EUROPEAN TELECOMMUNICATION STANDARD

ETS 300 468

October 1995

Source: EBU/ETSI JTC

Reference: DE/JTC-DVB-5

ICS: 33.020

Key words: digital television broadcasting, Service Information (SI), MPEG

European Broadcasting Union EBU UER Union Européenne de Radio-Télévision

Digital broadcasting systems for television, sound and data services; Specification for Service Information (SI) in Digital Video Broadcasting (DVB) systems

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 92 94 42 00 - Fax: +33 93 65 47 16

Contents

Foreword

This European Telecommunication Standard (ETS) was produced under the authority of the Joint Technical Committee (JTC) of the European Broadcasting Union (EBU) and the European Telecommunications Standards Institute (ETSI).

This ETS for Service Information (SI) in Digital Video Broadcasting (DVB) systems has been produced by Project Team 55V using as a basis, the DVB Steering Board approved specification TM1217 Rev 2.

NOTE: The EBU/ETSI JTC was established in 1990 to co-ordinate the drafting of ETSs in the specific field of radio, television and data broadcasting. The EBU is a professional association of broadcasting organisations whose work includes the co-ordination of its Members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has Active Members in about 60 countries in the European Broadcasting Area; its headquarters is in Geneva *.

* European Broadcasting Union
Case Postale 67
CH-1218 GRAND SACONNEX (Geneva)
Switzerland

Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

Transposition dates	
Date of adoption of this ETS:	1 September 1995
Date of latest announcement of this ETS (doa):	31 January 1996
Date of latest publication of new National Standard or endorsement of this ETS (dop/e):	31 July 1996
Date of withdrawal of any conflicting National Standard (dow):	31 July 1996

Blank page

1 Scope

This European Telecommunication Standard (ETS) specifies the Service Information (SI) data which forms a part of DVB bitstreams, in order that the user can be provided with information to assist in selection of services and/or events within the bitstream, and so that the Integrated Receiver Decoder (IRD) can automatically configure itself for the selected service. SI data for automatic configuration is mostly specified within ISO/IEC 13818-1 [1] as Program Specific Information (PSI). The ETS specifies additional data which complements the PSI by providing data to aid automatic tuning of IRDs, and additional information intended for display to the user. The manner of presentation of the information is not specified in this ETS, and IRD manufacturers have freedom to choose appropriate presentation methods.

It is expected that Electronic Programme Guides (EPGs) will be a feature of Digital TV transmissions. The definition of an EPG is outside the scope of the SI specification, but the data contained within the SI specified here may be used as the basis for an EPG.

2 Normative references

This ETS incorporates by dated and undated reference, provisions from other publications. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this ETS only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies.

[1] ISO/IEC 13818-1 (1994): "Information Technology - Generic Coding of Moving Pictures and Associated Audio Recommendation H.222.0 (systems)".

[2] ISO 3166: "Codes for the representation of names of countries".

[3] ISO 639: "Code for the representation of names of languages".

[4] EBU SPB 492 (1992): "Teletext specification (625 line Television Systems)".

[5] ISO 8859: "Information processing - 8-bit single-byte coded graphic character sets, Latin alphabets".

[6] ETR 162: "Digital broadcasting systems for television, sound and data services; Allocation of Service Information (SI) codes for Digital Video Broadcasting (DVB) systems".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this ETS, the following definitions apply:

bouquet: A collection of services marketed as a single entity.

broadcaster (service provider): An organisation which assembles a sequence of events or programmes to be delivered to the viewer based upon a schedule.

component (elementary stream): One or more entities which together make up an event, e.g. video, audio, teletext.

Conditional Access (CA) system: A system to control subscriber access to services, programmes and events e.g. Videoguard, Eurocrypt.

delivery system: The physical medium by which one or more multiplexes are transmitted e.g. satellite transponder, wide-band coaxial cable, fibre optics.

Entitlement Management Messages (EMM): Are private Conditional Access information which specify the authorization levels or the services of specific decoders. They may be addressed to individual decoder or groups of decoders.

event: A grouping of elementary broadcast data streams with a defined start and end time belonging to a common service, e.g. first half of a football match, News Flash, first part of an entertainment show.

forbidden: The term "forbidden" when used in the clauses defining the coded bit stream, indicates that the value shall never be used.

MPEG-2: Refers to the standard ISO/IEC 13818 [1]. Systems coding is defined in part 1. Video coding is defined in part 2. Audio coding is defined in part 3.

multiplex: A stream of all the digital data carrying one or more services within a single physical channel.

network: A collection of MPEG-2 Transport Stream multiplexes transmitted on a single delivery system, e.g. all digital channels on a specific cable system.

original_network_id: A unique identifier of a network.

programme: A concatenation of one or more events under the control of a broadcaster e.g. news show, entertainment show.

reserved: The term "reserved" when used in the clause defining the coded bit stream, indicates that the value may be used in the future for ISO defined extensions. Unless otherwise specified within this ETS all "reserved" bits shall be set to "1".

reserved_future_use: The term "reserved_future_use", when used in the clause defining the coded bit stream, indicates that the value may be used in the future for ETSI defined extensions. Unless otherwise specified within this ETS all "reserved_future_use" bits shall be set to "1".

section: A section is a syntactic structure used for mapping all service information defined in this ETS into ISO/IEC 13818 [1] Transport Stream packets.

service: A sequence of programmes under the control of a broadcaster which can be broadcast as part of a schedule.

service_id: A unique identifier of a service within a transport stream.

Service Information (SI): Digital data describing the delivery system, content and scheduling/timing of broadcast data streams etc. It includes MPEG-2 PSI together with independently defined extensions.

sub_table: A sub_table is collection of sections with the same value of table_id and:

for a NIT:	the same table_id_extension (network_id) and version_number;
for a BAT:	the same table_id_extension (bouquet_id) and version_number;
for a SDT:	the same table_id_extension (transport_stream_id), the same original_network_id and version_number;
for a EIT:	the same table_id_extension (service_id), the same transport_stream_id, the same original_network_id and version_number

The table_id_extension field is equivalent to the fourth and fifth byte of a section when the section_syntax_indicator is set to a value of "1".

table: A table is comprised of a number of sub_tables with the same value of table_id.

Transport Stream: A Transport Stream is a data structure defined in ISO/IEC 13818-1 [1]. It is the basis of the ETSI Digital Video Broadcasting (DVB) standards.

transport_stream_id: A unique identifier of a transport stream within an original network.

The relationships of some of these definitions are illustrated in the service delivery model in figure 1.

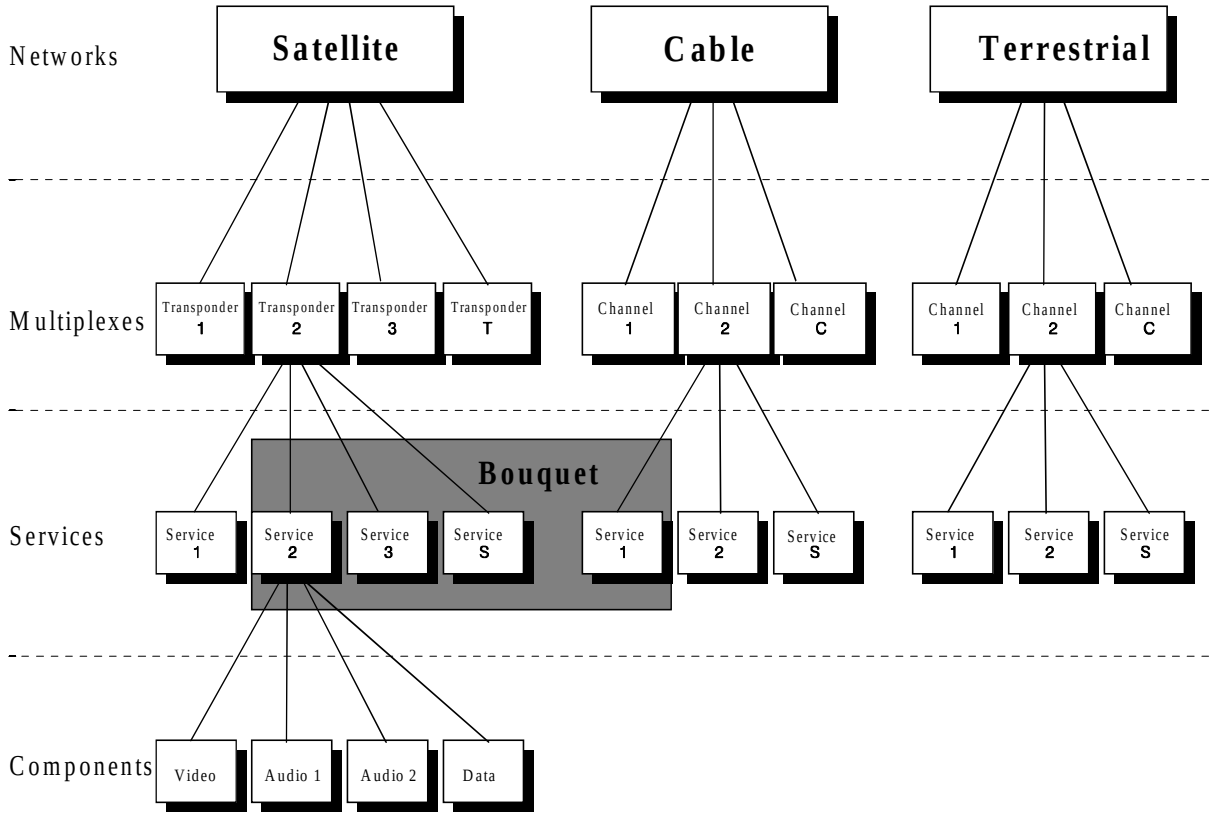


Figure 1: Digital broadcasting, service delivery model

3.2 Abbreviations

For the purposes of this ETS, the following abbreviations apply:

BAT	Bouquet Association Table
BCD	Binary Coded Decimal
CA	Conditional Access
CAT	Conditional Access Table
CRC	Cyclic Redundancy Check
DVB	Digital Video Broadcasting
EBU	European Broadcasting Union
EIT	Event Information Table
EMM	Entitlement Management Message
EPG	Electronic Programme Guide
ETS	European Telecommunication Standard
ETSI	European Telecommunications Standards Institute
FEC	Forward Error Correction
IEC	International Electrotechnical Commission
IRD	Integrated Receiver Decoder
ISO	International Organisation for Standardisation
JTC	Joint Technical Committee
LSB	Least Significant Bit
MJD	Modified Julian Date
MPEG	Moving Pictures Expert Group
NIT	Network Information Table
NVOD	Near Video On Demand
PAT	Program Association Table
PID	Packet IDentifier
PMT	Program Map Table
PSI	Program Specific Information
PSTN	Public Switched Telephone Network

QAM	Quadrature Amplitude Modulation
QPSK	Quaternary Phase Shift Keying
RS	Reed Solomon
RST	Running Status Table
SDT	Service Description Table
SI	Service Information
ST	Stuffing Table
TDT	Time and Date Table
UTC	Universal Time, Co-ordinated

bslbf	bit string, left bit first
rpchof	remainder polynomial coefficients, highest order first
uimsbf	unsigned integer most significant bit first

4 Service Information (SI) description

ISO/IEC 13818 [1] specifies SI which is referred to as PSI. The PSI data provides information to enable automatic configuration of the receiver to demultiplex and decode the various streams of programs within the multiplex.

The PSI data is structured as four types of table. The tables are transmitted in sections.

1) Program Association Table (PAT):

- for each service in the multiplex, the PAT indicates the location (the PID values of the Transport Stream packets) of the corresponding Program Map Table (PMT). It also gives the location of the Network Information Table (NIT).

2) Conditional Access Table (CAT):

- the CAT provides information on the Conditional Access (CA) systems used in the multiplex; the information is private (not defined within this ETS) and dependent on the CA system, but includes the location of the EMM stream, when applicable.

3) Program Map Table (PMT):

- the PMT identifies and indicates the locations of the streams that make up each service, and the location of the Program Clock Reference fields for a service.

4) Network Information Table (NIT):

- the location of the NIT is defined in this ETS in compliance with ISO/IEC 13818-1 [1] specification, but the data format is outside the scope of ISO/IEC 13818-1 [1]. It is intended to provide information about the physical network. The syntax and semantics of the NIT are defined in this ETS.

In addition to the PSI, data is needed to provide identification of services and events for the user. The coding of this data is defined in this ETS. In contrast with the PAT, CAT, and PMT of the PSI, which give information only for the multiplex in which they are contained (the actual multiplex), the additional information defined within this ETS can also provide information on services and events carried by different multiplexes, and even on other networks. This data is structured as six tables:

1) Bouquet Association Table (BAT):

- the BAT provides information regarding bouquets. As well as giving the name of the bouquet, it provides a list of services for each bouquet.

2) Service Description Table (SDT):

- the SDT contains data describing the services in the system e.g. names of services, the service provider, etc..

3) Event Information Table (EIT):

- the EIT contains data concerning events or programmes such as event name, start time, duration, etc.;

- the use of different descriptors allows the transmission of different kinds of event information e.g. for different service types.

4) Running Status Table (RST):

- the RST gives the status of an event (running/not running). The RST updates this information and allows timely automatic switching to events.

5) Time and Date Table (TDT):

- the TDT gives information relating to the present time and date. This information is given in a separate table due to the frequent updating of this information.

6) Stuffing Table (ST):

the ST is used to invalidate existing sections, for example at delivery system boundaries.

Where applicable the use of descriptors allows a flexible approach to the organisation of the tables and allows for future compatible extensions.

