SIST

SLOVENSKI STANDARD
SIST EN 50221:1999

01-april-1999

Common interface specification for conditional access and other digital video broadcasting decoder applications

Common interface specification for conditional access and other digital video broadcasting decoder applications

Festlegung der einheitlichen Schnittstelle für Zugriffsbeschränkung und andere digitale Fernsehrundfunkdecoder-Anwendungen

iTeh STANDARD PREVIEW
(standards.iteh.ai)

Spécification d'une interface commune pour l'accès conditionnel et d'autres applications dans un décodeur de télévision numérique

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

Ta slovenski standard je istoveten z: EN 50221:1997

ICS:

33.160.40	Video sistemi	Video systems

SIST EN 50221:1999 **en**

2003-01.Slovenski inštitut za standardizacijo. Razmnoževanje celote ali delov tega standarda ni dovoljeno.

SIST EN 50221:1999

iTeh STANDARD PREVIEW
(standards.iteh.ai)

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

SIST EN 50221:1999

EUROPEAN STANDARD

NORME EUROPÉENNE

EUROPÄISCHE NORM

EN 50221

February 1997

ICS 33.160.40

Descriptors: Telecasting, television receivers, decoders, interfaces, design, physical properties, open systems interconnection, physical layers, data link layer, transport layer, session layer, application layer, protocols, man-machine systems, specifications

English version

Common interface specification for conditional access and other digital video broadcasting decoder applications

Spécification d'une interface commune pour l'accès conditionnel et d'autres applications dans un décodeur de télévision numérique

Festlegung der einheitlichen Schnittstelle für Zugriffsbeschränkung und andere digitale Fernsehrundfunkdecoder-Anwendungen

iTeh STANDARD PREVIEW
(standards.iteh.ai)

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

This European Standard was approved by CENELEC on 1997-02-15. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the Central Secretariat has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and United Kingdom.

CENELEC

European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

Central Secretariat: rue de Stassart 35, B - 1050 Brussels

© 1997 CENELEC - All rights of exploitation in any form and by any means reserved worldwide for CENELEC members.

Ref. No. EN 50221:1997 E

SIST EN 50221:1999

Foreword

This draft European Standard was prepared by the Technical Committee CENELEC TC 206, Broadcast Receiving Equipment.

The text of the draft was submitted to the Unique Acceptance Procedure and was approved by CENELEC as EN 50221 on 1997-02-15.

The following dates were fixed:

- latest date by which the EN has to be implemented at national level by publication of an identical national standard or by endorsement	(dop)	1997-10-01
- latest date by which national standards conflicting with the EN have to be withdrawn	(dow)	1997-10-01

---

iTeh STANDARD PREVIEW
(standards.iteh.ai)

