II
Transport Stream Monitoring

This chapter starts with the discussion of the MPEG-2 Transport Stream (TS) structure and some issues related with the information transported by the same. There are in fact parameters, as it will be made clear, that can be monitored and will be helpful when trying to quantify transport impairments and hence, the effectiveness of the error correction code for given channel characteristics. Such Transport Stream derived parameters will actually indicate the quality of the stream arriving at the decoder, which may vary according to the channel coding technique employed. When transmitting video over IP, the delivered Transport Stream might be subject to impairments which include jitter or packet drops.

The Transport Stream structure understanding is fundamental for the experimental set up presented in Chapter IV. It is worth mentioning that this packing method, which organizes the information to be transported — video, audio and control data — structured as a TS, has been employed in most professional IP video transmission systems up to date.

MPEG-2 Transport Streams can be classified as *Single Program Transport Stream* (SPTS), carrying audio, video and data services associated to one program only or *Multi-Program Transport Stream* (MPTS), carrying services associated to multiple programs.

As far as the video bit rate is concerned, a TS can present data at either variable bit rate (VBR) or at constant bit rate (CBR) — these aspects will be discussed later on.

A description of the structure of a Transport Stream Packet (TSP) and its composition is next ensued.

II.1 Transport Stream Structure

A Transport Stream is a string of Transport Stream Packets, TSP's from now on. Each TSP is a 188 [1] bytes long packet, which encapsulates data provided by the source encoder, i.e., bytes drawn from, typically, an MPEG-2 or MPEG-4 compressed A/V data.

For the purpose of this study, the message generated by the source of information, which is to be transmitted to a remote user through the *IP channel*[2], consist of two sequences of bytes carrying audio and video (A/V) information. Let us designate this messages by the vectors $\mathbf{m}_A$ and $\mathbf{m}_V$, respectively.

Figure II.1 shows the components of the three main blocks concerning processing of audio and video (A/V) information into a Transport Stream and further into IP packets, subject to transmission.

In the *Source Coding* block, $\mathbf{m}_A$ and $\mathbf{m}_V$ are processed by the corresponding source encoders, typically MPEG-2 or MPEG-4, for the video sequence and AAC (Advanced Audio Coding) or AC3 (Audio Coding 3) for the audio sequence. The source coding block provides the so called *Elementary Streams*, represented by $\mathbf{e}_A$ and $\mathbf{e}_V$.

Next, in the *System Coding* block, $\mathbf{e_a}$ and $\mathbf{e_v}$ are fed to *Packetizers* which encapsulate this information into strings designated *Packetized Elementary Streams* (PES), $\mathbf{u}_A$ and $\mathbf{u}_V$, carrying audio and video information, respectively.

Each PES carries information from a single audio or video service and is fed to the *Transport Stream multiplexer* (TS MUX) sub-block, which distributes the symbols from $\mathbf{u}_A$ and $\mathbf{u}_V$ across the TSP's and adds the associated header and programming information.

The TS MUX sub-block also multiplexes TSP's carrying information from different services and auxiliary data into a single string of TSP's, originating the Transport Stream (TS), represented herein by the string $\mathbf{ts}$.

Finally, in the *Channel Coding and IP Packetizing* block, the multiple TSP's will be encoded with an erasure protection code, a Fountain Code herein and encapsulated into the network protocols (RTP/UDP/IP) for transmission.

[1] some standards define an intra-packet Reed-Solomon code, in which case the TSP will be larger than 188 bytes, i.e. 204 byte packets upon protection with an RS[204,188].

[2] The IP channel considered herein comprises single or multi-hop IP links, the Internet, a Local Area Network (LAN) or a Wide Area Network (WAN), all subject to packet drops and jitter.

