = (k' - j[S] - 1 + M)[M]$$

The iFDT column for a given modulo datagram burst number k' and a given section index number j is given as:

$$\text{ifdt_column}(k', j) = j$$

5.3.4 Generation of iFDT

For a given ADT, the generation of the iFDT follows the definition of [1] section 9.5.1.

Any $K < 191$ is achieved by shortening as specified in [1] section 9.3.3.1.

Any $N < 64$ is achieved by puncturing as specified in [1] section 9.3.3.2.

5.3.5 Memory considerations

see [5].

5.4 Generalized Encoding with Raptor code (informative)

5.4.1 General

The inclusion of a Raptor encoding scheme extends the flexibility compared to the sliding RS scheme as presented in section 5.3. This is achieved as the ADT size can be increased significantly and one MAY benefit from the possibility of a full software implementation of the IFEC decoding.

In case that this encoding scheme is used, this is signalled by the FEC identifier descriptor by T_code value 01.

5.4.2 Parameter Definitions

The following parameter restrictions apply:

- The T_code value SHALL be set to 01.
- The encoding period EP is signalled in the $id_selector_byte \rightarrow EP$, [0;255]
- The IFEC data spread, B , is signalled in the $id_selector_byte \rightarrow B$, [0;255]
- The IFEC spread, S , is signalled in the $id_selector_byte \rightarrow S$, [0;255]
- The data burst delay, D , is signalled in the $id_selector_byte \rightarrow D$, [0;255]
- The number of data columns in the ADST, C , is signalled in the $id_selector_byte \rightarrow C$, [4;8192].
- The maximum number of sections in one IFEC burst, R , is signalled in $id_selector_byte \rightarrow R$, [0;256]
- The maximum number of iFDT columns in one MPE-IFEC section, G , is signalled in the $id_selector_byte \rightarrow G_code$, whereby $G=2^{(G_code)}$;
- The number of rows in the ADST, ADT, and iFDT T is derived from the $frame_size \rightarrow mpe_fec_frame_rows \in \{256; 512; 768; 1024\}$ and the parameter G as $T=(frame_size \rightarrow mpe_fec_frame_rows)/G$
- The number of columns in an each ADT is given as $K=C*EP$, whereby it MUST be ensured that $K \leq 8192$.
- The number of columns in each iFDT is given as $N=R*G*EP$
- The number of encoding matrices, M , is defined as $M= \max(B, S, \text{ceil}(D/EP), B+S-\text{floor}(D/EP))$
- The modulo burst counter is set to $k_{\max} = \text{floor}(256/(M*EP))*M*EP$
- The modulo counter for the previous burst signalling is set to $j_{\max} = M*EP$

5.4.3 Mapping Functions

The following parameter mapping functions SHALL be used.

The ADT index for a given modulo datagram burst number k' and a given ADST column number j is given as:

$$\text{adt_index}(k', j) = (\text{floor}(k'/EP) + j[B])[M]$$

The ADT column for a given modulo datagram burst number k' and a given ADST column number j is given as:

$$\text{adt_column}(k', j) = \text{floor}((\text{floor}(s/EP)*C)/B) + C*(s[EP]) + \text{floor}(j/B),$$

with $s = (B - j + (k'[EP])[B] - 1)*EP + k'[EP]$

The iFDT index for a given modulo datagram burst number k' and a given section index number j is given as:

$$\text{ifdt_index}(k', j) = (\text{floor}(k'/EP) - j[S] - 1 + M)[M]$$

The iFDT index for a given modulo datagram burst number k' and a given section index number j is given as:

$$\text{ifdt_column}(k', j) = (k'[EP])*R + j*G$$

5.4.4 Generation of iFDT

The iFDT is generated using the Systematic Raptor Encoder as specified in [2], Annex C.4, whereby

- The ADT corresponds to a source block.
- Columns of the ADT correspond to source symbols
- Columns of the iFDT correspond to repair symbols

The source block size K is defined as $C*EP$, the symbol size T corresponds to the column size of the ADT and iFDT. Sub-blocking SHALL not be used.