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

SIST EN 50221:1999

Contents

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

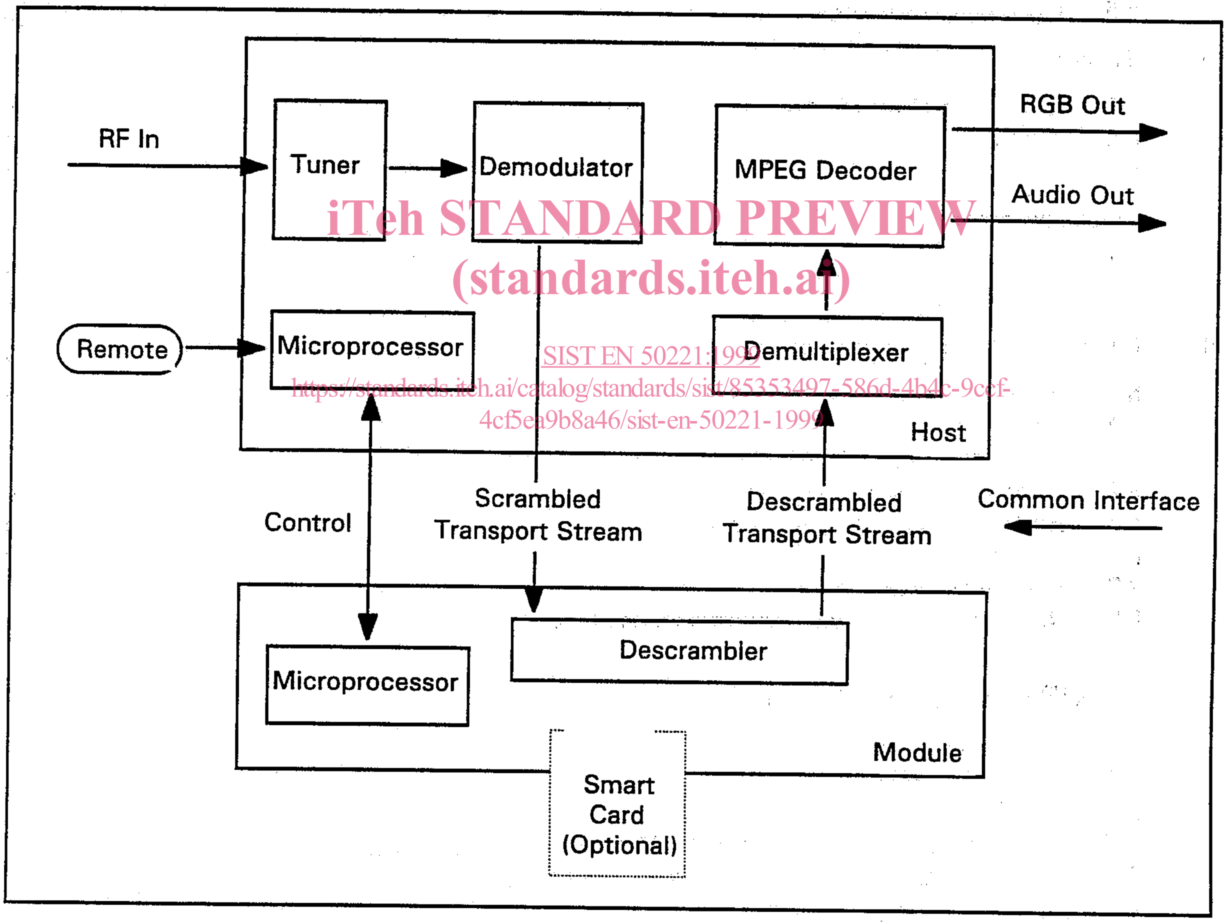
SIST EN 50221:1999

1 Introduction and scope

A set of standards has been designed to be used in digital video broadcasting. These standards include source coding, channel coding, service information and decoder interfaces. In addition, a conditional access system is used when there is a need to control access to a broadcast service. It has been decided that the conditional access system need not be standardised, although a common scrambling algorithm is provided. It remains for broadcasters to access decoders with different conditional access systems and to ensure that they have choice of supply of such systems. A solution is to use the common scrambling algorithm and to execute solutions for access based on commercial agreements between operators. This solution can operate with single CA systems embedded in decoders.

A second solution is based on a standardised interface between a module and a host where CA and more generally defined proprietary functions may be implemented in the module. This solution also allows broadcasters to use modules containing solutions from different suppliers in the same broadcast system, thus increasing their choice and anti-piracy options. The scope of this document is to describe this common interface.

The decoder, referred to in this specification as the host, includes those functions that are necessary to receive MPEG-2 video, audio and data in the clear. This specification defines the interface between the host and the scrambling and CA applications, which will operate on an external module.



iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9cef-4cf5ea9b8a46/sist-en-50221-1999

Figure 1: Example of single module in connection with host

Two logical interfaces, to be included on the same physical interface, are defined. The first interface is the MPEG-2 Transport Stream. The link and physical layers are defined in this specification and the higher layers are defined in the MPEG-2 specifications. The second interface, the command interface, carries commands between the host and the module. Six layers are defined for this interface. An example of a single module in connection with a host is shown in figure 1.

SIST EN 50221:1999

This specification only defines those aspects of the host that are required to completely specify the interactions across the interface. The specification assumes nothing about the host design except to define a set of services which are required of the host in order to allow the module to operate.

The specification does not define the operation or functionality of a conditional access system application on the module. The applications which may be performed by a module communicating across the interface are not limited to conditional access or to those described in this specification. More than one module may be supported concurrently.

