Society of Cable Telecommunications Engineers

ENGINEERING COMMITTEE
Digital Video Subcommittee

AMERICAN NATIONAL STANDARD

ANSI/SCTE 57 2003

(Formerly DVS 507)

System Information for Satellite Distribution of Digital Television for Cable and MMDS

NOTICE

The Society of Cable Telecommunications Engineers (SCTE) Standards are intended to serve the public interest by providing specifications, test methods and procedures that promote uniformity of product, interchangeability and ultimately the long term reliability of broadband communications facilities. These documents shall not in any way preclude any member or nonmember of SCTE from manufacturing or selling products not conforming to such documents, nor shall the existence of such standards preclude their voluntary use by those other than SCTE members, whether used domestically or internationally.

SCTE assumes no obligations or liability whatsoever to any party who may adopt the Standards. Such adopting party assumes all risks associated with adoption of these Standards or Recommended Practices, and accepts full responsibility for any damage and/or claims arising from the adoption of such Standards or Recommended Practices.

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. SCTE shall not be responsible for identifying patents for which a license may be required or for conducting inquires into the legal validity or scope of those patents that are brought to its attention.

Patent holders who believe that they hold patents which are essential to the implementation of this standard have been requested to provide information about those patents and any related licensing terms and conditions. Any such declarations made before or after publication of this document are available on the SCTE web site at http://www.scte.org.

140 Philips Road
Exton, PA 19341

Table of Contents

List of Figures

List of Tables

System Information for Satellite Distribution of Digital Television for Cable and MMDS

1 Scope

1.1 Purpose

This document defines a Standard for System Information (SI) compatible with MPEG-2 compliant digital multiplex bitstreams constructed in accordance with ISO/IEC 13818-1 (MPEG-2) and transmitted over satellite for distribution on cable and MMDS. The document defines the standard protocol that carries relevant System Information (SI) tables contained within packets carried in the transport multiplex. The term **SI** will be used to refer to system-wide information in the Network Packet Identifier (PID).

This standard is replacing the previous standards for satellite delivery (see References **Error! Reference source not found.** and **Error! Reference source not found.**). The reader should note that not all applications use this standard for SI delivery. SI for cable applications have been superceded by SCTE 65 2002 (Reference 20) and ATSC Standard A/65B (Reference 21). Other common standards for this application are written by the Digital Video Broadcasting Project (DVB) and standardized by ETSI. (See references 22 through 25).

1.2 Application

This document describes tables that are applicable to satellite broadcast signals carried over cable and MMDS. As noted below, some tables are carried in the Network PID in the MPEG-2 Transport Stream. Others tables are carried in transport streams packets carrying the MPEG-2 TS_program_map_section. This PID is referred to as the Program Map Table (PMT) PID. (The reader should note that portions of SI tables defined in this specification are delivered in messages. A message is a data structure that conforms to the ISO/IEC 13818-1 (MPEG-2) table section format.)

This document also defines certain descriptors that may appear in the satellite, cable and MMDS environments.

1.2.1 System Information for Satellite Transmission Medium

MPEG-2 Transport Streams compliant with this specification shall include, for the satellite application, System Information in the Network PID indicated in the Program Association Table carried in PID 0 per MPEG-2. Messages carrying data applicable to satellite transmission include the Network Information message, the Network Text message, and the Virtual Channel message. Each of these message types includes a field called **transmission_medium**, set to “satellite” for tables applicable to satellite use.

The System Time message shall be included in the Transport Stream to synchronize system events. Rate of transmission is typically once per minute. MPEG-2 Transport Streams carried on satellite shall include SI data appropriate to the satellite medium, including Satellite Information Table, Satellite Name Table, Transponder Data Tables, Transponder Name Tables, Carrier Definition Table, Modulation Mode Table, Source Name Table, Virtual Channel Tables, and Defined Channel Maps. Transmission of Map Name Tables, Inverse Channel Maps, Rating System Tables, and Ratings Text Tables is optional.

SI data shall at minimum describe the satellite and transponder carrying the Transport Stream, but may optionally describe other satellites and other transponders, both on the same and on other satellites.

SI data for transmission media other than satellite may be present in a Transport Stream delivered over satellite. These tables are present to support equipment downstream from the satellite downlink, including Decoders receiving Transport Streams on other media. The Decoder is expected to discard tables for media other than the one within which it is designed to operate.

.

1.2.2 Cable and MMDS Distribution of Satellite Transport Streams

For satellite distribution over cable and MMDS, MPEG-2 compliant Transport Streams shall include System Information in the Network PID indicated in the Program Association Table carried in PID 0. Messages carrying data applicable to cable and MMDS transmission include the Network Information message, the Network Text message, and the Virtual Channel message. Each of these message types includes a field called **transmission_medium**, set to "cable" for tables applicable to cable use.

The System Time message shall be included in the Transport Stream to synchronize system events. Rate of transmission is typically once per minute.

SI data for transmission media other than cable may be present in a Transport Stream delivered over cable and MMDS. The Decoder is expected to discard tables for media other than the one within which it is designed to operate.

1.3 Organization

The sections of this document are organized as follows:

- **Section 1** — Provides this general introduction.
- **Section 2** — Lists applicable documents.
- **Section 3** — Provides a list of acronyms and abbreviations used in this document.
- **Section 4** — Describes the basic structure of sections.
- **Section 5** — Describes formats of sections carried in the Network PID.
- **Section 6** — Describes formats of sections carried in the PMT PID.

- **Section 7** — Describes descriptors and stream types used.
- **Section 8** — Describes multilingual character strings.
- **Annex A** — Provides an overview of tables defined in this System Information Standard.
- **Annex B —** Provides usage guidelines.

2 References

The following documents are applicable to this System Information Standard:

2.1 Normative References

The following documents contain provisions that in whole or in part, through reference in this text, constitute provisions of this Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision and amendment, and parties to agreements based on this Standard are encouraged to investigate the possibility of applying the most recent editions of the documents listed below.

1. ATSC A/52A (2001), Digital Audio Compression (AC-3).
2. ATSC A/53B (2001), ATSC Digital Television Standard.
3. EBU Tech. 3264-E, Specification of the EBU Subtitling Data Exchange Format, European Broadcasting Union, February 1991.
4. ISO 639-2 — Code for the Representation of Names of Languages – Part 2: Alpha-3 code, 1998.
5. ISO 3166, Codes for the Representation of Names of Countries, 1988-08-15.
6. The Unicode Standard, Version 3.0, The Unicode Consortium, Addison-Wesley Pub., ISBN 0201616335
7. Unicode Technical Report #6, A Standard Compression Scheme for Unicode, Revision 3.0, 1999-11-12, The Unicode Consortium
8. ISO/IEC IS 13818-1, (2000), Information Technology — Generic coding of moving pictures and associated audio — Part 1: Systems.
9. SCTE 40 2003 Digital Cable Network Interface Standard
10. ITU-T Rec. J.83:1997, Digital multi-programme systems for television, sound and data services for cable distribution.
11. ITU-R Rec. BO.1211:1995, Digital multi-programme emission systems for television, sound and data services for satellites operating in the 11/12 GHz frequency range.
12. SCTE 27, 2003 Subtitling Methods for Broadcast Cable.
13. SCTE 19 2001 SCTE Proposed Standard Methods for Isochronous Data Services Transport
14. SCTE 53 2002 SCTE Proposed Standard Methods for Asynchronous Data Services Transport
15. SCTE 54 2003 Digital Video Service Multiplex and Transport System Standard for Digital Television
16. SCTE 35 2001 (DVS 253) Digital Program Insertion Cueing Message for Cable

2.2 Informative References

17. EIA/CEA-766-A: U.S. and Canadian Region Rating Table (RRT) and Content Advisory Descriptor for Transport of Content Advisory Information Using ATSC A/65, A66 and A67, Program and System Information Protocol (PSIP)
18. SCTE 65 2002: “Service Information Delivered Out-Of-Band For Digital Cable Television”
19. ATSC Standard A/65B: “Program and System Information Protocol for Terrestrial Broadcast and Cable”
20. ETSI EN 300 468: "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems".
21. ETSI TR 101 211: "Digital Video Broadcasting (DVB); Guidelines on implementation and usage of Service Information (SI)".
22. ETSI TR 101 154: "Digital Video Broadcasting (DVB); Implementation guidelines for the use of MPEG-2 Systems, Video and Audio in satellite, cable and terrestrial broadcasting applications".
23. ETSI ETR 162: "Digital Video Broadcasting (DVB); Allocation of Service Information (SI) codes for DVB systems".

3 Definitions

3.1 Compliance Notation

As used in this document, *"shall"* denotes a mandatory provision of the standard. *"Should"* denotes a provision that is recommended but not mandatory. *"May"* denotes a feature whose presence does not preclude compliance that may or may not be present at the option of the implementer.

3.2 Definition of Terms

The following terms are used throughout this document:

section: A data structure comprising a portion of an *ISO/IEC 13818-1*-defined table, such as the Program Association Table (PAT), Conditional Access Table (CAT), or Program Map Table (PMT). The term conforms to MPEG terminology. All sections begin with the table_ID and end with the CRC_32 field. Sections are carried in non-PES[1] streams; their starting points within a packet payload are indicated by the pointer_field mechanism defined in the *ISO/IEC 13818-1 Systems* document.

message*:* The more general term *message* is used interchangeably with *section*, especially to refer to non-table-oriented data structures such as, for example, the System Time message. Likewise, the term *message* is used to refer to a data structure that may deliver portions of various types of tables. The Network Information message, for example, defines portions of several types of network tables.

program element: A generic term for one of the elementary streams or other data streams that may be included in a program.

program: A collection of program elements. Program elements may be elementary streams. Program elements need not have any defined time base; those that do have a common time base and are intended for synchronized presentation. The term *program* is also used in the context of a "television program" such as a scheduled daily news broadcast. The distinction between the two usages should be understood by context.

service*: ISO/IEC 13818-1* uses the term *program* to refer to a collection of program elements without regard to time. In this System Information Standard, the term *service* is used in this same context to denote a collection of elementary components. Usage of the term *service* clarifies certain discussions that also involve the notion of the term *program* in its traditional meaning — in, for example, the statement, "A video service carries a series of programs."

stream: An ordered series of bytes. The usual context for the term *stream* involves specification of a particular PID (such as the "Program Map Table PID stream"), in

[1] Packetized Elementary Stream

which case the term indicates a series of bytes extracted from the packet multiplex from packets with the indicated PID value.

epoch: A period of time. A "program epoch" is the period of time during which a particular program is aired. Access control is defined in terms of program epochs, so that access requirements may change at program epoch boundaries, but not between.

3.3 Acronyms and Abbreviations

The following acronyms and abbreviations are used within this specification:

bslbf bit serial, leftmost bit first
CAT Conditional Access Table
CDT Carrier Definition Table
CRC cyclic redundancy check
ECM Entitlement Control Message
EMM Entitlement Management Message
GMT Greenwich Mean Time
GPS Global Positioning System
IPG Interactive Program Guide
IPPV Impulse Pay-Per-View
IRD Integrated Receiver-Decoder
ISO International Organization for Standardization
MCPT Multiple Carriers per Transponder
MMDS Multi-point Multi-channel Distribution System
MMT Modulation Mode Table
MPEG Moving Picture Experts Group
NVOD Near Video On Demand
PAT Program Association Table
PCR Program Clock Reference
PES Packetized Elementary Stream
PID Packet Identifier
PMT Program Map Table
PSI Program Specific Information
PTS Presentation Time Stamp
rpchof remainder polynomial coefficients, highest order first
RTT Rating Text Table
SECAM Sequential Couleur Avec Memoire
SIT Satellite Information Table
SMATV Satellite Master Antenna Television
TAI International Atomic Time[2]
TDT Transponder Data Table
TNT Transponder Name Table
TS Transport Stream

[2] Reversal of acronym letters is due to the translation from the French.

UTC Coordinated Universal Time[3]
uimsbf unsigned integer, most significant bit first
VCN Virtual Channel Number
VCT Virtual Channel Table[4]

3.4 Section and Data Structure Syntax Notation

This document contains symbolic references to syntactic elements. These references are typographically distinguished by the use of a different font (e.g., restricted), may contain the underscore character (e.g., sequence_end_code) and may consist of character strings that are not English words (e.g., dynrng).

The formats of sections and data structures in this document are described using a C-like notational method employed in *ISO/IEC 13818-1*. Extensions to this method are described in the following sections.

3.4.1 Field Sizes

Each data structure is described in a table format wherein the size in bits of each variable within that section is listed in a column labeled "Bits." The column adjacent to the bits column is labeled "Bytes" and indicates the size of the item in bytes. For convenience, several bits within a particular byte or multi-byte variable may be aggregated for the count. An example follows:

	Bits	Bytes	Description
example_section() {			
section_syntax_indicator	1	1	
...			
if (section_syntax_indicator) {			
table_extension	16	(2)	uimsbf
ISO_reserved	2	(1)	bslbf
version_number	5		uimsbf
current_next_indicator	1		bslbf {next, current}
...			
}			
...			
}			

In the byte count column, items that are conditional (because they are within a loop or conditional statement) are parenthesized. Nested parentheses are used if the loops or conditions are nested.

[3] Since unanimous agreement could not be achieved by the ITU on using the English word order, CUT, or the French word order, TUC, a compromise to use neither was reached.
[4] Note that the structure and semantics of this table as defined herein is restricted to this specification only.

4 Message Structure

This section defines the structure of tables and messages specified in this System Information Standard. This System Information standard defines tables and message sections that are carried in transport packets identified by the Network PID and the Program Map Table (PMT) PID.

4.1 Common Message Structure

4.1.1 Framing and Synchronization

Tables and messages defined in this System Information Standard are structured in the same manner used for carrying *ISO/IEC 13818-1* -defined PSI tables. The MPEG-defined 32-bit CRC shall be present in all cases.

4.1.2 Table ID Ranges and Values

Table 4.1 defines table_ID ranges and values.

Table 4.1. Table ID Ranges and Values

Table ID Value (hex)	System Information Sections	Stream	Reference
0xC0	PROGRAM INFORMATION	Program Map Table	Sec. 6.1
0xC1	PROGRAM NAME	Program Map Table	Sec. 6.2
0xC2	NETWORK INFORMATION	Network	Sec. 5.1
0xC3	NETWORK TEXT	Network	Sec. 5.2
0xC4	VIRTUAL CHANNEL	Network	Sec. 5.3
0xC5	SYSTEM TIME	Network	Sec. 5.4

The messages defined in this document, and any created as user extensions to it are considered “private” with respect to *ISO/IEC 13818-1*. For other table identifiers, refer to the ATSC registry.

4.1.3 Maximum Section Length

In accordance with *ISO/IEC 13818-1*, the maximum total length of any message section specified herein shall be 1024 bytes. This total includes table_ID, CRC, and everything in between.

4.1.4 Extensibility

This System Information Standard describes a number of tables and messages delivered in transport packets identified by either the Network PID or Program Map Table (PMT) PID. The System Information Standard is designed to be extensible via the following mechanisms:

1. **Reserved Fields:** Fields in this System Information Standard marked reserved are reserved for use either when revising this System Information Standard, or when another standard is issued that builds upon this one. See Section 4.1.5 below.

2. **User Private Descriptors:** Privately defined descriptors may be placed at designated locations throughout the messages described in this System Information Standard.

4.1.5 Reserved Fields

reserved — Fields in this System Information Standard marked "reserved" shall not be assigned by the user, but shall be available for future use. Decoders are expected to disregard reserved fields for which no definition exists that is known to that unit. Fields marked "reserved" shall be set to a value of zero until such time as they are defined and supported.

ISO_reserved — Fields in this System Information Standard marked "ISO_reserved" are reserved under *ISO/IEC 13818-1* and hence shall not be assigned by the user. All ISO_reserved bits shall be set to '1.'

zero — Indicates the bit or bit field shall be the value zero.

4.1.6 Protocol Version Field

protocol_version — A 5-bit unsigned integer field whose function is to allow, in the future, any defined table type to carry parameters that may be structured fundamentally differently than those defined in the current protocol. At present, all defined message types in this protocol are defined for protocol_version zero only. Nonzero values of protocol_version may only be processed by Decoders designed to accommodate the later versions as they become standardized.

4.2 Message Structure – Program Map Table PID Only

This section describes details of message structure and transport of program description messages carried within transport packets identified with PIDs containing the Program Map Table in the MPEG-2 multiplex.

4.2.1 Stuffing Table

Table ID 0xFD is reserved for use as a “stuffing table.” The Decoder is expected to ignore the contents of a stuffing table. In some applications it will be helpful for

generation equipment to include one or more stuffing tables in transport packets formatted in compliance with MPEG-2 PSI structures, so that downstream equipment may easily edit the Transport Stream to replace them with other data.

5 Message Formats — Network PID

The following sections define the formats of messages as they appear within the Network PID of the Transport Stream.

5.1 Network Information Message

The NETWORK INFORMATION message is carried in MPEG-2 transports stream packets identified by the Network PID (see Section 1.2.1 and 1.2.2) Stream, and delivers sections of non-textual tables applicable system-wide. The tables include:

- Carrier Definition Tables (CDTs) for each transmission medium (cable and satellite)
- Modulation Mode Tables (MMTs) for each transmission medium
- Satellite Information Table (SIT)
- Transponder Data Table (TDT)

The message format consists of

- The table index pointing to the first record in the table to be defined in this message
- The number of records being defined in this message
- The table type (CDT, MMT, and so on)
- If the table is a TDT type, the satellite ID

Figure 5.1 shows the format of the NETWORK INFORMATION message.

	Bits	Bytes	Description
network_info_message() {			
table_ID	8	1	uimsbf value 0xC2
zero	1	2	bslbf
reserved[5]	3		bslbf
section_length	12		uimsbf
zero	3	1	
protocol_version	5		uimsbf see section 4.1.6
first_index	8	1	uimsbf range 1-255
number_of_records	8	1	uimsbf
transmission_medium	4	1	uimsbf see Table 5.1
table_type	4		uimsbf see Table 5.2
if (table_type==TDT) {			
satellite_ID	8	(1)	uimsbf range 0-255
}			
for (i=0; i<number_of_records; i++) {			
if (table_type==CDT) {			
CDT_record()		((5))	
}			
if (table_type==MMT) {			
MMT_record()		((6))	
}			
if (table_type==SIT) {			
SIT_record()		((4))	
}			
if (table_type==TDT) {			
TDT_record()		((6))	
}			
descriptors_count	8	(1)	uimsbf range 0-255
for (i=0; i<descriptors_count; i++) {			
descriptor()	*	((*))	optional
}			
}			
for (i=0; i<N; i++) {			
descriptor()	*	(*)	optional
}			
CRC_32	32	4	rpchof
}			

Figure 5.1. Network information message format

5.1.1 Table ID

The table_ID of the NETWORK INFORMATION message shall be 0xC2.

5.1.2 Common Data

Certain data is common to all types of data delivered in the NETWORK INFORMATION message.

[5] Reserved field may indicate addressing mode in some deployed systems.

first_index — An 8-bit unsigned integer number in the range one to 255 that indicates the index of the first record to be defined in this message. If more than one record is provided, the additional records define successive table entries following first_index. The value zero is illegal and shall not be specified.

number_of_records — An 8-bit unsigned integer number that defines the number of records being defined in this message. The minimum allowed value is one. The maximum is limited by the maximum allowed length of the message.

transmission_medium — A 4-bit field that defines the transmission medium for which the data in this NETWORK INFORMATION message applies. Table 5.1 defines the coding.

Table 5.1. Transmission Medium

transmission_medium	meaning
0	cable
1	satellite
2	MMDS
3	SMATV
4-15	reserved

A NETWORK INFORMATION message received with transmission_medium indicating an unknown or unsupported medium is expected to be discarded.

table_type — A 4-bit value that defines the type of table delivered in the message. One instance of a NETWORK INFORMATION message can define entries within at most one type of table. The table_type parameter is defined in Table 5.2.

Table 5.2. Table Type

table_type	meaning
0	invalid
1	**CDT** — Carrier Definition Table
2	**MMT** — Modulation Mode Table
3	**SIT** — Satellite Information Table
4	**TDT** — Transponder Data Table
5-15	reserved

A value of zero for table_type is undefined, and may be used to indicate *unknown* or *invalid*.

A NETWORK INFORMATION message received with table_type indicating an unknown or unsupported table type is expected to be discarded.

5.1.2.1 Satellite ID

The satellite_ID is included in the message if the table_type is TDT.

satellite_ID — An 8-bit unsigned integer value in the range zero to 255 that identifies the satellite to which data in this message applies.

5.1.2.2 Descriptors Count

descriptors_count — An 8-bit unsigned integer value in the range zero to 255 representing the number of descriptor blocks to follow.

5.1.3 Carrier Definition Table (CDT)

Figure 5.2 defines the structure of the CDT_record(). The CDT defines sets of carrier frequencies to support cable and satellite media. Separate tables for each of the transmission media may be present in network data. If that is the case, the Decoder is expected to discard the inapplicable table or tables. A full frequency plan table for one medium is constructed from one or more CDT_record() structures, each defining a starting frequency, a number of carriers, and a frequency spacing for carriers in this group.

The specified carrier represents the nominal center of the spectral band for all modulation methods, including analog. Carrier frequencies in the table thus represent the data carrier frequency for digital transmissions modulated using QAM or PSK.

Each CDT_record represents a definition of N carriers. The first_index parameter reflects the index of the first carrier in the group. If the message includes more than one CDT_record(), the carrier index of the second group would be first_index plus the number of carriers defined in the first group. References to the Carrier Definition Table, such as the CDT_reference in the TDT_record(), are to the carrier's index (a carrier defined within a CDT_record()), *not* to the index of a CDT_record() itself.

Note that the carriers, as defined by one or more CDT_record()s, may or may not end up sorted in order of increasing carrier frequency. Certain frequency plans may be specified by overlapping two or more CDT_record()s, each of which defines equally-spaced carriers.

Note also that carriers may be defined that are currently not in use. To facilitate the compressed delivery format, carriers may be defined that do not reflect reality. An example: carriers at 1, 2, 4, 5, 7, 8 MHz could be defined as eight carriers at 1MHz spacing (3 MHz and 6 MHz do not really exist, or are not currently in use).

	Bits	Bytes	Description
CDT_record() {			
number_of_carriers	8	1	uimsbf
spacing_unit	1	2	bslbf see Table 5.3
reserved	1		bslbf reserved
frequency_spacing	14		uimsbf range 1-16,383 units of 10 or 125kHz
frequency_unit	1	2	bslbf see Table 5.4
first_carrier_frequency	15		uimsbf range 0-32,767 units of 10 or 125kHz
}			

Figure 5.2. CDT record format

number_of_carriers — An unsigned integer in the range one to 255 that represents the number of carriers whose frequency is being defined by this CDT_record().

spacing_unit — A 1-bit field identifying the units for the frequency_spacing field. Table 5.3 defines the coding for spacing_unit.

Table 5.3. Spacing Unit

spacing_unit	meaning
0	10 kHz spacing
1	125 kHz spacing

frequency_spacing — A 14-bit unsigned integer number in the range one to 16,383 that defines the frequency spacing in units of either 10 kHz or 125 kHz, depending upon the value of the spacing_unit parameter. If spacing_unit is zero, indicating 10 kHz, then a value of one indicates 10 kHz spacing, two indicates 20 kHz, and so on. If the number_of_carriers field is one, the frequency_spacing field is ignored. The maximum frequency that can be represented is ((2^14) -1) * 125 kHz = 2047.875 MHz. The minimum frequency spacing is 10 kHz.

frequency_unit — A 1-bit field identifying the units for the first_carrier_frequency field. Table 5.4 defines the coding for frequency_unit.

Table 5.4. Frequency Unit

frequency_unit	meaning
0	10 kHz units
1	125 kHz units

first_carrier_frequency — A 15-bit unsigned integer number in the range zero to 32,767 that defines the starting carrier frequency for the carriers defined in this group, in units of either 10 kHz or 125 kHz, depending on the value of frequency_unit. If only one carrier is defined for the group, the first_carrier_frequency represents its frequency. When the frequency_unit indicates 125 kHz, the first_carrier_frequency can be interpreted as a fractional frequency (1/8 MHz) in the least-significant 3 bits, and an integer number of megahertz in the upper 12 bits. The range of frequencies that can be represented is zero to ((2^15) - 1) * 125 kHz = 4095.875 MHz.