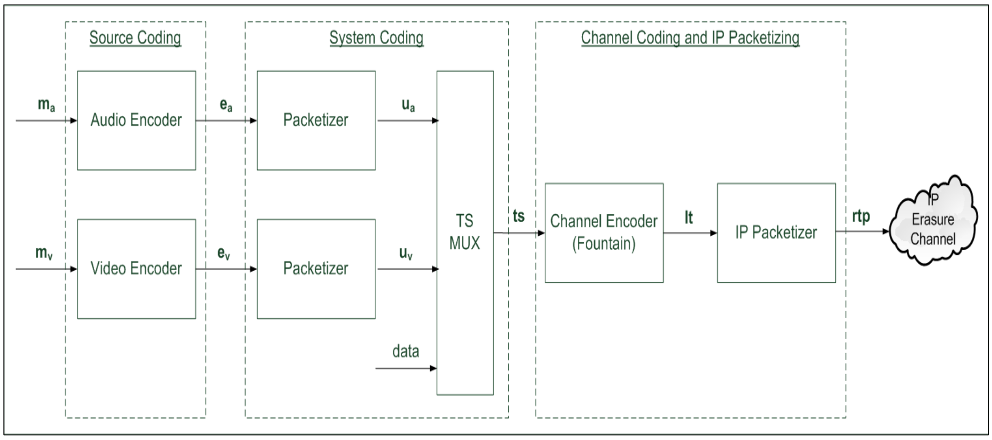

Figure II.1: Composition and packetizing of Transport Streams

Definition II.1 (Packetized Elementary Stream — PES) *The streams*

$$\mathbf{u}_A = \big(\mathbf{u}_a(0), \ldots, \mathbf{u}_a(i), \ldots, \mathbf{u}_a(L-1)\big)$$

and

$$\mathbf{u}_V = \big(\mathbf{u}_v(0), \ldots, \mathbf{u}_v(i), \ldots, \mathbf{u}_v(L-1)\big)$$

of packets of variable length derived from the streams $\mathbf{u}_v$ *and* $\mathbf{u}_a$ *are denominated Packetized Elementary Streams.*

Figure II.2 shows the TSP structure, represented by the concatenation of two main blocks: *header*, represented by $\mathbf{s}_h$ and *payload*, represented by $\mathbf{s}_p$. A third block might be present inside the payload and is denominated *Adaptation Field*. It carries relevant information concerning synchronization of the decoder and splicing [3].

The header part of the packet is 4 bytes long. In this section we are interested in the header elements which provide information concerning the quality of the incoming stream and these are listed as follows.

1. $(s_h(0), \ldots, s_h(7))$ — the first 8 bits of the header compose the *SyncByte*, which is a bit pattern equal to '47'HEX.

[3] Splicing consists of switching between different Transport Streams, creating a new stream that may be decoded with minimal artifacts around the splicing point. Splicing that results in an unbroken sequence of frames is said *seamless*, whereas splicing resulting in a broken sequence of frames is said *non-seamless*. More details on splicing can be found in [13].

2. $s_h(8)$ — *Transport Stream Error Indicator* (TEI), a single bit that may be set by a lower level protocol or a demodulator, if the TSP has uncorrectable errors.

3. $s_h(10)$ — *Transport Priority.* When this bit is set, it means that the current packet has a higher priority than other packets with the same PID.

4. $(s_h(11), \ldots, s_h(23))$ — *Packet Identifier* (PID), a 13 bits field identifying the type of information being carried by the incoming TSP. [4]

5. $\big(s_h(26), s_h(27)\big)$ — *Adaptation Field Presence.* When present, this field is carried in the payload block and provides information for synchronization of the decoder or splicing. The Adaptation Field is of high importance since its absence or corruption causes more severe impairment in the decoding of the incoming stream. We will review the significance of the information carried in this field in the next sub-sections of this document. The two-bit indicator can present the following states:

   - 01: no presence of adaptation fields, payload only;
   - 10: presence of adaptation field only;
   - 11: presence of both the adaptation field and the payload.

6. $\big(s_h(28), \ldots, s_h(31)\big)$ — *Continuity Counter*, a counter that carries the payload packet number.

Further in the TSP, in the Adaptation field, if present, the following fields are of interest:

1. $(s_a(0), \ldots, s_a(7))$ — *Adaptation field length*;

2. $s_a(10)$ — *Elementary Stream priority*, which is set when the ES being carried is of higher priority;

3. $s_a(11)$ — *Program Clock Reference (PCR)*[5] *flag*, indicating the presence (PCR) field. This field will be explained in detail in the next sub-section;

[4] All PIDs present in the stream and its associated meaning are described in Program Specific Information Tables (or PSI tables) which are also carried within the stream. Certain TSPs carry, in the Payload Field, information which are not really payload but tables and other types of control information. There are 4 types of PSI, namely, Program Association Table (PAT), Program Map Table (PMT), Conditional Access Table (CAT) and Network Information Table (NIT). Detailed explanation of these tables content can be found in [3].

[5] The OPCR is equally important when the stream is re-multiplexed. Since we are not working with re-multiplexed streams in the simulations herein, the OPCR will not be considered.

4. $(s_a(16), \ldots, s_a(58)$ — The *PCR* field;

The PCR fields are fundamental for the synchronization of the decoder and will be explained in the sub-section that follows. The TS so built is the information that will feed the erasure channel encoder, which in our case is a Fountain encoder.