2 Definitions

For the purposes of this standard, the following definitons apply:

application : An application runs in a module, communicating with the host, and provides facilities to the user over and above those provided directly by the host. An application may process the Transport Stream.

host : A device where module(s) can be connected, for example : an IRD, a VCR, a PC ...

module : A small device, not working by itself, designed to run specialised tasks in association with a host, for example : a conditional access sub system, an electronic program guide application module, or to provide resources required by an application but not provided directly by the host

resource : A unit of functionality provided by the host for use by a module. A resource defines a set of objects exchanged between module and host by which the module uses the resource.

service : A set of elementary streams offered to the user as a program. They are related by a common synchronisation. They are made of different data, i.e., video, audio, subtitles, other data...

iTeh STANDARD PREVIEW
(standards.iteh.ai)

transport stream : MPEG-2 Transport Stream.

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

3 Normative references

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

[1]	ISO/IEC 13818-1	Information technology - Generic coding of moving pictures and associated audio information: Systems
[2]	ISO 8824 1987	Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)
[3]	ISO 8825 1987	Open Systems Interconnection - Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1)
[4]	ETS 300 468	Specification for Service Information (SI) in Digital Video Broadcasting (DVB) systems
[5]	ETR 162	Allocation of Service Information (SI) codes for Digital Video Broadcasting (DVB) Systems
[6]	PC Card Standard	Volume 2 - Electrical Specification, February 1995, Personal Computer Memory Card International Association, Sunnyvale, California
[7]	PC Card Standard	Volume 3 - Physical Specification, February 1995, Personal Computer Memory Card International Association, Sunnyvale, California
[8]	PC Card Standard	Volume 4 - Metaformat Specification, February 1995, Personal Computer Memory Card International Association, Sunnyvale, California
[9]	prETS 300 743	DVB Subtitling Specification

SIST EN 50221:1999

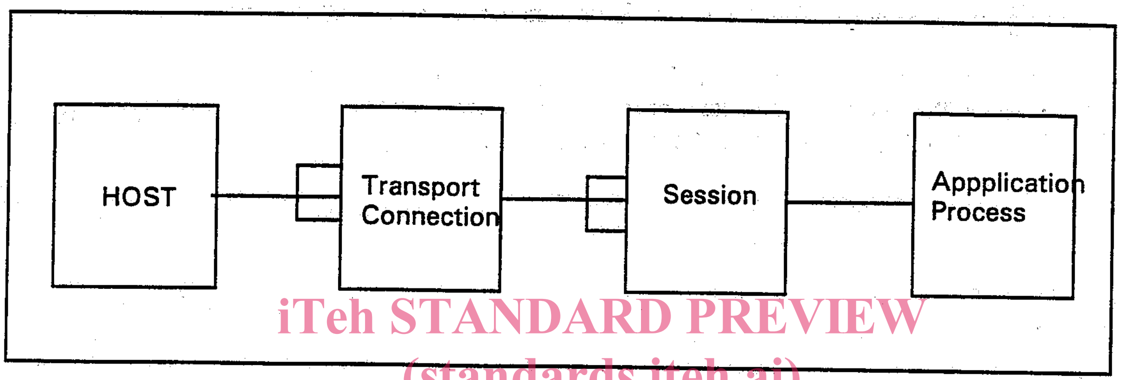
4 Design philosophy

4.1 Layering

The specification is described in layers in order to accommodate future variations in implementation. The application and session layers are defined for all applications of the common interface. The transport and link layers may be dependent on the physical layer used in a particular implementation. The physical interface is defined within this specification and includes the complete physical specification of the module

The layering of the specification allows flexibility in the use of the interface for a range of applications beyond CA. It also allows for multiple instance of CA processes to exist for the same host.

A representation of the basic layering on the command interface is shown in figure 2. The host may set up transport connections with more than one module, which may be connected directly or indirectly to the host. Each connection is maintained while the module is present. Each module may manage a number of different sessions with the host.



iTeh STANDARD PREVIEW
(standards.iteh.ai)

Figure 2: Layering on the command interface

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

4.2 Physical implementation

The baseline specification includes the implementation on a physical interface compatible with the PC Card standard used in the Personal Computer industry. Other physical implementations are allowed for in the future.

4.3 Client-server

The interface is designed on the principle that applications, as clients, use resources provided by a server. The applications reside on a module and resources can be served either by the host or another module in a way managed by the host. The term 'resources' has been used in preference to 'services' as that term is common in the broadcasting field for TV and radio services and there is a need to avoid confusion.