For satellite use, carrier frequencies specified in the CDT are defined relative to the point in the receiver following block conversion of the downlink signal — i.e., at the L-band input to the receiver. The Local Oscillator (L.O.) frequency offset is defined by the frequency_band parameter defined in the Satellite Information Table for each satellite. In addition, a convention for synthesis of the tuned frequency is assumed (either high side or low side), again based on frequency band. Table 5.5 defines the L.O. frequency and conversion method for each band.

To convert a frequency specified in the CDT to the corresponding downlink frequency, for the high-side conversion method, the CDT frequency is subtracted from the L.O. frequency appropriate to the satellite. For low-side conversion, the CDT frequency is added to the L.O. frequency.

Table 5.5. Local Oscillator Frequencies vs. Frequency Band

Frequency Band	L. O. Frequency	Conversion
C Band	5.150 GHz	High side
Ku (FSS)	10.750 GHz	Low side
Ku (BSS)	11.250 GHz	Low side

5.1.4 Modulation Mode Table (MMT)

Figure 5.3 defines the structure of the MMT_record().

	Bits	Bytes	Description
MMT_record() {			
transmission_system	4	1	uimsbf see Table 5.6
inner_coding_mode	4		uimsbf see Table 5.7
split_bitstream_mode	1	1	bslbf {no, yes}
reserved	2		bslbf reserved
modulation_format	5		uimsbf see Table 5.8
reserved	4	4	bslbf reserved
symbol_rate	28		uimsbf units: one symbol per sec.
}			

Figure 5.3. MMT record format

5.1.4.1 Transmission System

transmission_system — A 4-bit field that identifies the transmission standard employed for the waveform defined by this MMT record. Table 5.6 defines the coding for transmission_system.

Table 5.6. Transmission System

transmission_system	meaning
0	**unknown** — The transmission system is not known.
1	The transmission system conforms to Annex A of Reference [10].
2	The transmission system conforms to Annex B of Reference [10].
3	The transmission system conforms to Reference [11].
4	**ATSC** — The transmission system conforms to the ATSC Digital Television Standard (Reference [2]).
5	**DigiCipher** — The transmission system conforms to the General Instrument DigiCipher® II System for satellite distribution of compressed audio and video.
6-15	reserved

5.1.4.2 Inner Coding Mode

inner_coding_mode — A 4-bit field that indicates the coding mode for the inner code associated with the waveform described in this MMT record. The following values are currently defined: 5/11, 1/2, 3/5, 2/3, 3/4, 4/5, 5/6, and 7/8. Coding of the inner_coding_mode field is shown in Table 5.7.

Table 5.7. Inner Coding Mode

inner_coding_mode	meaning
0	rate 5/11 coding
1	rate 1/2 coding
2	reserved
3	rate 3/5 coding
4	reserved
5	rate 2/3 coding
6	reserved
7	rate 3/4 coding
8	rate 4/5 coding
9	rate 5/6 coding
10	reserved
11	rate 7/8 coding
12-14	reserved
15	none — indicates that the waveform does not use concatenated coding

5.1.4.3 Modulation Format

modulation_format — A 5-bit field that defines the basic modulation format for the carrier. Table 5.8 defines the parameter.

Table 5.8. Modulation Format

modulation_format	meaning
0	**unknown** — The modulation format is unknown.
1	**QPSK** — The modulation format is QPSK (Quadrature Phase Shift Keying).
2	**BPSK** — The modulation format is BPSK (Binary Phase Shift Keying).
3	**OQPSK** — The modulation format is offset QPSK.
4	reserved
5	reserved
6	**QAM 16** — The modulation format is 16-level Quadrature Amplitude Modulation (QAM).
7	**QAM 32** — 32-level QAM
8	**QAM 64** — 64-level QAM
9	**QAM 100** — 100-level QAM
10	**QAM 128** — 128-level QAM
11	**QAM 144** — 144-level QAM
12	**QAM 196** — 196-level QAM
13	**QAM 256** — 256-level QAM
14	**QAM 400** — 400-level QAM
15	**QAM 512** — 512-level QAM
16	**QAM 576** — 576-level QAM
17	**QAM 784** — 784-level QAM
18	**QAM 1024** — 1024-level QAM
19	**8-PSK**
20-31	reserved

5.1.4.4 Symbol Rate

symbol_rate — A 28-bit unsigned integer field that indicates the symbol rate in units of one symbol per second associated with the waveform described in this MMT record.

5.1.5 Satellite Information Table (SIT)

Figure 5.4 defines the structure of the SIT_record().

	Bits	Bytes	Description
SIT_record() {			
satellite_ID	8	1	uimsbf
you_are_here	1	2	bslbf {no, yes}
frequency_band	2		uimsbf see Table 5.9
out_of_service	1		bslbf {no, yes}
hemisphere	1		bslbf see Table 5.10
orbital_position	11		uimsbf units: 0.1 deg.
polarization_type	1	1	bslbf see Table 5.11
reserved	1		bslbf reserved
number_of_transponders	6		uimsbf range 0-63 (1-count)
}			

Figure 5.4. SIT record format

satellite_ID — An 8-bit unsigned integer number in the range zero to 255 that identifies the satellite associated with the data defined in this record.

you_are_here — A Boolean flag that indicates, when set, that the satellite_ID field reflects the satellite carrying the Transport Stream that delivered this NETWORK INFORMATION message. The you_are_here flag may be used to verify dish alignment during installation of a C-band or Ku-band IRD having a movable dish. All NETWORK INFORMATION messages having you_are_here bits set true must be locally originated at an Encoder, rather than being originated at a central site and carried in the Network stream.

frequency_band — A 2-bit field that indicates the frequency band associated with the CDT record. Table 5.9 defines the coding.

Table 5.9. Frequency Band

frequency_band	meaning
0	C Band
1	Ku Band (FSS)
2	Ku Band (BSS)
3	reserved

Refer to Table 5.5 for a description of how frequency_band is used to specify downlink frequencies for the satellite CDT.

out_of_service — A Boolean flag that, when set, indicates that the satellite given by satellite_ID is permanently out of service; i.e., it has failed or has been retired without replacement. When the flag is false, the satellite is currently in (at least partial) operation. The out_of_service flag may be used as a signal to delete satellite and transponder records from Decoder memory.

hemisphere—A 1-bit field indicating whether the satellite given by satellite_ID resides in the western (value zero) or eastern (value one) hemisphere. The coding for hemisphere is given in Table 5.10.

Table 5.10. Hemisphere

hemisphere	meaning
0	Western hemisphere
1	Eastern hemisphere

orbital_position — An 11-bit unsigned integer number in the range zero to 1800 representing the orbital position of the satellite given by satellite_ID, in units of 0.1° longitude. The longitudinal coordinates provided in the table are West longitude for products fielded in the western hemisphere, and East longitude for products fielded in the eastern hemisphere.

polarization_type — A one-bit field that indicates whether the satellite uses a linear polarization method (horizontal/vertical planes), or a circular method (left/right circular polarization). Table 5.11 defines polarization_type.

Table 5.11. Polarization Type

polarization_type	meaning
0	Linear polarization
1	Circular polarization

number_of_transponders — A 6-bit unsigned integer number in the range zero to 63 representing one less than the number of transponders (carriers) associated with the satellite being defined. A value of 23 in the field indicates that the satellite is associated with 24 transponders (carriers), for example.

5.1.6 Transponder Data Table (TDT)

Figure 5.5 defines the structure of the TDT_record().

	Bits	Bytes	Description
TDT_record() {			
transport_type	1	1	bslbf see Table 5.12
polarization	1		bslbf see Table 5.13
transponder_number	6		uimsbf range 0-63
CDT_reference	8	1	uimsbf range 1-255
if (transport_type==MPEG_2) {			
MMT_reference	8	(1)	uimsbf range 1-255
VCT_ID	16	(2)	uimsbf range 0-0xFFFF
reserved	8		bslbf reserved
} else { /* non-MPEG_2 */			
wide_bandwidth_video	1	(1)	bslbf {no, yes}
reserved	2		bslbf reserved
waveform_standard	5		uimsbf see Table 5.14
audio_mode()	24	(3)	
}			
}			

Figure 5.5. TDT record format

transport_type — A 1-bit field identifying the type of transport carried on this transponder as either being an MPEG-2 transport (value zero), or not (value one). Table 5.12 defines the coding.

Table 5.12. Transport Type

transport_type	meaning
0	MPEG-2 transport
1	Non-MPEG-2 transport

polarization — A 1-bit field identifying the polarization of the carrier associated with a satellite transponder. A value of zero indicates horizontal (or left) polarization; a value of one indicates vertical (or right) polarization. The polarization method used (linear or circular) is given in the SIT record for the satellite as polarization_type. Table 5.13 defines polarization.

Table 5.13. Polarization

polarization	meaning
0	Horizontal (for linearly polarized satellites) or Left (for circularly polarized satellites)
1	Vertical (for linearly polarized satellites) or Right (for circularly polarized satellites)

transponder_number — An unsigned 6-bit integer value in the range zero to 63 that indicates which transponder is being defined by this TDT_record(). A value of zero indicates the first transponder, and so on.

NOTE: Numbering in the TDT may or may not correspond to conventional transponder numbering, that for typical C-band satellites (for example) are numbered one to 24.

CDT_reference — An unsigned 8-bit integer value in the range one to 255 that associates the transponder being defined with a particular carrier frequency via this reference into the Carrier Definition Table. The value zero is illegal and shall not be specified. CDT_reference is used as an index value into the CDT.

If a frequency_spec_descriptor() (see Section 7.2.1) is present with the TDT record, the CDT_reference shall be discarded.

5.1.6.1 Standard MPEG-2 Transports

MMT_reference — An 8-bit unsigned integer value in the range one to 255 that references an entry into the Modulation Mode Table (MMT) for satellite transmission medium. For digital waveforms, the MMT_reference associates the transponder/carrier with a digital modulation mode. The value zero is illegal and shall not be specified. MMT_reference is used as an index value into the MMT.

VCT_ID — A 16-bit unsigned integer number in the range zero to 0xFFFF that indicates the Virtual Channel Table ID associated with this transponder.

5.1.6.2 Other Transport Types

wide_bandwidth_video — A Boolean flag that indicates, when set, that the video is modulated at 10.75 MHz peak deviation. When the flag is clear, the video is modulated in "narrow" format, or 8.2 MHz peak deviation.

waveform_standard — A 5-bit field that identifies the waveform standard associated with the transponder (carrier). Table 5.14 defines waveform_standard. If the waveform used has not been explicitly defined in the table, then it shall be coded as "unknown".

Table 5.14. Waveform Standard

waveform_standard	meaning
0	**unknown** — The waveform type is unknown or undefined..
1	**NTSC** — The waveform is standard NTSC
2	**PAL 625** — The waveform is standard 625-line PAL
3	**PAL 525** — The waveform is standard 525-line PAL
4	**SECAM** — The waveform is standard SECAM
5	**D2-MAC** — The waveform is D2-MAC
6	**B-MAC** — The waveform is B-MAC
7	**C-MAC** — The waveform is C-MAC
8	**DCI** — The waveform conforms to the General Instrument DigiCipher® I scrambling standard
9	**VideoCipher** — The waveform conforms to the General Instrument VideoCipher® scrambling standard
10	**RCA DSS** — The waveform conforms to the RCA DSS system
11	**Orion** — The waveform is scrambled using the TvCom (formerly Oak) Orion system
12	**Leitch** — The waveform is scrambled using the Leitch system
13-31	reserved

5.1.6.3 Audio Mode

Figure 5.6 describes the format of the audio_mode() structure.

	Bits	Bytes	Description
audio_mode() {			
wide_bandwidth_audio	1	3	bslbf see Table 5.15
companded_audio	1		bslbf {no, yes}
matrix_mode	2		uimsbf see Table 5.16
subcarrier_2_offset	10		uimsbf range 0-1023, units of 0.01MHz above 5.0 MHz
subcarrier_1_offset	10		uimsbf range 0-1023, units of 0.01MHz above 5.0 MHz
}			

Figure 5.6. Audio mode structure format

wide_bandwidth_audio — A 1-bit field indicating whether the audio subcarrier(s) defined for this transponder/carrier are modulated in narrow bandwidth (value zero) or wide bandwidth (value one) format. Table 5.15 defines the coding.

Table 5.15. Wide Bandwidth Audio

wide_bandwidth_audio	meaning
0	narrow bandwidth
1	wide bandwidth

companded_audio — A Boolean flag indicating whether audio is transmitted in companded format on this transponder/carrier. A value of zero indicates not companded; a value of one indicates companded audio.

matrix_mode — A 2-bit field that indicates the matrix mode associated with analog audio subcarriers on this transponder/carrier. The field is coded as shown in Table 5.16.

Table 5.16. Matrix Mode

matrix_mode	meaning
0	**mono** — Indicates that subcarrier 1 carries the L+R (mono) audio channel. Subcarrier 2, if present (specified as a different frequency than subcarrier 1), then subcarrier 2 carries cue tones or other audio.
1	**discrete_stereo** — Indicates that subcarrier 1 carries the left audio channel, and subcarrier 2 carries the right audio channel.
2	**matrix_stereo** — Indicates that subcarrier 1 carries the L+R (sum) vector and subcarrier 2 carries the L-R (difference) vector.
3	**reserved**

subcarrier_2_offset — A 10-bit unsigned integer field in the range zero to 1023 that defines the audio subcarrier offset for the second audio subcarrier (if any). The value is given in units of 0.01 MHz above 5.0 MHz. A value of one, for example, indicates a subcarrier offset of 5.01 MHz. If no second subcarrier exists, the value of subcarrier_2_offset is set to equal subcarrier_1_offset.

subcarrier_1_offset — A 10-bit unsigned integer field in the range zero to 1023 that defines the audio subcarrier offset for the primary audio subcarrier. The value is given in units of 0.01 MHz above 5.0 MHz.

5.1.7 Message-End Descriptors

The message may include at its end one or more structures of the form tag, length, data. The number of descriptors present is determined indirectly by processing the section_length field.[6]

[6] The total capacity of the CDT is 255 frequencies. As more Ku band satellites are launched that do not re-use existing frequency plans, this CDT may potentially overflow. One solution to a CDT overflow condition is a mechanism whereby the frequency can be specified directly in the TDT. This is done using the frequency_spec_descriptor.

5.2 Network Text Message

The NETWORK TEXT message is carried in the network data stream, and delivers sections of textual tables applicable system-wide. The NETWORK TEXT message is defined multilingually. The defined table types include:

- Transponder Name Table (TNT)
- Satellite Text Table (STT)
- Rating Text Table (RTT)
- Rating System Table (RST)
- Currency System Table (CST)
- Source Name Table (SNT)
- Map Name Table (MNT)

Figure 5.7 defines the format of the NETWORK TEXT message. Note that table type acronyms defined above parenthetically are used in Figure 5.7, as opposed to the actual integer table type values defined in Table 5.18. This is done to improve readability.

	Bits	Bytes	Description
network_text_message() {			
table_ID	8	1	uimsbf value 0xC3
zero	2	2	bslbf
ISO_reserved	2		bslbf
section_length	12		uimsbf
zero	3	1	
protocol_version	5		see section 4.1.6
ISO_639_language_code	24	3	per ISO 639 Part 2
transmission_medium	4	1	uimsbf see Table 5.17
table_type	4		uimsbf see Table 5.18
if (table_type==TNT) {			
satellite_ID	8	(1)	uimsbf range 0-255
first_index	8	(1)	uimsbf range 0-255
number_of_TNT_records	8	(1)	uimsbf
for (i=0; i<number_of_TNT_records; i++) {			
TNT_record()	*	((*))	
TNT_descriptors_count	8	((1))	range 0-255
for (i=0; i<TNT_descriptors_count; i++) {			
descriptor()	*	((*))	
}			
}			
}			
if (table_type==STT) {			
first_index	8	(1)	uimsbf range 0-255
number_of_STT_records	8	(1)	uimsbf
for (i=0; i<number_of_STT_records; i++) {			
STT_record()	*	((*))	
STT_descriptors_count	8	((1))	uimsbf range 0-255
for (i=0; i<STT_descriptors_count; i++) {			
descriptor()	*	(((*)))	
}			
}			
}			
if (table_type==RTT) {			
rating_region	8	(1)	uimsbf
rating_text_table()	*	(*)	
}			
if (table_type==RST)			
rating_system_table()	*	(*)	
if (table_type==CST)			
currency_system_table()	*	(*)	
if (table_type==SNT)			
source_name_table()	*	(*)	
if (table_type==MNT)			
map_name_table()	*	(*)	
for (i=0; i<N; i++) {			
descriptor()	*	(*)	optional
}			
CRC_32	32	4	rpchof
}			

Figure 5.7. Network text message format

5.2.1 Table ID

The table_ID of the NETWORK TEXT message shall be 0xC3.

5.2.2 Multilingual Text

The NETWORK TEXT message carries multilingual text strings, formatted as defined in Section 8. Text strings included in the NETWORK TEXT message shall not include format effectors (defined in Section 8.3). If format effectors are present in a text block, the Decoder is expected to disregard them.

5.2.3 Transmission Medium

transmission_medium — A 4-bit field that defines the transmission medium for which the data in this NETWORK TEXT message applies. Table 5.17 defines transmission_medium.

Table 5.17. Transmission Medium

transmission_medium	meaning
0	**Cable**
1	**Satellite**
2	MMDS
3	**SMATV**
4-14	reserved
15	**all** — Indicates the table is applicable to all transmission media

A NETWORK TEXT message received with transmission_medium indicating an unknown or unsupported medium is expected to be discarded.

5.2.4 Table Type

table_type — A 4-bit value that defines the type of table delivered in the message. One instance of a NETWORK TEXT message can define entries within at most one type of table. The table_type parameter is defined in Table 5.18.

Table 5.18. Table Type

table_type	meaning
0	Invalid
1	**TNT** — Transponder Name Table
2	**STT** — Satellite Text Table
3	**RTT** — Ratings Text Table
4	**RST** — Rating System Table
5	**SNT** — Source Name Table
6	**MNT** — Map Name Table
7	**CST –** Currency System Table
8-15	Reserved

A NETWORK TEXT message received with table_type indicating an unknown or unsupported table_type is expected to be discarded.

5.2.5 ISO 639 Language Code

ISO_639_language_code — A 3-byte language code per ISO 639 Part 2 defining the language associated with the text carried in this NETWORK TEXT message. The ISO_639_language_code field contains a three-character code as specified by ISO 639 Part 2. Each character is coded into 8 bits according to ISO 8859-1 (ISO Latin-1) and inserted in order into the 24-bit field. The value 0xFFFFFF shall be used in case the text is available in one language only

5.2.6 Transponder Name Table (TNT)

The Transponder Name Table consists of a series of TNT records defined in Figure 5.8 that provide textual names associated with satellite transponders, represented as multilingual character strings.

satellite_ID — An 8-bit unsigned integer number in the range zero to 255 that identifies the satellite associated with the data defined in this message.

first_index — An unsigned 8-bit integer number in the range zero to 255 that indicates the index of the first TNT record to be defined in this message. If more than one record is provided, the additional records define successive table entries following first_index.

number_of_TNT_records — An unsigned 8-bit integer number that defines the number of records being defined in this message. The minimum allowed value is one. The maximum is limited by the maximum allowed length of the message.

TNT_record() — A data structure defined as shown in Figure 5.8.

	Bits	Bytes	Description
TNT_record() {			
reserved	2	1	bslbf reserved
transponder_number	6		uimsbf range 0-63
reserved	3	1	bslbf reserved
transponder_name_length	5		uimsbf range 0-31 (N)
if (transponder_name_length>0) {			
transponder_name()	8*N	N	
}			
}			

Figure 5.8. TNT record format

transponder_number — A 6-bit unsigned integer value in the range zero to 63 that identifies the transponder number whose name is being defined by the transponder_name field to follow.

transponder_name_length — A 5-bit unsigned integer value in the range zero to 31 that defines the length, in bytes, of the transponder_name() field. If the value is zero, the transponder_name() field is omitted. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow. Note that the field length does not necessarily correspond with the number of *characters* comprising the name, given that some encoding modes are double-byte.

transponder_name() — The textual name of the transponder, represented in the multilingual format described in Section 8. The name field consists of one or more <mode><length> blocks.

5.2.7 Satellite Text Table (STT)

The Satellite Text Table consists of a series of records that provide textual names associated with satellites, in both abbreviated and full-length versions, represented as multilingual character strings. The first_index and number_of_records fields have the same definitions for the STT as for the TNT.

Figure 5.9 defines the structure of the STT_record().

	Bits	Bytes	Description
STT_record() {			
satellite_ID	8	1	uimsbf
reserved	4	1	bslbf reserved
sat_reference_name_length	4		uimsbf range 0,4-15 (R)
if (sat_reference_name_length>0) {			
sat_reference_name()	R*8	R	
}			
reserved	3	1	bslbf reserved
full_satellite_name_length	5		uimsbf range 0, 4-31 (L)
if (full_satellite_name_length>0) {			
full_satellite_name()	L*8	L	
}			
}			

Figure 5.9. STT record format

satellite_ID — An 8-bit unsigned integer number in the range zero to 255 that identifies the satellite associated with the satellite name data defined in this record.

sat_reference_name_length — A 4-bit unsigned integer value in the range zero or four to 15 that defines the length, in bytes, of the sat_reference_name() field. If the length is zero, the

sat_reference_name() field is omitted from the message. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow. Note that the field length does not necessarily correspond with the number of *characters* comprising the name, given that some encoding modes are double-byte.

sat_reference_name() — The abbreviated (reference) name of the satellite, represented in the multilingual format described in Section 8. The field consists of one or more <mode><length> blocks.

full_satellite_name_length — A 5-bit unsigned integer value in the range zero or four to 31 that defines the length, in bytes, of the full_satellite_name() field. If the length is zero, the full_satellite_name() field is omitted from the message. Note that the field length does not necessarily correspond with the number of *characters* comprising the name, given that some encoding modes are double-byte.

full_satellite_name() — The full name of the satellite, represented in the multilingual format identified in Section 8. The field consists of one or more <mode><length> blocks.

5.2.8 Rating Text Table (RTT)

The Rating Text Table provides textual representations for all rating dimensions for one particular rating region. Decoders are expected to disregard RTT definitions for regions other than the one to which they are currently assigned.

The Rating Text Table defines parental rating text associated with particular rating regions. Downloadable rating text allows definitions of ratings to change dynamically. Note that the RTT provides for dynamic redefinition of all aspects of parental ratings control, including:

- The number of defined dimensions (up to six)
- The name of a particular dimension
- The number of defined levels within one dimension
- The textual string representation of any level

For each rating region, the name of each of the dimensions may be defined. If a particular dimension is undefined, the number of levels defined for that dimension shall be specified as zero, and no text data shall follow in the message for that dimension.

For each defined dimension, the textual representation for each of as many as 16 levels is defined. Figure 5.10 describes the format of the Rating Text Table for one ratings region.

For parental rating limits (or ceilings) set by the user, level zero is typically not an option, since it would mean "disallow any program that has a rating defined for this dimension." For these reasons, the textual representation for level zero is typically a null string (zero length).

	Bits	Bytes	Description
rating_text_table() {			
for (i=0; I<6; i++) {			
levels_defined	8	(1)	uimsbf range 0-16
if (levels_defined>0) {			
dimension_name_length	8	((1))	uimsbf (D) range 0-255
dimension_name()	8*D	((D))	
for (i=0; i<levels_defined; i++) {			
rating_name_length	8	(((1)))	uimsbf (R) range 0-255
rating_name_text()	R*8	(((R)))	
}			
}			
}			
}			

Figure 5.10. Rating text table format

levels_defined — An unsigned 8-bit integer in the range zero to 16 that represents the number of levels defined for this particular dimension. If the number of defined levels is zero, the dimension_name_length and dimension_name() fields shall be omitted from the message.

dimension_name_length — An unsigned 8-bit integer in the range zero to 255 that represents the length, in bytes, of the dimension_name field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

dimension_name() — A text string, in standard multilingual format, describing one particular rating dimension for this rating region. One dimension in a rating region, for example, may be used to describe a program's MPAA rating. The dimension name for that dimension may be defined as "MPAA Rating."

rating_name_length — An unsigned 8-bit integer in the range zero to 255 that represents the length, in bytes, of the rating_name_text() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

rating_name_text() — A text string, in standard multilingual format, describing one particular rating level within the dimension being described in this loop in the message structure. One of the currently defined rating levels in a rating region in the MPAA Rating dimension, for example, is "PG-13."

