DIGITAL PAY-TV SYSTEMS

Confidential

CableSoft

Set-Top Box

Conditional Access Kernel

IRD Command Specification

V 1.3.16

NAGRAVISION S.A.
is a member of the KUDELSKI GROUP OF COMPANIES.

CH–1033 Cheseaux, Vaud, Switzerland.

First published, January 2004.

Revised, January 2005.
Part number: StbCakIrdSpe010316.pub.doc,

Nagravision S.A. is a member of the Kudelski Group of Companies.
Tel.: (41) (21) 732-0311 Fax: (41) (21) 732-0300

Polite notice and request to an unintended recipient

Should this document come into your possession and you are not the intended recipient: NagraVision kindly requests and thanks you in advance for making contact at your earliest convenience for instructions on how to proceed with its disposal.

Table Of Contents

1 Introduction 6
1.1 Purpose 6
1.2 Document History 6
1.3 Definitions, Acronyms, and Abbreviations 7
1.4 Notational Conventions 7
1.5 References 7
1.6 Trademarks 7
1.7 Overview 7
2 IRD Command Format 10
3 Generic IRD Commands 11
3.1 Reset PIN Code 11
3.2 Mail 12
3.3 Force Tune 13
3.4 Force Identification 14
3.5 Set Macrovision CPS 15
3.6 Configure STB 16
3.7 Set Network ID 18
3.8 Master/Slave 19
3.8.1 Continuous Mode Initialization 19
3.8.2 Cancellation 19
3.8.3 Single Shot 20
3.8.4 Automatic Master/Slave 21
3.9 Set PIN Code 22
3.10 Force Stand-by 23
3.11 Configure Camlock 24
3.12 Copy Protection 25
3.12.1 Validate POD_ID/Host_ID 25
3.12.2 Revoke POD_ID/Host_ID 25
3.12.3 Force Authentication 26
3.12.4 Set Key Session Period 27
3.13 Restore Factory Settings 28
3.14 Force Tune with Timeout 29
3.15 Pop-up 30

3.15.1 Display Pop-Up 30
3.15.2 Remove Pop-Up 31
3.16 MovieKey 32
3.17 Push-VOD........ 34
3.17.1 Content Download and Playback Configuration 34
3.17.2 Push-VOD Partition Formatting........ 35
3.17.3 Erase Asset 35
3.17.4 Erase Metadata File 36
3.17.5 Set Downloads Wake-Up 37
3.18 Force Software Download 39
3.19 Change Usage ID 40
3.20 Set Community Type 41
4 Specific IRD Commands 42

List Of Tables

Table 1 - Document History 7
Table 2 - Definitions, Acronyms, and Abbreviations 7
Table 3 - Commands Summary 9

1 Introduction

1.1 Purpose

This document defines the general format of an IRD command as well as generic NagraVision commands, such as "Reset PIN Code" or "Force Tune". It also defines the rules for defining manufacturer's or operator's specific commands.

1.2 Document History

Version	Date	Author	Description
1.3.16	20-Dec-2004	Jean-Luc Bussy	Added "Force Software Download", "Change Usage ID" and "Set Community Type" commands.
1.3.15	31-Aug-2004	Jean-Luc Bussy	Added "Pop-up" commands
1.3.14	05-Aug-2004	Jean-Luc Bussy	Added command "Force tune with timeout"
1.3.13	03-Aug-2004	Jean-Luc Bussy	Specified the max length for both DNASP2 and Aladin.
1.3.12	13-Jul-2004	Sébastien Robyr	• Corrected the max length of an IRD-command. • Accordingly corrected the MovieKey IRD-command.
1.3.11	29-Jun-2004	Sébastien Robyr	• Corrected the min/max length of the MovieKey IRD-command.
1.3.10	11-Jun-2004	Sébastien Robyr	• Added validity date into MovieKey IRD-command
1.3.9	18-May-2004	Sébastien Robyr	• Added Push-VOD commands (Section 3.17)
1.3.8	15-Mar-2004	Sébastien Robyr	• Corrected some typo • Updated MovieKey command (typo) • Updated all CRC in examples
1.3.7	05-Mar-2004	Sébastien Robyr	Added MovieKey command.
1.3.6	06-Oct-2003	Jean-Luc Bussy	Added 'Restore Factory Settings' and 'Automatic Master/Slave' commands.
1.3.5	23-Sep-2003	Serge Dubrova	Added the following Copy Protection commands 'Validate POD_ID/Host_ID', 'Revoke POD_ID/Host_ID', 'Force Authentication' and 'Set Key Session Period'.
1.3.4	12-Sep-2003	Jean-Luc Bussy	Added 'Configure Camlock' command.
1.3.3	21-Aug-2003	Jean-Luc Bussy	Added 'Master/Slave', 'Set PIN Code' and 'Force Stand-by' commands.
1.3.2	05-Jul-2002	Jean-Luc Bussy	Added the Set Network ID command.
1.3.1	02-Dec-2001	Jean-Luc Bussy	Improved description of the Mail command, added "Configure STB" command and changed the way to handle specific commands.
1.3.0	06-Juin-2001	Jean-Luc Bussy	Added Macrovision command and renamed the document StbCakIrdSpe.
1.2	06-Oct-2000	Jean-Luc Bussy	Added the Force Identification command and several manufacturer's command id.