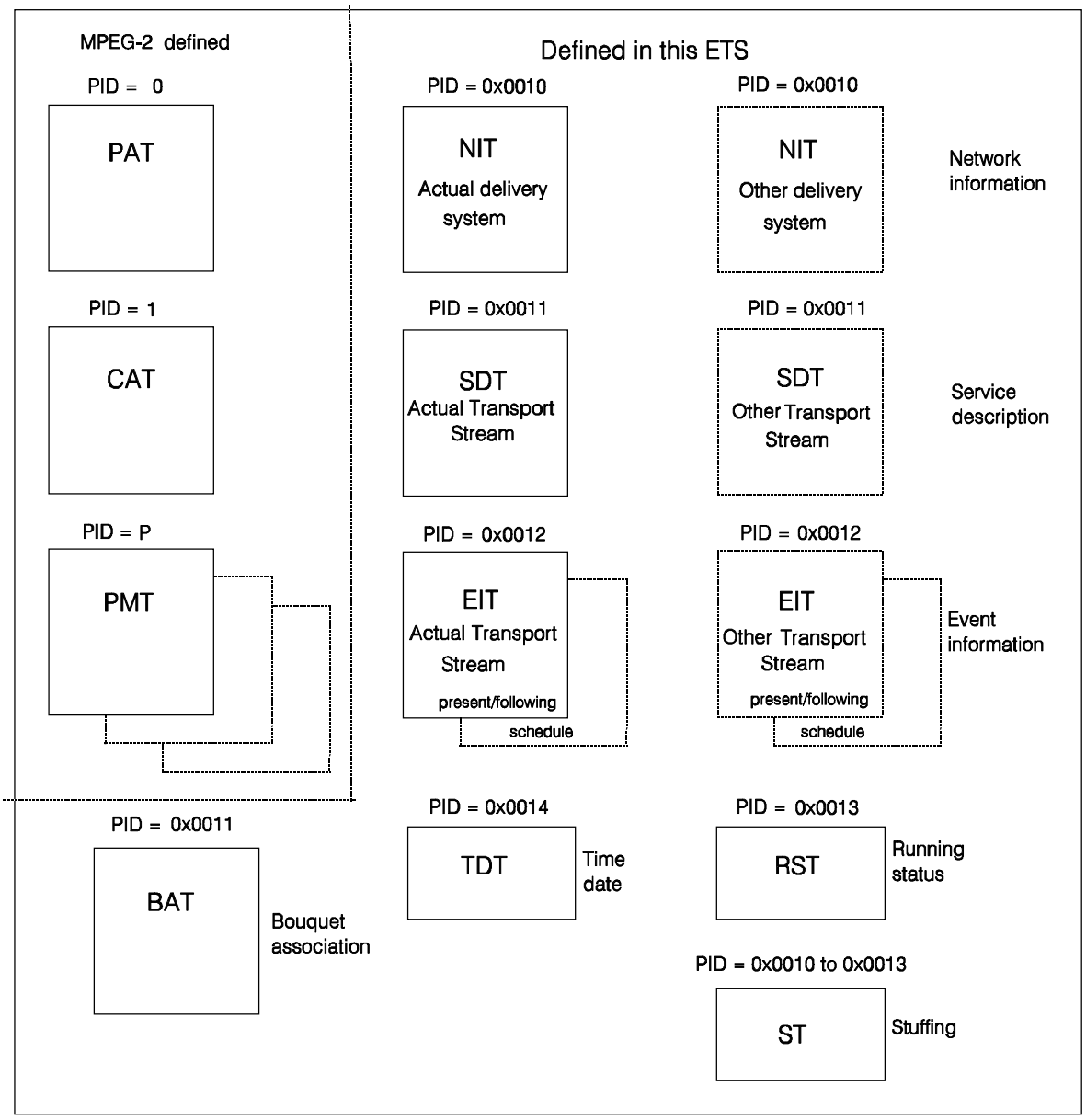


Figure 2: General organisation of the SI

5 The SI tables

5.1 SI table mechanism

The SI specified in this ETS and MPEG-2 PSI tables shall be segmented into one or more sections before being inserted into Transport Stream packets. The tables listed in clause 4 are conceptual in that they need never be regenerated in a specified form within an IRD. The tables, when transmitted shall not be scrambled, with the exception of the EIT, which may be scrambled if required (see subclause 5.1.5). A section is a syntactic structure that shall be used for mapping all MPEG-2 tables and SI tables specified in this ETS, into Transport Stream packets. These SI syntactic structures conform to the private section syntax defined in ISO/IEC 13818-1 [1].

5.1.1 Explanation

Sections may be variable in length. The sections within each table are limited to 1 024 bytes in length, except for sections within the EIT which are limited to 4 096 bytes. Each section is uniquely identified by the combination of the following elements:

a) table_id:
The table_id identifies to which table the section belongs.

Some table_ids have been defined by ISO and others by ETSI. Other values of the table_id can be allocated by the user for private purposes. The list of values of table_id is contained in table 2.

b) table_id_extension:
The table_id_extension is used for identification of a sub_table.

The interpretation of each sub_table is given in subclause 5.2.

c) section_number:
The section_number field allows the sections of a particular sub_table to be reassembled in their original order by the decoder. It is recommended, that sections are transmitted in numerical order, unless it is desired to transmit some sections of the sub_table more frequently than others, e.g. due to random access considerations.

For the SI tables as specified in this ETS, section numbering applies to sub_tables.

d) version_number:
When the characteristics of the Transport Stream described in the SI given in this ETS change (e.g. new events start, different composition of elementary streams for a given service), then new SI data shall be sent containing the updated information. A new version of the SI data is signalled by sending a sub_table with the same identifiers as the previous sub_table containing the relevant data, but with the next value of version_number.

For the SI tables specified in this ETS, the version_number applies to all sections of a sub_table.

e) Current_next_indicator:
Each section shall be numbered as valid "now" (current), or as valid in the immediate future (next). This allows the transmission of a future version of the SI in advance of the change, giving the decoder the opportunity to prepare for the change. There is however, no requirement to transmit the next version of a section in advance, but if it is transmitted, then it shall be the next correct version of that section.

5.1.2 Mapping of sections into Transport Stream packets

Sections shall be mapped directly into Transport Stream packets. Sections may start at the beginning of the payload of a Transport Stream packet, but this is not a requirement, because the start of the first section in the payload of a Transport Stream packet is pointed to by the pointer_field. There is never more than one pointer_field in a Transport Stream packet, as the start of any other section can be identified by counting the length of the first and any subsequent sections, since no gaps between sections within a Transport Stream packet are allowed by the syntax.

Within Transport Stream packets of any single PID value, one section is finished before the next one is allowed to be started, or else it is not possible to identify to which section header the data belongs. If a section finishes before the end of a Transport Stream packet, but it is not convenient to open another section, a stuffing mechanism may be used to fill up the space.

Stuffing may be performed by filling each remaining byte of the Transport Stream packet with the value "0xFF". Consequently the value "0xFF" shall not be used for the table_id. If the byte immediately following the last byte of a section takes the value of "0xFF", then the rest of the Transport Stream packet shall be stuffed with "0xFF" bytes. These bytes may be discarded by a decoder. Stuffing may also be performed using the adaptation_field mechanism.

For a more detailed description of the mechanism and functionality, refer to ISO/IEC 13818-1 [1], specifically section 2.4.4 and Annex C.

5.1.3 Coding of PID and table_id fields

Table 1 lists the PID values which shall be used for the Transport Stream packets which carry SI sections.

Table 1: PID allocation for SI

Table	PID value
PAT	0x0000
CAT	0x0001
reserved	0x0002 to 0x000F
NIT, ST	0x0010
SDT, BAT, ST	0x0011
EIT, ST	0x0012
RST, ST	0x0013
TDT	0x0014
reserved for future use	0x0015 to 0x001F

Table 2 lists the values which shall be used for table_id for the service information, defined in this ETS.

Table 2: Allocation of table_id values

Value	Description
0x00	program_association_section
0x01	conditional_access_section
0x02	program_map_section
0x03 to 0x3F	reserved
0x40	network_information_section - actual_network
0x41	network_information_section - other_network
0x42	service_description_section - actual_transport_stream
0x43 to 0x45	reserved for future use
0x46	service_description_section - other_transport_stream
0x47 to 0x49	reserved for future use
0x4A	bouquet_association_section
0x4B to 0x4D	reserved for future use
0x4E	event_information_section - actual_transport_stream, present/following
0x4F	event_information_section - other_transport_stream, present/following
0x50 to 0x5F	event_information_section - actual_transport_stream, schedule
0x60 to 0x6F	event_information_section - other_transport_stream, schedule
0x70	time_date_section
0x71	running_status_section
0x72	stuffing_section
0x73 to 0x7F	reserved for future use
0x80 to 0xFE	user defined
0xFF	reserved

5.1.4 Repetition rates and random access

In systems where random access is a consideration, it is recommended to re-transmit SI sections specified within this ETS several times, even when changes do not occur in the configuration.

For SI specified within this ETS the minimum time interval between the arrival of the last byte of a section to the first byte of the next transmitted section with the same PID, table_id and table_id_extension and with the same or different section_number shall be 25 milliseconds. This limit applies for Transport Streams with a total data rate of up to 100 Mbit/s.

5.1.5 Scrambling

With the exception of the EIT carrying schedule information, all tables specified in this ETS shall not be scrambled. One method for scrambling the EIT schedule table is given in the Bibliography 2). If a scrambling method operating over Transport Stream packets is used, it may be necessary to use a stuffing mechanism to fill from the end of a section to the end of a packet so that any transitions between scrambled and unscrambled data occur at packet boundaries.

In order to identify the CA streams which control the descrambling of the EIT data, a scrambled EIT schedule table shall be identified in the PSI. Service_id value 0xFFFF is allocated to identifying a scrambled EIT, and the program map section for this service shall describe the EIT as a private stream and shall include one or more CA_descriptors (defined in ISO/IEC 13818-1 [1]) which give the PID values and optionally, other private data to identify the associated CA streams. Service_id value 0xFFFF shall not be used for any other service.

5.2 Table definitions

The following subclauses describe the syntax and semantics of the different types of table.

NOTE: The symbols and abbreviations, and the method of describing syntax used in this ETS are the same as those defined in sections 2.2 and 2.3 of ISO/IEC 13818-1 [1].

5.2.1 Network Information Table (NIT)

The NIT (see table 3) conveys information relating to the physical organisation of the multiplexes/Transport Streams carried via a given network, and the characteristics of the network itself. The combination of original_network_id and transport_stream_id allow each Transport Stream to be uniquely identified throughout the ETS application area. Networks are assigned individual network_id values, which serve as unique identification codes for networks. The allocation of these codes may be found in ETR 162 [6]. In the case that the NIT is transmitted on the network on which the Transport Stream was originated, the network_id and the original_network_id shall take the same value.

IRDs may be able to store the NIT information in non-volatile memory in order to minimize the access time when switching between channels ("channel hopping"). It is also possible to transmit a NIT for other networks in addition to the actual network. Differentiation between the NIT for the actual network and the NIT for other networks is achieved using different table_id values (see table 2).

The NIT shall be segmented into network_information_sections using the syntax of table 3. Any sections forming part of an NIT shall be transmitted in Transport Stream packets with a PID value of 0x0010. Any sections of the NIT which describe the actual network (that is, the network of which the Transport Stream containing the NIT is a part) shall have the table_id value 0x40 with the network_id field taking the value assigned to the actual network in ETR 162 [6]. Any sections of an NIT which refer to a network other than the actual network shall take a table_id value of 0x41 and the network_id shall take the value allocated to the other network in ETR 162 [6].

Table 3: Network information section

Syntax	No. of bits	Identifier
network_information_section(){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
network_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
reserved_future_use	4	bslbf
network_descriptors_length	12	uimsbf
for(i=0;i<N;i++){		
descriptor()		
}		
reserved_future_use	4	bslbf
transport_stream_loop_length	12	uimsbf
for(i=0;i<N;i++){		
transport_stream_id	16	uimsbf
original_network_id	16	uimsbf
reserved_future_use	4	bslbf
transport_descriptors_length	12	uimsbf
for(j=0;j<N;j++){		
descriptor()		
}		
}		
CRC_32	32	rpchof
}		

Semantics for the network information section:

table_id: see table 2.

section_syntax_indicator: the section_syntax_indicator is a 1-bit field which shall be set to "1".

section_length: this is a 12-bit field, the first two bits of which shall be "00". It specifies the number of bytes of the section, starting immediately following the section_length field and including the CRC. The section_length shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

network_id: this is a 16-bit field which serves as a label to identify the delivery system, about which the NIT informs, from any other delivery system. Allocations of the value of this field are found in ETR 162 [6].

version_number: this 5-bit field is the version number of the sub_table. The version_number shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the current_next_indicator is set to "1", then the version_number shall be that of the currently applicable sub_table defined by the table_id and network_id. When the current_next_indicator is set to "0", then the version_number shall be that of the next applicable sub_table defined by the table_id and network_id.

current_next_indicator: this 1-bit indicator, when set to "1" indicates that the sub_table is the currently applicable sub_table. When the bit is set to "0", it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

section_number: this 8-bit field gives the number of the section. The section_number of the first section in the sub_table shall be "0x00". The section_number shall be incremented by 1 with each additional section with the same table_id and network_id.

last_section_number: this 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the sub_table of which this section is part.

network_descriptors_length: this 12-bit field gives the total length in bytes of the following network descriptors.

transport_stream_loop_length: this is a 12-bit field specifying the total length in bytes of the Transport Stream loops that follow, ending immediately before the first CRC-32 byte.

transport_stream_id: this is a 16-bit field which serves as a label for identification of this Transport Stream from any other multiplex within the delivery system.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system.

transport_descriptors_length: this is a 12-bit field specifying the total length in bytes of Transport Stream descriptors that follow.

CRC_32: this is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex B after processing the entire section.

5.2.2 Bouquet Association Table (BAT)

The BAT (see table 4) provides information regarding bouquets. A bouquet is a collection of services, which may traverse the boundary of a network.

The BAT shall be segmented into bouquet_association_sections using the syntax of table 4. Any sections forming part of a BAT shall be transmitted in Transport Stream packets with a PID value of 0x0011. The sections of a BAT sub_table describing a particular bouquet shall have the bouquet_id field taking the value assigned to the bouquet described in ETR 162 [6]. All BAT sections shall take a table_id value of 0x4A.