5.4.5 Example Parameters

This section provides recommendations for the setting of the transport parameters `id_selector_byte->G_code` and `id_selector_byte->C` at the sender. Note that other parameters MAY be applied and signaled based on specific criteria.

This recommendation is based on the following input parameters:

- N_{Burst} the maximum burst size as signalled by the `frame_size` value
- `nof_fec_rows`, the MPE-FEC frame rows as signalled by the `frame_size`
- r the target code rate between 0 and 1
- EP the encoding frequency
- K_{MAX} the maximum number of columns per ADT with $K_{\text{MAX}} = 8192$ according to the maximum systematic index in [2], Annex C.5.
- K_{MIN} the minimum number of columns per ADT with $K_{\text{MIN}} = 4$ according to the minimum systematic index in [2], Annex C.5.
- K_{TARGET} the target number of columns per ADT.
- T_{MIN} the minimum symbol size for the ADT

Then, $N_{\text{ADST}} = \text{ceil}(r* N_{\text{Burst}})$ is the expected maximum ADST size.

Then, $T = \max\{\bar{T}_{\text{MIN}}, \min_{i=0,1,2,\dots,7}\{ G=R/2^i \mid (EP * \text{ceil}(N_{\text{ADST}}/(R/2^i))) \leq K_{\text{TARGET}} \} \}$ is an appropriate symbol size, i.e. T is selected such that the Raptor source block has at least K_{TARGET} symbols, but the symbol size is always at least \bar{T}_{MIN} bytes and divides the number of MPE-FEC frame rows by a power of 2.

Then the proposed parameters are as follows:

$$G = \text{nof_fec_rows} / T \text{ and } \text{id_selector_byte} \rightarrow G_code = \log_2(G)$$

$$C = \text{ceil}(N_{\text{ADST}}/T) \text{ and } \text{id_selector_byte} \rightarrow C = C$$

Recommended settings for the input parameters are $K_{\text{TARGET}} = 2048$ and $\bar{T}_{\text{MIN}}=32$.

Assume a frame_size=0x03 with $N_{\text{Burst}}=2048$ kbits and nof_fec_rows=1024, then the above algorithm results in the following parameter settings.

Table 12 Recommended parameter settings for frame_size=0x03

	EP=1		EP=4		EP=8		EP=16		EP=32	
	C	G	C	G	C	G	C	G	C	G
$r=7/8$	3584	16	896	4	224	1	448	2	224	1
$r=3/4$	3072	16	768	4	192	1	384	2	192	1
$r=2/3$	2730	16	682	4	170	1	341	2	170	1
$r=1/2$	2048	16	512	4	128	1	256	2	128	1

6 Receiver Operation (informative)

6.1 Introduction

In this chapter we give informative background on how an MPE-IFEC decoder compliant to the coding profiles defined in sections 5.3 and 5.4 could be implemented:

- We give a general process overview
- Then we give more details on particular steps of this process
 - o parameters derivation from the time_slice_fec_descriptor
 - o burst number detection
 - o section reception
 - o padding ADST
 - o Main options in the decoding process

6.2 General Process

The sender operation takes as its input the sections received in one time-slice burst and generates at its output datagram bursts that contains a sequence of datagram. The basic steps of an example receiver operation are shown in Figure 6.