1.1	01-Feb-2000	Patrick Schyrr	Add ADB and Microsoft specific command id.
1.0	07-Sep-1999	Philippe Stransky	First issue. Extracted from the EMM and ECM descriptions. Includes a range for STB manufacturers.

Table 1 - Document History

1.3 Definitions, Acronyms, and Abbreviations

Acronym Abbreviation	Definition
CA	Conditional Access
CAK	Conditional Access Kernel
CRL	Certificate Revocation List
DVB	Digital Video Broadcasting
IRD	Integrated Receiver Decoder
MKY	MovieKey
NVM	Non-volatile memory
STB	Set-Top Box

Table 2 - Definitions, Acronyms, and Abbreviations

1.4 Notational Conventions

All source code occurrences appear in `courier` writing style

1.5 References

[1] Force Identification, Implementation Guidelines V1.0.0
[2] IRD Master/Slave, Solution Overview, Issue 1.0.0
[3] IRD Master/Slave, Implementation Guideline, Issue 0.0.3
[4] ANSI/STCE 41 2003, POD Copy Protection System
[5] NagraVision, Data Item Loader, Application Programming Interface, V 1.0.4 or higher.

1.6 Trademarks

Any company's or product name(s) found herein may be the trademarks or registered trademarks of their respective companies.

1.7 Overview

IRD commands allow the head-end to send messages to the set-top box in a secured way. IRD commands are carried by EMMs. They can benefit from EMM addressing mode. It means that a message can be addressed either to one single set-top box or to all set-top boxes.

The CA Kernel embedded in the set-top box is not dependent at all on IRD commands. It gets the command from the smartcard and forwards it to the set-top box application without additional processing. The set-top box application is completely responsible for IRD command management. Periodicity of commands (coming from the fact that commands are carried by EMMs) has to be managed by the set-top box application by means of the sequence number. If a command has to be split in several commands due to the EMM length limitation[1], it is also the responsibility of the set-top box application to re-build the original command.

NagraVision has defined a set of generic commands. The table below gives a synopsis of these commands along with the associated command identifier. Refer to §3 for a detailed description.

Name	**command_id**	**operation**
Reset PIN Code	0x12	0x01
Mail	0xC0	0x01
Force Tune	0xC1	0x01
Force Identification	0xC2	0x01
Set Macrovision CPS	0xC4	0x01
Configure STB	0xC5	0x01
Set Network ID	0xC6	0x01
Master/Slave Initialization	0xC7	0x01
Master/Slave Cancellation	0xC7	0x02
Master/Slave Single Shot	0xC7	0x03
Automatic Master/Slave	0xC7	0x04
Set PIN Code	0xC8	0x01..0xFF
Force Stand-by	0xC9	0x01
Configure Camlock	0xCA	0x00, 0x01
Copy Protection – Validate POD_ID/Host_ID	0xCB	0x00
Copy Protection – Revoke POD_ID/Host_ID	0xCB	0x01
Copy Protection – Force Authentication	0xCB	0x02
Copy Protection – Set Key Session Period	0xCB	0x03
Restore Factory Setting	0xCC	0x01
Force Tune with Timeout	0xCD	0x01
Pop-up	0xCF	0x00..0x01
MovieKey	0xD0	0x00
Push-VOD – Content Configuration	0xD1	0x00
Push-VOD – Partition Formatting	0xD1	0x01
Push-VOD – Erase Asset	0xD1	0x02
Push-VOD – Erase Metadata File	0xD1	0x03
Push-VOD – Set Downloads Wake-Up	0xD1	0x04
Force Software Download	0xD2	0x00

[1] The maximum size of the `command_body` that can be carried by one IRD-CMD is 75 Bytes for DNASP-3 and 61 Bytes for DNASP-2 (the complete IRD buffer returned by the ICC includes 3 more Bytes, the `EMM_command`, the `length` and the `checksum`).