Table 4: Bouquet association section

Syntax	No. of bits	Identifier
bouquet_association_section(){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
bouquet_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
reserved_future_use	4	bslbf
bouquet_descriptors_length	12	uimsbf
for(i=0;i<N;i++){		
descriptor()		
}		
reserved_future_use	4	bslbf
transport_stream_loop_length	12	uimsbf
for(i=0;i<N;i++){		
transport_stream_id	16	uimsbf
original_network_id	16	uimsbf
reserved_future_use	4	bslbf
transport_descriptors_length	12	uimsbf
for(j=0;j<N;j++){		
descriptor()		
}		
}		
CRC_32	32	rpchof
}		

Semantics for the bouquet association section:

table_id: see table 2.

section_syntax_indicator: the section_syntax_indicator is a 1-bit field which shall be set to "1".

section_length: this is a 12-bit field, the first two bits of which shall be "00". It specifies the number of bytes of the section, starting immediately following the section_length field and including the CRC. The section_length shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

bouquet_id: this is a 16-bit field which serves as a label to identify the bouquet. Allocations of the value of this field are found in ETR 162 [6].

version_number: this 5-bit field is the version number of the sub_table. The version_number shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the current_next_indicator is set to "1", then the version_number shall be that of the currently applicable sub_table defined by the table_id and bouquet_id. When the current_next_indicator is set to "0", then the version_number shall be that of the next applicable sub_table defined by the table_id and bouquet_id.

current_next_indicator: this 1-bit indicator, when set to "1" indicates that the sub_table is the currently applicable sub_table. When the bit is set to "0", it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

section_number: this 8-bit field gives the number of the section. The section_number of the first section in the sub_table shall be "0x00". The section_number shall be incremented by 1 with each additional section with the same table_id and bouquet_id.

last_section_number: this 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the sub_table of which this section is part.

bouquet_descriptors_length: this 12-bit field gives the total length in bytes of the following descriptors.

transport_stream_loop_length: this is a 12-bit field specifying the total length in bytes of the Transport Stream loop that follows.

transport_stream_id: this is a 16-bit field which serves as a label for identification of this Transport Stream from any other multiplex within the delivery system.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system.

transport_descriptors_length: this is a 12-bit field specifying the total length in bytes of Transport Stream descriptors that follow.

CRC_32: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex B after processing the entire private section.

5.2.3 Service Description Table (SDT)

Each sub_table of the SDT (see table 5) shall describe services that are contained within a particular Transport Stream. The services may be part of the actual Transport Stream or part of other Transport Streams, these being identified by means of the table_id (see table 2).

The SDT shall be segmented into service_description_sections using the syntax of table 5. Any sections forming part of an SDT shall be transmitted in Transport Stream packets with a PID value of 0x0011. Any sections of the SDT which describe the actual Transport Stream (that is, the Transport Stream containing the SDT) shall have the table_id value 0x42, and any sections of an SDT which refer to a Transport Stream other than the actual Transport Stream shall take a table_id value of 0x46.

Table 5: Service description section

Syntax	No. of bits	Identifier
service_description_section(){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
transport_stream_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
original_network_id	16	uimsbf
reserved_future_use	8	bslbf
for (i=0;i<N;i++){		
service_id	16	uimsbf
reserved_future_use	6	bslbf
EIT_schedule_flag	1	bslbf
EIT_present_following_flag	1	bslbf
running_status	3	uimsbf
free_CA_mode	1	bslbf
descriptors_loop_length	12	uimsbf
for (j=0;j<N;j++){		
descriptor()		
}		
}		
CRC_32	32	rpchof
}		

Semantics for the service description section:

table_id: see table 2.

section_syntax_indicator: the section_syntax_indicator is a 1-bit field which shall be set to "1".

section_length: this is a 12-bit field, the first two bits of which shall be "00". It specifies the number of bytes of the section, starting immediately following the section_length field and including the CRC. The section_length shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

transport_stream_id: this is a 16-bit field which serves as a label for identification of the Transport Stream, about which the SDT informs, from any other multiplex within the delivery system.

version_number: this 5-bit field is the version number of the sub_table. The version_number shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value "31", it wraps around to "0". When the current_next_indicator is set to "1", then the version_number shall be that of the currently applicable sub_table. When the current_next_indicator is set to "0", then the version_number shall be that of the next applicable sub_table.

current_next_indicator: this 1-bit indicator, when set to "1" indicates that the sub_table is the currently applicable sub_table. When the bit is set to "0", it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

section_number: this 8-bit field gives the number of the section. The section_number of the first section in the sub_table shall be "0x00". The section_number shall be incremented by 1 with each additional section with the same table_id, transport_stream_id, and original_network_id.

last_section_number: this 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the sub_table of which this section is part.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system.

service_id: this is a 16-bit field which serves as a label to identify this service from any other service within the Transport Stream. The service_id is the same as the program_number in the corresponding program_map_section.

EIT_schedule_flag: this is a 1-bit field which when set to "1" indicates that EIT schedule information for the service is present in the current Transport Stream (see Guidelines on Implementation of Service Information (see Bibliography) for information on maximum time interval between occurrences of an EIT schedule sub_table). If the flag is set to 0 then the EIT schedule information for the service should not be present in the Transport Stream.

EIT_present_following_flag: this is a 1-bit field which when set to "1" indicates that EIT_present_following information for the service is present in the current Transport Stream (see Guidelines on Implementation of Service Information (see Bibliography) for information on maximum time interval between occurrences of an EIT present/following sub_table). If the flag is set to 0 then the EIT present/following information for the service should not be present in the Transport Stream.

running_status: this is a 3-bit field indicating the status of the service as defined in table 6.

Table 6: SDT running_status

Value	Meaning
0	undefined
1	not running
2	starts in a few seconds (e.g. for video recording)
3	pausing
4	running
5 to 7	reserved for future use

free_CA_mode: this 1-bit field, when set to "0" indicates that all the component streams of the service are not scrambled. When set to "1" it indicates that access to one or more streams may be controlled by a CA system.

descriptors_loop_length: this 12-bit field gives the total length in bytes of the following descriptors.

CRC_32: this is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex B after processing the entire section.

5.2.4 Event Information Table (EIT)

The EIT (see table 7) provides information in chronological order regarding the events contained within each service. Four classifications of EIT have been identified, distinguishable by the use of different table_ids (see table 2):

1) actual Transport Stream, present/following event information = table_id = "0x4E";

2) other Transport Stream, present/following event information = table_id = "0x4F";

3) actual Transport Stream, event schedule information = table_id = "0x50" to "0x5F";

4) other Transport Stream, event schedule information = table_id = "0x60" to "0x6F".

The present/following table shall contain only information pertaining to the present event and the chronologically following event carried by a given service on either the actual Transport Stream or another Transport Stream, except in the case of a Near Video On Demand (NVOD) reference service where it may have more than two event descriptions. The event schedule tables for either the actual Transport Stream or other Transport Streams, contain a list of events, in the form of a schedule, namely, including events taking place at some time beyond the next event. The EIT schedule tables are optional. The event information shall be chronologically ordered.

The EIT shall be segmented into event_information_sections using the syntax of table 7. Any sections forming part of an EIT shall be transmitted in Transport Stream packets with a PID value of 0x0012.

Table 7: Event information section

Syntax	No. of Bits	Identifier
Event_Information_section(){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
service_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
transport_stream_id	16	uimsbf
original_network_id	16	uimsbf
segment_last_section_number	8	uimsbf
last_table_id	8	uimsbf
for(i=0;i<N;i++){		
event_id	16	uimsbf
start_time	40	bslbf
duration	24	uimsbf
running_status	3	uimsbf
free_CA_mode	1	bslbf
descriptors_loop_length	12	uimsbf
for(i=0;i<N;i++){		
descriptor()		
}		
}		
CRC_32	32	rpchof
}		

Semantics for the event information section:

table_id: see table 2.

section_syntax_indicator: the section_syntax_indicator is a 1-bit field which shall be set to "1".

section_length: this is a 12-bit field. It specifies the number of bytes of the section, starting immediately following the section_length field and including the CRC. The section_length shall not exceed 4 093 so that the entire section has a maximum length of 4 096 bytes.

service_id: this is a 16-bit field which serves as a label to identify this service from any other service within a Transport Stream. The service_id is the same as the program_number in the corresponding program_map_section.

version_number: this 5-bit field is the version number of the sub_table. The version_number shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the current_next_indicator is set to "1", then the version_number shall be that of the currently applicable sub_table defined by the table_id and service_id. When the current_next_indicator is set to "0", then the version_number shall be that of the next applicable sub_table defined by the table_id and service_id.

current_next_indicator: this 1-bit indicator, when set to "1" indicates that the sub_table is the currently applicable sub_table. When the bit is set to "0", it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

section_number: this 8-bit field gives the number of the section. The section_number of the first section in the sub_table shall be "0x00". The section_number shall be incremented by 1 with each additional section with the same table_id, service_id, transport_stream_id, and original_network_id. In this case, the sub_table may be structured as a number of segments. Within each segment the section_number shall increment by 1 with each additional section, but a gap in numbering is permitted between the last section of a segment and the first section of the adjacent segment.

last_section_number: this 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the sub_table of which this section is part.

transport_stream_id: this is a 16-bit field which serves as a label for identification of the Transport Stream, about which the EIT informs, from any other multiplex within the delivery system.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system.

segment_last_section_number: this 8-bit field specifies the number of the last section of this segment of the sub_table. For sub_tables which are not segmented, this field shall be set to the same value as the last_section_number field.

last_table_id: this 8-bit field identifies the last table_id used (see table 2). If only one table is used this is set to the table_id of this table. The chronological order of information is maintained across successive table_id values.

event_id: this 16-bit field contains the identification number of the described event (uniquely allocated within a service definition).

start_time: this 40-bit field contains the start time of the event in Universal Time, Co-ordinated (UTC) and Modified Julian Date (MJD) (see annex C). This field is coded as 16 bits giving the 16 LSBs of MJD followed by 24 bits coded as 6 digits in 4-bit Binary Coded Decimal (BCD). If the start time is undefined (e.g. for an event in a NVOD reference service) all bits of the field are set to "1".

EXAMPLE 1: 93/10/13 12:45:00 is coded as "0xC079124500"

duration: a 24-bit field containing the duration of the event in hours, minutes, seconds.

Format: 6 digits, 4-bit BCD = 24 bit.

EXAMPLE 2: 01:45:30 is coded as "0x014530"

r**unning_status:** this is a 3-bit field indicating the status of the event as defined in table 6.

free_CA_mode: this 1-bit field, when set to "0" indicates that all the component streams of the event are not scrambled. When set to "1" it indicates that access to one or more streams is controlled by a CA system.

descriptors_loop_length: this 12-bit field gives the total length in bytes of the following descriptors.

CRC_32: this is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex B after processing the entire private section.

5.2.5 Time and Date Table (TDT)

The TDT (see table 8) carries only the UTC-time and date information.

The TDT shall consist of a single section using the syntax of table 8. This TDT section shall be transmitted in Transport Stream packets with a PID value of 0x0014, and the table_id shall take the value 0x70.

Table 8: Time and date section

Syntax	No. of bits	Identifier
time_date_section(){		
table_id	8	uimsbf
section_syntax_indicator = 0	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
UTC_time	40	bslbf
}		

Semantics for the time and date section:

table_id: see table 2.

section_syntax_indicator: this is a one-bit indicator which shall be set to "0".

section_length: this is a 12-bit field, the first two bits of which shall be "00". It specifies the number of bytes of the section, starting immediately following the section_length field and up to the end of the section.

UTC_time: this 40-bit field contains the current time and date in UTC and MJD (see annex C). This field is coded as 16 bits giving the 16 LSBs of MJD followed by 24 bits coded as 6 digits in 4-bit BCD.

EXAMPLE: 93/10/13 12:45:00 is coded as "0xC079124500"

5.2.6 Running Status Table (RST)

The RST (see table 9) allows accurate and rapid updating of the timing status of one or more events. This may be necessary when an event starts early or late due to scheduling changes. The use of a separate table enables fast updating mechanism to be achieved.

The RST shall be segmented into running_status_sections using the syntax of table 9. Any sections forming part of an RST shall be transmitted in Transport Stream packets with a PID value of 0x0013, and the table_id shall take the value 0x71.

Table 9: Running status section

Syntax	No. of bits	Identifier
running_status_section(){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
for (i=0;i<N;i++){		
transport_stream_id	16	uimsbf
original_network_id	16	uimsbf
service_id	16	uimsbf
event_id	16	uimsbf
reserved_future_use	5	uimsbf
running_status	3	uimsbf
}		
}		

Semantics for the running status section:

table_id: see table 2.

section_syntax_indicator: this is a one-bit indicator which shall be set to "0".

section_length: this is a 12-bit field, the first two bits of which shall be "00". It specifies the number of bytes of the section, starting immediately following the section_length field and up to the end of the section. The section_length shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

transport_stream_id: this is a 16-bit field which serves as a label for identification of the Transport Stream, about which the RST informs, from any other multiplex within the delivery system.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system.

service_id: this is a 16-bit field which serves as a label to identify this service from any other service within the Transport Stream. The service_id is the same as the program_number in the corresponding program_map_section.

event_id: this 16-bit field contains the identification number of the related event.

running_status: this is a 3-bit field indicating the status of the event, as defined in table 6.

5.2.7 Stuffing Table (ST)

The purpose of this section (see table 10) is to invalidate existing sections at a delivery system boundary e.g. at a cable head-end. When one section of a sub_table is overwritten, then all the sections of that sub_table shall also be overwritten (stuffed) in order to retain the integrity of the section_number field.

Table 10: Stuffing section

Syntax	No. of bits	Identifier
stuffing_section(){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
for (i=0;i<N;i++){		
data_byte	8	uimsbf
}		
}		

Semantics for the stuffing section:

table_id: this 8-bit field shall take the value "0x72" according to table 2.

section_syntax_indicator: this 1-bit field may take either the value "1" or "0".

section_length: this is a 12-bit field. It specifies the number of bytes of the section, starting immediately following the section_length field and up to the end of the section. The section_length shall not exceed 4 093 so that the entire section has a maximum length of 4 096 bytes.

data_byte: this 8-bit field may take any value and has no meaning.

6 Descriptors

This clause describes the different descriptors that can be used within the SI (for further information refer to the document, Guidelines on Implementation of Service Information (see bibliography)).

6.1 Descriptor identification and location

Table 11 lists the descriptors defined within this ETS, giving the descriptors-tag values and the intended placement within the SI tables. This does not imply that their use in other tables is restricted.

Table 11: Possible locations of descriptors

Descriptor	Tag value	NIT	BAT	SDT	EIT
network_name_descriptor	0x40	*	-	-	-
service_list_descriptor	0x41	*	*	-	-
stuffing_descriptor	0x42	*	*	*	*
satellite_delivery_system_descriptor	0x43	*	-	-	-
cable_delivery_system_descriptor	0x44	*	-	-	-
reserved for future use	0x45	-	-	-	-
reserved for future use	0x46	-	-	-	-
bouquet_name_descriptor	0x47	-	*	*	-
service_descriptor	0x48	-	-	*	-
country_availability_descriptor	0x49	-	*	*	-
linkage_descriptor	0x4A	*	*	*	*
NVOD_reference_descriptor	0x4B	-	-	*	-
time_shifted_service_descriptor	0x4C	-	-	*	-
short_event_descriptor	0x4D	-	-	-	*
extended_event_descriptor	0x4E	-	-	-	*
time_shifted_event_descriptor	0x4F	-	-	-	*
component_descriptor	0x50	-	-	-	*
mosaic_descriptor +	0x51	-	-	-	-
stream_identifier_descriptor +	0x52	-	-	-	-
CA_identifier_descriptor	0x53	-	*	*	*
content_descriptor	0x54	-	-	-	*
parental_rating_descriptor	0x55	-	-	-	*
teletext_descriptor +	0x56	-	-	-	-
telephone_descriptor	0x57	-	-	*	*
reserved for future use	0x58 to 0x7F				
user defined	0x80 to 0xFE				
forbidden	0xFF				

* Possible location.
\+ Descriptors used in the PMT.