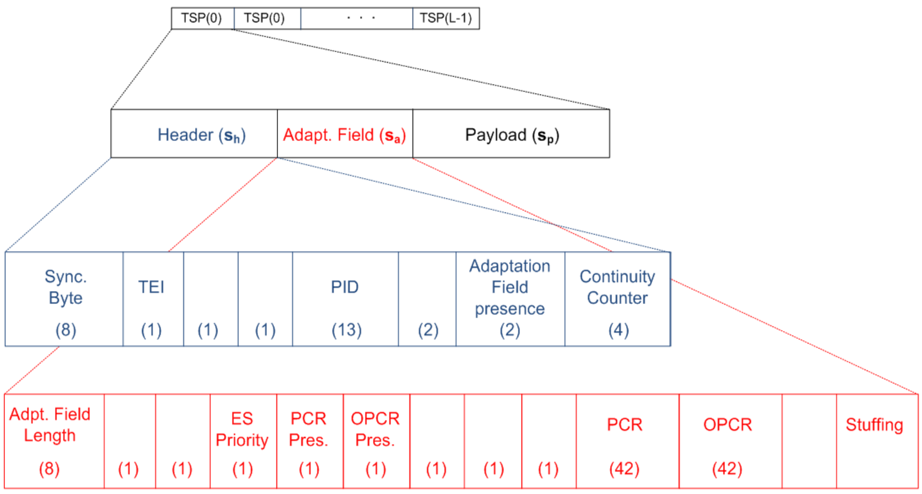

Figure II.2: Transport Stream structure

At the receiving end, the received TS is processed in order to retrieve the source A/V information. Fig.II.3 shows the receiver end with the decoding and de-multiplexing workflow. The channel encoded packet string that transverse the channel is the string that reaches the channel decoder — a Fountain decoder herein.

The output of the erasure channel decoder is the string of TSP's, which may have been corrupted by the channel, delivered to the Transport Stream de-Multiplexer. The role of the de-multiplexer is then to de-encapsulate the various PES's — or more formally, to retrieve the pair $(\mathbf{u}_A, \mathbf{u}_V)$ recovered at the receiver end. The $(\mathbf{u}_A, \mathbf{u}_V)$ pair is forwarded to the corresponding source decoders. Proper functioning of the source decoders is heavily dependent on synchronization information, which has been recovered from the received TS in the processing performed by the demultiplexer. This synchronization information is transferred to the clock control module, which interfaces with both the video and audio source decoders.

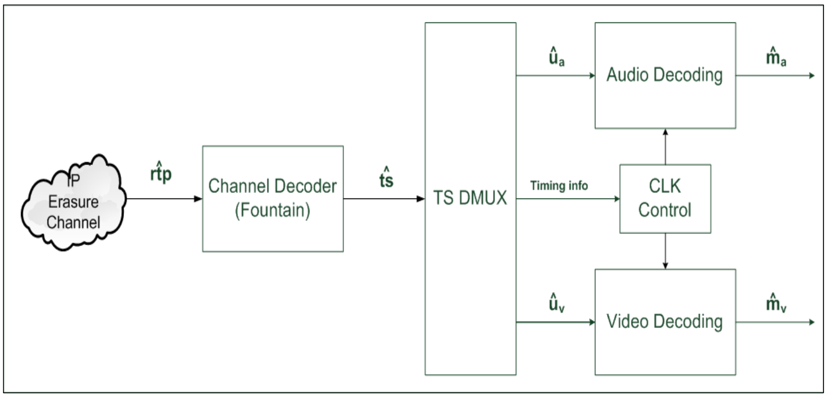

Figure II.3: Transport Stream de-multiplexing

As far as re-multiplexing capabilities concern, both the multiplexer and de-multiplexer presented provide the ability of re-multiplexing programs in the streams as demanded by the user. For instance, it is possible to select one service from an incoming MPTS and make a new SPTS out of it. It is also possible to select any subset of programs from an incoming MPTS and make a new different MPTS with these programs. Finally, it is possible to combine services from multiple streams into a new MPTS.

So far we reviewed the Transport Stream structure, its multiplexing and de-multiplexing workflows and its main fields for decoder synchronization and error indication at the receiving side. The next sub-section will present more details concerning synchronization.

II.2 Transport Stream Synchronization

Transport Streams are intended for transmission over mediums where the end-to-end delay is constant. That being said, the rate at which the signal is being output from the decoder is equal to the rate at which that same signal was input to the encoder. The Transport Stream decoder needs information present in particular positions of the incoming stream in order for it to synchronize itself and present the audio and video contents appropriately.

The end-to-end delay is the sum of the individual delays associated to each of the steps comprised in encoding, transmitting and decoding. Delays are incurred in the following steps:

- buffering in the TS multiplexer;
- transmission or storage of the Transport Stream in a media;
- buffering in the TS de-multiplexer;
- source decoding and A/V exhibition.