Name	command_id	operation
Change Usage ID	0xD3	0x00
Set Community Type	0xD4	0x00

Table 3 - Commands Summary

All commands required by a manufacturer or an operator that does not belong to this list may result in a specific command. Refer to §3.11 for a description of the procedure allowing the definition of a specific command.

2 IRD Command Format

Description

Defines the general format of an IRD command.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    7+N, max=71 for Aladin, max=55 for
                                            DNASP2
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf
    operation               8     uimsbf
    for(i=0; i<N; i++){                     Nmax=64 for Aladin, Nmax=48 for DNASP2
      data                  8     bslbf
    }
    checksum                8     bslbf
  }
}
```

Parameters

`sequence_number`

value incremented whenever a command is generated by the head-end.

Since IRD commands are carried by EMMs, the set-top box application may be notified of the same command several times. It is the responsibility of the set-top box application to process the sequence number in order to avoid a command to be run several times.

To do so the sequence number of the last *x* commands run by the application may be stored in NVM. The *x* value depends on the maximum number of different commands that could be broadcast at the same time on the network. It is operator dependent.

`command_id`

command identifier.

`operation`

used in conjunction with the command_id. The couple (command_id, operation) uniquely identifies a command.

`data`

additional data (optional)

`checksum`

two's complement of the sum of all bytes from the command_id to the last data byte. The sum of all bytes from the command_id to the checksum must be equal to 0.

For instance, the checksum of the reset PIN code command here after is equal to `ED (`12+01+ED=0)

`64 07 00000007 **12** **01** ED

3 Generic IRD Commands

3.1 Reset PIN Code

Description

Forces the STB to clear the parental code. This may be required if the subscriber lost the PIN code, or when reclaiming the STB from the field.

Format

```
IRD_command(){
  EMM_command                8      uimsbf    0x64
  length                     8      uimsbf    7
  command_body(){
    sequence_number          32     uimsbf
    command_id               8      uimsbf    0x12
    operation                8      uimsbf    0x01
    checksum                 8      bslbf     0xED
  }
}
```

Parameters

None.

Notes

1. If multiple PIN codes are available, then the operation field indicates the PIN code number.

Example

The following command reset the PIN code number 1.

```
IRD_command = `6407000000071201ED
```

3.2 Mail

Description

This command provides mail messages to the STB. The management of the messages is the STB responsibility.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    10+N
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xC0
    operation               8     uimsbf    0x01
    data{
      mail_id               10    bslbf     Mail message number
      total_segment         6     bslbf     Total number of segments
      priority              2     bslbf     0 normal priority
                                            1 high priority
                                            2 emergency
                                            3 reserved
      segment_number        6     bslbf
      for (i=0; i<N; i++){
        message             8     bslbf     Mail message body
      }
    }
    checksum                8     bslbf
  }
}
```

Parameters

`mail_id`	Unique mail number
`total_segment`	Total number of segments required to carry the whole message. It's a 6-bit variable covering the range [1..63]. Each segment may carry up to 45 bytes.
`priority`	Influences the STB behavior. For example, normal priority would not affect the display, while emergency mail would be displayed on the screen without manual intervention.
`segment_number`	Identifies the current segment. The first segment is equal to 0 and the last segment is equal to total_segment-1.

Notes

1. If the total length of a mail is larger than 45 bytes, then the message is split in several segments, each having the same mail id and consecutive segment numbers. As there is at the most 63 segments of 45 bytes per message, the maximum length of a message is equal to 2835 bytes.

3.3 Force Tune

Description

This command forces the STB to tune to a service defined by the network_id/transport_id/service_id. If the STB is able to query the access rights needed for the service, then the tuning should occur only if the subscriber has access to the service.

Format

```
IRD_command(){
  EMM_command                 8     uimsbf    0x64
  length                      8     uimsbf    13
  command_body(){
    sequence_number           32    uimsbf
    command_id                8     uimsbf    0xC1
    operation                 8     uimsbf    0x01
    data{
      network_id              16    uimsbf
      transport_id            16    uimsbf
      service_id              16    uimsbf
    }
    checksum                  8     bslbf
  }
}
```

Parameters

`network_id` corresponds to the network_id as described in the DVB Network Information Table (NIT).