6.2 Descriptor coding

When the construct 'descriptor ()' appears in the sections of subclause 5.2, this indicates that zero or more of the descriptors defined within this subclause shall occur.

The following semantics apply to all the descriptors defined in this subclause.

descriptor_tag: the descriptor tag is an 8-bit field which identifies each descriptor. Those values with MPEG-2 normative meaning are described in ISO/IEC 13818-1 [1]. The values of descriptor_tag are defined in table 11.

descriptor_length: the descriptor length is an 8-bit field specifying the total number of bytes of the data portion of the descriptor following the byte defining the value of this field.

6.2.1 Bouquet name descriptor

The bouquet name descriptor provides the bouquet name in text form, see table 12.

Table 12: Bouquet name descriptor

Syntax	No. of Bits	Identifier
bouquet_name_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for(i=0;i<N;i++){		
char	8	uimsbf
}		
}		

Semantics for the bouquet name descriptor:

char: this is an 8-bit field, a sequence of which conveys the name of the bouquet about which the BAT sub_table informs. Text information is coded using the character sets and methods described in annex A.

6.2.2 CA identifier descriptor

The CA identifier descriptor (see table 13) indicates whether a particular bouquet, service, event or component is associated with a conditional access system and identifies the CA system type by means of the CA_system_id.

Table 13: CA identifier descriptor

Syntax	No. of bits	Identifier
CA_identifier_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
CA_system_id	16	uimsbf
}		
}		

Semantics for the CA identifier descriptor:

CA_system_id: this 16-bit field identifies the CA system. Allocations of the value of this field are found in ETR 162 [6].

6.2.3 Component descriptor

The component descriptor identifies the type of component stream and may be used to provide a text description of the elementary stream (see table 14).

Table 14: Component descriptor

Syntax	No. of bits	Identifier
component_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
reserved_future_use	4	bslbf
stream_content	4	uimsbf
component_type	8	uimsbf
component_tag	8	uimsbf
ISO_639_language_code	24	bslbf
for (i=0;i<N;i++){		
text_char	8	uimsbf
}		
}		

Semantics for the component descriptor:

stream_content: this 4-bit field specifies the type (video, audio, or EBU-data) of stream. The coding of this field is specified in Table 15 below.

component_type: this 8-bit field specifies the type of the video, audio or EBU-data component. The coding of this field is specified in Table 15 below.

component_tag: this 8-bit field has the same value as the component_tag field in the stream identifier descriptor (if present in the PSI program map section) for the component stream.

ISO_639_language_code: this 24-bit field identifies the language of the component (in the case of audio or EBU-data) and of the text description which may be contained in this descriptor. The ISO_639_language_code contains a 3-character code as specified by ISO 639 Part 2 [3]. Each character is coded into 8 bits according to ISO 8859-1 [5] and inserted in order into the 24-bit field.

text_char: this is an 8-bit field. A string of 'text_char' fields specifies a text description of the component stream. Text information is coded using the character sets and methods described in annex A.

Table 15: stream_content and component_type

Stream_content	Component_type	Description
0x00	0x00 to 0xFF	reserved for future use
0x01	0x00	reserved for future use
0x01	0x01	video, 4:3 aspect ratio
0x01	0x02	video, 16:9 aspect ratio with pan vectors
0x01	0x03	video, 16:9 aspect ratio without pan vectors
0x01	0x04	video, > 16:9 aspect ratio
0x01	0x05 to 0xFF	reserved for future use
0x02	0x00	reserved for future use
0x02	0x01	audio, single mono channel
0x02	0x02	audio, dual mono channel
0x02	0x03	audio, stereo (2 channel)
0x02	0x04	audio, multi-lingual, multi-channel
0x02	0x05	audio, surround sound
0x02	0x06 to 0x3F	reserved for future use
0x02	0x40	audio description for the visually impaired
0x02	0x41	audio for the hard of hearing
0x02	0x42 to 0xAF	reserved for future use
0x02	0xB0 to 0xFE	user-defined
0x02	0xFF	reserved for future use
0x03	0x00	reserved for future use
0x03	0x01	EBU Teletext subtitles
0x03	0x02	associated EBU Teletext
0x03	0x03 to 0xFF	reserved for future use
0x04 to 0x0F	0x00 to 0xFF	reserved for future use

6.2.4 Content descriptor

The intention of the content descriptor (see table 16) is to provide classification information for an event.

Table 16: Content descriptor

Syntax	No. of bits	Identifier
content_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++) {		
content_nibble_level_1	4	uimsbf
content_nibble_level_2	4	uimsbf
user_nibble	4	uimsbf
user_nibble	4	uimsbf
}		
}		

Semantics of the content descriptor:

content_nibble_level_1: this 4-bit field represents the first level of a content identifier. This field shall be coded according to table 17.

content_nibble_level_2: this 4-bit field represents the second level of a content identifier. This field shall be coded according to table 17.

user_nibble: this 4-bit field is defined by the broadcaster.

Table 17: Content_nibble level 1 and 2 assignments

Content_nibble_level_1	Content_nibble_level_2	Description
0x0	0x0 to 0xF	undefined content
		Movie:
0x1	0x0	movie (general)
0x1	0x1	detective/thriller
0x1	0x2	adventure/western/war
0x1	0x3	science fiction/fantasy/horror
0x1	0x4	comedy
0x1	0x5	soap/melodrama/folkloric
0x1	0x6	romance
0x1	0x7	serious/classical/religious/historical drama
0x1	0x8	adult movie
0x1	0x9 to 0xE	reserved for future use
0x1	0xF	user defined
		News/Current affairs:
0x2	0x0	news/current affairs (general)
0x2	0x1	news/weather report
0x2	0x2	news magazine
0x2	0x3	documentary
0x2	0x4	discussion/interview/debate
0x2	0x5 to 0xE	reserved for future use
0x2	0xF	user defined
		Show/Game show:
0x3	0x0	show/game show (general)
0x3	0x1	game show/quiz/contest
0x3	0x2	variety show
0x3	0x3	talk show
0x3	0x4 to 0xE	reserved for future use
0x3	0xF	user defined
		Sports:
0x4	0x0	sports (general)
0x4	0x1	special events (Olympic Games, World Cup etc.)
0x4	0x2	sports magazines
0x4	0x3	football/soccer
0x4	0x4	tennis/squash
0x4	0x5	team sports (excluding football)
0x4	0x6	athletics
0x4	0x7	motor sport
0x4	0x8	water sport
0x4	0x9	winter sports
0x4	0xA	equestrian
0x4	0xB	martial sports
0x4	0xC to 0xE	reserved for future use
0x4	0xF	user defined

(continued)

Table 17 (continued): Content_nibble level 1 and 2 assignments

Content_nibble_level_1	Content_nibble_level_2	Description
		Children's/Youth programmes:
0x5	0x0	children's/youth programmes (general)
0x5	0x1	pre-school children's programmes
0x5	0x2	entertainment programmes for 6 to14
0x5	0x3	entertainment programmes for 10 to 16
0x5	0x4	informational/educational/school programmes
0x5	0x5	cartoons/puppets
0x5	0x6 to 0xE	reserved for future use
0x5	0xF	user defined
		Music/Ballet/Dance:
0x6	0x0	music**/**ballet**/**dance (general)
0x6	0x1	rock/pop
0x6	0x2	serious music/classical music
0x6	0x3	folk/traditional music
0x6	0x4	jazz
0x6	0x5	musical**/**opera
0x6	0x6	ballet
0x6	0x7 to 0xE	reserved for future use
0x6	0xF	user defined
		Arts/Culture (without music):
0x7	0x0	arts/culture (without music, general)
0x7	0x1	performing arts
0x7	0x2	fine arts
0x7	0x3	religion
0x7	0x4	popular culture/traditional arts
0x7	0x5	literature
0x7	0x6	film/cinema
0x7	0x7	experimental film/video
0x7	0x8	broadcasting/press
0x7	0x9	new media
0x7	0xA	arts/culture magazines
0x7	0xB	fashion
0x7	0xC to 0xE	reserved for future use
0x7	0xF	user defined
		Social/ Political issues/Economics:
0x8	0x0	social/political issues/economics (general)
0x8	0x1	magazines/reports/documentary
0x8	0x2	economics/social advisory
0x8	0x3	remarkable people
0x8	0x4 to 0xE	reserved for future use
0x8	0xF	user defined

(continued)

Table 17 (concluded): Content_nibble level 1 and 2 assignments

Content_nibble_level_1	Content_nibble_level_2	Description
		Education/ Science/Factual topics:
0x9	0x0	education/science/factual topics (general)
0x9	0x1	nature/animals/environment
0x9	0x2	technology/natural sciences
0x9	0x3	medicine/physiology/psychology
0x9	0x4	foreign countries/expeditions
0x9	0x5	social/spiritual sciences
0x9	0x6	further education
0x9	0x7	languages
0x9	0x8 to 0xE	reserved for future use
0x9	0xF	
		user defined
		Leisure hobbies:
0xA	0x0	leisure hobbies (general)
0xA	0x1	tourism/travel
0xA	0x2	handicraft
0xA	0x3	motoring
0xA	0x4	fitness & health
0xA	0x5	cooking
0xA	0x6	advertisement/shopping
0xA	0x7 to 0xE	reserved for future use
0xA	0xF	user defined
0xB to 0xE	0x0 to 0xF	reserved for future use
0xF	0x0 to 0xF	user defined

6.2.5 Country availability descriptor

In order to identify various combinations of countries efficiently, the descriptor may appear twice for each service, once giving a list of countries and/or groups of countries where the service is intended to be available, and the second giving a list of countries and/or groups where it is not. The latter list overrides the former list. If only one descriptor is used, which lists countries where the service is intended to be available, then it indicates that the service is not intended to be available in any other country. If only one descriptor is used, which lists countries where the service is not intended to be available, then it indicates that the service is intended to be available in every other country. If no descriptor is used, then it is not defined for which countries the service is intended to be available. See table 18.

Table 18: Country availability descriptor

Syntax	No. of bits	Identifier
country_availability_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
country_availability_flag	1	bslbf
reserved_future_use	7	bslbf
for (i=0;i<N;i++){		
country_code	24	bslbf
}		
}		

Semantics for the country availability descriptor

c**ountry_availability_flag:** this 1-bit field indicates whether the following country codes represent the countries in which the reception of the service is intended or not. If country_availability_flag is set to "1" the following country codes specify the countries in which the reception of the service is intended. If set to "0", the following country codes specify the countries in which the reception of the service is not intended.

country_code: this 24-bit field identifies a country using the 3-character code as specified in ISO 3166 [2]. Each character is coded into 8-bits according to ISO 8859-1 [5] and inserted in order into the 24-bit field. In the case that the 3 characters represent a number in the range 900 to 999, then country_code specifies a ETSI defined group of countries. These allocations are found in ETR 162 [6].

6.2.6 Delivery system descriptors

The delivery system descriptors all have the same overall length of 13 bytes. This facilitates the interchange of these descriptors when a Transport Stream is transcoded from one delivery system to another, e.g. satellite to cable.

6.2.6.1 Cable delivery system descriptor

See table 19.

Table 19: Cable delivery system descriptor

Syntax	No. of bits	Identifier
cable_delivery_system_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
frequency	32	bslbf
reserved_future_use	12	bslbf
FEC_outer	4	bslbf
modulation	8	bslbf
symbol_rate	28	bslbf
FEC_inner	4	bslbf
}		

Semantics for cable delivery system descriptor

frequency: the frequency is a 32-bit field giving the 4-bit BCD values specifying 8 characters of the frequency value. For the cable_delivery_system_descriptor, the frequency is coded in MHz, where the decimal occurs after the fourth character (e.g. 0312.0000 MHz).

FEC_outer: the FEC_outer is a 4-bit field specifying the outer FEC scheme used according to table 20:

Table 20: Outer FEC scheme

FEC_outer bit 3210	Description
0000	not defined
0001	no outer FEC coding
0010	RS(204/188)
0011 to 1111	reserved for future use

modulation: this is an 8-bit field. It specifies the modulation scheme used on a cable delivery system according to table 21:

Table 21: Modulation scheme for cable

Modulation (hex)	Description
0x00	not defined
0x01	16 QAM
0x02	32 QAM
0x03	64 QAM
0x04	128 QAM
0x05	256 QAM
0x06 to 0xFF	reserved for future use

symbol_rate: the symbol_rate is a 28-bit field giving the 4-bit BCD values specifying 7 characters of the symbol_rate in Msymbol/s where the decimal point occurs after the third character (e.g. 027.4500).

FEC_inner: the FEC_inner is a 4-bit field specifying the inner FEC scheme used according to the following table 22:

Table 22: Inner FEC scheme

FEC_inner bit 3210	Description
0000	not defined
0001	1/2 conv. code rate
0010	2/3 conv. code rate
0011	3/4 conv. code rate
0100	5/6 conv. code rate
0101	7/8 conv. code rate
1111	No conv. coding
0110 to 1110	reserved for future use

6.2.6.2 Satellite delivery system descriptor

See table 23.

Table 23: Satellite delivery system descriptor

Syntax	No. of bits	Identifier
satellite_delivery_system_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
frequency	32	bslbf
orbital_position	16	bslbf
west_east_flag	1	bslbf
polarisation	2	bslbf
modulation	5	bslbf
symbol_rate	28	bslbf
FEC_inner	4	bslbf
}		

Semantics for satellite delivery system descriptor

frequency: the frequency is a 32-bit field giving the 4-bit BCD values specifying 8 characters of the frequency value. For the satellite_delivery_system_descriptor the frequency is coded in GHz, where the decimal point occurs after the third character (e.g. 011.75725 GHz).

orbital_position: the orbital_position is a 16-bit field giving the 4-bit BCD values specifying 4 characters of the orbital position in degrees where the decimal point occurs after the third character (e.g. 019.2 degrees).

west_east_flag: the west_east_flag is a 1-bit field indicating if the satellite position is in the western or eastern part of the orbit. A value "0" indicates the western position and a value "1" indicates the eastern position.

polarisation: the polarisation is a 2-bit field specifying the polarisation of the transmitted signal. The first bit defines whether the polarisation is linear or circular. See table 24.