5.2.9 Rating System Table (RST)

The Rating System Table provides textual representations for each rating region in use in the network. A Decoder may use text from the RST to present the user with a menu of choices of rating system. Figure 5.11 defines the format of the rating_system_table().

	Bits	Bytes	Description
rating_system_table() {			
regions_defined	8	1	uimsbf range 1-63
for (i=0; i<regions_defined; i++) {			
data_length	8	(1)	uimsbf range 4-255
rating_region	8	(1)	uimsbf
string_length	8	(1)	uimsbf (L)
rating_system_text()	8*L	(L)	uimsbf
for (i=0; i<N; i++) {			
descriptor()	*	((*))	as indicated by data_length
}			
}			
}			

Figure 5.11. Rating system table format

regions_defined — An 8-bit unsigned integer number in the range one to 63 that represents the number of rating regions being textually defined in this NETWORK TEXT message.

data_length — An 8-bit unsigned integer number in the range four to 255 that defines the number of bytes of data to follow in this iteration of the for loop. The data_length parameter is provided to allow an optional variable number of descriptor blocks to be included with each region definition.

rating_region — An 8-bit unsigned integer number that defines the rating region to be associated with the text in rating_system_text().

string_length — An 8-bit unsigned integer number that defines the total length in bytes of the rating_system_text() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

rating_system_text() — A data structure containing a multilingual text string defining the rating system given by rating_region. Text strings are formatted according to the rules outlined in Section 8.

5.2.10 Currency System Table (CST)

The Currency System Table provides textual representations for each currency region in use in the network. A Decoder may use text from the CST to present the user with a

menu of choices of currency system. Figure 5.12 defines the format of the currency_system_table().

	Bits	Bytes	Description
currency_system_table() {			
regions_defined	8	1	uimsbf range 1-63
for (i=0; i<regions_defined; i++) {			
data_length	8	(1)	uimsbf range 4-255
currency_region	8	(1)	uimsbf
string_length	8	(1)	uimsbf (L)
currency_system_text()	8*L	(L)	uimsbf
for (i=0; i<N; i++) {			
descriptor()	*	((*))	as indicated by data_length
}			
}			
}			

Figure 5.12. Currency system table format

regions_defined — An 8-bit unsigned integer number in the range one to 63 that represents the number of currency regions being textually defined in this NETWORK TEXT message.

data_length — An 8-bit unsigned integer number in the range four to 255 that defines the number of bytes of data to follow in this iteration of the for loop. The data_length parameter is provided to allow an optional variable number of descriptor blocks to be included with each region definition.

currency_region — An 8-bit unsigned integer number that defines the currency region to be associated with the text in currency_system_text().

string_length — An 8-bit unsigned integer number that defines the total length in bytes of the currency_system_text() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

currency_system_text() — A data structure containing a multilingual text string defining the currency system given by currency_region. Text strings are formatted according to the rules outlined in Section 8.

5.2.11 Source Name Table (SNT)

The format of the source_name_table() is given in Figure 5.13.

	Bits	Bytes	Description
source_name_table() {			
number_of_SNT_records	8	1	uimsbf range 1-255
for (i=0; i<number_of_SNT_records; i++) {			
application_type	1	(1)	bslbf {false, true}
reserved	7		bslbf
if (application_type) {			
application_ID	16	((2))	uimsbf
} else {			
source_ID	16	((2))	uimsbf
}			
name_length	8	(1)	size of source_name() (L)
source_name()	L*8	(L)	multilingual text
SNT_descriptors_count	8	(1)	uimsbf range 0-255
for (i=0; i<SNT_descriptors_count; i++) {			
descriptor()	*	((*))	
}			
}			
}			

Figure 5.13. Source name table format

number_of_SNT_records — An unsigned 8-bit integer number in the range one to 255 that defines the number of records being defined in this message.

application_type — A Boolean flag that indicates, when set, that the name string being defined is for an application whose ID is given in application_ID. When the flag is clear, the name string being defined is for a source whose ID is given in source_ID.

application_ID — A 16-bit unsigned integer value identifying the application associated with the name string to follow.

source_ID — A 16-bit unsigned integer value identifying the programming source associated with the source name to follow.

name_length — An unsigned 8-bit integer number in the range one to 255 that defines the number of bytes in the name() structure to follow.

source_name() — A multilingual text string defining the name of the source or application, formatted according to the rules defined in Section 8.

SNT_descriptors_count — An unsigned 8-bit integer number in the range zero to 255 that defines the number of descriptors to follow.

5.2.12 Map Name Table (MNT)

The Map Name Table (MNT) provides a textual name for each Virtual Channel Table in the network. The names may be provided multilingually. Figure 5.14 describes the format.

	Bits	Bytes	Description
map_name_table() {			
number_of_MNT_records	8	1	uimsbf
for (i=0; i<number_of_MNT_records; i++) {			
VCT_ID	16	(2)	uimsbf
map_name_length	8	(1)	uimsbf (N)
map_name()	N*8	(N)	multilingual text
MNT_descriptors_count	8	(1)	uimsbf range 0-255
for (i=0; i<MNT_descriptors_count; i++) {			
descriptor()	*	((*))	
}			
}			
}			

Figure 5.14. Map name table format

number_of_MNT_records — An 8-bit unsigned integer number in the range one to 255 that identifies the number of data records containing VCT_ID and MNT data to follow in the message. The number of records included is further limited by the allowed maximum message length.

VCT_ID — A 16-bit unsigned integer value which defines the ID of the Virtual Channel Table to be associated with the name string given in the map_name() to follow.

map_name_length — An 8-bit unsigned integer value which defines the length, in bytes, of the map_name() structure to follow.

map_name() — A multilingual text string defining the textual name of the Virtual Channel Table whose ID is given in VCT_ID, given in the language defined in ISO_639_language_code.

MNT_descriptors_count — An 8-bit unsigned integer value in the range zero to 255 that defines the number of descriptors to follow.

5.2.13 Message-End Descriptors

The message may include at its end one or more structures of the form tag, length, data. The number of descriptors present is determined indirectly by processing the section_length field.

5.3 Virtual Channel Message

The VIRTUAL CHANNEL message delivers portions of the Virtual Channel Table, and also delivers the Defined Channel Map (DCM) and Inverse Channel Table (ICT). The VIRTUAL CHANNEL message accommodates the needs of all transmission media.

Figure 5.15 defines the format of the VIRTUAL CHANNEL message body.

	Bits	Bytes	Description
virtual_channel_message() {			
table_ID	8	1	uimsbf value 0xC4
zero	2	2	bslbf
ISO_reserved	2		bslbf
section_length	12		uimsbf
zero	3	1	
protocol_version	5		see section 4.1.6
transmission_medium	4	1	uimsbf see Table 5.1
table_subtype	4		uimsbf see Table 5.19
VCT_ID	16	2	uimsbf
if (table_subtype==DCM) {			
DCM_structure()	*	(*)	
}			
if (table_subtype==VCT) {			
VCT_structure()	*	(*)	
}			
if (table_subtype== ICT) {			
ICT_structure()	*	(*)	
}			
for (i=0; i<N; i++) {			
descriptor()	*	(*)	optional
}			
CRC_32	32	4	rpchof
}			

Figure 5.15. Virtual channel message format

5.3.1 Table ID

The table_ID of the VIRTUAL CHANNEL message shall be 0xC4.

5.3.2 Transmission Medium

transmission_medium — A 4-bit field that defines the transmission medium for which the data in this VIRTUAL CHANNEL message applies. Table 5.1 above defines the coding.

A VIRTUAL CHANNEL message received with transmission_medium indicating an unknown or unsupported medium is expected to be discarded.

5.3.3 Table Subtype

table_subtype — A 4-bit field that indicates the table subtype being delivered in this VIRTUAL CHANNEL message. Three subtypes are defined, one delivering the Defined Channels Map (DCM), one delivering the Virtual Channel Table (VCT) itself, and one defining the Inverse Channel Table (ICT). Table 5.19 defines table_subtype.

Table 5.19. Table Subtype

table_subtype	meaning
0	**VCT** — Virtual Channel Table
1	**DCM** — Defined Channels Map
2	**ICT** — Inverse Channel Table
3-15	reserved

A VIRTUAL CHANNEL message received with table_subtype indicating an unknown or unsupported table subtype is expected to be discarded.

5.3.4 Virtual Channel Table ID

VCT_ID — A 16-bit unsigned integer value in the range 0x0000 to 0xFFFF indicating the VCT to which the channel definitions in this message apply.

5.3.5 Defined Channels Map Structure

Figure 5.16 shows the format of the DCM_structure().

	Bits	Bytes	Description
DCM_structure() {			
reserved	4	2	bslbf reserved
first_virtual_channel	12		uimsbf range 0-4095
reserved	1	1	bslbf reserved
DCM_data_length	7		uimsbf range 1-127
for (i=0; i<DCM_data_length; i++) {			
range_defined	1	(1)	bslbf {no, yes}
channels_count	7		uimsbf range 1-127
}			
}			

Figure 5.16. DCM structure format

5.3.5.1 First Virtual Channel

first_virtual_channel — An unsigned 12-bit integer reflecting the first virtual channel whose existence is being provided by this message, for the map identified by the VCT_ID field. The range is zero to 4095.

5.3.5.2 DCM Data Length

DCM_data_length — A 7-bit unsigned integer number in the range one to 127 that defines the number of DCM_data_fields to follow in the message.

5.3.5.3 DCM Data Bytes

The DCM data bytes taken as a whole define which virtual channels, starting at the channel number defined by first_virtual_channel, are defined and which are not. Each DCM_data_field defines two pieces of data: a flag indicating whether this block of channels is defined or not, and the number of channels in the block. The bytes are interpreted in an accumulative way, with a pointer into the Virtual Channel Table assumed initialized to first_virtual_channel. As each byte is processed, the pointer is incremented by the number of channels indicated by the channel count field.

For example, if channels 2-90, 200-210, 400-410, 600-610, 800-810, and 999 were defined, and first_virtual_channel was zero, the DCM data sequence (in decimal) would be the following, where underlined numbers have the range_defined bit set: 2, 89, 109, 11, 127, 62, 11, 127, 62, 11, 127, 62, 11, 127, 61, 1.

5.3.5.4 Range Defined

range_defined — A Boolean flag that indicates, when true, that the number of channels given by channel_count is defined in the VCT, starting at the current pointer value. When the flag is clear, the number of channels equal to channels_count are currently not defined starting at the current pointer value.

5.3.5.5 Channels Count

channels_count — An unsigned 7-bit integer number in the range one to 127 that indicates the number of defined (or undefined) channels in a group.

5.3.6 Virtual Channel Table

Figure 5.17 defines the format of the VCT_structure().

	Bits	Bytes	Description
VCT_structure() {			
reserved	1	2	bslbf [7]
reserved	1		bslbf[8]
descriptors_included	1		bslbf {no, yes}
reserved	13		bslbf reserved
activation_time	32	4	uimsbf units: GPS sec.
number_of_VC_records	8	1	
for (i=0; i<number_of_VC_records; i++) {			
if (transmission_medium==satellite {			
satellite_virtual_channel()	*	(*)	
} else if (transmission_medium==SMATV) {			
SMATV_virtual_channel()	*	(*)	
} else if (transmission_medium==cable) or			
transmission_medium== MMDS) {			
virtual_channel()	*	(*)	
}			
}			
}			

Figure 5.17. VCT structure format

5.3.6.1 Descriptors Included

descriptors_included — A Boolean flag that indicates, when set, that one or more descriptors are present in the message. When the flag is clear, the descriptor block is absent.

5.3.6.2 Processing of Local Access Virtual Channel Records (Informative)

The VIRTUAL CHANNEL message defines two basic types of virtual channels, primary and "local access." The primary virtual channel records are stored in the Decoder so that any channel can be acquired given its channel number. The idea of a local access point is that a program associated with a particular channel may be carried within a Transport Stream other than the one referenced by the primary virtual channel definition. The local access virtual channel gives the location on *this* TS of a program associated with the primary one.

An example is an IPG service, where one (hidden) virtual channel defines the point of access to an MPEG program carrying one or more streams of IPG data. If it is desired to carry IPG on other Transport Streams, programs on those other Transport Streams could be defined to carry lower-rate data streams carrying IPG data. A "local access" virtual

[7] Bit formerly known as frequency_spec_included.
[8] Bit formerly known as symbol_rate_included.

channel definition may be defined and included in network data generated from the uplink control system. In order to allow the Decoder to associate these programs with the primary one, the local access virtual channel would quote the same virtual channel number as the primary entry point.

Local access records do not overwrite or replace records in the Virtual Channel Table. Processing of local access virtual channel records involves building a list upon entry to a transponder or carrier. If programs can be processed in the background, this list is consulted to allow the Decoder to select and decrypt a stream as appropriate.

5.3.6.3 Activation Control and Time

The activation time indicates the time at which the data delivered in the message will be valid.

activation_time — A 32-bit unsigned integer field providing the absolute second the virtual channel data carried in the message will be valid, defined in terms of the number of GPS seconds since January 1, 1980 00:00:00. Refer to Annex A for a discussion of time in this System Information Standard. If the activation_time is in the past, the data in the message shall be considered valid immediately. An activation_time value of zero shall be used to indicate immediate activation.

5.3.6.4 Number of Virtual Channel Records

number_of_VC_records — An 8-bit unsigned integer number in the range one to 255 that identifies the number of virtual_channel() records to follow in the message. The number of records included is further limited by the allowed maximum message length.

5.3.6.5 Virtual Channel Structure for Satellite Transmission Medium

Figure 5.18 defines the format of the satellite_virtual_channel() structure.

	Bits	Bytes	Description
satellite_virtual_channel() {			
HDTV_channel	1	2	bslbf {no, yes}
reserved	3		bslbf reserved
virtual_channel_number	12		uimsbf range 0-4095
application_virtual_channel	1	1	bslbf {no, yes}
bitstream_select	1		bslbf see Table 5.20
tone_select	1		bslbf (off, on}
transport_type	1		bslbf see Table 5.21
channel_type	4		uimsbf see Table 5.22
if (application_virtual_channel) {			
application_ID	16	(2)	uimsbf
} else {			
source_ID	16	(2)	uimsbf
}			
if (channel_type ==NVOD_access) {			
reserved	4	(2)	bslbf reserved
NVOD_channel_base	12		uimsbf range 1-4095
reserved	16	(2)	bslbf reserved
} else			
if (transport_type==MPEG_2) {			
satellite	8	((1))	uimsbf range 0-255
reserved	2	((1))	bslbf reserved
transponder	6		uimsbf range 0-63
program_number	16	((2))	uimsbf
} else if (transport_type !=MPEG_2) {			
satellite	8	((1))	uimsbf range 0-255
reserved	2	((1))	bslbf reserved
transponder	6		uimsbf range 0-63
suppress_video	1	((2))	bslbf
reserved	13		bslbf reserved
audio_selection	2		blsbf see Table 5.23
}			
}			
if (descriptors_included) {			
descriptors_count	8	(1)	uimsbf
for (i=0; i<descriptors_count; i++) {			
descriptor()	*	((*))	
}			
}			
}			

Figure 5.18. Virtual channel format for satellite transmission medium

5.3.6.5.1 HDTV Channel

HDTV_channel — A binary flag that indicates, when set the virtual channel is carrying a High Definition video signal.

5.3.6.5.2 Virtual Channel Number

virtual_channel_number — An unsigned 12-bit integer in the range zero to 4095 reflecting the virtual channel whose definition is being provided by this virtual channel record, for the map identified by the VCT_ID field.

5.3.6.5.3 Application Virtual Channel

application_virtual_channel — A binary flag that indicates, when set, that this virtual channel defines an access point for an application whose ID is given in application_ID. When the flag is clear, the channel is not an application access point, and the 16-bit application_ID/source_ID field is the source_ID.

bitstream_select — A one-bit field that identifies the bit stream associated with programs referenced by this virtual channel number, when the Transport Stream carrying the virtual channel is MPEG-2 compatible, and the Transport Stream is carried on a split bit stream. When the TS is a combined TS, the bitstream_select bit shall indicate stream 1. Table 5.20 defines bitstream_select.

Table 5.20. Bitstream Select

bitstream_select	meaning
0	stream_0
1	stream_1

tone_select — A one-bit field that identifies whether the 22kHz control tone superimposed on the IRD LNB power output shall be energized.

5.3.6.5.4 Transport Type

transport_type — A 1-bit field identifying the type of transport carried on this transponder as either being an MPEG-2 transport (value zero), or not (value one). Table 5.21 defines transport_type. MPEG-2 transports shall not violate *ISO/IEC 13818-1* (Reference [8]).

Table 5.21. Transport Type

transport_type	meaning
0	MPEG-2 transport
1	non-MPEG-2 transport

5.3.6.5.5 Channel Type

channel_type — A 4-bit field defining the channel type. Table 5.22 defines channel_type. Control streams associated with background programs may exist on multiple Transport Streams. A Transport Stream other than the one identified in the virtual channel defining the program may also carry the same (or related) data. When that occurs, a virtual channel record tagged as local_access, associated with the same virtual channel number,

will be present to define local access to the program. Decoders supporting background program processing are expected to store local_access virtual channel definitions separately from the working VCT, and use the local definition when not on the Transport Stream identified in the working map.

Table 5.22. Channel Type

channel_type	meaning
0	**normal** — Indicates that the record is a regular virtual channel record, and does not define a local access, hidden, or NVOD channel. For non-MPEG-2 transports, the **channel_type** shall be specified as "**normal**."
1	**hidden** — Indicates that the record identifies a virtual channel that may not be accessed by the user by direct entry of the channel number (hidden). Hidden channels are skipped when the user is channel surfing, and appear as if undefined if accessed by direct channel entry. Programs constructed for use by specific applications (such as NVOD theaters) utilize hidden virtual channels.
2	**local access** — Indicates that the virtual channel record is not stored in the permanent (or working) map, but overrides the definition in the working map only on the Transport Stream upon which it was received.
3	**NVOD access** — Indicates that the virtual channel is an access point to a number of hidden virtual channels comprising a near video on demand (NVOD) group. Instead of containing a regular virtual channel definition indicating the location of a Transport Stream, the virtual channel record points to the base of a group of hidden virtual channels comprising the theater cine-plex.
4-15	**reserved** — Decoders shall treat virtual channel records of unknown **channel_type** the same as non-existent (undefined) channels.

5.3.6.5.6 Application ID

application_ID — A 16-bit unsigned integer number in the range 0x0001 to 0xFFFF that identifies the application associated with the virtual channel, on a system-wide basis. One particular program guide application, for example, may look for a program carrying data in its native transmission format by searching through the Virtual Channel Table for a match on its assigned application_ID. In some cases, one application may be able to process streams associated with more than one application ID. The application ID may be used to distinguish content as well as format, for the benefit of processing within the application. The value zero for application_ID shall not be assigned; if specified in a VIRTUAL CHANNEL message, the value zero indicates "unknown" or "inapplicable" for the application_ID/source_ID field.

Refer to Annex A for a discussion of the use of application_ID.

5.3.6.5.7 Source ID

source_ID — A 16-bit unsigned integer number in the range 0x0000 to 0xFFFF that identifies the programming source associated with the virtual channel. The source ID is expected to be unique at each receiving location utilizing this standard. In this context, a *source* is one specific source of video, text, data, or audio programming. For the purposes of referencing virtual channels to the program guide database, each such program source is associated with a unique value of source_ID. The source_ID itself may appear in an IPG

database, where it tags entries to associate them with specific services. The value zero for source_ID, if used, shall indicate the channel is not associated with a source ID.

5.3.6.5.8 NVOD Channel Base

NVOD_channel_base — A 12-bit unsigned integer number in the range zero to 4095 that defines, for NVOD base channels, the index of the virtual channel record defining the access path for NVOD theatre zero.

5.3.6.5.9 Satellite and Transponder

satellite — An 8-bit unsigned integer number in the range zero to 255 that identifies the satellite that carries the programs referenced by this virtual channel. The satellite combined with the transponder identify the point of access for programs accessed through this virtual channel number.

transponder — A 6-bit unsigned integer number in the range zero to 63 that identifies the transponder number on the satellite identified by the satellite field that carries the programs referenced by this virtual channel.

5.3.6.5.10 Audio Selection

Virtual channels may be associated with analog services such as clear NTSC or VideoCipher II+ waveforms. The satellite and transponder references index into the SIT and TDT, which define the location of analog subcarriers, if any. Normally, a virtual channel associated with a VideoCipher II+ waveform would use the digital audio included with VCII+. Some operators, however, carry alternate language audio on analog subcarriers in the same transponder that carries the VCII+ signal. The audio_selection parameter is used to allow a virtual channel to use the subcarrier audio in place of the usual digital audio.

Audio-only services carried on analog subcarriers are supported as well via the suppress_video flag.

suppress_video — A binary flag that indicates, when set, that this virtual channel is described entirely by its non-video components. An example of a virtual channel in which video is suppressed is one used to identify a radio service carried on one or more subcarriers on a transponder. When the flag is clear, the video component should not be suppressed.

audio_selection — A 2-bit field that associates one or two audio subcarriers with the virtual channel, or alternatively, with the standard audio carried by a scrambled waveform. Table 5.23 defines the coding.

Table 5.23. Audio Selection

Value	Meaning
0	**default_audio**—Indicates the audio for the virtual channel is processed in the standard way appropriate to the waveform standard.
1	**subcarrier_1_mono**—indicates the audio should be taken from subcarrier 1, the location of which is defined in the TDT for the referenced transponder.
2	**subcarrier_2_mono**—indicates the audio should be taken from subcarrier 2, the location of which is defined in the TDT for the referenced transponder.
3	**subcarrier_stereo**—indicates the stereo audio should be processed from two subcarriers; subcarrier locations and matrixing mode are defined in the TDT for the referenced transponder.

5.3.6.5.11 Program Number

program_number — A 16-bit unsigned integer number that associates the virtual channel number being defined with services defined in the Program Association Table (PAT) and TS_program_map_section. Access to elementary streams defined in each virtual channel record involves first acquiring the Transport Stream on the satellite and transponder associated with the virtual channel, then referencing the Program Association Table (PAT) in PID 0 to find the PID associated with the Program Map Table (PMT) for this program_number. PIDs for each elementary stream are then found by acquisition of the PMT.

A program_number with value 0x0000 (invalid as a regular program number) is reserved to indicate that the Decoder is expected to discard the corresponding virtual channel record from the queue of pending virtual channel changes. Records are identified in the pending queue by their activation_time, VCT_ID, and virtual_channel_number. If no pending virtual channel change is found in the Decoder's queue, no action should be taken for this virtual channel (i.e. the record is expected to be discarded).

5.3.6.5.12 Frequency Specification in the Virtual Channel Table

The current System Information definition allows any virtual channel definition to optionally include a frequency specification by an overlay called the “frequency specification included” overlay. In the absence of a frequency specification, the IRD finds the frequency associated with the channel by using the satellite and transponder references given in the VCT record, and going into the TDT (which in turn supplies a reference to the CDT).

For the satellite transmission medium, this optional frequency specification is provided to support the needs of multiple carriers per transponder (MCPT) applications. In MCPT applications, carriers can be located virtually anywhere within the full width of a transponder, and no standard frequency plans exist.

The CDT would therefore contain the center frequency for all physical transponders, and would be sufficient for consumer IRDs (which don't support MCPT modes). Non-standard frequency plans may be employed by newly launched satellites, however, such that some or all of these frequencies are not present in the CDT.

For consistency of approach, the frequency_spec_descriptor() (see Section 7.2.1) method shall be used to optionally specify frequency in the VCT, bypassing frequency reference through the TDT and CDT. The freq_spec_included bit becomes reserved for all transmission media as shown in Figure 5.17.

5.3.6.5.13 Modulation Parameters Specification in the Virtual Channel Table

In some satellite applications, multiple carriers are present on one physical transponder. One or more of these carriers may be modulated using different parameters than those defined in the TDT. The modulation_params_descriptor() described in Section 7.2.2 may be placed in the virtual channel record to define parameters in use for the given carrier. When present, the descriptor overrides modulation parameters provided indirectly through reference to the TDT and MMT.

For consistency of approach, the modulation_params_descriptor() method shall be used to optionally specify modulation parameters in the VCT, bypassing the reference to these parameters through the TDT and MMT. The symbol_rate_included bit becomes reserved for all transmission media as shown in Figure 5.17.