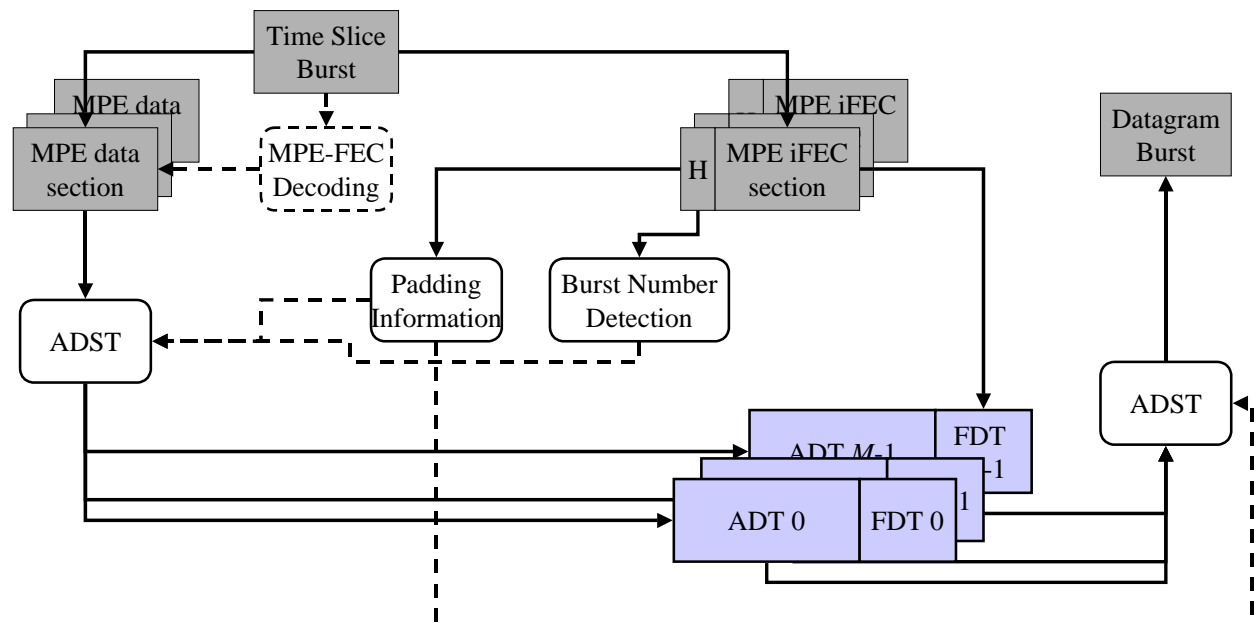


Figure 6: Example Receiver Operation

Assume that we have detected the reception of a time-slice burst that MAY contain

- None, one or several MPE data sections
- None, one or several MPE-FEC sections
- None, one, or several MPE-IFEC sections

In case that MPE data sections are detected to be missing, and if MPE-FEC sections for this time-slice burst have been received, an MPE-FEC decoding

MAY be initiated to recover the MPE data sections for this time-slice burst. In any case, even if everything is correct, the following IFEC decoding process should be started because the data can be used to correct other burst.

The following steps are given for information; other alternatives MAY be given to optimize display time (e.g. fast zapping):

0. Detect current burst number; different techniques are presented in 6.4
1. Use received IFEC sections to perform IFEC decoding; this is presented in 6.7. If a combination of ADT and iFDT has sufficient data to recover, an ADT MAY be recovered and corresponding datagram bursts MAY be recovered.
2. Use received MPE section and, using ADST function, map them onto the ADTs; this is detailed in section 6.5 and 6.6
3. Go to 0

When one or several successive complete burst losses happen, it MAY be impossible to detect burst numbering straight away. When this is possible (for instance when another burst is received and the loss is detected) then step 2 is applied for each previously lost burst.

Usually one IFEC section is enough to reconstruct the burst number, reconstruct some padding information, and to insert the IFEC section parity bytes in the relevant positions in the iFDTs (see section reception).

6.3 Parameters

With the availability of an INT and `time_slicing_fec_descriptor`, the following parameters can be determined following the same procedure as at the sender.

Table 13: parameters for receiver operations

Static Parameters	Description
EP	IFEC Encoding period
B	IFEC Data Interleaving
S	IFEC Spread
D	Data delay at sender
C	Maximum number of data columns per ADST
T	Symbol size of code (Number of rows in ADST/ADT/iFDT)
G	Maximum number of symbols per MPE-IFEC section
M	Number of Concurrent ADT and iFDT
K	Number of columns in ADT
N	Number of columns in iFDT
R	maximum number of IFEC sections in an IFEC burst
k_{max}	Modulo counter for burst
T_{bmax}	maximum burst duration

6.4 Burst number detection

Since MPE sections do not carry the time-slice burst number, the first thing that a DVB-IFEC capable receiver must do is to determine to which time-slice burst the section belongs to. This can be done through different ways:

- An MPE-IFEC section must have already been received beforehand so that we have a reference number of burst in the past;
- We can derive the current burst number using real time parameters of the MPE section
 - o Let $I_p/T_p/\Delta T_p$ be the burst number and time of the preceding MPE-IFEC section
 - o Let $I_n/T_n/\Delta T_n$ be the burst number and time of the following MPE-IFEC section
 - o Let T and ΔT the time and delta-t information of the current MPE section

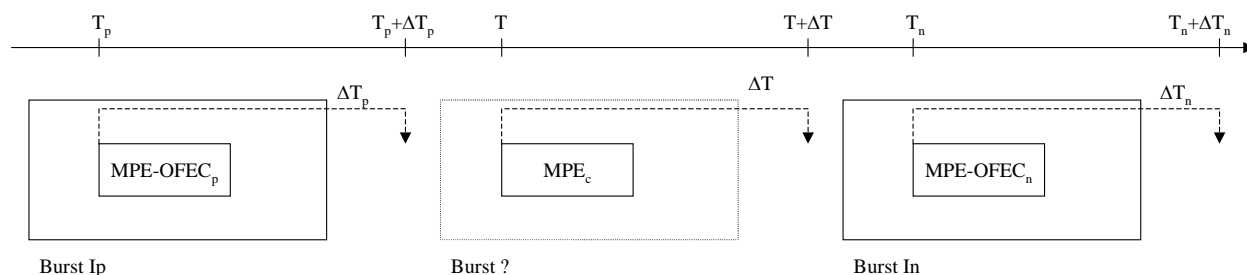


Figure 7: burst numbering, general case

- o We are looking for the current burst number:
 - If $(I_p = I_n)$ burst_number = $I = I_p = I_n$
 - Otherwise, the result MAY be based on some timing heuristic:
 - By definition, $T_p < T$ and $T < T_n$
 - If $T_p + \Delta T_p < T$ and $T_n < T + \Delta T$ THEN $I = I_n$