Table 24: Polarisation

Polarisation	Description
00	linear - horizontal
01	linear - vertical
10	circular - left
11	circular - right

modulation: this is a 8-bit field. It specifies the modulation scheme used on a satellite delivery system according to table 25:

Table 25: Modulation scheme for satellite

Modulation bit 4 3210	Description
0 0000	not defined
0 0001	QPSK
0 0010 to 1 1111	reserved for future use

symbol_rate: the symbol_rate is a 28-bit field giving the 4-bit BCD values specifying 7 characters of the symbol_rate in Msymbol/s where the decimal point occurs after the third character (e.g. 027.4500).

FEC_inner: the FEC_inner is a 4-bit field specifying the inner FEC scheme used according to table 22.

6.2.7 Extended event descriptor

The extended event descriptor provides a detailed text description of an event, which may be used in addition to the short event descriptor. More than one extended event descriptor can be associated to allow information about one event greater in length than 256 bytes to be conveyed. Text information can be structured into two columns, one giving an item description field and the other the item text. A typical application for this structure is to give a cast list, where for example the item description field might be "Producer" and the item field would give the name of the producer.

Table 26: Extended event descriptor

Syntax	No. of bits	Identifier
extended_event_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
descriptor_number	4	uimsbf
last_descriptor_number	4	uimsbf
ISO_639_language_code	24	bslbf
length_of_items	8	uimsbf
for (i=0;i<N;i++){		
item_description_length	8	uimsbf
for (j=0;j<N;j++){		
item_description_char	8	uimsbf
}		
item_length	8	uimsbf
for (j=0;j<N;j++){		
item_char	8	uimsbf
}		
}		
text_length	8	uimsbf
for (i=0;i<N;i++){		
text_char	8	uimsbf
}		
}		

Semantics for the extended event descriptor:

descriptor_number: this 4-bit field gives the number of the descriptor. It is used to associate information which cannot be fitted into a single descriptor. The descriptor_number of the first extended_event_descriptor of an associated set of extended_event_descriptors shall be "0x00". The descriptor_number shall be incremented by 1 with each additional extended_event_descriptor in this section.

last_descriptor_number: this 4-bit field specifies the number of the last extended_event_descriptor (that is, the descriptor with the highest value of descriptor_number) of the associated set of descriptors of which this descriptor is part.

ISO_639_language_code: this 24-bit field identifies the language of the following text fields. The ISO_639_language_code contains a 3-character code as specified by ISO 639 Part 2 [3]. Each character is coded into 8 bits according to ISO 8859-1 [5] and inserted in order into the 24-bit field.

length_of_items: this is an 8-bit field specifying the length in bytes of the following items.

item_description_length: this 8-bit field specifies the length in bytes of the item description.

item_description_char: this is an 8-bit field. A string of 'item_description_char' fields specify the item description. Text information is coded using the character sets and methods described in annex A.

item_length: this 8-bit field specifies the length in bytes of the item text.

item_char: this is an 8-bit field. A string of 'item_char' fields specify the item text. Text information is coded using the character sets and methods described in annex A.

text_length: this 8-bit field specifies the length in bytes of the non itemised extended text.

text_char: this is an 8-bit field. A string of 'text_char' fields specify the non itemised extended text. Text information is coded using the character sets and methods described in annex A.

6.2.8 Linkage descriptor

The linkage descriptor (see table 27) identifies a service that can be presented if the consumer requests for additional information related to a specific entity described by the SI system. The location of the linkage descriptor in the syntax indicates the entity for which additional information is available. For example a linkage descriptor located within the NIT shall point to a service providing additional information on the network, a linkage descriptor in the BAT shall provide a link to a service informing about the bouquet, etc.

A CA replacement service can also be identified using the linkage descriptor. This service may be selected automatically by the IRD if the CA denies access to the specific entity described by the SI system.

Table 27: Linkage descriptor

Syntax	No. of bits	Identifier
linkage_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
transport_stream_id	16	uimsbf
original_network_id	16	uimsbf
service_id	16	bslbf
linkage_type	8	uimsbf
for (i=0;i<N;i++){		
private_data_byte	8	bslbf
}		
}		

Semantics for the linkage descriptor:

transport_stream_id: this is a 16-bit field which identifies the Transport Stream containing the information service indicated.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system of the information service indicated.

service_id: this is a 16-bit field which uniquely identifies an information service within a Transport Stream. The service_id is the same as the program_number in the corresponding program_map_section.

linkage_type: this is an 8-bit field specifying the type of linkage e.g. to information (see table 28).

Table 28: Linkage type coding

Linkage_type	Description
0x00	reserved for future use
0x01	information service
0x02	Electronic Programme Guide (EPG) service
0x03	CA replacement service
0x04 to 0x7F	reserved for future use
0x80 to 0xFE	user defined
0xFF	reserved for future use

private_data_byte: this is an 8-bit field, the value of which is privately defined.

6.2.9 Mosaic descriptor

A mosaic component is a collection of different video images to form a coded video component. The information is organized so that each specific information when displayed appears on a small area of a screen.

The mosaic descriptor gives a partitioning of a digital video component into elementary cells, the allocation of elementary cells to logical cells, and gives a link between the content of the logical cell and the corresponding information (e.g.: bouquet, service, event etc.); see table 29.

Table 29: Mosaic descriptor

Syntax	No. of bits	Identifier
mosaic_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
mosaic_entry_point	1	bslbf
number_of_horizontal_elementary_cells	3	uimsbf
reserved_future_use	1	bslbf
number_of_vertical_elementary_cells	3	uimsbf
for (i=0,i<N; i++) {		
logical_cell_id	6	uimsbf
reserved_future_use	7	bslbf
logical_cell_presentation_info	3	uimsbf
elementary_cell_field_length	8	uimsbf
for (i=0,i<elementary_cell_field_length;i++) {		
reserved_future_use	2	bslbf
elementary_cell_id	6	uimsbf
}		
cell_linkage_info	8	uimsbf
If (cell_linkage_info ==0x01){		
bouquet_id	16	uimsbf
}		
If (cell_linkage_info ==0x02){		
original_network_id	16	uimsbf
transport_stream_id	16	uimsbf
service_id	16	uimsbf
}		
If (cell_linkage_info ==0x03){		
original_network_id	16	uimsbf
transport_stream_id	16	uimsbf
service_id	16	uimsbf
}		
If (cell_linkage_info ==0x04){		
original_network_id	16	uimsbf
transport_stream_id	16	uimsbf
service_id	16	uimsbf
event_id	16	uimsbf
}		
}		
}		

Semantics for the Mosaic Descriptor

Mosaic_entry_point: this is a 1-bit field which when set to a value of "1" indicates that the mosaic is the highest mosaic in a hierarchy. A complete mosaic system could be organized in a tree structure, the flag being set to identify the entry point in the tree.

Number_of_horizontal_elementary_cells: this 3-bit field indicates the number of cells of horizontal screen display, see table 30 for coding.

Table 30: Coding of horizontal_elementary_cells

Value	Meaning
0x00	one cell
0x01	two cells
0x02	three cells
0x03	four cells
0x04	five cells
0x05	six cells
0x06	seven cells
0x07	eight cells

Number_of_vertical_elementary_cells: this 3-bit field indicates the number of cells of vertical screen display, see table 31 for coding.

Table 31: Coding of vertical_elementary_cells

Value	Meaning
0x00	one cell
0x01	two cells
0x02	three cells
0x03	four cells
0x04	five cells
0x05	six cells
0x06	seven cells
0x07	eight cells

Logical_cell_id: this 6-bit field is coded in binary form.

Different adjacent (see figure 3) elementary cells may be grouped together to form a logical cell. A logical_cell_number is associated to such a group of adjacent elementary_cell_ids. The total number of logical cells shall not exceed the number of elementary cells (maximum = 64). Each elementary cell shall be allocated to one logical cell. More than one elementary cell may belong to one logical cell.

A	B	C
D	E	F
G	H	I

Cells B, D, H, F are adjacent to cell E; C is not adjacent to A or D; D is not adjacent to H.

Figure 3: Adjacent cells

Logical_cell_presentation_info: this 3-bit field identifies the type of presentation for a logical cell. The logical_cell_presentation information allows an identification of presentation styles, which are defined in table 32.

Table 32: Coding of logical_cell_presentation_info

Value	Meaning
0x00	undefined
0x01	video
0x02	still picture (Note)
0x03	graphics/text
0x04 to 0x07	reserved for future use

NOTE 1: Still picture: A coded still picture consists of a video sequence containing exactly one coded picture which is intra-coded.

Elementary_cell_field_length: the elementary_cell_field_length is an 8-bit field specifying the number of bytes following this field up to and including the last elementary_cell_id in this logical_cell_id loop.

Elementary_cell_id: this 6-bit field indicates in binary form the number of the cell. The value of this field is in the range 0 to N.

NOTE 2: The elementary cells are implicitly numbered from 0 to N. The value 0 is allocated to the cell of the first row (top left corner). This number is incremented from left to right and from top to bottom in such a way that the number N is allocated to the cell of the last position of the last row (bottom right corner).

Cell_linkage_info: this 8-bit field identifies the type of information carried in a logical cell, see table 33 for coding.

Table 33: Coding of cell_linkage_info

Value	Meaning
0x00	undefined
0x01	bouquet related
0x02	service related
0x03	other mosaic related
0x04	event related
0x05 to 0xFF	reserved for future use

bouquet_id: this is a 16-bit field which serves as a label to identify the bouquet described by the cell.

original_network_id: this 16-bit field is a label (see subclause 5.2) which in conjunction with the following fields uniquely identifies a service, event or mosaic.

transport_stream_id: this is a 16-bit field which serves as a label identifying the transport stream which contains the service, event or mosaic described by the cell.

service_id: this is a 16-bit field which identifies a service within a transport stream. The service_id is the same as the program_number in the corresponding program_map_section.

The interpretation of this field is context sensitive, dependent on the value of cell_linkage_info:

When cell_linkage_info = "0x02", this is the service_id of the service described by the cell.

When cell_linkage_info = "0x03", this is the service_id of the mosaic service described by the cell.

When cell_linkage_info = "0x04", this is the service_id of the service to which the event described by the cell belongs.

event_id: this is a 16-bit field containing the identification number of the described event.

6.2.10 Near Video On Demand (NVOD) reference descriptor

This descriptor, in conjunction with the time shifted service and time shifted event descriptors, provides a mechanism for efficiently describing a number of services which carry the same sequence of events, but with the start times offset from one another. Such a group of time-shifted services is referred to as Near Video On Demand, since a user can at any time access near to the start of an event by selecting the appropriate service of the group.

The NVOD reference descriptor (see table 34) gives a list of the services which together form a NVOD service. Each service is also described in the appropriate SDT sub_table by a time shifted service descriptor, see subclause 6.2.20. The time shifted service descriptor associates a time shifted service with a reference_service_id. The reference_service_id is the label under which a full description of the NVOD service is given, but the reference_service_id does not itself correspond to any program_number in the program_map_section.

The time shifted event descriptor is used in the event information for each time shifted service. Instead of duplicating the full information for each event, the time shifted event descriptor points to a reference_event_id in the reference service. The full event information is provided in the event information for the reference service.

The services which make up an NVOD service need not all be carried in the same Transport Stream. However, a reference service shall be described in the SI in each Transport Stream which carries any services of the NVOD service.

Table 34: NVOD reference descriptor

Syntax	No. of bits	Identifier
NVOD_reference_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++) {		
transport_stream_id	16	uimsbf
original_network_id	16	uimsbf
service_id	16	uimsbf
}		
}		

Semantics for the NVOD reference descriptor:

transport_stream_id: this is a 16-bit field which identifies the Transport Stream.

original_network_id: this 16-bit field gives the label identifying the network_id of the originating delivery system.

service_id: this is a 16-bit field which uniquely identifies a service within a Transport Stream. The service_id is the same as the program_number in the corresponding program_map_section.

6.2.11 Network name descriptor

The network name descriptor provides the network name in text form. See table 35.

Table 35: Network name descriptor

Syntax	No. of bits	Identifier
network_name_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
char	8	uimsbf
}		
}		

Semantics for the network name descriptor:

char: this is an 8-bit field. A string of char fields specify the name of the delivery system about which the NIT informs. Text information is coded using the character sets and methods described in annex A.

6.2.12 Parental rating descriptor

This descriptor (see table 36) gives a rating based on age and allows for extensions based on other rating criteria.

Table 36: Parental rating descriptor

Syntax	No. of bits	Identifier
parental_rating_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
country_code	24	bslbf
rating	8	uimsbf
}		
}		

Semantics for the parental rating descriptor:

country_code: this 24-bit field identifies a country using the 3-character code as specified in ISO 3166 [2]. Each character is coded into 8-bits according to ISO 8859-1 [5] and inserted in order into the 24-bit field. In the case that the 3 characters represent a number in the range 900 to 999, then country_code specifies a ETSI defined group of countries. These allocations are found in ETR 162 [6].

rating: this 8-bit field is coded according to table 37, giving the recommended minimum age in years of the end user.

Table 37: Parental rating descriptor, rating

Rating	Description
0x00	undefined
0x01 to 0x0F	minimum age = rating + 3 years
0x10 to 0xFF	defined by the broadcaster

EXAMPLE: 0x04 implies that end users should be at least 7 years old.

6.2.13 Service descriptor

The service descriptor (see table 38) provides the names of the service provider and the service in text form together with the service_type.

Table 38: Service descriptor

Syntax	No. of bits	Identifier
service_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
service_type	8	uimsbf
service_provider_length	8	uimsbf
for (i=0;i<N;i++){		
char	8	uimsbf
}		
service_name_length	8	uimsbf
for (i=0;i<N;i++){		
char	8	uimsbf
}		
}		

Semantics for the service descriptor:

service_type: this is an 8-bit field specifying the type of the service. It shall be coded according to table 39.