5.3.6.5.14 Optional Descriptors

descriptors_count — An 8-bit unsigned integer value in the range zero to 255 that defines the number of descriptors to follow.

5.3.6.6 Virtual Channel Record for SMATV

Figure 5.19 defines the format of the SMATV_virtual_channel() record, used to define VCTs for the SMATV transmission medium. The definitions of most parameters are the same for this data structure as for the satellite_virtual_channel() record. Exceptions and additions are listed below.

5.3.6.6.1 Audio Mode

audio_mode() — A data structure defining the location and format of analog audio subcarriers associated with the virtual channel. Refer to Section 5.1.6.3 for the format of the audio_mode() structure.

5.3.6.6.2 CDT Reference

CDT_reference — An unsigned 8-bit integer number in the range zero to 255 that identifies the frequency associated with this virtual channel. Values one to 255 of CDT_reference are used as indices into the Carrier Definition Table appropriate to the transmission medium to find a frequency to tune to acquire the virtual channel. The value zero is reserved to indicate that the referenced service is carried on *all referenced* digital multiplexes.

5.3.6.6.3 MMT Reference

MMT_reference — An 8-bit unsigned integer value in the range one to 255 that references an entry in the Modulation Mode Table (MMT) appropriate to the transmission medium. The value zero is illegal and shall not be specified. For digital waveforms, the MMT_reference associates the carrier with a digital modulation mode. For Decoder implementations that support only one set of modulation parameters, in systems in which one modulation method is used for all carriers, storage and processing of the MMT_reference is unnecessary.

	Bits	Bytes	Description
SMATV_virtual_channel() {			
reserved	4	2	bslbf reserved
virtual_channel_number	12		uimsbf range 0-4095
application_virtual_channel	1	1	bslbf {no, yes}
bitstream_select	1		bslbf see Table 5.20
reserved	1		bslbf reserved
transport_type	1		bslbf see Table 5.21
channel_type	4		uimsbf see Table 5.22
if (application_virtual_channel) {			
application_ID	16	(2)	uimsbf
} else {			
source_ID	16	(2)	uimsbf
}			
if (channel_type ==NVOD_access) {			
reserved	4	(2)	bslbf reserved
NVOD_channel_base	12		uimsbf range 1-4095
reserved	24	(3)	bslbf reserved
} else			
if (transport_type==MPEG_2) {			
CDT_reference	8	((1))	uimsbf range 0-255
program_number	16	((2))	
MMT_reference	8	((1))	uimsbf range 1-255
reserved	8	((1))	bslbf reserved
} else { /* non-MPEG-2 */			
CDT_reference	8	((1))	uimsbf range 0-255
scrambled	1	((1))	bslbf {false, true}
reserved	3		bslbf
video_standard	4		uimsbf see Table 5.24
audio_mode()	24	((3))	
}			
if (descriptors_included) {			
descriptors_count	8	(1)	
for (I=0; i<descriptors_count; i++) {			
descriptor()	*	((*))	
}			
}			
}			

Figure 5.19. Virtual channel format for SMATV transmission medium.

5.3.6.6.4 Non-Standard Channels

scrambled — A 1-bit Boolean flag that indicates, when set, that the associated non-Standard waveform is scrambled. When the flag is clear, the non-Standard waveform is in the clear.

video_standard — A 4-bit field that indicates the video standard associated with this non-Standard virtual channel. Table 5.24 defines video_standard.

Table 5.24. Video Standard

video_standard	meaning
0	**NTSC** — The video standard is NTSC
1	**PAL 625** — The video standard is 625-line PAL
2	**PAL 525** — The video standard is 525-line PAL
3	**SECAM** — The video standard is SECAM
4	**MAC** — The video standard is MAC
5-15	reserved

5.3.6.7 Virtual Channel Record for Cable and MMDS

Figure 5.20 defines the format of the virtual_channel() record, used to define VCTs for cable and MMDS. The definitions of all parameters except for path_select are as defined for the satellite_virtual_channel() structure.

	Bits	Bytes	Description
virtual_channel() {			
HDTV_channel	1	2	bslbf {no, yes}
reserved	2		bslbf reserved
preferred_source	1		bslbf {no, yes}
virtual_channel_number	12		uimsbf range 0-4095
application_virtual_channel	1	1	bslbf {no, yes}
bitstream_select	1		bslbf see Table 5.20
path_select	1		bslbf see Table 5.25
transport_type	1		bslbf see Table 5.21
channel_type	4		uimsbf see Table 5.22
if (application_virtual_channel) {			
application_ID	16	(2)	
} else {			
source_ID	16	(2)	
}			
if (channel_type ==NVOD_access) {			
reserved	4	(2)	bslbf reserved
NVOD_channel_base	12		uimsbf range 1-4095
reserved	16	(2)	bslbf reserved
} else			
if (transport_type==MPEG_2) {			
CDT_reference	8	((1))	uimsbf range 1-255
program_number	16	((2))	
MMT_reference	8	((1))	uimsbf range 1-255
} else { /* non-MPEG-2 */			
CDT_reference	8	((1))	uimsbf range 0-255
scrambled	1	((1))	bslbf {no, yes}
reserved	3		bslbf reserved
video_standard	4		uimsbf see Table 5.24
reserved	16	((2))	uimsbf
}			
if (descriptors_included) {			
descriptors_count	8	(1)	uimsbf
for (i=0; i<descriptors_count; i++) {			
descriptor()	*	((*))	
}			
}			
}			

Figure 5.20. Virtual channel format

5.3.6.7.1 HDTV Channel

HDTV_channel — A binary flag that indicates, when set the virtual channel is carrying a High Definition video signal.

5.3.6.7.2 Preferred Source

preferred_source — A 1-bit field that is used in some models of satellite IRDs that include cable tuners. When set (=yes), the flag indicates that the cable virtual channel carrying programming indicated by source_ID shall take precedence over any virtual channel available by satellite with the same source_ID. When the flag is clear (=no), no such preference is indicated.

5.3.6.7.3 Path Select

path_select — A 1-bit field that associates the virtual channel with a transmission path. For cable transmission medium, path_select identifies the physical cable that carries the Transport Stream associated with this virtual channel. Table 5.25 defines path_select.

Table 5.25. Path Select

path_select	meaning
0	path 1
1	path 2

5.3.7 Inverse Channel Table

The inverse channel table, once reconstructed in the Decoder from a sequence of VIRTUAL CHANNEL messages (ICT subtype), consists of a list of source_ID/virtual_channel pairs, ordered by source_ID. The Decoder may use this table to quickly find the virtual channel carrying the program given by a particular value of source_ID (by binary search), if such a virtual channel exists. One Inverse Channel Table is defined per Virtual Channel Table. The ICT may be constructed from the VCT, or linear searches may be done to resolve source_ID references. Transmission of the ICT is therefore optional.

Virtual channels that provide access points for applications (i.e., ones tagged with the application_virtual_channel flag) are not included in the ICT.

Figure 5.21 describes the format of the ICT_structure().

	Bits	Bytes	Description
ICT_structure() {			
reserved	4	2	bslbf reserved
first_map_index	12		uimsbf range 0-4095
reserved	1	1	bslbf reserved
record_count	7		uimsbf range 1-127
for (i=0; i<record_count; i++) {			
source_ID	16	(2)	
reserved	4	(2)	bslbf reserved
virtual_channel	12		uimsbf range 0-4095
}			
}			

Figure 5.21. ICT structure format

5.3.7.1 First Map Index

first_map_index — A 12-bit unsigned integer in the range zero to 4095 that represents the index into the Inverse Channel Table where data carried in this ICT_structure() should be stored.

5.3.7.2 Record Count

record_count — A 7-bit unsigned integer value in the range one to 127 that represents the total number of source_ID/ virtual_channel pairs defined in this message.

5.3.7.3 Source ID and Virtual Channel

source_ID — A 16-bit unsigned integer number in the range 0x0001 to 0xFFFF that identifies the source associated with the virtual channel. The source ID is expected to be unique at each receiving location utilizing this standard. In this context, a "source" is one specific source of video, text, data, or audio programming. For the purposes of referencing virtual channels to the program guide database, each such source is associated with a unique value of source_ID. Value zero for source_ID is illegal.

virtual_channel — A 12-bit unsigned integer value in the range zero to 4095 that represents the virtual channel, in the Virtual Channel Table given by VCT_ID, associated with the given source_ID. A virtual_channel value of zero indicates that the program given by source_ID is currently not carried in this Virtual Channel Table. Such placeholders are useful in the case where a certain program's existence within a virtual channel table may come and go.

5.3.8 Message-End Descriptors

The message may include at its end one or more structures of the form tag, length, data. The number of descriptors present is determined indirectly by processing the section_length field.

5.4 System Time Message

The SYSTEM TIME message is used to synchronize Decoders with accurate calendar time, as well as to define IPPV reportback slot timing. Minimal rate of transmission shall be once per minute, at second *00* of each minute.

The processing of the SYSTEM TIME message in the Decoder is time-critical. Delays between reception of the message and processing it increase the inaccuracy of events specified via system time (GPS) seconds. Processing delays should be kept below a maximum of approximately 200 milliseconds.

Figure 5.22 shows the format of the SYSTEM TIME message.

	Bits	Bytes	Description
system_time_message() {			
table_ID	8	1	uimsbf value 0xC5
zero	2	2	bslbf
ISO_reserved	2		bslbf
section_length	12		uimsbf
zero	3	1	
protocol_version	5		see section 4.1.6
reserved	8	1	bslbf
system_time	32	4	uimsbf GPS seconds
GPS_UTC_offset	8	1	uimsbf seconds
for (i=0; i<N; i++) {			
descriptor()	*	(*)	optional
}			
CRC_32	32	4	rpchof
}			

Figure 5.22. System time message format

5.4.1 Table ID

The table_ID of the System Time message shall be 0xC5.

5.4.2 System Time

system_time — A 32-bit unsigned integer quantity representing the current system time as the number of GPS seconds since January 6, 1980 00:00:00.

5.4.3 GPS to UTC Offset

GPS_UTC_offset — An 8-bit unsigned integer that defines the current offset in whole seconds between GPS and UTC time standards. To convert GPS time to UTC, the GPS_UTC_offset is subtracted from GPS time. Whenever the International Bureau of Weights and Measures decides that the current offset is too far in error, an additional leap second may be added (or subtracted), and the GPS_UTC_offset shall contain the new value.

5.4.4 Message-End Descriptors

The message may include at its end one or more structures of the form tag, length, data. The number of descriptors present is determined indirectly by processing the section_length field.

6 Message Formats - PMT PID

The messages in this section describe aspects of programs carried on a particular MPEG service. The PROGRAM INFORMATION and PROGRAM NAME messages are carried on the PID that carries the service's Program Map Table. The maximum spacing that is allowed between occurrences of a PROGRAM INFORMATION message is 500 milliseconds. The maximum spacing that is allowed between occurrences of a PROGRAM NAME message is 1000 milliseconds.

6.1 Program Information Message

The PROGRAM INFORMATION message provides program-specific information related to user interface functions in the Decoder. Epoch start and end times are defined, as is a definition of exactly when the program itself begins, given that the beginning portion of a program epoch may be used for *interstitial* material such as promotions and advertising.

In addition, the message delivers antitaping override information, special event status, parental rating, program cost strings, and NVOD information (when applicable).

Figure 6.1 shows the format of the PROGRAM INFORMATION message.

	Bits	Bytes	Description
program_info_message() {			
table_ID	8	1	uimsbf value 0xC0
zero	2	2	bslbf
ISO_reserved	2		bslbf reserved
section_length	12		uimsbf
zero	3	1	bslbf zero
protocol_version	5		see Section 4.1.6
program_number	16	2	uimsbf
reserved	8	1	bslbf reserved
program_epoch_number	8	1	uimsbf
NVOD_data_included	1	2	bslbf {no, yes}
epoch_start_defined	1		bslbf {no, yes}
epoch_end_defined	1		bslbf {no, yes}
if (epoch_end_defined) {			
special_event	1		bslbf {disabled, enabled}
reserved	1		bslbf {disabled, enabled}
schedule_update	1		bslbf {no, yes}
reserved	10		bslbf reserved
} else {			
reserved	13		bslbf reserved
}			
if (epoch_start_defined) {			
epoch_start_time	32	(4)	uimsbf GPS seconds
} else {			
reserved	32	(4)	bslbf reserved
}			
rating_data()	*	(*)	
if (epoch_end_defined) {			
program_record()	*	(*)	
}			
for (i=0; i<N; i++) {			
descriptor()	*	(*)	optional
}			
CRC_32	32	4	rpchof
}			

Figure 6.1. Program Information Message Format

6.1.1 Table ID

The **table_ID** for the PROGRAM INFORMATION message shall be 0xC0.

6.1.2 Program Number

program_number — A 16-bit unsigned integer number that associates the program information carried in this message with a program referenced by the MPEG-2 Program Association and TS_program_map_section.

6.1.3 Program Epoch Number

program_epoch_number — An 8-bit counter of program epochs. The program_epoch_number parameter may be used to associate status reported by an access control subsystem with

status generated by the Decoder for a given program. If no epoch start time is defined, the program_epoch_number is undefined and may be disregarded.

6.1.4 NVOD Data Included

NVOD_data_included — A Boolean flag that indicates, when set, that the program is part of a Near Video On Demand (NVOD) program group, and that NVOD data is included in the program record. When the flag is clear, the program is not part of an NVOD group, and the overlay is not included in the message. For NVOD data to be valid, the program must have a defined start and end time: if either epoch_start_defined or epoch_end_defined is false, NVOD_data_included shall be false.

Refer to Annex B, Section 4 for a discussion of NVOD.

6.1.5 Epoch Start Defined

epoch_start_defined — A Boolean flag that indicates, when set, that this PROGRAM INFORMATION message defines a program epoch with a specific start time. When the flag is clear, no program is currently defined beyond the default parental rating level. If the epoch has no defined start time, the end time is also assumed to be undefined (epoch_end_defined shall be false). A program with a defined start time may or may not have a defined end time.

6.1.6 Epoch End Defined

epoch_end_defined — A Boolean flag that indicates, when set, that this PROGRAM INFORMATION message defines a program epoch with a known endpoint. When the flag is clear, the program is undefined except for the rating data provided. A program with a defined end time shall also have a start time defined.

6.1.7 Special Event

special_event — If a program epoch end is defined, the special_event bit, when set, indicates that the program is a special event. Special events are programs that initiate immediate telephone reportback if purchased via IPPV. The special_event flag is reserved if the program epoch is not defined.

6.1.8 Schedule Update

schedule_update — A Boolean flag that indicates, when set, that the program information for this epoch (including information carried in the PROGRAM NAME message) has been changed recently and may not reflect scheduled programming as described in a paper or electronic program guide. For systems employing electronic program guides, this bit may be used to signal the Decoder to use program information from the PROGRAM INFORMATION and PROGRAM NAME messages in lieu of information from the program guide database. When the bit is clear, the program information matches that found in the guide database. The schedule_update flag shall not be set if PROGRAM INFORMATION and/or PROGRAM NAME messages are not currently being supplied.

6.1.9 Epoch Start Time

epoch_start_time — A 32-bit unsigned integer field providing the second of the scheduled start of the program epoch, measured as the number of GPS seconds since January 1, 1980 00:00:00. Refer to Annex A for a discussion of GPS time. A value of zero for epoch_start_time indicates that the display of start time should be suppressed in the Decoder.

6.1.10 Rating Data

The parental rating data carried in this PROGRAM INFORMATION message is used to allow the Decoder to compare the program's ratings with the unit's user preferences for parental rating limits. The user interface related to parental ratings is driven by text defined in the Rating Text Table (RTT), delivered in the rating_text_table subtype of the NETWORK INFORMATION message.

The rating_data() structure defines parental rating levels for the program. The system defined here supports multiple *rating regions*. A program may specify its parental rating definition according to the standards defined for each region. Programs without ratings specified for the Decoder's region are assumed to mean the program is not subject to rating-restricted viewing. Ratings may be given for any or all of the defined regions. See Section 5 in Annex B for a discussion on the use of rating regions.

Informative note: the Rating Data information contained in this section was in use prior to industry standard methods currently defined.

Figure 6.2 shows the format of the rating_data() structure.

	Bits	Bytes	Description
rating_data() {			
reserved	2	1	bslbf reserved
rating_regions_count	6		uimsbf range 1-63
for (I=0; i<rating_regions_count; i++) {			
rating_region	8	(1)	(see text)
program_ratings()	24	(3)	see Figure 6.3
rating_description_length	8	(1)	uimsbf (L)
rating_description()	8*L	(L)	
}			
}			

Figure 6.2. Rating Data Format

6.1.10.1 Rating Region Count

rating_region_count — A 6-bit unsigned integer value that indicates the number of rating region specifications to follow in the message.

6.1.10.2 Rating Region

rating_region — An unsigned 8-bit integer that specifies the rating region for which the data in the three bytes to follow is defined. The encoding of the rating_region parameter is shown in Table 6.1.

Table 6.1. Rating Region Encoding

rating_region	meaning
0x00	United States
0x01-0xFF	Reserved

6.1.10.3 Program Ratings

Figure 6.3 defines the program_ratings() data structure.

	Bits	Bytes	Description
program_ratings() {			
rating_dimension_1	4	(1)	uimsbf { (see text) }
rating_dimension_2	4		uimsbf { (see text) }
rating_dimension_3	4	(1)	uimsbf { (see text) }
rating_dimension_4	4		uimsbf { (see text) }
rating_dimension_5	4	(1)	uimsbf { (see text) }
rating_dimension_6	4		uimsbf { (see text) }
}			

Figure 6.3. Program Ratings Format

rating_dimension_1..6 — Each of these six 4-bit fields represents a different dimension of parental rating control for the region given in rating_region**.** The format for each is basically the same, in that value zero means the program has no restrictions within that dimension, or the dimension is undefined, and levels above zero represent increasing levels of rated content within the dimension. Each dimension can have up to fifteen levels.

6.1.10.4 Rating Description Text

rating_description_length — An 8-bit unsigned integer value in the range zero to 80 that represents the length of the rating_description() field to follow.

rating_description() — A text string, in the standard multilingual format, describing the program's ratings, for the rating region indicated in rating_region**.** An example rating text string might be "R (MV, AL)."[9] Multilingual text strings are formatted according to the rules outlined in Section 8. If the rating_description_length field is zero, a rating_description() is not present in the message for the indicated region. Format effectors, as described in Section 8.3, shall not be present in the rating_description().

[9] Subsequent to the development of this standard, a voluntary parental advisory system was defined in Reference [17].

6.1.11 Program Record

Figure 6.4 shows the format of the program_record() structure.

	Bits	Bytes	Description
program_record() {			
epoch_end_time	32	4	uimsbf GPS seconds
interstitial_duration	16	2	uimsbf units: seconds
free_preview_duration	16	2	uimsbf units: seconds
IPPV_window_duration	16	2	uimsbf units: seconds
service_provider_ID	16	2	uimsbf
name_display_time	8	1	uimsbf units: tenths of seconds
if (NVOD_data_included) {			
stagger_start_time	16	(2)	uimsbf units: seconds
reserved	2	(1)	bslbf reserved
theater_count	6		uimsbf range: 2-63
last_showing	1	(1)	bslbf {no, yes}
reserved	1		bslbf reserved
highest_theater	6		uimsbf range: 0-theater_count
}			
reserved	5	1	bslbf reserved
package_count	3		uimsbf range 0-7
if (package_count>0) {			
package_availability	8	(1)	uimsbf
package_special_event	8	(1)	uimsbf
}			
for (I=0; i<package_count; i++) {			
package_provider_ID	8	(1)	uimsbf
package_ID	16	(2)	uimsbf
}			
cost_string_record()	*	(*)	
}			

Figure 6.4. Program Record Format

6.1.11.1 Epoch End Time

epoch_end_time — A 32-bit unsigned integer field providing the second of the scheduled end of the program epoch, measured as the number of GPS seconds since January 6, 1980 00:00:00.

6.1.11.2 Interstitial Duration

Interstitial duration provides the time duration between the scheduled start of the program epoch and the scheduled start of the actual program itself. The interstitial time is typically used for promotional material, and is provided as a parameter to the Decoder for use in construction of text screens that reference time to (or since) the actual start of the program. The parameter is also used in determining which of a group of NVOD programs will next reach its starting point.

interstitial_duration — A 16-bit unsigned integer number that represents the number of seconds of time between the scheduled start of the program epoch and the actual start of the program.

6.1.11.3 Free Preview Duration

free_preview_duration — A 16-bit unsigned integer number that provides the number of seconds between the scheduled start of the program epoch and the scheduled start of the pay portion of the program. The free preview usually includes the interstitial period[10]. The value is provided to the Decoder for use in construction of text strings related to free preview, such as "FreeView time left: 12 min."

6.1.11.4 IPPV Window Duration

IPPV_window_duration — A 16-bit unsigned integer that specifies the time in seconds from the start of the program epoch that IPPV purchases are allowed (given that other access control requirements are met). A value of zero would prevent IPPV purchases from occurring, were it to be specified.

6.1.11.5 Service Provider ID

service_provider_ID — A 16-bit unsigned integer value that defines the service provider associated with the program for purposes of impulse pay per view. The service_provider_ID is part of the purchase record reported in store and forward pay-per-view implementations. The values of program_provider_IDs are assigned by the IPPV system operator in order to be unique throughout each closed system. One business entity will have more than one service_provider_ID assigned if more than one purchasable program may be broadcast simultaneously by one provider.

Value 0x0000 for service_provider_ID is reserved to signify "not applicable." The zero value is used when the program is not offered for pay-per-view.

6.1.11.6 Name Display Time

name_display_time — An 8-bit unsigned integer in the range 10 to 255 that represents the time, in units of tenths of one second that the program name "banner" display shall be displayed.

6.1.11.7 NVOD Data Overlay

The NVOD data overlay is used to deliver information pertaining to NVOD services. Refer to Annex B, Section 4 for a discussion of NVOD.

The Decoder processes data in the NVOD data overlay to determine which service in the NVOD group should be selected, whether during initial acquisition of the NVOD

[10] That is, free preview duration is usually longer than interstitial duration.

program, or return to it after prior purchase. The contents of the message are also used to determine appropriate actions to take when the user requests pause, jump forward, and jump backwards playback functions.

stagger_start_time — An unsigned 16-bit integer number of seconds in the range one to 3600 that indicates the repeat interval for the NVOD group. A two-hour movie, for example, might be offered starting on the half-hour. The stagger start time interval would be 30 minutes for the four programs in the NVOD group. The Decoder uses the stagger_start_time to determine which program in the group will next reach its starting point.

theater_count — A 6-bit unsigned integer number in the range two to 63 that defines the number of "theaters" that are part of the NVOD group of which this program is a part.

last_showing — A Boolean flag that indicates, when true, that the instance of the program described by this PROGRAM INFORMATION message is the last showing of this particular program on this service. When the flag is false, the next epoch's program is another showing of this same program.

highest_theater — A 6-bit unsigned integer value used to control NVOD access during the transition period between one program (movie) and the next. Setting of the highest_theater allows the program provider to selectively disallow purchases of NVOD programs too close to the boundary between one program offering and another. Purchasing close to the boundary limits the user's ability to use the VCR-like functions, specifically (in this case) pause and jump back. The Decoder uses the highest_theater to determine whether or not to grant initial access to an instance of a program when the instance in theater zero is the last showing.

If T is defined as the theater number where the program of interest will next start the soonest, the rule used by the Decoder in determining access is: if the program in theater zero is the last showing, then purchase is allowed if T is less than or equal to highest_theater. For example, a highest_theater of zero means that, for the last showing, the only instance of the movie allowed for purchase is in theater zero. If highest_theater is 4, any instance in theaters zero through four may be purchased as the initial purchase.

If purchase is disallowed because of the application of the highest_theater limit, then the user should be prompted that the next available purchase will be the next-epoch's movie, whenever it starts in theater zero. When a transition from one movie (program) title to another is made in the theater complex, the first showing of the new feature shall be made in theater zero.