`transport_id` corresponds to the network_id as described in the DVB Network Information Table (NIT).

`service_id` corresponds to the service_id as described in the DVB Service Description Table (SDT). It may also correspond to the program number found in the MPEG Program Map Table (PMT).

3.4 Force Identification

Description

Forces the STB to display its Nagra S/N along with the UA of its smartcard on the screen for a while.

Format

```
IRD_command(){
  EMM_command                8     uimsbf    0x64
  length                     8     uimsbf    7
  command_body(){
    sequence_number          32    uimsbf
    command_id               8     uimsbf    0xC2
    operation                8     uimsbf    0x01
    checksum                 8     bslbf     0x3D
  }
}
```

Parameters

None

Note

See document [1] for more information about this command.

3.5 Set Macrovision CPS

Description

The Macrovision system uses a chip inside the set-top box that acts on the analog video output to prevent the recording, but not the viewing. The chip accepts configuration data and operational data. Configuration data allows to parameters the different ways to mess up the signal: how long to rotate the colors, how high is the peak in the signal, and so on. This is a 136-bit string called CPS by Macrovision. Operational data tell us which way has to be applied: turn color stripe on, turn v sync off, and so on. This is a 8-bit string called the Mode byte by Macrovision.

The purpose of this IRD command is to provide the CPS string to a set-top box, in order to parameter its Macrovision chip. The Mode byte is not transmitted through this command and will be part of a private descriptor present in the EIT.

Format

```
IRD_command(){
  EMM_command                8     uimsbf    0x64
  length                     8     uimsbf    7+N
  command_body(){
    sequence_number          32    uimsbf
    command_id               8     uimsbf    0xC4
    operation                8     uimsbf    0x01
    data{
      for (i=0; i<N; i++){
        cps                  8     uimsbf    CPS string
      }
    }
    checksum                 8     bslbf
  }
}
```

Parameters

cps

CPS (Copy Protection Setup) string defined by Macrovision. The actual Macrovision chip expects this string to be 136 bits long (17 bytes). However the current specification defines it of variable length in order to support future version. The length can be deduced from global "length" field of the IRD command.

3.6 Configure STB

Description

This command allows the head-end to enable or disable features in a set-top box. Each feature is associated to a single bit set to 1 when enabled and 0 when disabled. All features are disabled by default. The features configuration has to be stored in NVM so that no information is lost after power-cycling the set-top box.

Format

```
IRD_command(){
  EMM_command                8     uimsbf    0x64
  length                     8     uimsbf    9+N
  command_body(){
    sequence_number          32    uimsbf
    command_id               8     uimsbf    0xC5
    operation                8               0x01
    data{
      compatible_mode        1     bslbf
      video                  1     bslbf
      audio                  1     bslbf
      smartcard_1            1     bslbf
      smartcard_2            1     bslbf
      harddisk               1     bslbf
      dvd                    1     bslbf
      serial_port_1          1     bslbf
      serial_port_2          1     bslbf
      parallel_port          1     bslbf
      usb_port               1     bslbf
      1394_port              1     bslbf
      spare_port_1           1     bslbf
      spare_port_2           1     bslbf
      peripheral_1           1     bslbf
     peripheral_2            1     bslbf
    }
    for(i=0; i<N; i++){
      pattern                8     bslbf     optional
    }
    checksum                 8     bslbf
  }
}
```

Parameters

compatible_mode — Usually set to 1 when the set-top box is fully DVB compliant and set to 0 when the set-top box usage is restricted to a specific network only. In case the set-top box is configured for a specific network but is connected to another network, the set-top box application shall display a proper message and all features shall be disabled. The compatible mode could be used by operators willing to avoid rented set-top boxes to be used in other networks.

video — Video decoding shall be disabled when set to 0.

audio — Audio decoding shall be disabled when set to 0.

smartcard_1 — Smartcard reader 1 shall be disabled when set to 0.

smartcard_2	Smartcard reader 2 shall be disabled when set to 0.
harddisk	Hard disk shall be disabled when set to 0.
dvd	Dvd shall be disabled when set to 0.
serial_port_1	Access to serial port 1 shall be disabled when set to 0.
serial_port_2	Access to serial port 2 shall be disabled when set to 0.
parallel_port	Access to parallel port shall be disabled when set to 0.
usb_port	Access to usb port shall be disabled when set to 0.
1394_port	Access to IEEE 1394 port shall be disabled when set to 0.
spare_port_1	Access to spare port 1 shall be disabled when set to 0.
spare_port_2	Access to spare port 2 shall be disabled when set to 0.
peripheral_1	Peripheral 1 shall be disabled when set to 0.
peripheral_2	Peripheral 2 shall be disabled when set to 0.
pattern	Optional additional bit fields. Their absence shall be interpreted as value 1 by the set-top box application.