Table 39: Service type coding

Service_type	Description
0x00	reserved for future use
0x01	digital television service
0x02	digital radio sound service
0x03	teletext service
0x04	NVOD reference service
0x05	NVOD time-shifted service
0x06	mosaic service
0x07	PAL coded signal
0x08	SECAM coded signal
0x09	D/D2-MAC
0x0A	FM Radio
0x0B to 0x7F	reserved for future use
0x80 to 0xFE	user defined
0xFF	reserved for future use

service_provider_length: this 8-bit field specifies the number of bytes that follow the service_provider_length field for describing characters of the name of the service provider.

char: this is an 8-bit field. A string of char fields specify the name of the service provider or service. Text information is coded using the character sets and methods described in annex A.

service_name_length: this 8-bit field specifies the number of bytes that follow the service_name_length field for describing characters of the name of the service.

6.2.14 Service list descriptor

The service list descriptor (see table 40) provides a means of listing the services by service_id and service type.

Table 40: Service list descriptor

Syntax	No. of bits	Identifier
service_list_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
service_id	16	uimsbf
service_type	8	uimsbf
}		
}		

Semantics for the service list descriptor:

service_id: this is a 16-bit field which uniquely identifies a service within a Transport Stream. The service_id is the same as the program_number in the corresponding program_map_section, except that in the case of service_type = 0x04 (NVOD reference service) the service_id does not have a corresponding program_number.

service_type: this is an 8-bit field specifying the type of the service. It shall be coded according to table 39.

6.2.15 Short event descriptor

The short event descriptor provides the name of the event and a short description of the event in text form. See table 41.

Table 41: Short event descriptor

Syntax	No. of bits	Identifier
short_event_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
ISO_639_language_code	24	bslbf
event_name_length	8	uimsbf
for (i=0;i<event_name_length;i++){		
event_name_char	8	uimsbf
}		
text_length	8	uimsbf
for (i=0;i<text_length;i++){		
text_char	8	uimsbf
}		
}		

Semantics for the short event descriptor:

ISO_639_language_code: this 24-bit field contains the ISO 639 [3] three character language code of the language of the following text fields. Each character is coded into 8 bits according to ISO 8859-1 [5] and inserted in order into the 24-bit field.

event_name_length: an 8-bit field specifying the length in bytes of the event name.

event_name_char: this is an 8-bit field. A string of 'char' fields specifies the event name. Text information is coded using the character sets and methods described in annex A.

text_length: this 8-bit field specifies the length in bytes of the following text describing the event.

text_char: this is an 8-bit field. A string of 'char' fields specify the text description for the event. Text information is coded using the character sets and methods described in annex A.

6.2.16 Stream identifier descriptor

The stream identifier descriptor (see table 42) may be used in the PSI PMT to label component streams of a service so that they can be differentiated, e.g. by text descriptions given in component descriptors in the EIT if present. The stream identifier descriptor shall be located following the relevant ES_info_length field.

Table 42: Stream identifier descriptor

Syntax	No. of bits	Identifier
stream_identifier_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
component_tag	8	uimsbf
}		

Semantics for the stream identifier descriptor:

component_tag: this 8-bit field identifies the component stream for associating it with a description given in a component descriptor. Within a program map section each stream identifier descriptor shall have a different value for this field.

6.2.17 Stuffing descriptor

The stuffing descriptor provides a means of invalidating previously coded descriptors or inserting dummy descriptors for table stuffing. See table 43.

Table 43: Stuffing descriptor

Syntax	No. of bits	Identifier
stuffing_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i= 0;i<N;i++){		
stuffing_byte	8	bslbf
}		
}		

Semantics for the stuffing descriptor:

stuffing_byte: this is an 8-bit field. Each occurrence of the field may be set to any value. The IRDs may discard the stuffing bytes.

6.2.18 Telephone descriptor

The telephone descriptor may be used to indicate a telephone number. which may be used in conjunction with a modem (PSTN or cable) to exploit narrowband interactive channels. Further information is given in Implementation Guidelines for the use of Telecommunications Interfaces in Digital Video Broadcasting Systems (see bibliography).

The telephone descriptor syntax is specified in table 44.

Table 44: Telephone descriptor

Syntax	No. of bits	Identifier
telephone_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
reserved_future_use	2	bslbf
foreign_availability	1	bslbf
connection_type	5	uimsbf
reserved_future_use	1	bslbf
country_prefix_length	2	uimsbf
international_area_code_length	3	uimsbf
operator_code_length	2	uimsbf
reserved_future_use	1	bslbf
national_area_code_length	3	uimsbf
core_number_length	4	uimsbf
for (i=0;i<N;i++){		
country_prefix_char	8	uimsbf
}		
for (i=0;i<N;i++){		
international_area_code_char	8	uimsbf
}		
for (i=0;i<N;i++){		
operator_code_char	8	uimsbf
}		
for (i=0;i<N;i++){		
national_area_code_char	8	uimsbf
}		
for (i=0;i<N;i++){		
core_number_char	8	uimsbf
}		
}		

Semantics for the telephone descriptor:

foreign_availability: this is a 1-bit flag. When set to "1" it indicates that the number described can be called from outside of the country specified by the country_prefix. When set to "0" it indicates that the number can only be called from inside the country specified by the country_prefix.

connection_type: this is a 5-bit field which indicates connection types. One example of the use of the connection type is to inform the IRD that when, if an interaction is initiated, it the connection is not made within 1 minute, then the connection attempt should be aborted.

country_prefix_length: this 2-bit field specifies the number of 8-bit alphanumeric characters in the country prefix.

international_area_code_length: this 3-bit field specifies the number of 8-bit alphanumeric characters in the international area code.

operator_code_length: this 2-bit field specifies the number of 8-bit alphanumeric characters in the operator code.

national_area_code_length: this 3-bit field specifies the number of 8-bit alphanumeric characters in the national area code.

core_number_length: this 4-bit field specifies the number of 8-bit alphanumeric characters in the core number.

country_prefix_char: this 8-bit field which shall be coded in accordance with ISO 8859-1 [5] gives one alphanumeric character of the country prefix.

international_area_code_length: this 8-bit field which shall be coded in accordance with ISO 8859-1 [5] gives one alphanumeric character of the international area code.

operator_code_length: this 8-bit field which shall be coded in accordance with ISO 8859-1 [5] gives one alphanumeric character of the operator code.

national_area_code_length: this 8-bit field which shall be coded in accordance with ISO 8859-1 [5] gives one alphanumeric character of the national area code.

core_number_length: this 8-bit field which shall be coded in accordance with ISO 8859-1 [5] gives one alphanumeric character of the core number.

6.2.19 Teletext descriptor

The teletext descriptor (see table 45) shall be used in the PSI PMT to identify streams which carry EBU teletext data. The descriptor is to be located in a program map section following the relevant ES_info_length field.

Table 45: Teletext descriptor

Syntax	No. of bits	Identifier
teletext_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
ISO_639_language_code	24	bslbf
teletext_type	5	uimsbf
teletext_magazine_number	3	uimsbf
teletext_page_number	8	uimsbf
}		
}		

Semantics for the teletext descriptor:

ISO_639_language_code: this 24-bit field contains the 3 character ISO 639 [3] language code of the language of the teletext. Each character is coded into 8 bits according to ISO 8859-1 [5] and inserted in order into the 24-bit field.

teletext_type: this 5-bit field indicates the type of teletext page indicated. This shall be coded according to table 46.

Table 46: Teletext descriptor, teletext_type

Teletext_type	Description
0x00	reserved for future use
0x01	initial teletext page
0x02	teletext subtitle page
0x03	additional information page
0x04	programme schedule page
0x05 to 0x1F	reserved for future use

teletext_magazine_number: this is a 3-bit field which identifies the magazine number as defined in EBU SPB 492 [4].

teletext_page_number: this is an 8-bit field giving two 4-bit hex digits identifying the page number as defined in EBU SPB 492 [4].

6.2.20 Time shifted event descriptor

The time shifted event descriptor (see table 47) is used in place of the short_event_descriptor to indicate an event which is a time shifted copy of another event.

Table 47: Time shifted event descriptor

Syntax	No. of bits	Identifier
time_shifted_event_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
reference_service_id	16	uimsbf
reference_event_id	16	uimsbf
}		

Semantics for the time shifted event descriptor:

reference_service_id: this 16-bit field identifies the reference service of a NVOD collection of services. The reference service can always be found in this Transport Stream. The service_id here does not have a corresponding program_number in the program_map_section.

reference_event_id: this 16-bit field identifies the reference event of which the event described by this descriptor is a time shifted-copy.

6.2.21 Time shifted service descriptor

This descriptor is used in place of the service descriptor to indicate services which are time shifted copies of other services. See table 48.

Table 48: Time shifted service descriptor

Syntax	No. of bits	Identifier
time_shifted_service_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
reference_service_id	16	uimsbf
}		

Semantics for the time shifted service descriptor:

reference_service_id: this 16-bit field identifies the reference service of a NVOD collection of services. The reference service can always be found in this Transport Stream. The service_id here does not have a corresponding program_number in the program_map_section.

Annex A (normative): Coding of text characters

Text items can optionally include information to select a wide range of character tables as indicated below. For the European languages a set of five character tables are available. If no character selection information is given in a text item, then a default character set is assumed.

A.1 Control codes

The codes in the range 0x80 to 0x9F are assigned to control functions as shown in table A.1.

Table A.1: Control codes

Control code	Description
0x80 to 0x85	reserved for future use
0x86	character emphasis on
0x87	character emphasis off
0x88 to 0x89	reserved for future use
0x8A	CR/LF
0x8B to 0x9F	user defined

A.2 Selection of character table

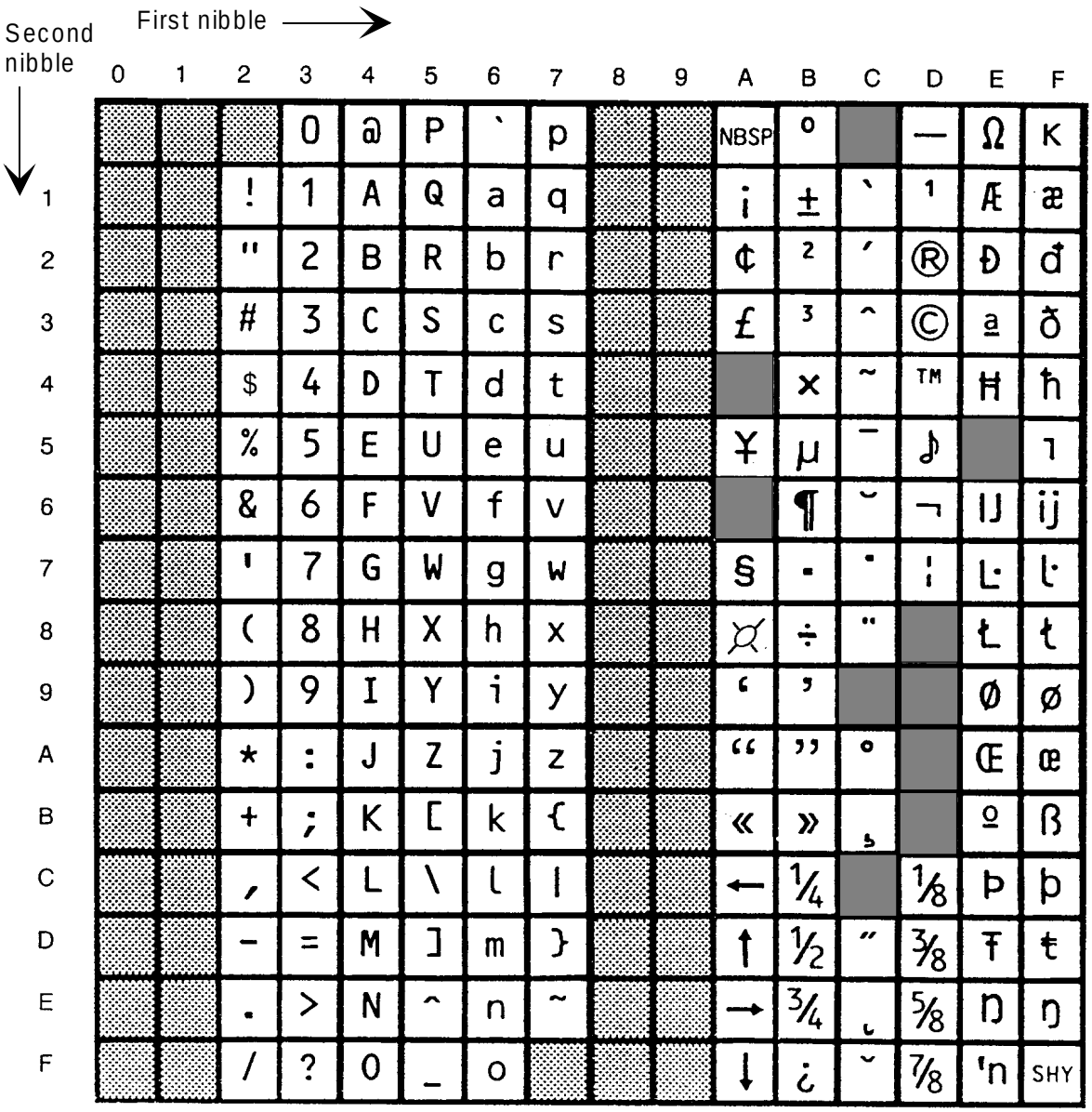
Text fields can optionally start with non-spacing, non-displayed data which specifies the alternative character table to be used for the remainder of the text item. The selection of character table is indicated as follows:

- if the first byte of the text field has a value in the range "0x20" to "0xFF" then this and all subsequent bytes in the text item are coded using the default character coding table (table 00 - Latin alphabet) of figure A.1;

- if the first byte of the text field has a value in the range "0x01" to "0x05" then the remaining bytes in the text item are coded in accordance with character coding tables 01 to 05 respectively, which are given in figures A.2 to A.6 respectively;

- if the first byte of the text field has a value "0x10" then the following two bytes carry a 16-bit value (uimsbf) N to indicate that the remaining data of the text field is coded using the character code table specified by ISO Standard 8859-1 to 9 [5].

Values for the first byte of "0x00", "0x06" to "0x0F", and "0x11" to "0x1F" are reserved for future use.



NOTE 1: The SPACE character is located in position 20h of the code table.