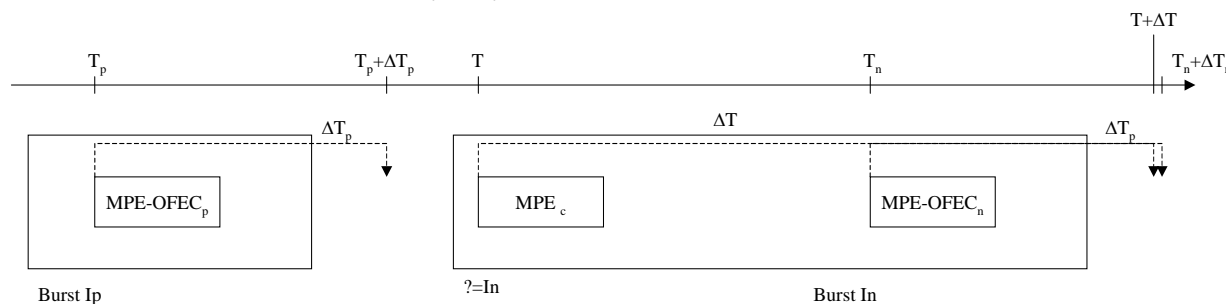


Figure 8: burst numbering, case of next burst

- ELSE $T < T_p + \Delta T_p$ THEN $I = I_p$

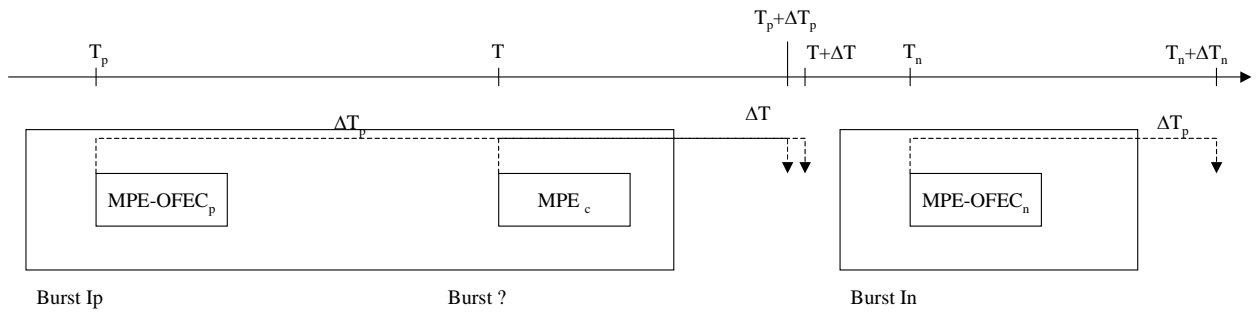


Figure 9: burst numbering, case of previous burst

- ELSE IGNORE that MPE section for MPE-IFEC decoding
 - Obviously, the discovery of the $l=l_p$ can be achieved only if there are MPE-IFEC sections preceding the MPE data sections, the latter being completely implementation dependent.

6.5 Section Reception

Once the burst numbering has been done, it is possible to place each received section inside their ADT or iFDT:

- MPE-IFEC sections:
 - o Each MPE-IFEC section carries a `section_number` that enables to position this IFEC section inside one iFDT according to the algorithm described in section 3.5.
 - o Note that in any case (and contrary to MPE-FEC), even if all sections in current datagram burst have been received correctly, the receiver SHALL continue to receive remaining MPE-IFEC sections since they can be used in different FDTs than the ones used to decode the current ADST, or they could be used in the same FDTs as those used to decode the current ADST but for decoding other more erroneously received ADSTs.
 - o The number of received sections in the current burst cannot exceed the signalled `fec_burst_size` in every IFEC section received. If there are fewer sections delivered, this means that some IFEC sections have been lost.
 - o When reception is over, in case one section is missing, there MAY be a decision from the receiver with regard to decoding because it has learnt the total number of IFEC columns transmitted for an iFDT using `max_iFDT_column` signalled in the sections carrying the columns of this iFDT. If losses are too severe on the ADT, it MAY be impossible to decode.
 - o The IFEC sections carry also `prev_burst_size` information enabling to delineate the padding part within the previously received burst.
- MPE sections:
 - o Each correctly received (id est with a correct CRC32) MPE-section can be placed at the right position inside the ADST since each MPE section carries in the section header a start address

for the payload. The receiver will then be able to put the received datagram in the right byte positions in the ADST and mark these positions as "reliable".

- o When all possible MPE sections correctly received have been processed, the padding or erased bytes are added according to chapter 6.6. Erased bytes are positioned as unreliable whereas padding bytes are positioned as reliable. Then the C columns are interleaved inside their relevant ADT using same algorithm as in the sender described in section 3.6.

6.6 Padding in ADST mapping

The receiver introduces the number of padding columns during the ADST mapping by marking these padding bytes as reliable. Some of the following guidelines MAY be used to recover the padding information whereby the description is supported by some examples in Figure 10:

- If the receiver has received the last MPE section of the time-slice burst correctly, it can mark any remaining padding bytes as reliable (case $k'-2$ in the figure below).
- If the receiver did not receive the last section correctly, it has to check if the `prev_burst_size` or this burst is available; if yes, this implies that all byte positions after the last correctly received section until the burst size signal by `prev_burst_size` are lost data and will mark them as unreliable (case $k'-1$).
- If the receiver has received neither the signalling of the padding size using `prev_burst_size`, nor the last MPE section, then it will have to mark all bytes positions after the last correctly received section as lost data and mark the corresponding byte positions as unreliable (case $k'-3$).
- If there are missing sections before the last section, the receiver can set as unreliable the corresponding bytes (case $k'-4$).