4.4 Coding of data

The communication of data across the command interface is defined in terms of objects. The objects are coded by means of a general Tag-Length-Value coding derived from that used to code ASN.1 syntax (see [2] and [3]). This is generally extensible. There is a particular transport layer coding for the PC Card implementation but it may be different in other physical implementations. However the semantics would be identical.

4.5 Extensibility

The higher layers have been designed to be extensible. As indicated above, the TLV coding used is extensible so that new objects can be added, and existing objects can be extended. There is no problem about running out of tag coding space, or length restrictions on the values. The Resource Manager resource provides a mechanism for extending the range of resources provided by hosts, both for CA purposes and for other module-based applications.

SIST EN 50221:1999

4.6 Incorporation of existing standards

Existing standards have been used, where possible and appropriate, as building blocks for this specification. This gives important time-to-market benefits, as all the standards development work has already been done. It also gives implementation benefits in that software and hardware already developed for existing standards may be re-used here, with potential cost benefits.

5 Description and architecture

5.1 Overview

A partial logical architecture has been assumed for a host in order to define the place in the host where the common interface can logically occur. The impact upon the freedom of choice for host designers in other respects has been minimised. Figure 1 shows a simplified picture of a typical host architecture and the positioning of the interface within it. Note that there can be more than one instance of the interface on a host.

The common interface consists of two components, the Transport Stream Interface and the Command Interface. Both are layered to make the overall interface design and implementation easier. The upper layers are common to all implementations but alternative lower-layer implementations are possible. This specification includes one based upon the PC Card standard but others may be included in future versions.

5.2 Transport Stream Interface

The Transport Stream Interface carries MPEG-2 transport packets in both directions. If the module gives access to any services in the transport stream and those services have been selected by the host, then the packets carrying those services will be returned descrambled, and the other packets are not modified. On the Transport Stream Interface a constant delay through the module and any associated physical layer conditioning logic is preserved under most conditions (see 5.4.2). The Transport Stream Interface layers are shown in figure 3 below. The Transport Layer and all upper layers are defined in the MPEG-2 specification - ISO 13818.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85553497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

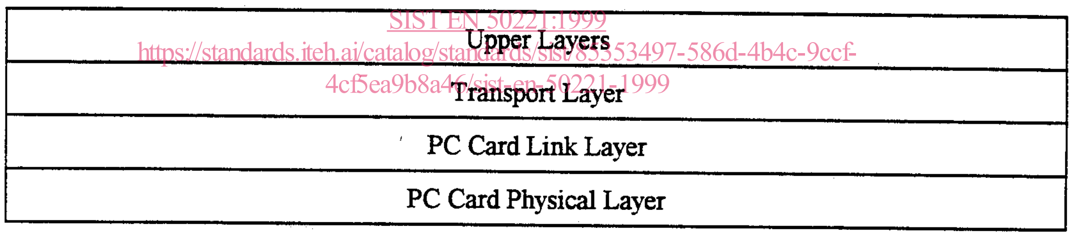


Figure 3: Transport Stream Interface Layers

5.3 Command Interface

The Command Interface carries all the communication between the application(s) running in the module and the host. The communication protocols on this interface are defined in several layers in order to provide the necessary functionality. This functionality includes: the ability to support multiple modules on one host, the ability to support complex combinations of transaction between module and host, and an extensible set of functional primitives (objects) which allow the host to provide resources to the module. The layering is shown in figure 4 below.

The PC Card implementation described in this specification has its own Physical and Link layers, and also its own Transport lower sublayer. A future different physical implementation is likely to differ in these layers and any difference will be restricted to these layers. The implementation-specific features of the Transport lower sublayer are limited to coding and specific details of the message exchange protocol, and the common upper sublayer defines identification, initiation and termination of Transport layer connections. The Session, Resource and Application layers are common to all physical implementations.