The constant rate constraint enforcement while decoding a given program, has to be calculated at the decoders, directly from the received stream (without any further external reference) by means of data obtained at specific fields in the stream. This aspect will be explained in the sequel.

The program being decoded is a sequence of segments with a bit rate, measured in terms of the System Clock Frequency, specific to the time interval associated to each segment. The notation System Clock Frequency is used to refer the frequency of a clock, with value $f_{scr} = 27$ MHz, constrained to a maximum allowed deviation of 810Hz and a deviation rate that has to be smaller than 0.075.

The time t_i at which the i-th byte has to enter the decoder is defined by the bits located in fields designated by *Program Clock Reference* (PCR). The bits of the PCR are located in the appropriate field within the Adaptation Field of the Transport Stream packets which carry the PCR field, referred to as PCR TSP. It should be recalled that PSI tables included in the stream identify the PCR PIDs. The specific location is indicated in figure Fig. II.2.

(a) Rate calculation at the decoder

The PCR provides sufficient information to the decoder for estimating the arrival time for each byte. This arrival time information is used by the decoder for managing the input buffer. Network jitter may cause the bytes to arrive in the wrong time intervals, not expected by the decoder, which may cause the input buffer of the decoder to underflow or overflow.

The equation

$$t(\ell) = \frac{PCR(\ell'')}{f_{scr}} + \frac{\ell - \ell''}{Trate(\ell)}. \quad (1)$$

yields $t(\ell)$, the correct time of entrance in the decoder of the ℓ-th payload byte, bearing in mind that:

- ℓ' is the index of the byte containing the last bit of the immediately following Program Clock Reference Base field (`PCRbase`), with value PCR_b, applicable to the program being decoded;

- ℓ'' is the index of the byte containing the last bit of the most recent `PCRbase` field applicable to the program being decoded;
- ℓ, $\ell'' \leq \ell \leq \ell'$, is the index of any byte in the TS;
- $PCR(\ell'')$ is the time obtained from the values stored on the fields known as `PCRbase` and `PCRext`.

The time interval between two PCR (two time stamps) cannot exceed 40ms. The PCR is used to reconstruct the system clock employed in the encoder at the decoder. The values of two consecutive PCRs, namely $PCR(\ell)$ and $PCR(\ell+1)$, do provide the means for the decoder to calculate the Transport Rate $Trate(\ell)$ associated to a particular segment (the ℓ-th segment) of the program:

$$Trate(\ell) = \frac{(\ell' - \ell'')f_{scr}}{2} PCR_(\ell') - PCR_(\ell'') \tag{2}$$

(b) Transport Stream Measurements

The set of Transport Stream measurements of the Standard presented in [23], simply known as *ETR101290*, is widely adopted by measurement and probing equipment manufacturers. The standard also defines measurements at RF levels, but these are not of our interest for the present work. The parameters specified provide information about the quality and decodability of the incoming TS. In [6] and [26] practical PCR measurement guides are presented. The parameters concerning Transport Streams presented by the *Advanced Television Systems Committee, Inc.* in [2] are also useful for the purpose herein.

The *ETR101290* groups the Transport Stream related measurements in three levels of priority:

Priority 1 comprises a list of parameters considered indispensable for proper decoding;

Priority 2 comprises the parameters considered recommended for periodic monitoring;

Priority 3 comprises parameters which monitoring might be of interest depending on the application.

The first two alarms of our interest are listed as Priority 1 alarms and denominated TS_sync_loss and TS_sync_error. Both are associated to the presence of the Sync Byte in the header of the TSP. The Sync Byte was defined in the previous section about Transport Stream structure. If two or more consecutive Sync Bytes are corrupted, a *TS_sync_loss* alarm is declared by the Transport Stream analyzer. If only one Sync Byte is corrupted or missing, a *TS_sync_error* is declared. The Sync Byte measurement is critical, since this field is necessary for the decoder to "lock" itself to the incoming stream and start decoding the packets sequence. According to the system standard presented in [3], five consecutive Sync Bytes should be enough for the decoder to synchronize itself to the incoming stream.

The next alarm considered herein is denominated *Continuity Error* and is also classified as Priority 1. This parameter monitors the incrementing of the Continuity Counter, explained in the previous sub-section. It is expected to increment by one as a new packet for a given PID reaches the decoder and loops from the value 'one' through 'fifteen'. The Continuity Counter alarm is useful in our case because it indicates the absence of a particular TSP, what will happen in the event of an IP packet drop. The Continuity Error also indicates that packets are arriving out of order, which may not be handled by the buffer in the decoder. Finally, the Continuity Error identifies packets that are transmitted duplicated, indicating a problem at the encoding side.