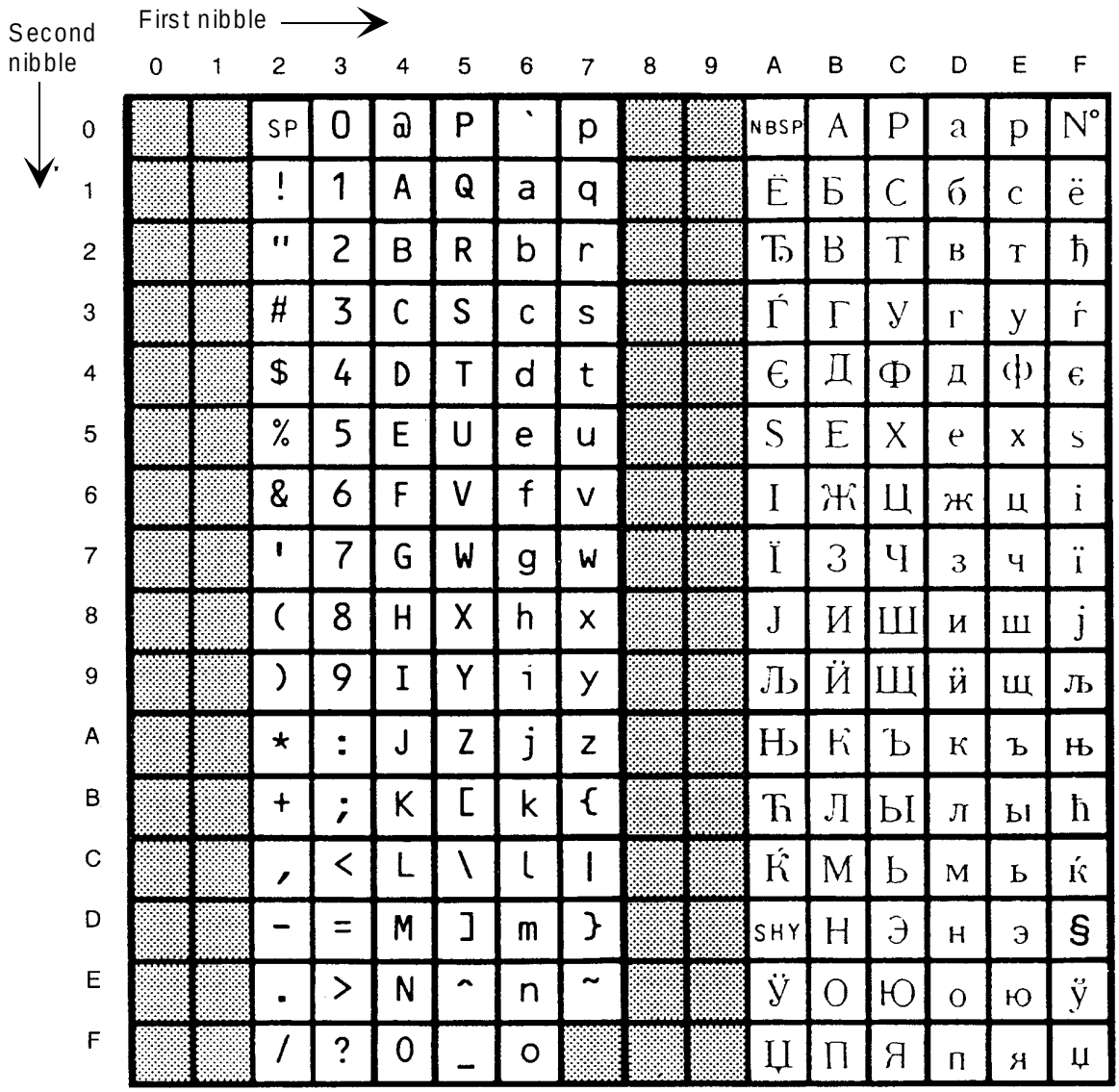
NOTE 2: NBSP = no-break space.

NOTE 3: SHY = soft hyphen.

NOTE 4: Table reproduced from ISO 6937 (1994).

NOTE 5: All characters in column C are non-spacing characters (diacritical marks).

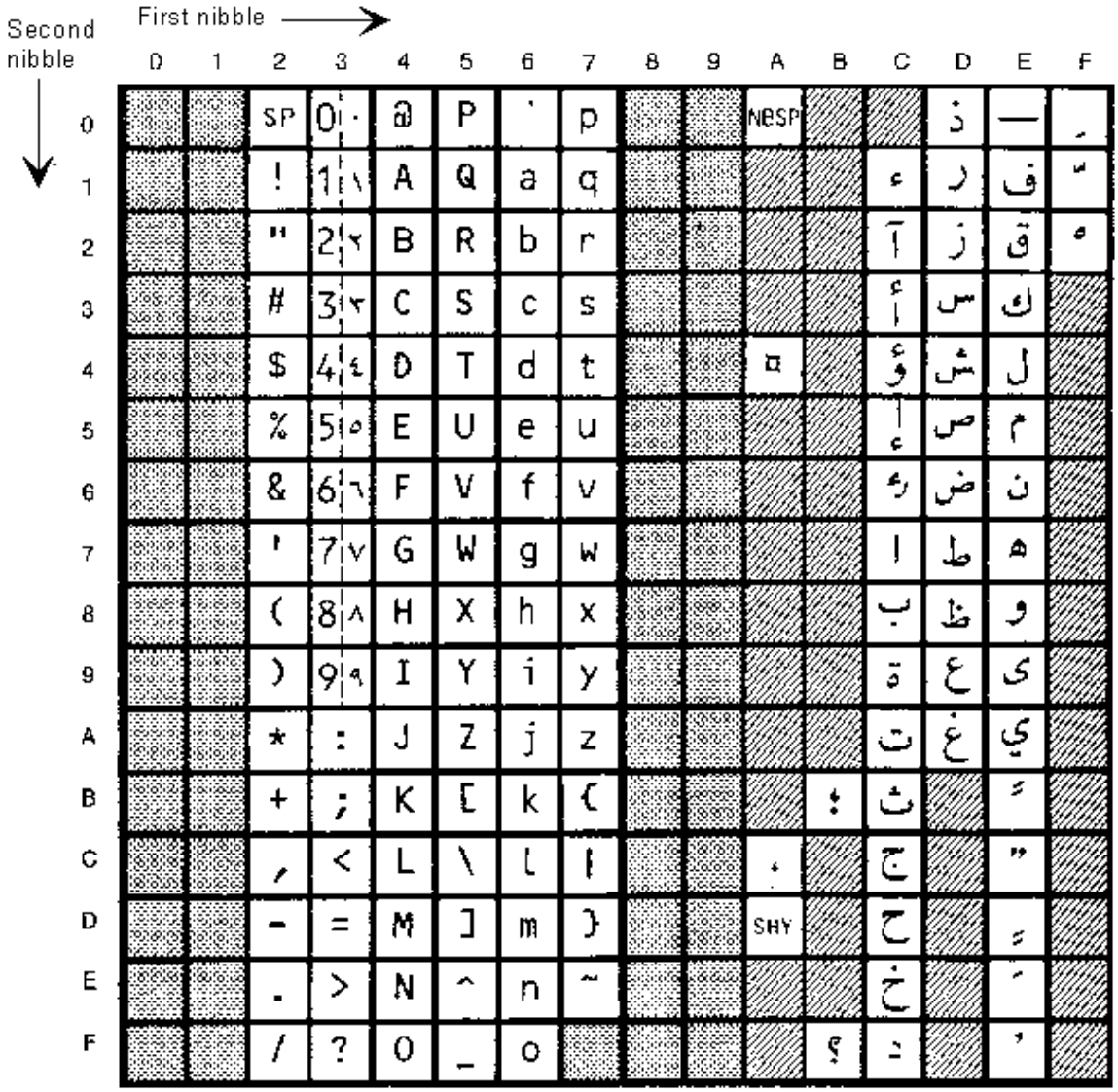
Figure A.1: Character code table 00 - Latin alphabet



NOTE 1: For the Ruthenian language, the characters in code positions Ah/5h (S) and Fh/5h (s) are replaced by Ґ and ґ, respectively.

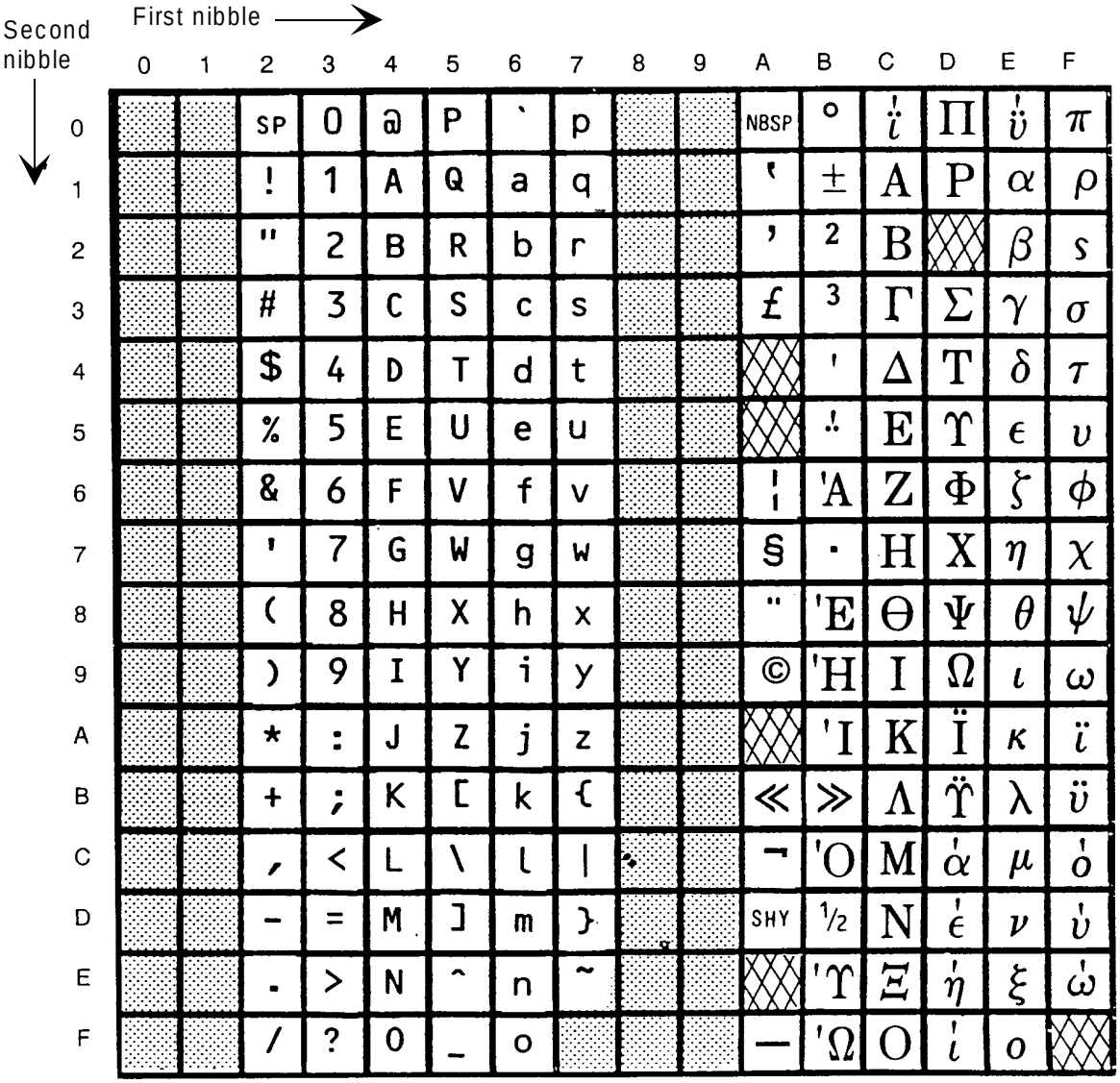
NOTE 2: Table reproduced from ISO 8859-5 (1988).

Figure A.2: Character code table 01 - Latin/Cyrillic alphabet



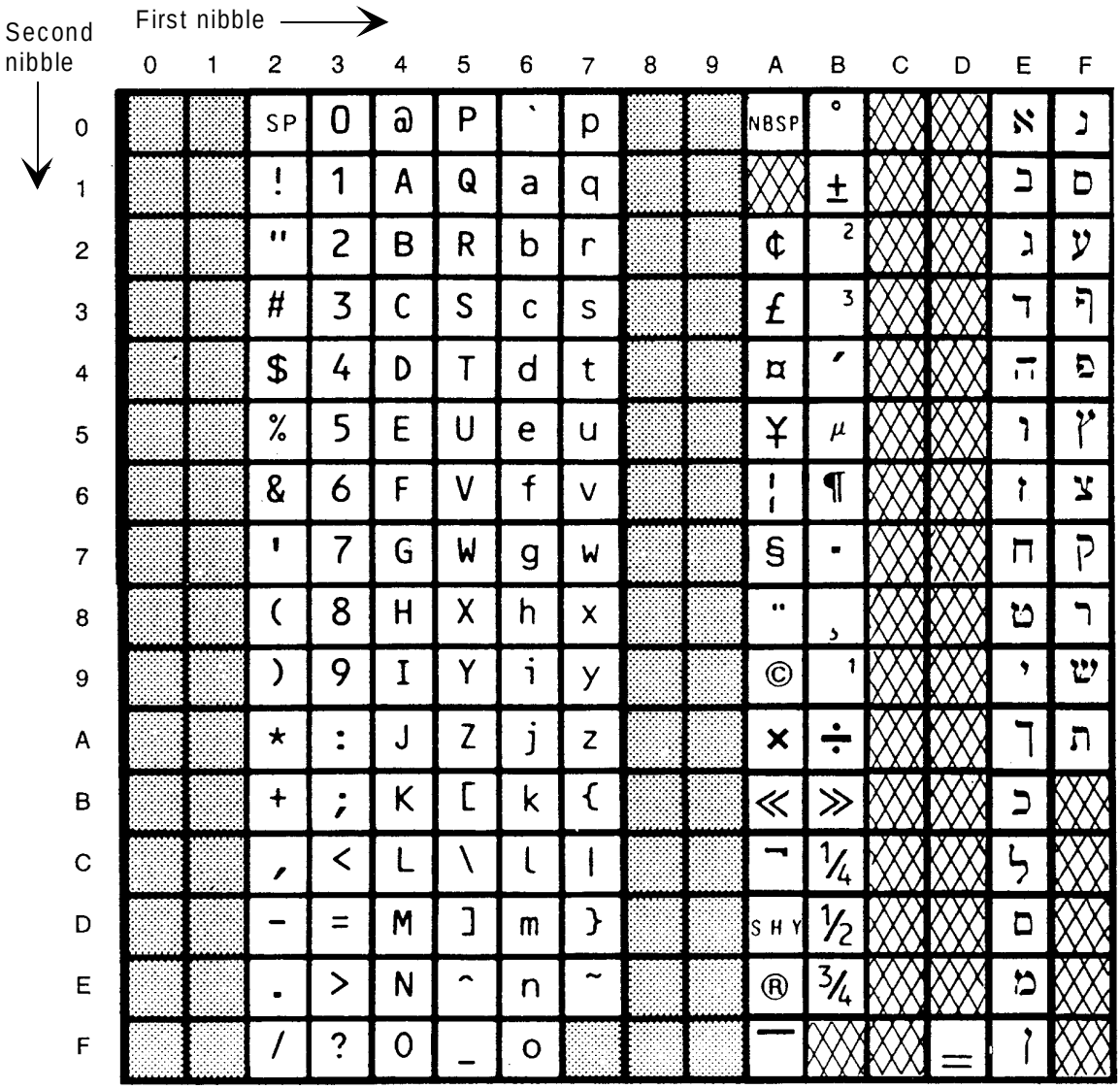
NOTE: Table reproduced from ISO 8859-6 (1987).