SIST EN 50221:1999

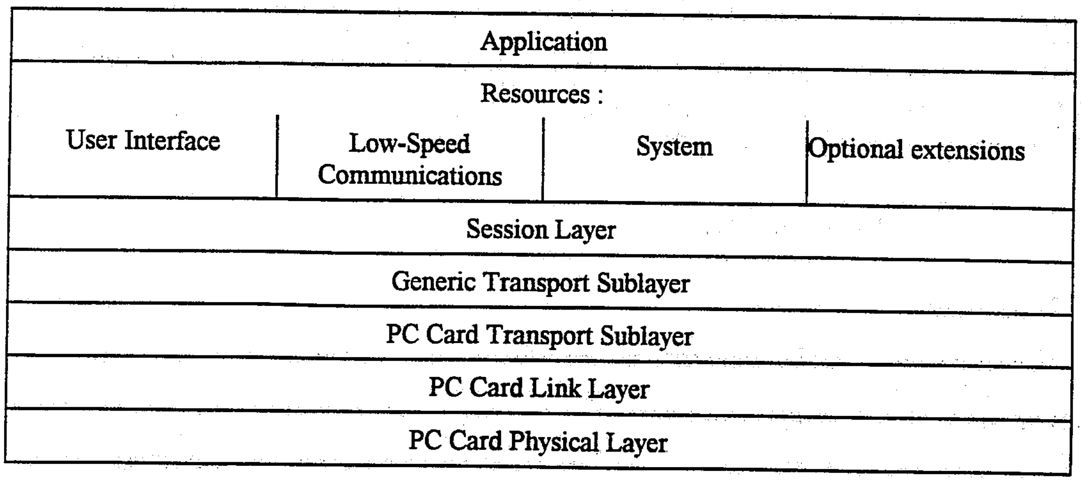


Figure 4: Command Interface Layers

As far as possible the Application layer of the interface has been designed to be free of specific application semantics. Communication is in terms of resources, such as User Interface interaction, and low-speed communications, that the host provides to the application(s) running on a module. This strategy makes it very much easier to provide modules performing other tasks than just Conditional Access.

5.4 Physical requirements

5.4.1 Introduction

iTeh STANDARD PREVIEW
(standards.iteh.ai)

This clause defines the requirements the Physical Layer must meet in order to carry out all the required functions. The following Physical Layer characteristics are not constrained here, although the specification for any Physical Layer used will define them: mechanical and electrical connection between the host and the module, i.e. socket type & size, number of pins, voltages, impedances, power limits.

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85555497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

Requirements and limits on the following Physical Layer characteristics are defined here:

- Transport Stream and Command logical connections;
- data rates;
- connection & disconnection behaviour;
- low-level initialisation;
- use of multiple modules.

5.4.2 Data and Command logical connections

The Physical Layer shall support independent both-way logical connections for the Transport Stream and for commands.

The Transport Stream Interface shall accept an MPEG-2 Transport Stream, consisting of a sequence of Transport Packets, either contiguously or separated by null data. The returned Transport Stream may have some of the incoming transport packets returned in a descrambled form. The Transport Stream Interface is subject to the following restrictions:

1 When the module is the source of a transport stream its output shall comply with ISO/IEC 13818-9.

2 Each output packet shall be contiguous if the module is the source of the packet or the input packet is contiguous.

SIST EN 50221:1999

3 A module shall introduce a constant delay when processing an input transport packet, with a maximum delay variation (tmdv) applied to any byte given by the following formula:

$$tmdv_{max} = (n * TMCLKI) + (2 * TMCLKO).$$

and

$$tmdv_{max} <= 1 \text{ microsecond when } n = 0$$

where:

tmdv	=	Module Delay Variation
n	=	Number of gaps present within the corresponding input transport packet
TMCLKI	=	Input data clock period
TMCLKO	=	Output data clock period

* A `gap' is defined to be one MCLKI rising edge for which the MIVAL signal is inactive.
* All hosts are strongly recommended to output contiguous transport packets.
* Hosts may only output non-contiguous transport packets if they implement less than 3 common interface sockets.
* Inter packet gaps may vary considerably.

4 A CI compliant host should be designed to support Nm modules. Nm is the greater of the number of CI sockets implemented by the host or 16. It should tolerate the jitter resulting from Nm modules plus the jitter in the input transport stream. The worst case jitter may arise either from the host's own input followed by Nm modules or an input module with a ISO/IEC 13818-9 compliant output followed by (Nm - 1) modules.

5 All interfaces shall support a data rate of at least 58 Mb/s averaged over the period between the sync bytes of successive transport packets.

6 All interfaces shall support a minimum byte transfer clock period of 111 ns.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/...

The Command Interface shall transfer commands as defined by the appropriate Transport Layer part of this specification in both directions. The data rate supported in each direction shall be at least 3,5 Megabits/sec.