6.1.11.8 Program Package Data

The program record includes information defining, for the program identified by program_number and program_epoch_number, which program packages (if any) may be available for impulse purchase. Certain program package IDs may be associated with the program (as reported by the access control subsystem) but not currently be offered for sale. Refer to Annex B, Section 3 for a discussion of program packages.

package_count — A 3-bit unsigned integer number in the range zero to seven that indicates the number of packages that may be available for pay-per-view purchase in association with this program. Actual authorization for purchase is determined within the access control subsystem. If the package_count is zero, package costs, package_provider_ID and list of package_IDs are omitted from the program_record.

package_availability — A byte containing seven Boolean values (bit 7 is reserved) that each indicates whether the associated package is available for purchase in the program being defined by this message. Bit zero refers to the first package in the list, bit 1 to the second, and so on. If the bit is true, the package is available; if false, the package is unavailable for new purchasers. Note: the Decoder may use the existence of a non-null cost string for the associated package as evidence of package availability. The package_availability field may be disregarded if this method is used.

package_special_event — A byte containing seven Boolean values (bit 7 is reserved) that each indicates whether a pay-per-view purchase of the associated package initiates a *special event* reportback (if the Decoder supports and is enabled for reportback via modem of PPV events). Bit zero refers to the first package in the list, bit 1 to the second, and so on. If the bit is true, the purchase of the package triggers a reportback in the next available reportback window; if false, purchase of the package does not affect the reportback algorithm.

package_provider_ID — An 8-bit unsigned integer number that identifies the entity defining the package_ID values. The package_provider_ID taken together with a package_ID uniquely identify a program package within the system described in this Standard. All package_IDs in the list are associated with the one package_provider_ID given.

package_ID — A 16-bit tag used to group programs into packages. The grouping of programs into packages is managed by assigning each a common package_ID.

6.1.12 Cost String Record

The PROGRAM INFORMATION message delivers textual representations of the costs of the program and any packages of which the program is a member. The message may also define the textual cost to buy the program and also have the right to tape it. To support the simultaneous offering of one program to regions using differing currency standards, cost strings may be supplied for several different currency regions. The cost_string_record() structure is described in Figure 6.5.

	Bits	Bytes	Description
cost_string_record() {			
reserved	2	1	bslbf reserved
cost_string_count	6		uimsbf range 0-63
for (i=0; i<cost_string_count; i++) {			
currency_region	8	(1)	uimsbf
cost_string_length	8	(1)	(L) max. 16
cost_string()	8*L	(L)	
}			
reserved	2	1	bslbf reserved
override_cost_string_count	6		uimsbf range 0-63
for (i=0; i<override_cost_string_count; i++) {			
override_currency_region	8	(1)	uimsbf
override_cost_string_length	8	(1)	uimsbf (D) max. 16
override_cost_string()	8*D	(D)	
}			
for (i=0; i<package_count; i++) {			
reserved	2	(1)	bslbf reserved
package_cost_string_count	6		uimsbf range 0-63
for (i=0; i<package_cost_string_count; i++) {			
package_currency_region	8	((1))	uimsbf
package_cost_string_length	8	((1))	uimsbf (S) max. 16
package_cost_string()	8*S	((S))	
}			
}			
}			

Figure 6.5. Cost String Record Structure

6.1.12.1 Program Cost Strings

cost_string_count — A 6-bit unsigned integer value in the range zero to 63 that identifies the number of currency regions for which program cost strings are defined in this message.

currency_region — An 8-bit unsigned integer value identifying the currency region to which the program cost information to follow applies.

cost_string_length — An 8-bit unsigned integer value in the range one to 16 that defines the length in bytes of the cost_string() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

cost_string() — A data structure containing a multilingual text string defining the program cost for the currency region identified in currency_region. Text strings are formatted according to the rules outlined in Section 8. Format effectors, as described in Section 8.3 shall not be present in the cost_string().

6.1.12.2 Override Cost Strings

The override cost strings and counts are formatted the same as the regular program cost strings and counts, except that they define the cost of the program including right to tape.

6.1.12.3 Package Cost Strings

package_cost_string_count —A 6-bit unsigned integer value in the range zero to 63 that identifies the number of currency regions for which package cost strings are defined in this message.

package_currency_region — An 8-bit unsigned integer value identifying the package currency region to which the package cost information to follow applies.

package_cost_string_length — An 8-bit unsigned integer value in the range zero to 16 that defines the length in bytes of the package_cost_string() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

package_cost_string() — A data structure containing a multilingual text string defining the program cost for the currency region identified in currency_region. Text strings are formatted according to the rules outlined in Section 8. Format effectors, as described in Section 8.3, shall not be present in the package_cost_string().

6.1.13 Message-End Descriptors

The message may include at its end one or more structures of the form tag, length, data. The number of descriptors present is determined indirectly by processing the section_length field.

6.2 Program Name Message

The PROGRAM NAME message is a language-tagged message that defines the program and package names associated with a given program epoch. The names may be referenced by scripts defining program-related user screens, or (in case the unit does not support scripting) may be used to construct on-screen display data directly. The name string is also used as the reference name when a program or package is impulse purchased—it is the name that appears in the view history record.

Figure 6.6 defines the format of the PROGRAM NAME message. The parameters in this message that are common to all messages are defined in Reference [1] and Reference [2].

	Bits	Bytes	Description
program_name_message() {			
table_ID	8	1	uimsbf value 0xC1
zero	2	2	bslbf
ISO_reserved	2		bslbf
section_length	12		uimsbf
zero	3	1	bslbf zero
protocol_version	5		see section 4.1.6
ISO_639_language_code	24	3	uimsbf
program_number	16	2	uimsbf
reserved	8	1	bslbf reserved
sequence	8	1	uimsbf
program_epoch_number	8	1	uimsbf
display_name_when_not_auth	1	1	bslbf {no, yes}
use_alt_name_in_purchase_history	1		bslbf {no, yes}
use_alt_name_if_not_auth	1		bslbf {no, yes}
display_ratings	1		bslbf {no, yes}
reserved	4		bslbf reserved
program_name_length	8	1	uimsbf (N)
program_name()	8*N	N	uimsbf
alternate_program_name_length	8	1	uimsbf (A)
alternate_program_name()	8*A	A	
reserved	5	1	bslbf reserved
package_count	3		uimsbf range 0-7
for (I=0; i<package_count; i++) {			
package_name_length	8	(1)	uimsbf (P)
package_name()	8*P	(P)	
}			
for (I=0; i<N; i++) {			
descriptor()	*	(*)	optional
}			
CRC_32	32	4	rpchof
}			

Figure 6.6. Program Name Message Format

6.2.1 Table ID

The **table_ID** of the PROGRAM NAME message shall be 0xC1.

6.2.2 ISO 639 Language

ISO_639_language — A 3-byte language code per ISO 639/2 Part B defining the language associated with the text carried in this PROGRAM NAME message. The ISO_639_language_code field contains a three character code as specified by ISO 639.2/B. Each character is coded into 8 bits according to ISO 8859-1 (ISO Latin-1) and inserted in order into the 24 bit field. The value 0xFFFFFF shall be used in case the text is available in one language only. In the Decoder, value 0xFFFFFF shall represent a "wild card" match when filtering by language.

6.2.3 Program Number

program_number — a 16-bit unsigned integer number that associates the program name information carried in this message with a service referenced by the Program Association Table and TS_program_map_section.

6.2.4 Sequence

sequence — An 8-bit unsigned integer value in the range zero to 255 used to indicate changes in the associated program_name_data() structure. If the Decoder has stored name data with sequence X, it may ignore subsequent messages for the same program epoch also tagged with sequence X. The sequence number is only unique within messages of a given language. I.e. each language may be delivered using an independent sequence numbering scheme.

6.2.5 Program Epoch Number

program_epoch_number — An 8-bit counter of program epochs. The program_epoch_number parameter is used to associate status reported by the access control subsystem with status generated by the Decoder for a given program. Program epoch number also ties the program described in the PROGRAM INFORMATION message together with name data in given in the PROGRAM NAME message.

6.2.6 Display Options

The following fields are provided as a means to control the appearance of on-screen displays relating to program name information.

display_name_when_not_auth — A Boolean flag that indicates, when set, that the program name may be displayed when the program is not authorized. Some programmers wish to avoid displaying the names of programs not available; when the flag is zero, the program name shall be suppressed when not authorized.

use_alt_name_in_purchase_history — A Boolean flag that indicates, when set, that the alternate program name (defined in the PROGRAM NAME message) shall be used to record the purchase of this program, instead of the regular program name. When the flag is false, the regular name shall be used if the program is impulse-purchased.

use_alt_name_if_not_auth — A Boolean flag that indicates, when set, that the alternate program name shall be used instead of the regular name for all program name displays (such as the name displayed upon acquisition of the program) when the authorization state for that program does not enable access—for example in the "no subscription" case, or in the case of an IPPV program that has not yet been purchased. When the flag is clear, the regular name shall be used in these cases. If display_name_when_not_authorized is false, no name (alternate or otherwise) shall be displayed in the "not authorized" case.

display_ratings — A Boolean flag that indicates, when set, that the program ratings text may be displayed as appropriate. When the flag is clear, ratings information shall be suppressed.

6.2.7 Program Name

program_name_length — An 8-bit unsigned integer value in the range zero to 80 that defines the length in bytes of the program_name() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

program_name() — A data structure containing a multilingual text string defining the program name. Text strings are formatted according to the rules outlined in Section 8. The name field consists of one or more <mode><length> blocks. Format effectors, as described in Section 8.3 shall not be present in the program_name().

6.2.8 Alternate Program Name

Some programmers wish to suppress the actual program name under certain circumstances, including its use in the view history display for IPPV, and for display upon channel acquisition (VIEW displays). The Decoder shall record the alternate name in the view history stack in place of the regular program name if the use_alt_name_in_purchase_history flag is set in the PROGRAM INFORMATION message. The Decoder shall display the alternate name in place of the regular name for VIEW displays when not authorized, if the use_alt_name_if_not_auth flag is set.

alternate_program_name_length — An 8-bit unsigned integer value in the range zero to 80 that defines the length in bytes of the alternate_program_name() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

alternate_program_name() — A data structure containing a multilingual text string defining the "alternate" program name. The field consists of one or more <mode><length> blocks. Format effectors, as described in Section 8.3 shall not be present in the alternate_program_name().

6.2.9 Package Count

package_count — A 3-bit unsigned integer value in the range zero to 7 that indicates the number of package name blocks to follow in the message. If no package name data is provided in the message, package_count is zero.

6.2.10 Package Name

package_name_length — An 8-bit unsigned integer value in the range zero to 80 that defines the length in bytes of the package_name() field to follow. The length represents the sum total of all <mode><length> blocks comprising the multilingual text string to follow.

package_name() — A data structure containing a multilingual text string defining the package name. The name field consists of one or more <mode><length> blocks. Format effectors, as described in Section 8.3 shall not be present in the package_name().

6.2.11 Message-End Descriptors

The message may include at its end one or more structures of the form tag, length, data. The number of descriptors present is determined indirectly by processing the section_length field.

7 Descriptors and Stream Types

7.1 Descriptor Tag Definitions

Table 7.1 defines descriptor tag values and ranges as used by this Standard. Descriptor structures for which the tag value is unknown to the Decoder are expected to be disregarded.

Table 7.1. Descriptors

Tag	Descriptor Name	Where Used	Reference
0x80-0xBF	**Standard**		
0x80	stuffing_descriptor()	Network, PMT	7.3.3
0x81	AC3_audio_descriptor()	PMT	See Reference [1]
0x82	frame_rate_descriptor()	PMT	7.3.4
0x83	extended_video_descriptor()	PMT	7.3.5
0x84	component_name_descriptor()	PMT	7.3.6
0x90	frequency_spec_descriptor()	VCT, TDT	7.2.1
0x91	modulation_params_descriptor()	VCT	7.2.2
0x92	transport_stream_ID_descriptor()	VCT, TDT	7.2.3

7.2 Descriptors – Network PID

Three descriptors are standardized for use in the tables defined in the Network Information message: the frequency_spec_descriptor(), the modulation_params_descriptor(), and the transport_stream_ID_descriptor().

7.2.1 Frequency Specification Descriptor

The frequency_spec_descriptor() is defined for use in the Transponder Data Table record and in the satellite Virtual Channel Table record. Figure 7.1 defines the format of the frequency_spec_descriptor(). The tag type shall be 0x90. The carrier frequency given in the frequency_spec_descriptor() shall override any carrier frequency given by regular means. If the frequency_spec_descriptor() is present in the TDT record, the CDT_reference shall be disregarded. If the frequency_spec_descriptor() is present in the VCT record, the CDT_reference given in the TDT record indicated by the channel's satellite and transponder references shall be disregarded.

	Bits	Bytes	Description
frequency_spec_descriptor() {			
descriptor_tag	8	1	value 0x90
descriptor_length	8	1	uimsbf
frequency_unit	1	2	bslbf see Table 7.2
carrier_frequency	15		uimsbf units: 10 or 125kHz
}			

Figure 7.1. Frequency Specification Descriptor Format

frequency_unit — A 1-bit field identifying the units for the carrier_frequency field. Table 7.2 defines the coding for frequency_unit.

Table 7.2. Frequency Unit

frequency_unit	meaning
0	10 kHz units
1	125 kHz units

carrier_frequency — A 15-bit unsigned integer number in the range zero to 32,767 that defines the carrier frequency for the transponder, in units of either 10 kHz or 125 kHz, depending on the value of frequency_unit. When the frequency_unit indicates 125 kHz, the carrier_frequency can be interpreted as a fractional frequency (1/8 MHz) in the least-significant 3 bits, and an integer number of megahertz in the upper 12 bits. The range of frequencies that can be represented is zero to ((2^15) -1) * 125 kHz = 4095.875 MHz.

7.2.2 Modulation Parameters Descriptor

In some satellite applications, multiple carriers are present on one physical transponder. One or more of these carriers may be modulated using different parameters than those defined in the TDT for the indicated transponder. The modulation_params_descriptor() may be placed in the virtual channel record to define parameters in use for the given carrier. When present, the descriptor overrides modulation parameters provided indirectly through reference to the TDT and MMT. If the modulation_params_descriptor() is present in the VCT record, the MMT_reference given in the TDT record indicated by the channel's satellite and transponder references shall be disregarded.

Figure 7.2 defines the format of the modulation_params_descriptor(). The fields are as defined for the MMT_record() defined in Section 5.1.4. The tag type shall be 0x91.

	Bits	Bytes	Description
modulation_params_descriptor() {			
descriptor_tag	8	1	value 0x91
descriptor_length	8	1	uimsbf
transmission_system	4	1	uimsbf See Table 5.6
inner_coding_mode	4		uimsbf See Table 5.7
split_bitstream_mode	1	1	bslbf {combined, split}
reserved	2		bslbf reserved
modulation_format	5		uimsbf See Table 5.8
reserved	4	4	bslbf reserved
symbol_rate	28		uimsbf units: one symbol per sec
}			

Figure 7.2. Modulations Parameters Descriptor Format

7.2.3 Transport Stream ID Descriptor

Figure 7.3 defines the format of the transport_stream_ID_descriptor(). The tag value shall be 0x92. The transport_stream_ID_descriptor() may appear within the satellite_virtual_channel() record or within the TDT_record().

The transport_stream_ID_descriptor() acts to associate the transponder or virtual channel with an MPEG-2 Transport Stream by identifying the transport_stream_ID of that Transport Stream (TS). According to MPEG-2 syntax, the transport_stream_ID is indicated in the Program Association Table carried in PID 0 of any MPEG-2-compliant multiplex bitstream. In the normal case, accessing a target TS by using knowledge of its frequency is sufficient. In cases where multiple carriers are present spaced closely together however, knowledge in the receiver of the target transport_stream_ID is desirable because it increases confidence in the receiver that the correct carrier has been acquired.

	Bits	Bytes	Description
transport_stream_ID_descriptor() {			
descriptor_tag	8	1	value 0x92
descriptor_length	8	1	uimsbf
target_transport_stream_ID	16	2	uimsbf
}			

Figure 7.3. Transport Stream ID Descriptor Format

target_transport_stream_ID — A 16-bit unsigned integer quantity that identifies the target MPEG-2 transport_stream_ID of the Transport Stream associated with the virtual channel or transponder record in which this descriptor appears.

7.3 Descriptors – PMT PID

This section describes descriptors that may appear in the PMT to help define digital services offered on cable and satellite.

7.3.1 Registration Descriptor

The registration_descriptor() is defined in section 2.6.8 of the *ISO/IEC 13818-1* standard (Reference [8]). Programs that conform to this specification will be identified by the 32-bit identifier in the section of the Program Map Table detailed in Section 2.4.4.8 of *ISO/IEC 13818-1*. The identifier will be coded according to Section 2.6.8.

7.3.2 ISO 639 Language Descriptor

The ISO_639_language_descriptor() is defined in MPEG-2 Systems (Reference [8]), and indicates the language or languages associated with the service or a component of the service.

7.3.2.1 ISO 639 Language Descriptor—Service Level

When placed at the service level, the *ISO/IEC 13818*-1 ISO_639_language_descriptor() indicates the languages in which textual program information (for example, PROGRAM NAME messages) are available within the service. The descriptor contains N ISO-defined 24-bit language codes.

When the ISO_639_language_descriptor() is used to describe languages available for textual program information, the audio_type field may be disregarded.

7.3.2.2 ISO 639 Language Descriptor—Component Level

At the component level, the ISO_639_language_descriptor() shall be present on all audio and subtitle streams to define the language of these elementary streams carried in Transport Streams constructed in accordance with this Standard.

In the context of the definition of an elementary audio stream, the ISO_639_language_descriptor() indicates either one or two languages. If the component is stereo, a single language is indicated. If the component is formatted as two monophonic tracks (mono-mono), two languages are indicated. The first language indicates the language of the left audio track, the second the right track. In the case that one of the monophonic tracks is undefined or unused, the value of zero ("unknown or undefined") is used. The presence of two language codes thus indicate mono-mono audio format.

In the context of the definition of a subtitle stream, the ISO_639_language_descriptor() indicates one or more languages for which subtitle data is available within that elementary stream. Multilingual subtitle data may be combined into a single stream when common access control is applied.

7.3.2.3 Audio Type for Subtitle Application

The MPEG-2 Systems document defines an audio_type field associated within each instance of ISO_639_language_code in the "for" loop of the ISO_639_language_descriptor(). The "undefined" value (0x00) shall be used for regular audio and subtitling. Value 0x02, "hearing impaired," may be used as well to signify audio or subtitling tailored to the needs of hearing impaired viewers.

7.3.3 Stuffing Descriptor

For certain applications it is necessary to define a block of N bytes as a placeholder. The N bytes themselves are not to be processed or interpreted. The stuffing_descriptor() is specified for this purpose. The stuffing_descriptor() is simply a descriptor type for which the contents are to be disregarded. The tag type for the stuffing descriptor shall be 0x80. The stuffing_descriptor() may appear where descriptors are allowed in any standard message defined in this System Information Standard.

7.3.4 Frame Rate Descriptor

The frame_rate_descriptor() may be used to identify the frame rate associated with video components. The format of the frame_rate_descriptor() is defined in Figure 7.4.

	Bits	Bytes	Description
frame_rate_descriptor() {			
descriptor_tag	8	1	uimsbf value 0x82
descriptor_length	8	1	uimsbf
multiple_frame_rate_flag	1	1	bslbf {false, true}
frame_rate_code	4		uimsbf
reserved	3		bslbf
}			

Figure 7.4. Frame Rate Descriptor Structure

multiple_frame_rate_flag — A binary flag that indicates, when set, that multiple frame rates may be present in the video stream. When set to a value of zero, only a single frame rate is present.

frame_rate_code — A 4-bit value indicating the frame rate present in the video stream. When multiple_frame_rate_flag is true, additional frame rates may also be present. Table 7.3 defines the encoding.

Table 7.3. Frame Rate Code Values

frame_rate_code	Meaning	Also Includes
'0000'	forbidden	
'0001'	24.000 ÷ 1.001 = 23.976...	-
'0010'	24.000	23.976
'0011'	25.000	-
'0100'	30.000 ÷ 1.001 = 29.97...	23.976
'0101'	30.000	23.976, 24.000, 29.97
'0110'	50.000	25.000
'0111'	60.000 ÷ 1.001 = 59.94...	23.976, 29.97
'1000'	60.000	23.976, 24.000, 29.97, 30.000, 59.94
'1000'-'1111'	Reserved	

7.3.5 Extended Video Descriptor

The extended_video_descriptor() may be present in conjunction with a video elementary component to flag certain attributes that may be needed for processing. Figure 7.5 describes the format of the extended_video_descriptor().

	Bits	Bytes	Description
extended_video_descriptor(){			
descriptor_tag	8	1	uimsbf value 0x83
descriptor_length	8	1	uimsbf
catalog_mode_flag	1	1	bslbf {false, true}
video_includes_setup	1		bslbf {false, true}
reserved	6		bslbf
}			

Figure 7.5. Extended Video Descriptor Format

catalog_mode_flag — A binary flag that indicates, when set, that the associated video stream supports applications that select and display (video hold) single frames from the processed bitstream, as referenced by the temporal reference field. When the bit is clear, or the descriptor is not present, the bitstreams are decoded and displayed normally.

video_includes_setup — A binary flag that indicates, when set, that the associated video includes video setup data. When the flag is false, the associated video stream does not include setup. Video setup is the difference between video blanking and video black in PAL and NTSC video standards. The Decoder may decide to remove or insert video setup depending on its presence in the video and the decoder's chosen output video standard[11].

7.3.6 Component Name Descriptor

Figure 7.6 defines the component_name_descriptor(), which serves to define an optional textual name tag for any component of the service. N strings may be provided, each tagged with its associated language via an ISO 639 3-byte language code.

	Bits	Bytes	Description
component_name_descriptor() {			
descriptor_tag	8	1	uimsbf value 0x84
descriptor_length	8	1	uimsbf
reserved	2	1	bslbf
string_count	6		uimsbf range 1-63
for (I=0; i<string_count; I++) {			
ISO-639_language	24	(3)	uimsbf
string_length	8	(1)	uimsbf (L) range 1-31
name_string()	8*L	(L)	
}			
}			

Figure 7.6. Component Name Descriptor Format

string_count — A 6-bit unsigned integer number in the range one to 63 that represents the number of name strings being defined in this descriptor.

ISO_639_language_code — A 3-byte language code per ISO 639.2/B defining the language associated with the name_string() to follow. The ISO_639_language_code field contains a three character code as specified by ISO 639.2/B. Each character is coded into 8 bits according to ISO 8859-1 (ISO Latin-1) and inserted in order into the 24 bit field.

string_length — An 8-bit unsigned integer in the range one to 31 that represents the length in bytes of the multilingual name string to follow.

name_string() — A multilingual text string characterizing the component. Text strings are formatted according to the rules outlined in Section 8.

[11] For NTSC video whose characteristics are defined by SMPTE 170M, video setup is 7.5 IRE.

7.3.7 User Private Descriptors

The use of private descriptors is outside the scope of this document.

7.3.8 Processing of Unknown Descriptor Types

In general, tables defined in Reference [1] and this specification may carry descriptor types that may not be understood by any particular model of Decoder. If a descriptor is encountered whose type is unknown to the Decoder, that descriptor shall be disregarded, by skipping the number of bytes indicated by the length field in the descriptor. All descriptors follow the same structure, which is tag, length, and data.

7.4 Stream Types – PMT PID

The stream_type is an 8-bit field in the PMT that indicates the type of component whose PID and language are being associated with the service. Table 7.4 defines the coding.

Table 7.4. Stream Component Types

Value	Description
0x00-0x7F	Defined in *ISO/IEC 13818-1* Systems (Reference [8])
0x80	DigiCipher® II video[12]
0x81	ATSC A/53 audio (Reference [2])
0x82	Standard subtitle (Reference [12])
0x83-0x85	Reserved - standard
0x86	Cueing (Reference [16])
0x87-0x9F	Reserved - standard
0xA0	IP data
0xA1-0xBF	Reserved – managed by the ATSC code point registry.
0xC0-0xC1	Reserved – User Private
0xC2	Isochronous data (Reference [13])
0xC3	Asynchronous data (Reference [14])
0xC4-0xFF	User Private

User private stream types shall be designated by stream_type values in the range 0xC0 through 0xFF. The owner of a user private stream type shall be identified by an instance of an MPEG-2 registration_descriptor().

When it is used to tag the owner of a user private stream type, in the message syntax the registration_descriptor() follows the stream_type and elementary_PID. If more than one registration_descriptor() is included in the "for" loop for one stream_type, the first one shall indicate the owner of that stream type.

[12] *DigiCipher* is a registered trademark of General Instrument Corporation.