Figure A.3: Character code table 02 - Latin/Arabic alphabet



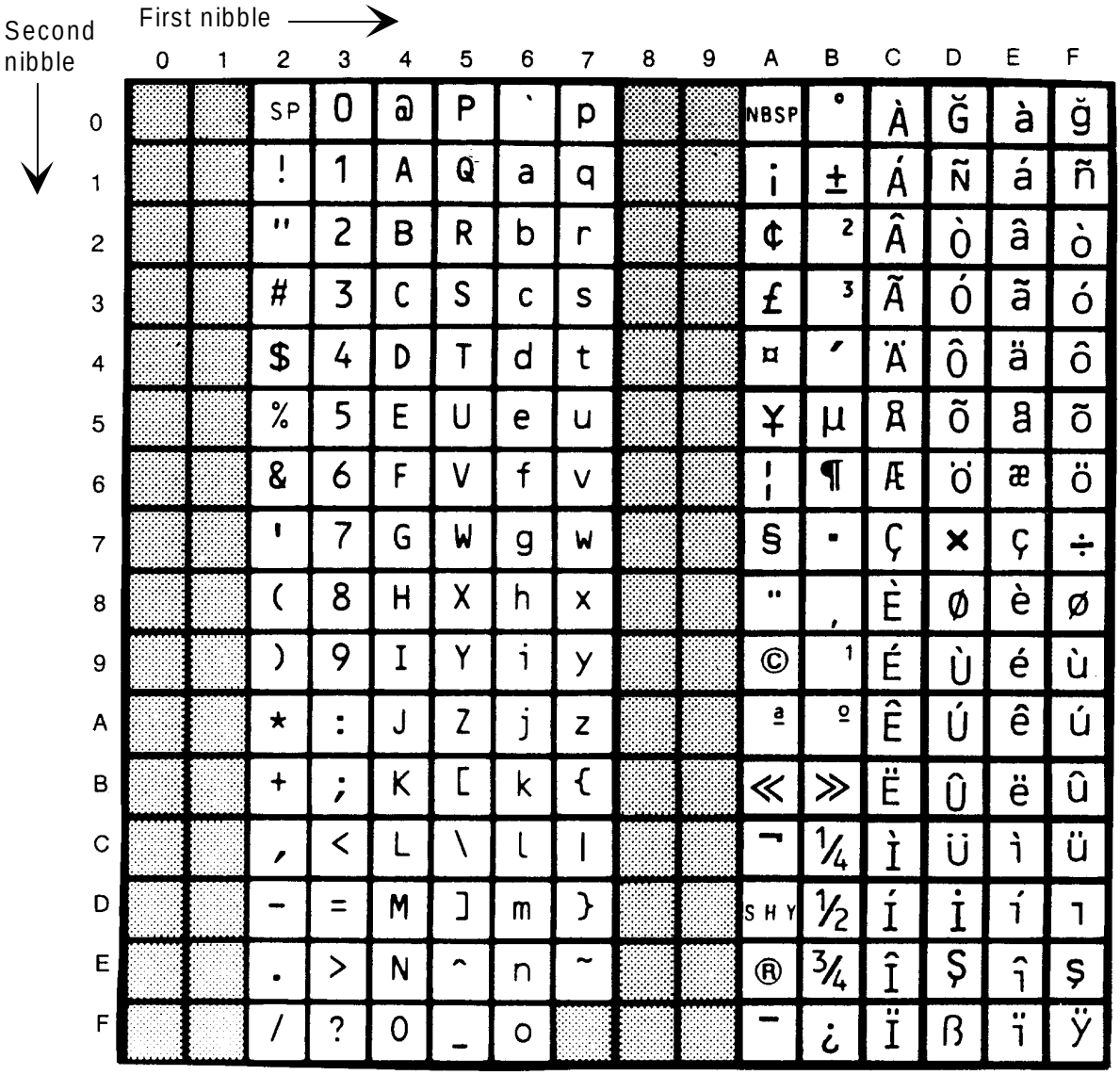
NOTE: Table reproduced from ISO 8859-7 (1987).

Figure A.4: Character code table 03 - Latin/Greek alphabet



NOTE: Table reproduced from ISO 8859-8 (1988).

Figure A.5: Character code table 04 - Latin/Hebrew alphabet



NOTE: Table reproduced from ISO 8859-9.

Figure A.6: Character code table - Latin alphabet number 5

Annex B (normative): CRC decoder model

The 32-bit CRC decoder is specified in figure B.1

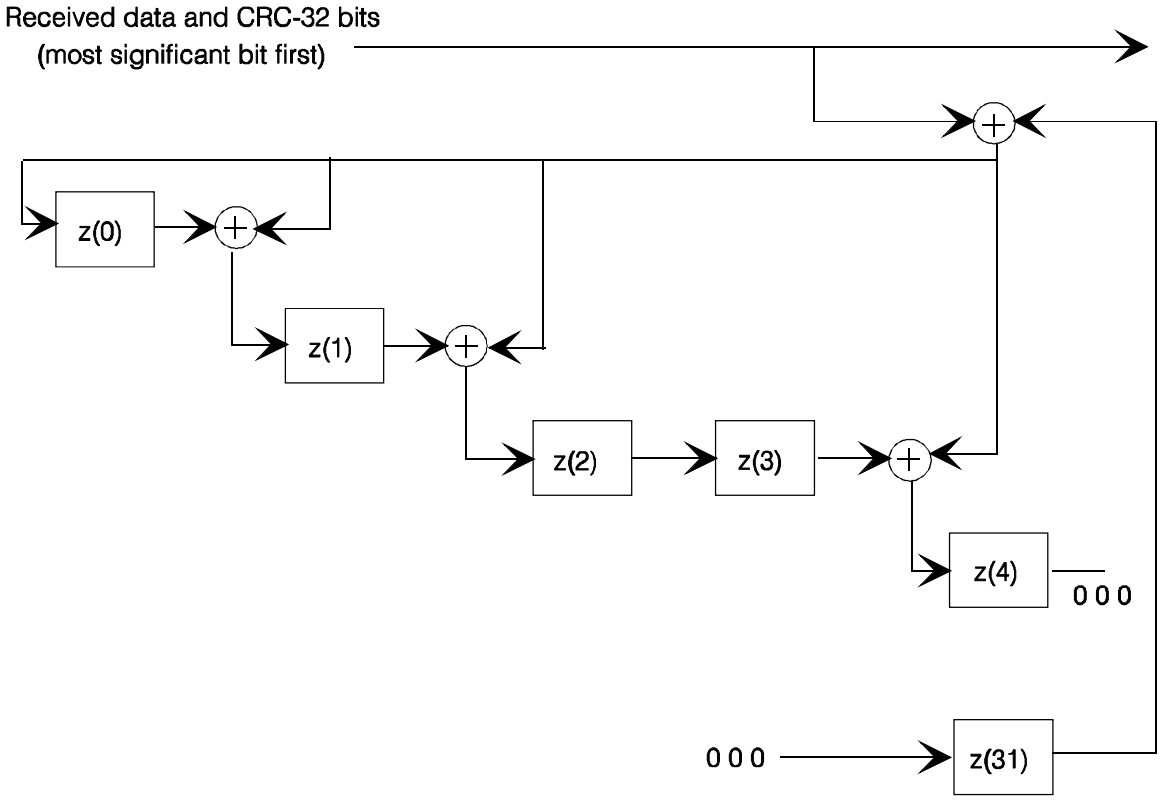


Figure B.1: 32-bit CRC decoder model

The 32 bit CRC decoder operates at bit level and consists of 14 adders + and 32 delay elements z(i). The input of the CRC decoder is added to the output of z(31), and the result is provided to the input z(0) and to one of the inputs of each remaining adder. The other input of each remaining adder is the output of z(i), while the output of each remaining adder is connected to the input of z(i+1), with i = 0, 1, 3, 4, 6, 7, 9, 10, 11, 15, 21, 22, and 25. See figure above.

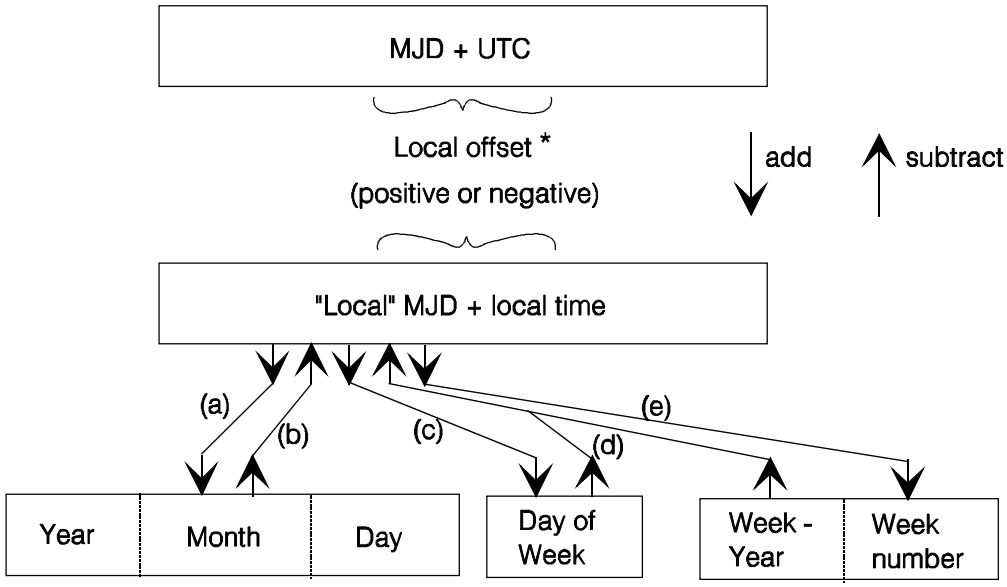
This is the CRC calculated with the polynomial:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

At the input of the CRC decoder bytes are received. Each byte is shifted into the CRC decoder one bit at a time, with the most significant bit (msb) first, i.e. from byte 0x01 (the last byte of the startcode prefix), first the seven "0"s enter the CRC decoder, followed by the one "1". Before the CRC processing of the data of a section the output of each delay element z(i) is set to its initial value "1". After this initialization, each byte of the section is provided to the input of the CRC decoder, including the four CRC_32 bytes. After shifting the last bit of the last CRC_32 byte into the decoder, i.e. into z(0) after the addition with the output of z(31), the output of all delay elements z(i) is read. In case of no errors, each of the outputs of z(i) has to be zero. At the CRC encoder the CRC_32 field is encoded with such value that this is ensured.

Annex C (informative): Conversion between time and date conventions

The types of conversion which may be required are summarized in the diagram below.



* Offsets are positive for Longitudes East of Greenwich and negative for longitudes West of Greenwich.

Figure C.1: Conversion routes between Modified Julian Date (MJD) and Co-ordinated Universal Time (UTC)

The conversion between MJD + UTC and the "local" MJD + local time is simply a matter of adding or subtracting the local offset. This process may, of course, involve a "carry" or "borrow" from the UTC affecting the MJD. The other five conversion routes shown on the diagram are detailed in the formulas below.

Symbols used:

MJD:	Modified Julian Day
UTC:	Co-ordinated Universal Time
Y:	Year from 1900 (e.g. for 2003, Y = 103)
M:	Month from January (= 1) to December (= 12)
D:	Day of month from 1 to 31
WY:	"Week number" Year from 1900
MN:	Week number according to ISO 2015
WD:	Day of week from Monday (= 1) to Sunday (= 7)
K, L ,M' , W, Y':	Intermediate variables
x:	Multiplication
int:	Integer part, ignoring remainder
mod 7:	Remainder (0-6) after dividing integer by 7

a) To find Y, M, D from MJD

Y' = int [(MJD - 15 078,2) / 365,25]
M' = int { [MJD - 14 956,1 - int (Y' x 365,25)] / 30,6001 }
D = MJD - 14 956 - int (Y' x 365,25) - int (M' x 30,6001)
If M' = 14 or M' = 15, then K = 1; else K = 0
Y = Y' + K
M = M' - 1 - K x 12

b) To find MJD from Y, M, D

If M = 1 or M = 2, then L = 1; else L = 0
MJD = 14 956 + D + int [(Y - L) x 365,25] + int [(M + 1 + L x 12) x 30,6001]

c) To find WD from WJD

WD = [(MJD + 2) mod 7] + 1

d) To find MJD from WY, WN, WD

WJD = 15 012 + WD + 7 x { WN + int [(WY x 1 461 / 28) + 0,41] }

e) To find WY, WN from MJD

W = int [(MJD / 7) - 2 144,64]
WY = int [(W x 28 / 1 461) - 0,0079]
WN = W - int [(WY x 1 461 / 28) + 0,41]

EXAMPLE:

MJD	=	45 218	W	=	4 315
Y	=	(19)82	WY	=	(19)82
M	=	9 (September)	WN	=	36
D	=	6	WD	=	1 (Monday)

NOTE: These formulas are applicable between the inclusive dates 1 900 March 1 to 2 100 February 28.

Annex D (informative): Bibliography

- DVB Project office: "Guidelines on Implementation of Service Information".

- DVB Project office: Final technical report of the Conditional access Group.

- Implementation guidelines for use of telecommunications interfaces in the Digital Broadcasting systems.

History

Document history			
December 1994	Unified Approval Procedure	UAP 23:	1994-12-19 to 1995-04-14
June 1995	Vote	V 81:	1995-06-12 to 1995-08-18
October 1995	First Edition		
February 1996	Converted into Adobe Acrobat Portable Document Format (PDF)		

ISBN 2-7437-0310-5
Dépôt légal : Octobre 1995