5.4.3 Connection and disconnection behaviour

The Physical layer shall support connection and disconnection of the module at any time, whether the host is powered or not. Connection or disconnection shall not cause any electrical damage to either module or host, and shall not cause any spurious modification of stored non-volatile data in the module. When a module is not connected the Transport Stream Interface shall bypass the module, and the Command Interface to that module shall be inactive. On connection of a module, the host shall initiate a low-level initialisation sequence with the module. This will carry out whatever low-level connection establishment procedures are used by the particular Physical Layer, and then establish that the module is a conformant DVB module. If successfully completed, the host shall establish the Transport Stream connection by inserting the module into the host's Transport Stream path. It is acceptable that some Transport Stream data is lost during this process. At the same time a Transport Layer connection shall be established on the Command Interface to allow Application Layer initialisation to take place and normal Application Layer communications to proceed.

If the Physical Layer is used in other applications than as a DVB-conformant module connection, and if a non-conformant module is connected to the host, no damage shall be caused to the module or the host, and the host shall not attempt to complete initialisation as though it were a DVB-conformant module. Optionally, the host may signal to the user that an unrecognised module has been connected.

On disconnection of the module, the host shall remove the module from the Transport Stream data path. It is acceptable that some Transport Stream data is lost during this process. Also, the Command Interface connection shall be terminated by the host.

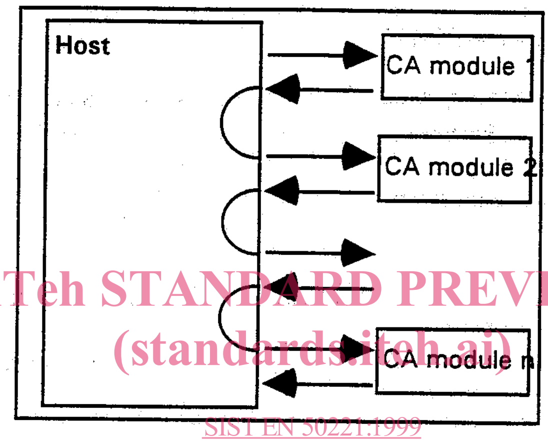
SIST EN 50221:1999

5.4.4 Multiple modules

The Application Layer places no limit on the number of modules which may be connected to the host at any time. However, particular Physical Layers and particular host design choices may do so. The Physical Layer specification must allow there to be several modules connected simultaneously to the host, even though a minimum host design may only provide for one connection. Ideally the Physical Layer specification should place no hard limit on the number of modules, but if a limit is imposed, then it shall be set at no less than 15 modules.

Where there is provision for more than one module to be connected, the Transport Stream Interface connection shall be daisy-chained through each module in turn, as illustrated in figure 5 below. The host shall maintain separate and simultaneous Command Interface connections to each module, so that transactions between host and module are treated independently for each module. When a module is unplugged the Command Interface transport layer connection to any other module shall not be disturbed or terminated.

When several modules are connected to a host, the host should be able to select the module(s) relevant for the descrambling of the selected service(s).



iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9ccf-4cf5ea9b8a46/sist-en-50221-1999

Figure 5: Transport Stream Interface chaining between modules

5.5 Operational example

To illustrate some of the features described above consider this example of the processes that occur when a PC Card module is plugged into a host. The PC Card initialisation commences with sensing of the module being plugged in by sense pins on the interface. The host then reads the Card information Structure residing in the Attribute Memory of the module. This contains low-level configuration information for the module, such as PC Card read and write addresses used by the module, and indicates to the host that it is a DVB-conformant module. The host now turns off the Transport Stream Interface bypass link and allows the transport packets to flow through the module. This introduces a delay, and consequently a short gap in the Transport Stream data, but this is unavoidable. At the same time the physical layer interface initialisation process takes place to negotiate the buffer size to be used for communication. At this point the physical layer initialisation process is complete and the upper-layer initialisation process, common to all physical implementations, commences with the host creating a Transport Layer connection to the module. This process and the rest of the upper-layer initialisation process are described elsewhere in this document.

The initialisation process will be logically similar for other physical implementations though the details will differ.

6 Transport Stream Interface (TSI)

6.1 TSI - physical, link layers

These layers depend on the physical implementation of the module.