7.5 Ordering of Audio Elementary Streams

The first audio stream listed in the TS_program_map_section is assumed to be the *default* audio for the service. The first audio stream may be monophonic (in that case, the left component shall be considered “first”). The default audio may be chosen by the Decoder if the desired language track is unavailable, and is the preferred track to use with subtitles, since it reflects the native language of the source material (it is the language being spoken by the actors in the program).

7.6 Dynamic Changes to Program Map Table

Changes to the Program Map Table Section occur routinely, for example at program boundaries in which the number of available audio channels (languages) changes. The PMT section should be refreshed at least every five seconds to track changes.

8 Multilingual Character Strings

This section describes the format of multilingual character strings in this System Information Standard.

8.1 General Format

The format of multilingual text strings adheres to the following structure. Items in square brackets may be repeated one or more times:

<mode><length> [<mode><length>]

A **string_length** field always precedes the one or more instances of mode**,** length**,** segment. This field is described in each instance where multilingual text is used, and may be either 8- or 16-bits in length, as appropriate. The value of string_length represents the sum total of all mode**,** length**,** segment blocks comprising the multilingual text string to follow, and serves to indicate the end of the text string structure.

The multilingual text data structure is designed to accommodate the need to represent a text string composed of characters from a variety of alphabets, as well as ideographic characters. Whereas characters could be represented using 16- or 32-bit character codes (as does Unicode (*Reference [6]*)), that form is inefficient and wasteful of transmission bandwidth for strings composed primarily of alphabetic rather than ideographic characters. To accommodate the need to handle Chinese, Japanese, and Korean, modes are defined that allow 16-bit (double byte) character representations in standard formats.

References below to Unicode shall be to the Basic Multilingual Plane (BMP) within that standard (see Reference [6]).

mode — An 8-bit value representing the text mode to be used to interpret characters in the segment to follow. See Table 8.1 for definition. Mode bytes in the range zero through 0x3E select Unicode character code pages. Mode byte value 0x3F selects 16-bit Unicode character coding. Mode bytes in the rage 0x40 through 0xFF represent selection of a format effector function such as *underline ON* or *new line*. If mode is in the range 0x40 to 0x9F, then the length**/**segment portion is omitted. Format effector codes in the range 0x40 through 0x9F involve no associated parametric data; hence the omission of the length**/**segment portion. Format effector codes in the range 0xA0 through 0xFF include one or more parameters specific to the particular format effector function.

length — An 8-bit unsigned integer number representing the number of bytes in the segment to follow in this block.

segment — An array of bytes representing a character string formatted according to the mode byte.

Figure 8.1 describes the format of the **multilingual_text_string()**.

Figure 8.1. Multilingual text string format.

	Bits	Bytes	Description
multilingual_text_string() {			
for (I=0; i<N; i++) {			
mode	8	(1)	uimsbf
if (mode < 0x3F) {			
eightbit_string_length	8	((1))	uimsbf
for (i=0; i<eightbit_string_length; I++) {			
eightbit_char	8	(((1)))	uimsbf
}			
} else if (mode==0x3F) {			
sixteenbit_string_length	8	((1))	uimsbf (even)
for (i=0; i<(sixteenbit_string_length); i+=2) {			
sixteenbit_char	16	(((2)))	uimsbf
}			
} else if (mode >= 0xA0) {			
format_effector_param_length	8	((1))	uimsbf
for (i=0; i<(format_effector_param_length); i++) {			
format_effector_data	8	(((1)))	
}			
}			
}			
}			

8.2 Mode Byte Definition

The mode byte is used either to select a *Unicode* code page from the BMP (exact mapping, or in the case of page zero, an extended mapping as defined herein), or to indicate that the text segment is coded in one of a number of standard double-byte formats. Table 8.1 shows the encoding of the mode byte. Values in the zero to 0x33 range select Unicode code pages.

Value 0x3F selects double-byte forms used with non-alphabetic script systems, where the segment consists of a sequence of 16-bit character codes according to the *Unicode* standard. Byte ordering is high-order byte first, also known as *big-endian*.

Table 8.1. Mode Byte Encoding

Mode Byte	Meaning	Language(s) or Script
0x00	Select Unicode Page 0x00	ASCII, ISO Latin-1 (Roman)
0x01	Select Unicode Page 0x01	European Latin (many)[13]
0x02	Select Unicode Page 0x02	Standard Phonetic
0x03	Select Unicode Page 0x03	Greek
0x04	Select Unicode Page 0x04	Russian, Slavic
0x05	Select Unicode Page 0x05	Armenian, Hebrew
0x06	Select Unicode Page 0x06	Arabic[14]
0x07-0x08	Reserved	-
0x09	Select Unicode Page 0x09	Devanagari[15], Bengali
0x0A	Select Unicode Page 0x0A	Punjabi, Gujarti
0x0B	Select Unicode Page 0x0B	Oriya, Tamil
0x0C	Select Unicode Page 0x0C	Telugu, Kannada
0x0D	Select Unicode Page 0x0D	Malayalam
0x0E	Select Unicode Page 0x0E	Thai, Lao
0x0F	Reserved	-
0x10	Select Unicode Page 0x10	Tibetan, Georgian
0x11-0x1F	Reserved	-
0x20	Select Unicode Page 0x20	Miscellaneous[16]
0x21	Select Unicode Page 0x21	Misc. symbols, arrows
0x22	Select Unicode Page 0x22	Mathematical operators
0x23	Select Unicode Page 0x23	Misc. technical
0x24	Select Unicode Page 0x24	OCR, enclosed alpha-num.
0x25	Select Unicode Page 0x25	Form and chart components
0x26	Select Unicode Page 0x26	Miscellaneous dingbats
0x27	Select Unicode Page 0x27	Zapf dingbats
0x28-0x2F	Reserved	-
0x30	Select Unicode Page 0x30	Hiragana, Katakana
0x31	Select Unicode Page 0x31	Bopomopho, Hangul elem.
0x32	Select Unicode Page 0x32	Enclosed CJK Letters, ideo.
0x33	Select Unicode Page 0x33	Enclosed CJK Letters, ideo.
0x34-0x3E	Reserved	-
0x3F	Select 16-bit Unicode mode	all
0x40-0x9F	Format effector (single byte)	see Table 8.2
0xA0-0xFF	Format effector (with parameter[s])	-

[13] When combined with page zero (ASCII and ISO Latin-1), covers Afrikaans, Breton, Basque, Catalan, Croatian, Czech, Danish, Dutch, Esperanto, Estonian, Faroese, Finnish, Flemish, Firsian, Greenlandic, Hungarian, Icelandic, Italian, Latin, Latvian, Lithuanian, Malay, Maltese, Norwegian, Polish, Portuguese, Provencal, Ghaeto-Romanic, Romanian, Romany, Slovak, Slovenian, Serbian, Spanish, Swedish, Turkish, and Welsh.

[14] Also Persian, Urdu, Pashto, Sindhi, and Kurdish.

[15] Devanagari script is used for writing Sanskrit and Hindi, as well as other languages of northern India (such as Marathi) and of Nepal (Nepali). In addition, at least two dozen other Indian languages use Devanagari script.

[16] General punctuation, superscripts and subscripts, currency symbols, and other diacritics.

8.3 Format Effectors

Mode bytes in the 0x40 to 0xFF range are defined as format effectors. Table 8.2 defines the encoding for currently defined single-byte values. Format effectors in the range 0x40 through 0x9F are self-contained, and do not have a length or data field following them. Format effectors in the range 0xA0 through 0xFF include a multi-byte parameter field. No multi-byte format effectors are currently defined.

8.3.1 Line Justification

Values 0x80, 0x81, and 0x82 signify the end of the current line of displayed text and the beginning of a new line. Value 0x80 indicates that the text is displayed left justified within an enclosing rectangular region (defined outside the scope of the text string). Value 0x81 indicates that the text is displayed right justified. Value 0x82 indicates that the text is centered on the line. The dimensions and location on the screen of the box into which text is placed is defined outside the scope of the text string itself.

Table 8.2. Format Effector Function Codes

Mode Byte	Meaning
0x40-0x7F	reserved
0x80	begin new line, left justify
0x81	begin new line, right justify
0x82	begin new line, center
0x83	italics ON
0x84	italics OFF
0x85	underline ON
0x86	underline OFF
0x87	bold ON
0x88	bold OFF
0x89-0x9F	reserved

8.3.2 Italics, Underline, Bold Attributes

These format effectors toggle *italics*, underline, and **bold** display attributes. The italics, underline, and bold format effectors indicate the start or end of the associated formatting within a text string. Formatting extends through new lines. For example, to display three lines of bold text, only one instance of the *bold ON* format effector is required.

8.3.3 Processing of Unknown or Unsupported Format Effectors

Receivers must discard format effectors that are unknown, or known not to be supported within a specific receiver model. If a parameter value carries an undefined value, that format effector is expected to be discarded.

8.4 Default Attributes

Upon entry to a multilingual text string, all mode toggles (bold, underline, italics) shall be assumed “OFF”.

8.5 Mode Zero

Unicode page zero (U+0000 through U+00FF) includes ASCII in the lower half (U+0000 through U+007F), and Latin characters from ISO 8859-1, *Latin-1*, in U+0090 through U+00FF. This set of characters covers Danish, Dutch, Faroese, Finnish, French, German, Icelandic, Irish, Italian, Norwegian, Portuguese, Spanish and Swedish. Many other languages can be written with this set of letters, including Hawaiian, Indonesian/Malay, and Swahili.

Table 8.3 shows encodings of page zero characters in the range 0x80 through 0x9F (these are undefined within *Unicode*).

Table 8.3. Encodings of Columns 8 and 9 of Mode Zero Latin Character Set

	8	9
0	<RESERVED>	<RESERVED>
1	<RESERVED>	<RESERVED>
2	<RESERVED>	<RESERVED>
3	<RESERVED>	<RESERVED>
4	<RESERVED>	<RESERVED>
5	<RESERVED>	<RESERVED>
6	<RESERVED>	<RESERVED>
7	<RESERVED>	<RESERVED>
8	<RESERVED>	U+2030 — <PER MILLE>
9	<RESERVED>	<RESERVED>
A	<RESERVED>	U+266A — <MUSICAL NOTE>
B	<RESERVED>	<RESERVED>
C	<RESERVED>	U+2190 — <LEFT ARROW>
D	<RESERVED>	U+2191 — <UP ARROW>
E	<RESERVED>	U+2192 — <RIGHT ARROW>
F	<RESERVED>	U+2193 — <DOWN ARROW>

8.6 Supported Characters

Support for specific characters and languages depends upon the specific model of Standard-compatible Decoder. Not all Decoders support all defined character sets or character codes. Use of multilingual text must be predicated on the knowledge of limitations in character rendering inherent in different Decoder models for which text is available.

Annex A

Informative
System Information Overview

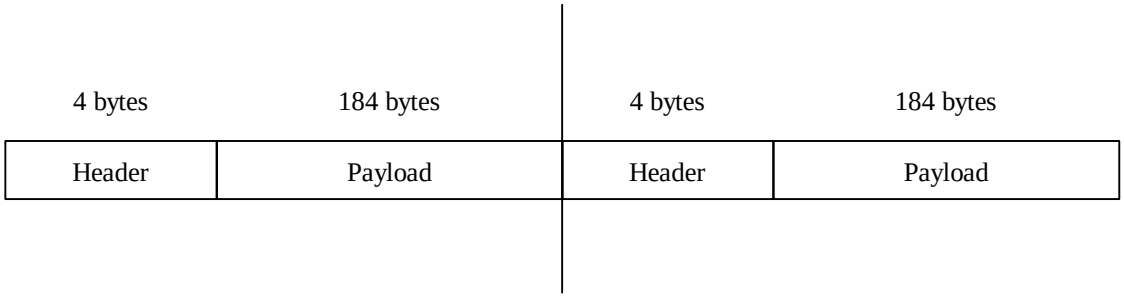
1 System Information Overview

In transmission systems compliant with this System Information Standard, the Transport Stream (TS) delivers services such as audio, video, text and data, as well as control information used for access control and the creation of user interface screens. Transport Streams compliant with this System Information Standard consist of a continuous sequence of fixed-length transport packets, per *ISO/IEC 13818-1*.

The following sections are informative only, and reflect compliance with packet format and transport multiplex methods and formats described in *ISO/IEC 13818-1,* which may be referenced for details.

2 Packet Format

Packets are formatted byte-aligned, and may be considered to be composed of a four-byte header followed by a 184-byte payload portion, as shown below.



For purposes of understanding this SI message stream protocol, the header contents may be simplified to the following items:

- **Packet ID (PID)**—A 13-bit number uniquely identifying the packet contents. A sequence of packets bearing the same PID is called a *stream* or *PID stream*.
- **Encrypted/Clear**—A two-bit field indicating whether the packet payload is encrypted or in the clear, and also indicating packet parity (used to select one of two keys for decryption).
- **Payload unit start indicator**—A bit indicating whether or not the first byte of payload is a pointer into the payload reflecting the start of the next message. A value of zero in the pointer indicates that a message starts immediately following the pointer byte.

Normally, messages are placed back-to-back in the packet payload, with the first byte of another message directly following the last byte of the previous one. If another message is not to start until the following packet, the remainder of the packet must be padded with

bytes of value 0xFF. If the first byte of a message is found to be 0xFF, the remainder of the packet is discarded; at that time the pointer mechanism is used to realign with the start of the next message.

Messages starting in one packet may end in the following packet of the same PID (see Figure A.1); that is, messages may span packets.

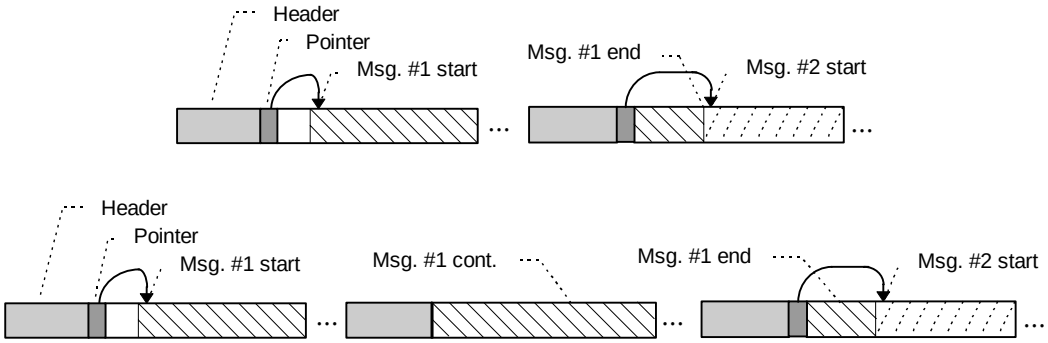


Figure A.1. Messages can span packets

2.1 Service Concept

For the purposes of this System Information Standard, a *service* (MPEG uses the term *program)* is defined as a collection of *elementary* streams, where elementary streams are streams carrying raw compressed video data, compressed audio, subtitles, textual data, or privately defined data channels. Services may consist of several stream components. The most typical is the case in that a service is composed of one video, one or more audio tracks, and zero or more subtitle tracks. A service could consist of just one elementary stream; one single unencrypted data channel is an example. An audio service could consist of one audio stream plus one text information stream.

All the component streams comprising a service are typically access controlled and encrypted together. That is, if access is granted to one component, access is granted to all components of the service. If a service is encrypted, it includes a component that carries the parameters related to access control and keys used for decryption. The encryption/control stream is called the ECM stream because it carries ENTITLEMENT CONTROL messages, or ECMs.

Elementary component streams are illustrated in Figure A.2.

A service may be defined such that certain components are access controlled differently from others. For example, a data channel may accompany a baseball game and provide player statistics. The data channel stream may be encrypted and access controlled differently from the video/audio components.

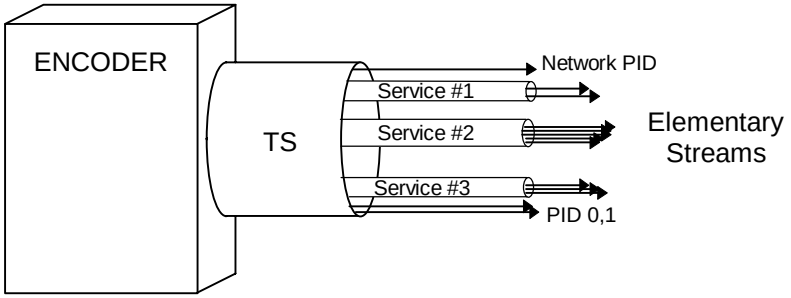


Figure A.2. Elementary component streams

2.2 Stream Types

In Transport Streams compliant with this System Information Standard, the following types of streams are present:

- **Network Stream** — The Network Stream is carried on the Network PID in every TS, and carries data distributed system-wide. The Network Stream carries the NETWORK INFORMATION message, which defines frequency plans, modulation mode tables, satellite and transponder data, and name tables. In addition, the Network Stream carries the VIRTUAL CHANNEL message used to manage access to services for user selection. The Network Stream also carries the SYSTEM TIME message, used to synchronize calendar time.
- **PID 0 (Program Association Table) Stream** — PID 0 is defined by the *ISO/IEC 13818-1 Systems* document as the stream carrying the Program Association Table (PAT). This is the only data structure that may be present on the stream. Every valid Transport Stream has a PID 0. The Program Association Table provides a simplified directory of services on the TS by listing each by MPEG program_number and PID carrying the TS_program_map_section. The PAT also defines the Network PID.
- **PID 1 (CONDITIONAL ACCESS) Stream** — PID 1 is also defined by *ISO/IEC 13818-1*, and carries only the CONDITIONAL ACCESS message, used to define the PID streams carrying Entitlement Management Messages (EMMs) on the TS.
- **Program Map Table Stream** — Every service on the TS has an associated MPEG Program Map Table PID stream. The Program Map Table stream carries the *ISO/IEC 13818-1* TS_program_map_section, which identifies the PIDs, types, and languages for all elementary streams associated with the program. In some applications, the Program Map Table stream also carries program-related messages that deal with user interface functionality, such as text defining the program name or offering the program on a pay-per-view basis.
- **ENTITLEMENT CONTROL Message (ECM) Stream** — Entitlement control messages define the program's access requirements.

- **ENTITLEMENT MANAGEMENT Message (EMM) Stream** — ENTITLEMENT MANAGEMENT messages, or EMMs, define access rights for each individual Decoder.
- **Component Streams** — Each service is composed of a number of component streams. Typically these include the video stream, one or more audio streams, one or more subtitle streams, and possibly a data stream. The composition of some services may not include video; an example would be an audio service. One or more subtitle streams or a text stream may be present as well.

2.3 Overview of Relationship Between Streams

Figure A.3 shows the relationship between the Network Stream, the ISO-defined streams, the Program Map Table stream, PID 0 and 1, the EMM streams, and the component streams.

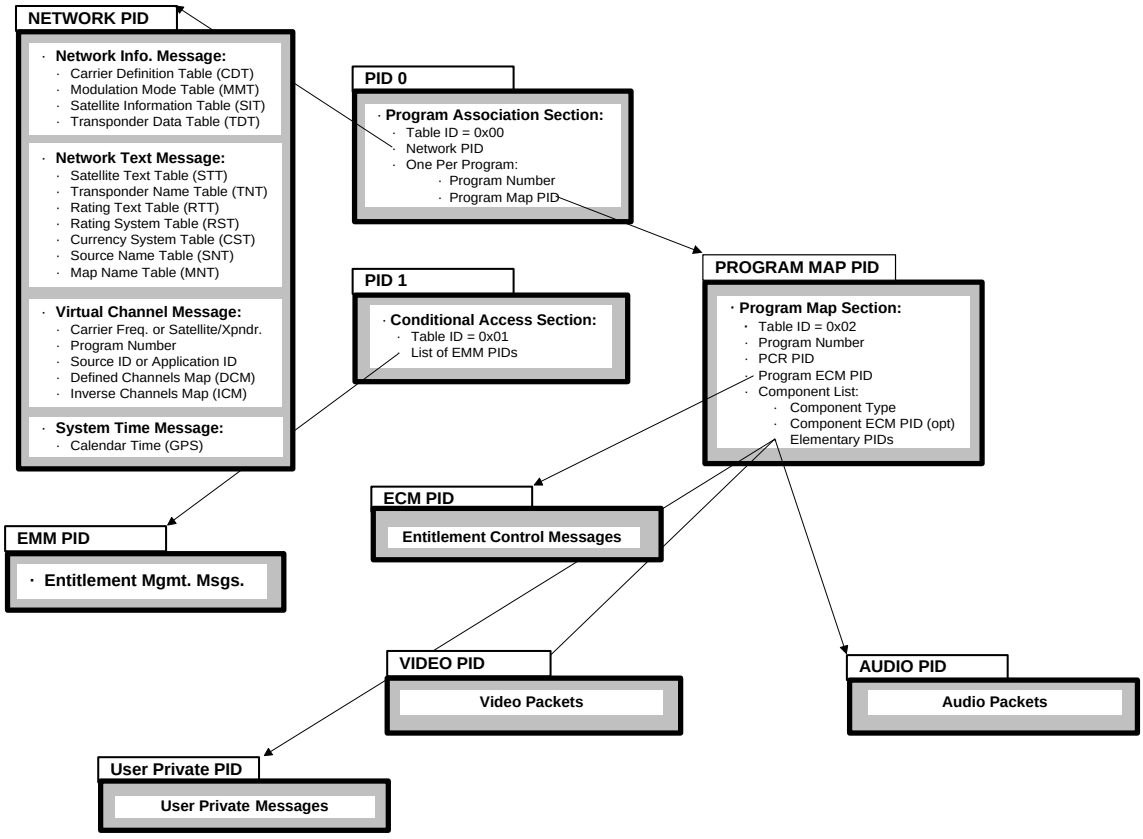


Figure A.3. Transport stream relationships

3 Network Information

Data delivered on the Network Stream defines information of system-wide applicability such as frequency plans, satellite names and orbital locations, and transponder-specific data such as waveform type carried and name. NETWORK INFORMATION, NETWORK TEXT,

and VIRTUAL CHANNEL messages each carry a portion of one of the following types of tables:

- Carrier Definition Table (CDT)
- Modulation Mode Table (MMT)
- Satellite Information Table (SIT)
- Satellite Text Table (STT)
- Transponder Data Table (TDT)
- Transponder Name Table (TNT)
- Virtual Channel Tables (VCTs)
- Ratings Text Tables (RTTs)
- Rating System Table (RST)
- Currency System Table (CST)
- Source Name Table (SNT)
- Map Name Table (MNT)

These tables are described in the following sections.

3.1 Carrier Definition Table (CDT)

The Carrier Definition Table provides a foundation for the definition of frequency plans by defining a set of carrier frequencies appropriate to a particular transmission medium. The CDT is stored in the Decoder as an array of as many as 255 CDT records, each consisting of:

- Carrier frequency, 15 bits, in units of 10 or 125 kHz

Frequencies in excess of 4 GHz can be represented. For satellite carriers, the specified frequency represents the downconverted value. If a carrier on one satellite shares the definition with one on another satellite (as they often will), those two carriers will refer to one common definition (they share a common CDT record).

Changes to the table are rare, and will only occur in conjunction with the launching of new satellites providing carriers at nonstandard frequencies, or introduction of new frequency allocations by the FCC.

3.2 Modulation Mode Table (MMT)

The Modulation Mode Table provides a foundation for quick acquisition of digitally modulated waveforms. A separate MMT is transmitted in Network data for each transmission medium supported by that network. An MMT is stored in the Decoder as an array of up to 255 MMT records, each consisting of:

- Modulation format: QAM, QPSK, BPSK, offset QPSK
- Transmission system: ITU-T (ETSI Standard), ITU-T (North America), ITU-R (ETSI Standard), DigiCipher®, ANSI
- Symbol rate, in units of 1 Hz
- Inner coding mode, expressed as either “none” or an integer ratio such as 1/2 or 3/4
- For QAM modulation, the number of levels

- Flag indicating whether or not the bit stream is split into separate I and Q components

Each MMT contains entries for each modulation mode currently in use by any digital waveform, plus entries for any modes anticipated to be used. As with the CDT, changes to the table are rare.

Parameters defined within the MMT are not specifically manipulated by Decoders compliant with this System Information Standard, but are referenced by the Decoder when attempting to acquire a digitally encoded and modulated waveform.