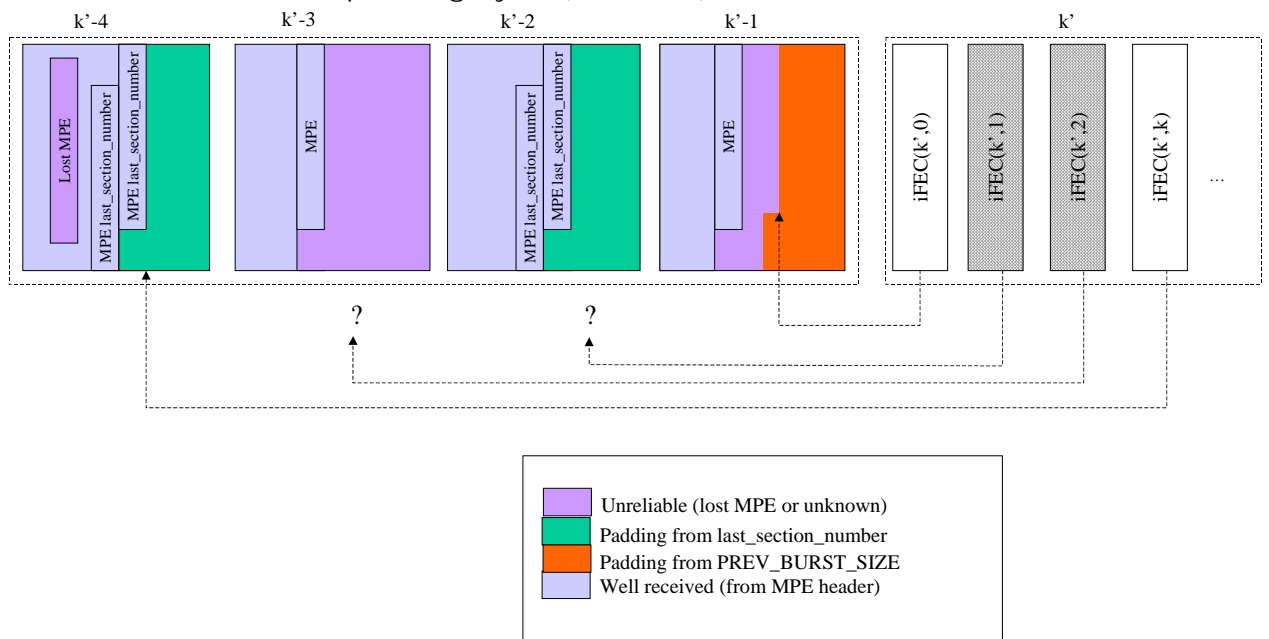


Figure 10: padding strategy

6.7 Decoding

The scope of this section is to exemplify typical receiver procedures for the reconstruction of a flow of IP datagram bursts from a sequence of received time-slice bursts. The reference receiver presented in this section SHALL be intended as to meet minimum performance requirement and therefore its structure is intentionally kept simple. Specific implementations MAY differ in some implementation aspects, but they SHALL still be considered fully compliant as long as they fulfil at least the minimum performance of the reference receiver.

6.7.1 Input

The IFEC decoding attempts to recover an ADT with use of the information in the ADT, the padding information in the ADT, as well as the corresponding iFDT. The ADT consists of K columns and T rows, the iFDT consists of $R \cdot EP$ columns and T rows. Only CRC32 valid sections are positioned: all the bytes that constitute the section are signalled as unreliable; in order to save memory, each column in the ADT is MAY be marked as unreliable, if any of the T bytes in the column is erased. Then, if the total number of non-erased columns in ADT and iFDT is at least K , IFEC decoding SHALL be applied.

6.7.2 RS decoding

For the case of RS codes, at every burst, one (ADT , iFDT) couple can be decoded using the same principles as for MPE-FEC presented in [1] section 9.3.3.

6.7.3 Raptor decoding

For the case of Raptor codes, an appropriate decoding algorithm is provided in [2], Section C.7, whereby

- The source block is determined by the number of symbols K and the symbol size T
- The ensemble of encoding symbols consists of
 - o The non-erased columns in the ADT, whereby the encoding symbol ID (ESI) corresponds column number in the ADT
 - o The available columns in the iFDT, whereby the ESI corresponds to the column number in iFDT plus K .

If IFEC decoding is successful, the ADT columns are updated by the reconstructed encoding symbols.

6.7.4 Output

Depending on the number of erroneous bytes after frame decoding, several options are possible:

- All missing bytes have been corrected, all datagrams are forwarded to the application;
- Some bytes have not been corrected; only correct datagrams are forwarded to the application.
- Some bytes have not been corrected; the erroneous datagrams are also forwarded to the application.

END OF DOCUMENT