SIST EN 50221:1999

6.2 TSI - transport layer

The transport layer used is the same as the MPEG-2 System transport layer. Data travelling over the transport stream interface is organised in MPEG-2 Transport Packets. The whole MPEG-2 multiplex is sent over this transport stream interface and is received back fully or partly descrambled. If the packet is not scrambled, the module returns it as is. If it is scrambled and the packet belongs to the selected service and the module can give access to that service, then the module returns the corresponding descrambled packet with the transport_scrambling_control flag set to '00'.

If scrambling is performed at Packetised Elementary Stream (PES) level, then the module reacts in the same way and under the same conditions as above, and returns the corresponding descrambled PES with the PES_scrambling_control flag set to '00'.

The transport packet and the PES packet are completely defined in the MPEG-2 System specification [1].

6.3 TSI - upper layers

Apart from the Packetised Elementary Stream, any layering or structure of the MPEG-2 data above the Transport Stream layer is not relevant to this specification. However the specification does assume that the module will find and extract certain data required for its operation, such as ECM and EMM messages, directly from the Transport Stream.

7 Command Interface - Transport & Session Layers

The communication of data across the command interface is defined in terms of objects. The objects are coded by means of a general Tag-Length-Value coding derived from that used to code ASN.1 syntax.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

Table 1: Length field used by all Protocol Data Units at Transport, Session & Application Layers

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85353497-586d-4b4c-9c4f-4cf5ea9b8a46/sist-en-50221-1999

Syntax	No. of bits	Mnemonic
length_field() {		
size_indicator	1	bslbf
if (size_indicator == 0)		
length_value	7	uimsbf
else if (size_indicator == 1) {		
length_field_size	7	uimsbf
for (i=0; i<length_field_size; i++) {		
length_value_byte	8	bslbf
}		
}		
}		

This clause describes the ASN.1 objects for the Transport and Session Layers that travel over the command interface. For all these objects, and for the Application Layer objects in clause 8, the coding in table 1 applies for the Length field, which indicates the number of bytes in the following Value field.

Size_indicator is the first bit of the length_field. If size_indicator = 0, the length of the data field is coded in the succeeding 7 bits. Any length from 0 to 127 can thus be encoded on one byte. If the length exceeds 127, then size_indicator is set to 1. In this case, the succeeding 7 bits code the number of subsequent bytes in the length field. Those subsequent bytes shall be concatenated, first byte at the most significant end, to encode an integer value. Any value field length up to 65535 can thus be encoded by three bytes.

The indefinite length format specified by the basic encoding rules of ASN.1 (see [3]) is not used.

SIST EN 50221:1999

7.1 Generic Transport Layer

7.1.1 Introduction

The Transport Layer of the Command Interface operates on top of a Link Layer provided by the particular physical implementation used. For the baseline PC Card physical implementation the Link Layer is described in annex A. The transport protocol assumes that the Link Layer is reliable, that is, data is conveyed in the correct order and with no deletion or repetition of data.

The transport protocol is a command-response protocol where the host sends a command to the module, using a Command Transport Protocol Data Unit (C_TPDU) and waits for a response from the module with a Response Transport Protocol Data Unit (R_TPDU). The module cannot initiate communication: it must wait for the host to poll it or send it data first. The protocol is supported by eleven Transport Layer objects. Some of them appear only in C_TPDUs from the host, some only in R_TPDUs from the module and some can appear in either. Create_T_C and C_T_C_Reply, create new Transport Connections. Delete_T_C and D_T_C_Reply, clear them down. Request_T_C and New_T_C allow a module to request the host to create a new Transport Connection. T_C_Error allows error conditions to be signalled. T_SB carries status information from module to host. T_RCV requests waiting data from a module and T_Data_More and T_Data_Last convey data from higher layers between host and module. T_Data_Last with an empty data field is used by the host to poll regularly for data from the module when it has nothing to send itself.

A C_TPDU from the host contains only one Transport Protocol Object. A R_TPDU from a module may carry one or two Transport Protocol Objects. The sole object or second object of a pair in a R_TPDU is always a T_SB object.

7.1.2 Transport protocol objects

iTeh STANDARD PREVIEW
(standards.iteh.ai)

All transport layer objects contain a transport connection identifier. This is one octet, allowing up to 255 Transport Layer connections to be active on the host simultaneously. Transport connection identifier value 0 is reserved. The identifier value is always assigned by the host. The protocol is described in detail here as it is common to all physical implementations but the objects are only described in general terms. The detailed coding of the objects depends upon the particular physical layer used. The coding for the PC Card physical implementation is described in annex A.