3.3 Satellite Information Table (SIT)

The Satellite Information Table provides information on each satellite defined in the network, including:

- Orbital position, 11 bits, 0.1° resolution
- Number of transponders/carriers (maximum 64)
- Frequency band, 2 bits: C or Ku (BSS) or Ku (FSS)

Satellite name information is delivered in the Satellite Text Table (STT).

Satellite Integrated Receiver-Decoders (IRDs) may process SIT data to provide the user access to new satellites as they are launched, or to track their new locations if orbital positions are moved.

3.4 Satellite Text Table (STT)

The Satellite Text Table is structured much like the Satellite Information Table, except that it consists of only the abbreviated (reference) name of the satellite (usually two or three characters), as well as the full satellite name. The STT is delivered multilingually, filtered by language in the Decoder.

3.5 Transponder Data Table (TDT)

For each transponder on each satellite, the TDT defines:

- The carrier frequency (in the form of a reference to a satellite CDT)
- Polarization: left/right or horizontal/vertical (as appropriate)
- The transport type: MPEG-2 or not
- For carriers not carrying MPEG-2 transport, the type of waveform carried (NTSC, PAL, or other digital or analog standard)
- The modulation mode data for the transponder, in the form of an MMT reference; applicable only for transponders carrying digitally modulated waveforms.
- An association to a specific VCT, in the form of a VCT ID reference; also applicable for compliant waveforms only
- For clear channels, the audio mode (subcarrier, matrix, mono, digital, narrow, wide, etc.)

Note that, for the sake of completeness, this System Information Standard defines carriers on transponders carrying non-MPEG waveforms, for the benefit of IRDs capable of

processing these waveforms. Such IRDs may make use of network data carried on a Standard MPEG-2 TS to assist them in navigation and processing of all available video services available via satellite, not just those carried on MPEG-2 Transport Streams.

The system defined in this System Information Standard supports multiple carriers per transponder (MCPT) operation, hence the total number of carriers on one satellite can exceed the number of transponders on that satellite. For satellite in which MCPT carriers are defined, the carriers are numbered such that the first 24 (for C band) or 32 (for Ku band) transponders defined correspond to the actual physical transponders. Any additional "transponders" defined are actually additional carriers on one or more of the transponders. Up to 64 transponders/carriers can be defined per satellite. Transponder name information is delivered in the Transponder Name Table (TNT).

3.5.1 Satellite ID

SIT records are delivered tagged with a *satellite ID*, an 8-bit number used to index the stored array. Where other data structures refer to a satellite, the satellite ID is used in place of the satellite name, for convenience and storage efficiency.

3.5.2 Waveform Type

The waveform type indicates the type of waveform primarily carried by the transponder, if known. For those transponders that change waveform type during the course of a day, the waveform type will be less useful.

3.6 Transponder Name Table (TNT)

The transponder name is the textual name associated with the transponder (or MCPT carrier), and can apply to either scrambled or (if the system supports them) clear transponders.

The TNT is delivered multilingually in the NETWORK TEXT message.

3.7 Virtual Channel Table (VCT)

The Virtual Channel Table (VCT) is a data structure consisting of up to 4096 channel definition records. Section 4 of this Annex provides an overview of virtual channels and their purpose and usage. Each VCT is associated with a 16-bit reference ID field. Each record in the VCT consists of:

- For satellite applications, the channel's transponder, indicating which transponder carries the channel
- For terrestrial applications, the channel's carrier frequency (via reference to the CDT) and modulation mode (via reference to the MMT)
- For satellite applications, the channel's satellite ID
- The MPEG program number, associating the virtual channel record with a service defined in the PAT and TS_program_map_sections
- For virtual channels associated with programs carried in a program guide, the *source ID*, a number that may be used to link the virtual channel to entries in the Interactive Program Guide (IPG) database

- For virtual channels used as access paths to application code or data (such as IPG), the *application ID*[17]
- An optional frequency specification descriptor, in case the CDT does not include an entry for the desired frequency (typically used in some satellite MCPT applications)
- An optional descriptor specifying Transport Stream symbol rate, in case the MMT does not include an appropriate rate
- An optional descriptor identifying the MPEG-2 transport stream ID associated with the MPEG-2 Transport Stream carrying the service targeted by the virtual channel

3.7.1 Virtual Channel Table ID

Incoming VCT data is tagged with a 16-bit VCT *ID*, a number identifying the specific VCT to which the data is to be associated.

Many VCTs may be defined in the satellite system, although for terrestrial applications only one will be used per system. If all programmers carrying digital services on a particular satellite could agree to a map definition, there could be just one VCT on that satellite. If programmers do not agree to cooperate, each may define their own VCT.

3.7.2 Transponder

The transponder field in the virtual channel record identifies which transponder carries the virtual channel.

3.7.3 Satellite

For consumer Decoders, VCTs generally do not span satellites; that is, all channels defined in one map are carried on the same satellite. If a particular virtual channel references a satellite other than the root satellite, and that satellite is not available to the receiver without dish movement (that is, it is not co-located), a consumer receiver should treat the channel as unavailable.

Commercial receivers can accept multiple L-band inputs, selected via relay. Satellite references within the VCT accommodate this case.

3.8 Source ID

Source ID is a 16-bit number associated with each program source, defined in such a way that every programming source offered anywhere in the system described in this System Information Standard is uniquely identified. For example, HBO/W has a different assigned source ID than HBO/E, and both are different than HBO-2 or HBO-3. Uniqueness is necessary to maintain correct linkages between an IPG database and

[17] Source ID and application ID need never be defined in the same virtual channel record, therefore they share a common 16-bit field in the stored map. Channels are defined as for “application access” or not; if they are application access, the field defines the application ID, if not, it defines the source ID.

virtual channel tables. See Section 4.9 of this Annex for a discussion of the relationship between **source_ID**, virtual channels, and an IPG database.

3.9 Source Names and Source Name Table (SNT)

The Source Name is a variable length multilingual text string associating a source ID with a textual name. The Source Name Table (SNT) is delivered within the NETWORK TEXT message.

Source name information is delivered in a message format separate from the message containing other information comprising the virtual channel table. Name information is not strictly necessary for channel acquisition, and (depending on the memory management scheme employed in the receiver) may not always be available from memory at acquisition time. Source name information is refreshed often, and is typically available within several seconds of acquisition.

An IPG database may define textual reference names associated with given program sources (referenced by source ID). Such a database may be used to derive virtual channel names in some applications, though in an IPG database the name is generally abbreviated due to display considerations.

Name data is, unlike the regular VCT data, language tagged, so that multilingual source names may be defined. Transmission format for multilingual text is defined to include references to multiple phonetic and ideographic character sets.

3.10 Defined Channels Map (DCM) and Inverse Channels Table (ICT)

For a given Standard-compliant channel, DCM data consists of a series of bytes that, taken as a whole, define which channels in the map are defined, and which are not.

Each Virtual Channel Table has associated with it a table listing **source_ID**s and their associated virtual channel numbers. The **source_ID** values are sorted by value from lowest to highest in the table, to facilitate (using a binary search) lookup of a virtual channel given a source ID.

3.11 Ratings Text Table (RTT)

Programs defined in conformance with this System Information Standard may be characterized by their “ratings” parameters. Ratings information is typically utilized in the Decoder’s user interface to provide parental control over the kinds of programming for which viewing is allowed. A typical rating parameter in the United States is the one provided by the Motion Picture Association of America (MPAA) for particular movie titles. The system described in this System Information Standard supports up to 256 “rating regions,” where a rating region is defined as one subset of the full population of Decoders that conform to a particular standard definition of rating definitions and interpretations.

The MPAA rating is one possible rating dimension; other dimensions are possible. HBO and others, for example, are using a set of program ratings specified in a publication called the *Expanded EDS Program Ratings Specification,* which includes ratings for adult language, and sexual and violence contents.

Decoders designed in conformance with this System Information Standard should adapt to changes in the definition of ratings in a dynamic fashion. The user interface, which allows the user to set ceilings on each of the various dimensions, is table driven from the RTT, so that changes are automatically accommodated. The change might involve addition of a new dimension, addition of a new level in one dimension, or a change of name of any level or dimension.

3.12 Rating System Table (RST)

Each Decoder is assigned to a rating region, either through an installation or user-preference option, via factory configuration, or via a private message individually addressed to the Decoder.

To support installation or user setup dialogs within the Decoder, each rating region is associated with a descriptive text string via the Ratings System Table. A Standard-compliant Decoder may use the RST to offer the user a choice from among several rating regions. The RST defines a list of text strings, each of which is associated (by its index into the table) with a particular rating_region currently in use in the system.

3.13 Currency System Table (CST)

Each Decoder is assigned to a currency region, again either through an installation or user-preference option, via factory configuration, or via a private Decoder-specific message.

The Currency System Table defines a list of text strings, each of which is associated (by its index) with a particular currency_region currently in use in the system. If more than one currency region is supported in a given network, Decoders may use the CST to create a user interface dialog to allow the user to select a preference for currency system. In many applications, there will be a one-to-one correspondence between currency region and country, but such a correspondence is not required.

3.14 Map Name Table (MNT)

The Map Name Table provides a multilingual textual name for a given Virtual Channel Table.

3.15 Overview of Downloaded Tables

Figure A.4 and Figure A.5 show the relationships between the tables of the network data stream for the satellite and cable media cases, respectively.

4 Virtual Channels

Each Standard-compliant Transport Stream may deliver a large number of video, audio, text, and data services. Virtual channels offer the user a consistent view of services available on any particular Standard-compliant TS, and provide a convenient reference mechanism for various combinations of services (such as video with stereo or secondary audio).

Each virtual channel number defines the access point for a particular programming source. The access point for the target programming is described in terms of its Transport Stream location and *ISO/IEC 13818-1* program_number. Transport Stream location is given as satellite and transponder (for satellite IRDs) or its carrier frequency (for cable). The virtual channels concept also supports non-video components such as data, audio-only (radio), and text (or text plus audio).

4.1 Virtual Channel Table

Upon acquisition of an MPEG-2 Transport Stream compliant with this System Information Standard, the Decoder is provided information defining the VCT, a data structure that maps MPEG-2 and non-MPEG-2 services to user channel numbers, and provides the access path to those services. For satellite transmission, the VCT takes the form of a satellite/transponder reference and (for MPEG-2 programs) MPEG program number. For terrestrial applications, the VCT takes the form of a reference to the Carrier Definition Table (CDT), Modulation Mode Table (MMT), and (for MPEG-2 programs) MPEG program number. The Decoder may store one or more VCTs in RAM to allow direct acquisition of any given channel.

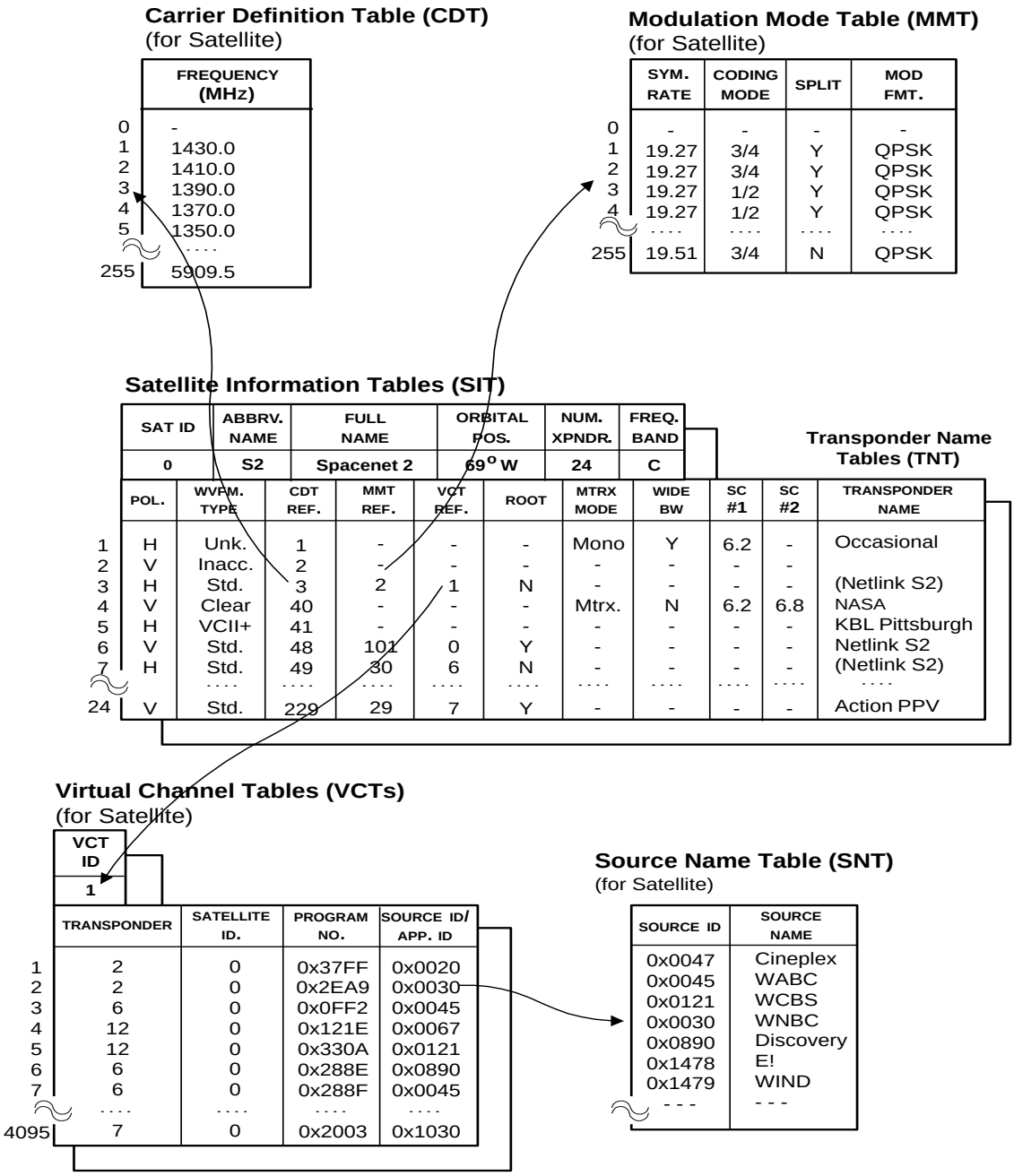


Figure A.4. Network data relationships — satellite case

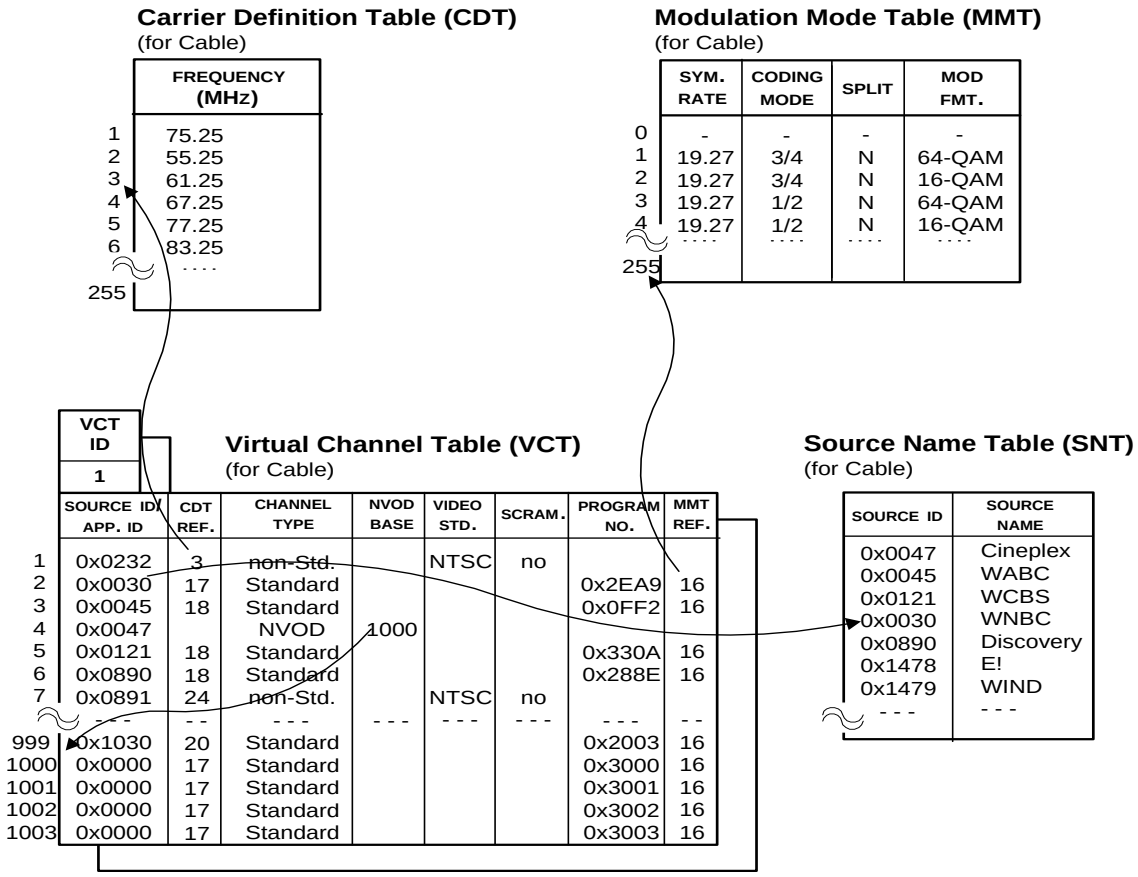


Figure A.5. Network data relationships — cable case

4.2 Defined Channels Map

VCT records are downloaded for defined channels only. Virtual channels for which no definition is available are assumed to be *null* (unavailable) channels. Given that certain channels may cease to exist at some point in time, a mechanism is provided to *undefine* channels no longer in use. The Defined Channels Map (DCM) performs this function by indicating, for a given map, which channels are defined and which are not. The receiver can use this information to eliminate channels no longer in use from the specified VCT.

4.3 Multiple Virtual Channel Tables

In typical cable networks, any single Decoder will store and process one single VCT. The VCT used within Decoders in one cable system must differ from that used in another, of course, if the systems offer different channel lineups or number their channels differently. The C/Ku-band movable dish satellite IRD, on the other hand, must be able to store and process multiple VCTs. The 16-bit VCT_ID is used to differentiate one VCT from another in this System Information Standard.

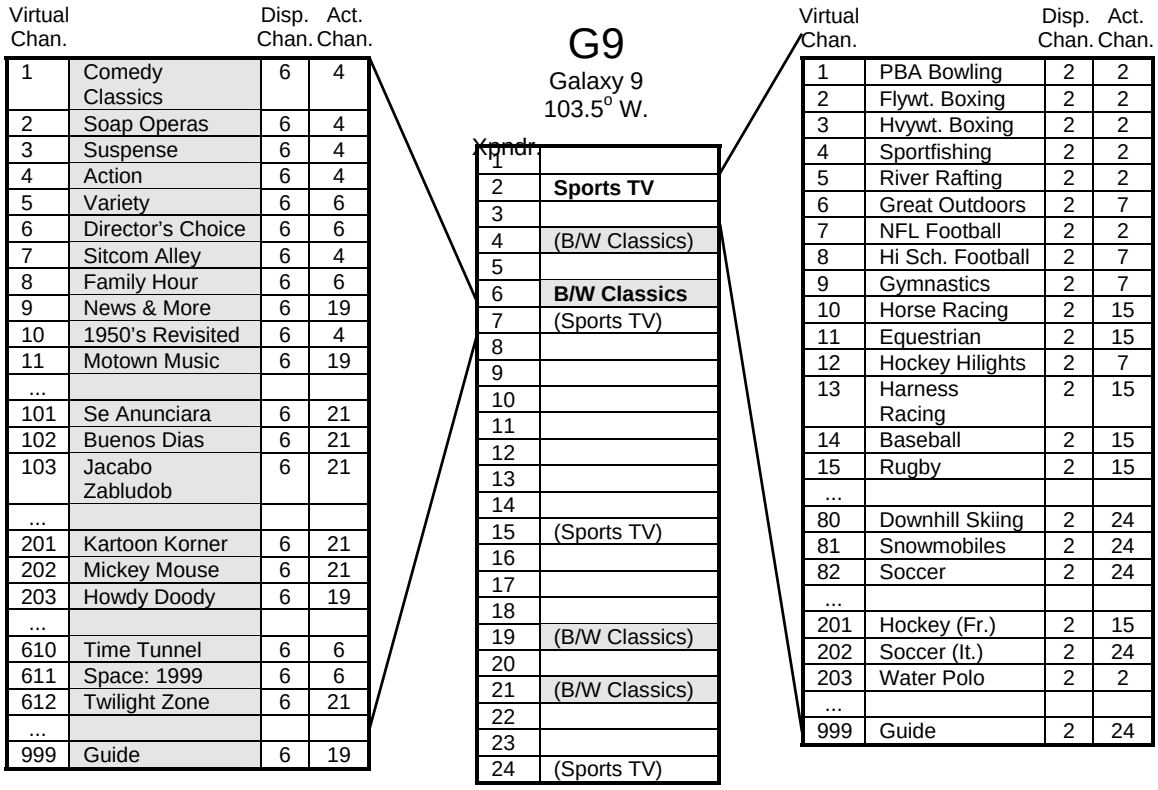


Figure A.6. Satellite virtual channels example

One particular programmer may own and operate several uplinks carried on several transponders, aimed at one particular satellite. With the virtual channel definition, the user can access any of the services carried by any of this programmer's uplinks by selecting its *root* transponder and then identifying the channel number of interest. Hidden from the user, the receiver may actually tune to a different transponder to access the requested virtual channel.

4.3.1 Consumer Satellite Virtual Channel Tables

Consumer C/Ku-band movable dish satellite IRDs navigate using Virtual Channel Tables designed in a way that simplifies the user's view of the world. The user first decides which satellite he wishes to surf. The IRD then determines the one VCT that pertains to that satellite, moves the dish to that satellite (if necessary) and uses the VCT to select digital and analog services.[18]

[18] Past versions of this protocol defined a scheme that supported the notion of multiple consumer VCTs per satellite. This method of navigation is abandoned, as it was determined that it was possible to gain agreement among programmers to combine digital channel offerings into one map per satellite.

Prior to MPEG-2 and digital compression, the typical C-band consumer navigated the sky only by satellites and transponders. For this reason, the first N virtual channels of the satellite VCT are mapped to point to the transponder corresponding to the virtual channel number. For example, access to virtual channel 2 takes the user to transponder 2. If desired, the VCT may be designed so that channels that are inaccessible or are digital multiplexes are skipped. Digital channels are typically mapped into the part of the VCT that carries channels numbered 100 and above.

Figure A.6 shows an example transponder table in which there is a mixture of digital, analog, and unavailable channels. The right side of the figure shows a corresponding Virtual Channel Table constructed according to the above-described scheme. The lowest 24 virtual channels correspond to their like-numbered transponder counterparts. All the digital services are mapped into channels above 100, and have been grouped by category (movies, sports, networks, etc.).

Note that the IRD does not assume anything special about these VCTs; they are used for navigation just like any other channel map. The multi-map consumer satellite IRD does, however, need to find one and only one VCT per satellite, and only one must be sent.

4.4 Changes to a Virtual Channel Table

An important and powerful aspect of the virtual channel idea is the notion that virtual channel definitions can change dynamically. A programmer may want to routinely reconfigure an uplink Encoder to carry more or fewer video services. When this happens, by making changes to the definitions carried in the VCTs, the programmer can cause any Decoder that has acquired a virtual channel that will be moving to retune to the service's new destination. Or, if the service will be unavailable, to a channel carrying an informational text screen.

Using the virtual channel concept, a program provider is free to move services from one physical place to another on the satellite while keeping the virtual channel number constant. For example, a video service may be moved from one transponder or cable carrier to another without changing the user's method of accessing the channel. If the definition of the currently acquired virtual channel changes, the Decoder reacquires using the changed data.

4.5 Hidden Virtual Channels

A virtual channel may be defined as *hidden* if it is necessary to define a channel that is accessible via means other than the usual direct channel number entry or channel up/down (*surfing*). The GUIDE button on the remote control could be, for example, tied to such a hidden channel to provide the access path to the service carrying the guide database (see the following section).

4.6 Access to Application Data Streams via Virtual Channel

Each virtual channel record provides, in addition to the physical location of the associated Transport Stream, and the *ISO/IEC 13818-1* program number within that TS, a 16-bit identifying tag. Depending upon the type of channel, this tag indicates either the channel's associated "source ID" or its associated "application ID." The source ID is present for channels that carry programming described by records in a program guide database, and acts as a linkage to an IPG. Refer to Section 4.9 of this Annex for a description of this linkage and its use.