SIST EN 50221:1999
https://standards.iteh.ai/catalog/standards/sist/85553497-586d-484c-9cef-4cf5ea9b8a46/sist-en-50221-1999

The host shall allow at least 16 transport connections to be created per module socket supported but preferably all 255 connections distributed amongst the module sockets.

1 Create_T_C creates the Transport Connection. It is only issued by the host and carries the transport connection identifier value for the connection to be established.

2 C_T_C_Reply is the response from the target module to Create_T_C and carries the transport connection identifier for the created connection.

3 Delete_T_C deletes an existing Transport Connection. It has as a parameter the transport connection identifier for the connection to be deleted. It can be issued by either host or module. If issued by the module it does so in response to a poll or data from the host.

4 D_T_C_Reply is the reply to the delete. In some circumstances this reply may not reach its destination, so the Delete_T_C object has a time-out associated with it. If the time-out matures before the reply is received then all actions which would have been taken on receipt of the reply can be taken at the time-out.

5 Request_T_C requests the host to create a new Transport Connection. It is sent on an existing Transport Connection from that module. It is sent in response to a poll or data from the host.

SIST EN 50221:1999

6 New_T_C is the response to Request_T_C. It is sent on the same Transport Connection as the Request_T_C object, and carries the transport connection identifier of the new connection. New_T_C is immediately followed by a Create_T_C object for the new connection, which sets up the Transport Connection proper.

7 T_C_Error is sent to signal an error condition and carries a 1-byte error code specifying the error. In this version this is only sent in response to Request_T_C to signal that no more Transport Connections are available.

8 T_SB is sent as a reply to all objects from the host, either appended to other protocol objects or sent on its own, as appropriate. It carries one byte which indicates if the module has data available to send.

9 T_RCV is sent by the host to request that data the module wishes to send (signalled in a previous T_SB from the module) be returned to the host.

10 T_Data_More and T_Data_Last convey data between host and module, and can be in either a C_TPDU or a R_TPDU. From the module they are only ever sent in response to an explicit request by a T_RCV from the host. T_Data_More is used if a Protocol Data Unit (PDU) from a higher layer has to be split into fragments for sending due to external constraints on the size of data transfers. It indicates that at least one more fragment of the upper-layer PDU will be sent after this one. T_Data_Last indicates the last or only fragment of an upper-layer PDU.

7.1.3 Transport protocol

Figures 6 and 7 show the state transition diagrams for connection set-up and clear-down on the host side and the module side respectively. Each state transition arc is labelled with the event that causes that transition. If the transition also causes an object to be sent, then this is indicated with boxed text.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

When the host wishes to set up a transport connection to a module, it sends the Create_T_C object and moves to state 'In Creation'. The module shall reply directly with a C_T_C_Reply object. If after a time-out period the module does not respond, then the host returns to the idle state (via the 'Timeout' arc). The host will not transmit or poll again on that particular transport connection, and a late C_T_C_Reply will be ignored. If, subsequently, the host re-uses the same transport connection identifier, then the module will receive Create_T_C again, and from this it shall infer that the old transport connection is dead, and a new one is being set up.

When the module replies with C_T_C_Reply the host moves to the 'Active' state of the connection. If the host has data to send, it can now do so, but otherwise it issues a poll and then polls regularly thereafter on the connection.

If the host wishes to terminate the transport connection, it sends a Delete_T_C object and moves to the 'In Deletion' state. It then returns to the 'Idle' state upon receipt of a D_T_C_Reply object, or after a time-out if none is received. If the host receives a Delete_T_C object from the module it issues a D_T_C_Reply object and goes directly to the idle state. Except for the 'Active' state, any object received in any state which is not expected is ignored.

In the 'Active' state the host issues polls periodically, or sends data if it has an upper-layer PDU to send. In response it receives a T_SB object, preceded by a Request_T_C or Delete_T_C object if that is what the module wants to do.

In the 'Active' state, data can be sent by the host at any time. If the module wishes to send data it must wait for a message from the host - normally data or a poll - and then indicate that it has data available in the T_SB reply. The host will then at some point - not necessarily immediately - send a T_RCV request to the module to which the module responds by sending the waiting data in a T_Data object. Where T_Data_More is used, each subsequent fragment must wait for another T_RCV from the host before it can be sent.