The application ID tag is present for channels marked as "application access" type. One example of an application that uses this tag is a built-in interactive program guide function. Each application that may be resident in the system (either ROM-based or downloaded dynamically) has assigned to it a unique 16-bit application ID.

To follow the IPG example further, if guide provider X wishes to transport messages carrying his program guide database on a service within a Transport Stream somewhere in the system, he may create a MPEG program and one or more component streams within it, on a selected TS. A hidden virtual channel of the "application access" type is also created, and tagged with an application ID unique to provider X and this specific application (program guide data, in provider X's transmission format).

The application code can gain access to its data stream by searching the Virtual Channel Table for application access channels that match its application ID, and then acquiring the service pointed to by that virtual channel record. The application ID for any given application is hard-coded into the application itself. As mentioned above, if the Decoder's remote control unit has a GUIDE button, that button could be hard-wired to one specific application ID.

More than one IPG data service may be available in the system. If this is the case, it may be that a single application (capable of handling streams of different formats, or streams in a common format but originating from different providers) can find and process services associated with more than one application ID. The user may be offered a choice of providers. Alternatively, if one program guide application can be replaced in the Decoder with another, whichever guide application is currently present will acquire its own data stream.

4.7 Replicated Services

The transmission system described in this protocol supports the notion that more than one service may be decrypted simultaneously, even when each are separately access controlled. The service that carries the electronic program guide data, for example, may be replicated on many Transport Streams; a Decoder may be designed to access and decrypt this data in a "background" manner during processing of a standard audio/video service.

Definition of a replicated service involves the following:

- One virtual channel record (per map) defines the service's *primary* entry point.
- Access to the primary entry point, like normal channel access, involves tuning to the transponder associated with the virtual channel.
- If a service is carried on a Transport Stream in addition to the one defined as the primary entry point, a virtual channel defining a local access point may be defined. The local access record:
  - Is identified with the primary service in that the channel number is the same as the primary entry point virtual channel number.
  - Provides a definition used to access a copy of the service within a Transport Stream (defined within the local access record) other than the primary one.

4.8 Virtual Channels and an IPG Database

As mentioned, virtual channels may be tagged with a source_ID, used to associate the channel with entries in an IPG database. Any virtual channel that has (or may have) associated records in an IPG database that utilizes the source ID to uniquely associate the channel with its program source *must* have a source_ID specified. The defined association between each programming source and its source_ID is consistent and defined at the system level for the benefit of any IPG databases which may with to utilize the relationship.

The source_ID can link an IPG database to the Virtual Channel Tables in the following way. If a user is browsing through the interactive guide looking for current programs, he may stop, highlight a certain program, and select it for viewing. The selected program is associated with other programs offered by that programmer, and with the textual name of the programmer or channel itself by the source_ID fields within the database.

The selection process therefore results in the need to find a virtual channel associated with the selected source_ID. The Decoder must search through the VCT looking for a match on source_ID, or (because it is faster) search through the inverse channel maps using a binary search algorithm. Once a match is found, the service can be acquired by following the access path defined in the virtual channel record.

5 Representation of Time

The International Bureau of Weights and Measures maintains International Atomic Time (TAI)[19], a constant rate time standard based on a number of cesium-based atomic clocks. Cesium beam frequency standards are also used onboard the Global Positioning Satellites (GPS), whose coordinated outputs provide *GPS Time*, another constant rate time standard.

At its inception, GPS time was usable directly to set time-of-day clocks throughout the world. Due to the gradual slowdown of the Earth as it moves through its solar orbit, however, the length of a year in GPS time has drifted away from the length of a year as

[19] The ordering of the letters in the acronym is based on the French language representation.

measured by the Earth's rotation. Based on observed rotational anomalies, leap seconds are inserted or deleted as required from TAI to yield Universal Coordinated Time, or UTC. Time of day clocks throughout the world are aligned to UTC time. GPS satellites also broadcast the offset between the constant rate GPS time and UTC. Currently, GPS runs approximately 10 seconds ahead of UTC. UTC is what we currently think of as *standard* time (formerly called *Greenwich Mean Time* [GMT]), and is referenced to the time zone at Greenwich, England.

This protocol defines various time-related events and activities, including starting times for programs, text display, changes to VCTs, and others. Standard-compatible headend equipment is synchronized to GPS time by the incorporation of one or more GPS time clocks.[20] The Standard-compliant Decoder is synchronized to system time by the SYSTEM TIME message, which provides time in the form of GPS seconds since week zero of GPS time. The message also provides the current GPS/UTC offset, in whole seconds.

5.1 System Time

GPS satellites typically output GPS time in a format consisting of a week count (Tw) and a seconds within the week count (Ts), where week zero is defined as starting January 1, 1980 00:00:00. For purposes of building the SYSTEM TIME message, the following formula may be used:

$$T = (Tw * 604{,}800) + Ts$$

There are 604,800 seconds per week.

When converting between GPS seconds and current local time in hours/minutes/seconds, the following factors must be taken into account:

- **GPS to UTC offset** — Given a time represented as GPS seconds, the Decoder first subtracts the GPS/UTC offset to convert to UTC.
- **1980** — The first year of GPS time started on January 6th, yielding 361 days in the first year (1980 was also a leap year).
- **Leap years** — The number of leap years that occurred between the current GPS second and 1980 must be accounted for. A leap year is a year whose number is evenly divisible by four, or, in the case of century years, by 400.

NOTE: According to this rule, the year 2000 *will* be a leap year even though it is a century year, because it is also divisible by 400.

- **Time zones** — Time zones are signed integer values in the range -12 to +13 hours, where positive numbers represent zones east of the Greenwich meridian and negative numbers west of it. Pacific Standard Time (PST) is 8 hours behind standard time, and Eastern Standard Time (EST) is 5 hours behind. The system defined by this System Information Standard accommodates time zones that are not an integral number of hours offset from Greenwich by defining time zone as an 11-bit signed integer number in units

[20] Companies making such equipment include Trimble Navigation, Trak Microwave, Datum Inc., Socket Communications, and many others.

of minutes. To convert to local time, the time zone is added to Greenwich time using signed integer arithmetic.

- **Daylight savings time** — If applicable, daylight savings time must be taken into account. On a unit by unit basis, each Decoder may be given a definition for when daylight savings time is entered into in Spring, and when it is exited in Fall. Entry/exit points are given as absolute times (GPS seconds), and hence are given in one second resolution.

5.2 Transmission Format for Event Times

In this messaging protocol, the absolute time of action is specified for most events in terms of an unsigned 32-bit integer number, the count of GPS seconds since January 1, 1980 00:00:00. This count will not wrap until after the year 2116[21].

[21] Prior to that time, all initial Decoders will surely be out of service, and new ones can be designed to handle the wrap condition.

Annex B

Informative
Usage Guidelines

1 Scope

This section provides some informative text relating to additional features supported in this System Information Standard.

2 Programs and Program Epochs

In the system described in this standard, within the context of one single ECM stream, access control is provided on the basis of program epochs. A program epoch is defined as the period of time that the access requirements for one specific program are in effect. One program epoch directly follows another, with no gap between.

Within one program epoch, a number of specific events and characteristics are defined:

- **Interstitial duration**—Defines the amount of time between the beginning of the program epoch and the time the movie actually starts; the interstitial material is typically advertising and previews of the current or coming attractions
- **IPPV window duration**—Defines the period of time at the beginning of the epoch that an IPPV purchase may be made, given that all other requirements for IPPV are met. The programmer may elect to restrict IPPV purchases to the first 15 minutes of a movie, for example, to reduce customer complaints arising from, "I didn't watch the first part of the movie but I got billed anyway." Note that purchase prior to the start of the epoch is not restricted.
- **Initial free preview duration**—Defines the period of time at the beginning of the epoch that is given away free (no subscription or IPPV purchase required); used for promotional purposes. Initial free preview is optional and may be set to zero.

These characteristics modify the behavior of the user interface, and are used in the creation of certain user screens. Figure B.1 describes a typical program epoch in which these three milestones are indicated. Each is independent of the other; for example, the figure shows that the free preview extends beyond the start of the actual movie, but this relationship is not enforced.

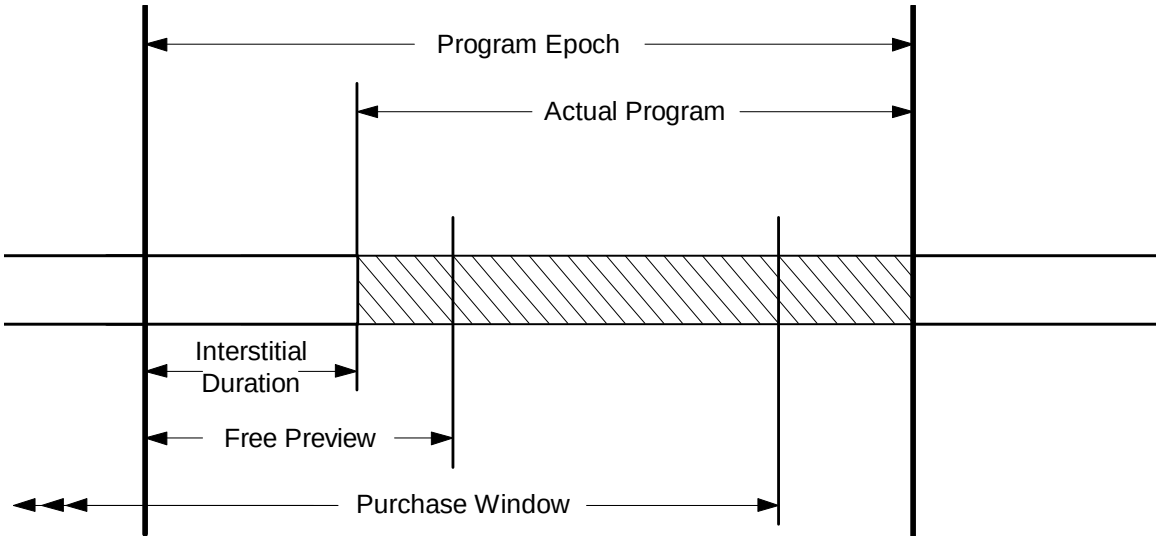


Figure B.1. Program Epoch Definition

3 Program Packages

Program packaging provides a way to group programs in a cryptographically secure manner for access control. Any program may be defined as being associated with a number of different program packages, each identified by package ID, package cost, and package IPPV tier. Through an appropriate user interface, either the individual program may be offered for IPPV purchase (normal IPPV), or any one of the program packages may be offered (program package IPPV).

Once a program or package has been purchased, the secure processor records the program or package ID so that, if the same program or package is encountered again later, the user is not required to re-buy it. Note that many programs may be tagged with the same package ID and still be encrypted using different program keys and have different access requirements. Programs tied together using a common package ID are in all other ways distinct and independent.

If a program that would be authorized because it is part of a bought package is subject to a circular blackout, the Access Control Processor (ACP) grants authorization for that program only if the program access requirements allow for “buy through blackout.” If buy through blackout is not offered, access to the program is denied even though it is part of a bought package.

If a program that would be authorized because it is part of a bought package is subject to a regional blackout, access to that program is denied.

4 Near Video On Demand (NVOD)

From the Decoder user’s point of view, a number of movies may be offered for sale or viewing at any given time. The user does not wish to be bothered that a particular movie

started over an hour ago; all he or she knows is that this is the movie of interest. With true "video on demand," the system would be able to create an instance of the desired movie, which is started at the moment the user is ready for it.

Because it is impractical to provide 120 instances of a two hour movie, one starting every minute, the idea of "near" video on demand is introduced: with NVOD, the user may have to wait 10 or 15 minutes for a movie to start. Instead of 120 instances of the movie, there may be four or perhaps as many as ten. The actual number is based on the programmer's choice of use of video bandwidth resources.

4.1 Design Goals of NVOD

This protocol is designed with the following system-level goals in mind:

- To the user, the movie must look like a single channel, not N channels each carrying the same movie. The NVOD scheme must be integrated with the virtual channel scheme, so that the user sees a single channel number for the movie, regardless of which instance is actually selected by the Decoder.
- If one movie is purchased, access control must allow viewing of movies that started before or after, to allow a "skip ahead" and "skip back" function analogous to the VCR fast forward/reverse.
- The access control scheme could support a model analogous to videotape rental, where, once a showing is purchased, any instance of the movie shown during any day may be viewed without further charges.
- The Decoder User Processor needs to determine which of the N movies is starting the soonest, and offer that instance initially.

4.2 System Design of NVOD

The NVOD design involves a number of system features, including program packaging, virtual channel grouping, and NVOD user interface. These are described in the following sections.

4.2.1 Program Packaging for NVOD

NVOD operation involves use of the program packaging concept. The way program packaging is used for NVOD is that, at the discretion of the programmer, NVOD movie programs may be grouped to allow access to instances of the movie in addition to the one purchased. For example, the programmer may wish to emulate the video rental store model, where the user is free to watch the movie as many times as desired for a 24 hour period. In that case, any instance of a particular title during a particular day would carry a package ID unique to that day.

For another example, say the programmer wished to allow one 15-minute pause and one 15-minute skip-ahead. Each movie would be a member of three package groups (one in which it was the first, one where it was the middle, and one in which it was the last). Flexibility is ultimately limited by the total number of packages that can be associated with one movie or event (currently four).

4.2.2 Virtual Channel Grouping

Each virtual channel number (VCN) is associated with a record structure that includes a bit that indicates whether the VCN is an NVOD access channel or not. If the VCN is an NVOD access channel, instead of identifying the components of the service, the record instead points to a contiguous block of VCNs. The block is described by two parameters, the NVOD base VCN and the NVOD channel count.

The NVOD base VCN is an index into the virtual channel table where the first of N VCN records describing the NVOD channels in this group can be found. The number of channels associated with this particular NVOD channel is given by NVOD channel count.
Each of the N VCN records describing the set of NVOD channels are tagged as a "hidden channel" so that access to them can only be made via the NVOD VCN.

4.2.3 NVOD User Interface

From the user's perspective, there is just one NVOD channel per movie; alternatively each NVOD channel could be thought of as a movie theater.[22] The (non-subscribing) user is first presented with the IPPV screen, and if the purchase is made, one of two things occurs next. Of the N movies in the NVOD group, one will be next to start, and the movie previously playing in that theater may or may not be over:

1. If the prior running of the instance of the movie with the closest start time is still is progress, a screen may be presented saying something like, "Thank you for choosing Action Movie. The movie starts in 14:34."
2. If the instance of the movie with the closest start time is already over, the promotional (interstitial) material may be shown.

To create the user interface, the Decoder first tunes to the NVOD base VCN and collects the PROGRAM INFORMATION message describing the instance of the movie currently playing in that theater (call it theater 0). The "staggered start interval" is defined as the time between the start of a program on one NVOD channel, and the start of the same program on the next NVOD channel in the group. The message defines the staggered start interval, valid for all instances of the movie in the NVOD group, and gives the interstitial duration. From the interstitial duration, the Decoder can derive the time until the movie starts, or the time elapsed since the movie started, as appropriate.

The Decoder then determines which of the N instances of the movie in the NVOD group will next reach its starting point.

[22] Analogous to the TVN "theaters" on C-band today.

In the following discussion, the following definitions apply:

1. T = theater number
2. M = min. since movie in this epoch started in theater zero (if within interstitial period, M **< 0)**
3. S = stagger time for current movie in theater zero
4. N = number of theaters playing current movie in theater zero
5. D = total epoch duration
6. L = last showing flag

The general formula used by the Decoder to determine the theater next starting a showing of the movie is:

1) if M < 0 then T = 0, else T = (1 + M div S)

2) if T >= N then T = 0

The value of *I* **div** *J* is the mathematical quotient of *I/J*, rounded in the direction of zero to an integer result.

4.2.4 Summary of Rules for Setting Up NVOD Groups

The following guidelines should be followed when setting up NVOD groups:

1. Each instance of a movie should be defined identically, with regard to:
    - Program epoch duration
    - Interstitial duration
    - Program cost, tiers, and all other access control parameters
2. Given the definitions of the previous section, and if D is the program epoch duration, the relationship D >= N*S holds true. The values for D and S are specified in whole minutes.

The latter guideline says that it is not allowable, for example, to have three instances of a one-hour movie and a 25 minute staggered start. It would be allowable, however, to use a 10-minute staggered start in this example. The D >= N*S rule ensures that as movies are played back-to-back, the pattern started at theater zero repeats with every subsequent showing of the same movie in theater zero, and that next starting times within the NVOD group are always in consecutive order.

Note that the rules for NVOD group definition result from the fact that it is desired to be able to derive knowledge of all instances of an NVOD program by acquisition of just the NVOD base VCN. While it would be possible to acquire, in turn, each of the VCNs in the group, this would be time consuming and interfere with the creation of a smooth user interface.

4.2.5 Example of NVOD Entry

Figure B.2 shows an example of a transition between an 80-minute program having a 10-minute interstitial period, shown in six theaters with a 15-minute staggered start time, with a 65-minute program having a 5-minute interstitial period, shown in seven theaters with a 10-minute staggered start time. Six example access points, labeled “a” through “f” are shown.

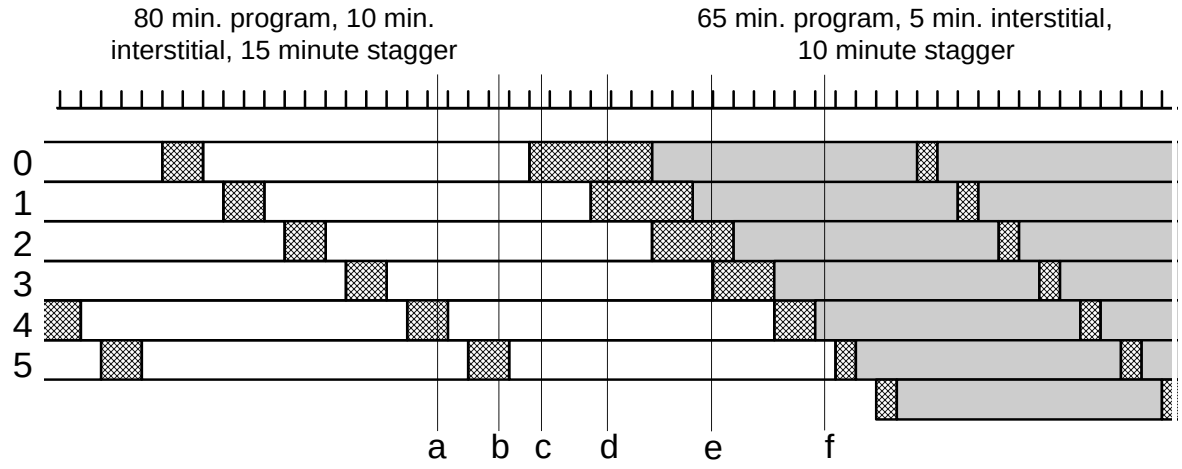


Figure B.2. NVOD Movie Transition Period Example #1

Applying the formula given above to the six example entry points:

a: T = (1 + 58 div 15) = 4 (Movie A)
b: T = (1 + 73 div 15) = 5 (Movie A)
c: T = 0 (Movie B) because within interstitial
d: T = 0 (Movie B) because within interstitial
e: T = (1 + 15 div 10) = 2 (Movie B)
f: T = (1 + 43 div 10) = 5 (Movie B)

Figure B.3 provides a second example, this time where movie B's stagger time is longer, and movie B is shown in fewer theaters than movie A.

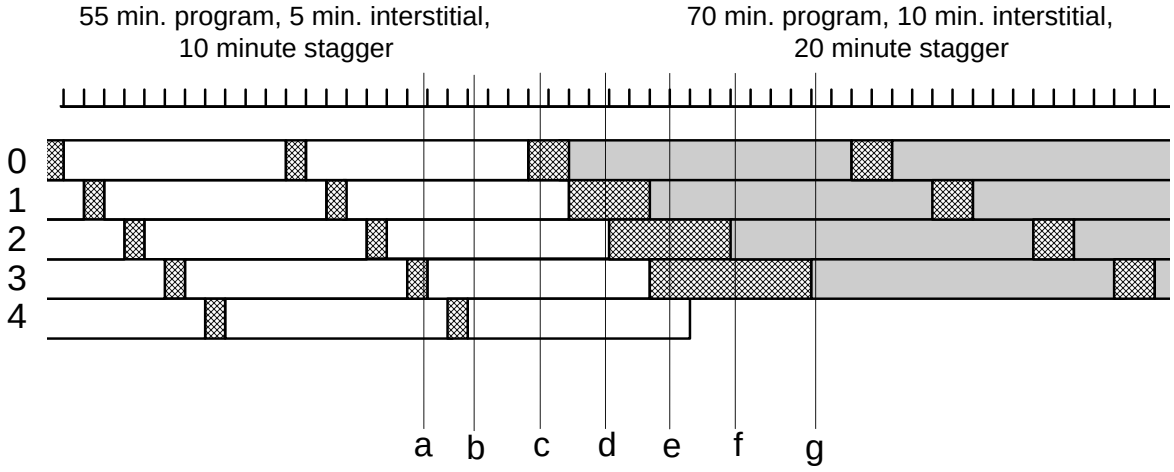


Figure B.3. NVOD Movie Transition Example #2

For example #2, the calculations derive:

a: T = (1 + 29 div 10) = 3 (Movie A)
b. T = (1 + 42 div 10) = 5; 5>=N so T = 0
c: T = 0 (Movie B) because within interstitial
d: T = (1 + 9 div 20) = 1 (Movie B)
e: T = (1 + 25 div 20) = 2

f: T = (1 + 46 div 20) = 3
g: T = (1 + 66 div 20) = 4; 4>=N so T = 0

5 Program Ratings

Programs conformant to this specification may include parental rating advisory data in the rating_data() structure delivered in the PROGRAM INFORMATION message. The data is used in conjunction with the Rating Text Table, delivered in the NETWORK INFORMATION message.

5.1 Rating Regions

To ensure maximum flexibility and applicability, the RTT defines rating dimensions and levels for multiple "rating regions." Rating regions provide another dimension to the rating system supported in this protocol by allowing any program to carry rating advisories according to several rating standards. For example, if an organization in the United States develops a rating system that simply uses an age-level advisory ("suitable for persons above the age of 12"), programs could be tagged with the age-appropriateness rating in addition to the standard "default" rating (MPAA). Or, if Canada or Mexico standardizes a rating system, those rating attributes could be carried along with their U.S. counterparts.

The Decoder may be designed to be flexible enough to offer the user a choice of rating methods (actually region selection), by a user preference option. For some applications, the Decoder's rating region may be set by the system operator (through an EMM, for example) or via a procedure done by a professional installer.

5.2 Definition of Rating Dimensions and Levels

The definitions of each field in the program_ratings() structure may be changed dynamically by redefining the text strings associated with each rating dimension. The text defining ratings is carried in the in the RTT, carried in the Network PID. The following dimension definitions represent an informative example only.

Figure B.4 defines the rating dimensions currently defined for the US rating region.

	Bits	Bytes	Description
program_ratings(){			
MPAA_rating	4	1	uimsbf range 0-15
violence_content_advisory	4		uimsbf range 0-15
language_content_advisory	4	1	uimsbf range 0-15
sexual_content_advisory	4		uimsbf range 0-15
Reserved	4	1	bslbf reserved
Reserved	4		bslbf reserved
}			

Figure B.4. Example US Region Ratings Dimensions

All program content analysis is the function of parties involved in program production or distribution. No precise criteria for establishing content ratings or advisories is known to

exist, or are to be implied here. The codes are provided for the convenience of consumers and the programming industry.

MPAA_rating — A 4-bit number in the range zero through 15 that indicates the program's parental rating level. Zero is a special case that indicates that the program does not carry a rating designation, and is not subject to rating lockout. The uncontrolled designation (value zero) is also used when motion picture ratings are not applicable to the program (for example, made-for-TV movies).

violence_content_advisory — A 4-bit enumerated type that defines the program's violence content on a graduated scale. Value zero indicates no violent content.

sexual_content_advisory — A 4-bit enumerated type that defines the program's sexual content on a graduated scale. Value zero indicates no sexual content.

language_content_advisory — A 4-bit enumerated type that defines the program's language content on a graduated scale. Value zero indicates no language advisory.