3.7 Set Network ID

Description

This command sets the set-top box network ID to a specific value. This allows the set-top box to retrieve the Network Information Table (NIT) defining the topology of a particular local area. This command can also be used to assign testing network ID to specific set-top boxes.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    11
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xC6
    operation               8     uimsbf    0x01
    data{
      network_id            16    uimsbf    Network ID
      original_network_id   16    uimsbf    Original network ID
    }
    checksum                8     bslbf
  }
}
```

Parameters

network_id	Unique identifier indicating the network ID.
original_network_id	Unique identifier indicating the original network ID.

3.8 Master/Slave

Refer to document [2] for a Master/Slave feature solution overview and document [3] for implementation guidelines.

3.8.1 Continuous Mode Initialization

Description

This command is used to set the parameters in order to initialise the Master/Slave continuous mode.

Format

```
IRD_command(){
  EMM_command              8     uimsbf    0x64
  length                   8     uimsbf    14
  command_body(){
    sequence_number        32    uimsbf
    command_id             8     uimsbf    0xC7
    operation              8     uimsbf    0x01
    data{
      masterSmartcard      32    uimsbf
      validationPeriod     8     uimsbf    in days
      randomPeriod         8     uimsbf    in days
      timeout              8     uimsbf    in hours
    }
    checksum               8     bslbf
  }
}
```

Parameters

`masterSmartcard` this is the Smartcard ID of the master Smartcard without checksum.

`validationPeriod` this value define the average time, expressed in days, between two validation procedures.

`randomPeriod` the next validation procedure will occur in `validationPeriod` days +/- `randomPeriod` days. The targeted day will be randomly chosen in this bracket of time.

`timeout` the timeout is the period of time during which the customer has to succeed with the validation procedure (insert the master Smartcard in the slave STB). At the end of the `timeout` period, the STB will stop playing video and/or audio signal.

3.8.2 Cancellation

Description

This command id used to disable the IRD Master/Slave mode continuous and single shot mode.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    7
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xC7
    operation               8     uimsbf    0x02
    checksum                8     bslbf     0x37
  }
}
```

Parameters

None

3.8.3 Single Shot

Description

This command is used to set the parameters in order to initialise the single shot Master/Slave command. This is not possible to disable this command only; all Master/Slave modes must be disabled in order to cancel it. In other words, it's not possible to cancel a single shot command without cancelling the continuous mode.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    12
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xC7
    operation               8     uimsbf    0x03
    data{
      masterSmartcard       32    uimsbf
      timeout               8     uimsbf    in hours
    }
    checksum                8     bslbf
  }
}
```

Parameters

`masterSmartcard`	this is the Smartcard ID of the master Smartcard without checksum.
`timeout`	the timeout is the period of time during which the customer has to succeed with the validation procedure (insert the master Smartcard in the slave STB). At the end of the `timeout` period, the STB will stop playing video and/or audio signal.

3.8.4 Automatic Master/Slave

Description

This command is used to set the parameters in order to initialise the automatic Master/Slave feature.

Format

```
IRD_command(){
  EMM_command                8     uimsbf    0x64
  length                     8     uimsbf    14
  command_body(){
    sequence_number          32    uimsbf
    command_id               8     uimsbf    0xC7
    operation                8     uimsbf    0x04
    data{
      stbMode                2     bslbf     0 master
                                             1 slave
                                             2 stand-alone
                                             3 reserved
      reserved               6     bslbf     all bits set to 1
      masterSmartcard        32    uimsbf
      timeout                16    uimsbf    in seconds
    }
    checksum                 8     bslbf
  }
}
```

Parameters

`stbMode` This is the mode in which the STB is running

`masterSmartcard` This is the Smartcard ID of the master Smartcard without checksum.

`timeout` This is the period of time during which a slave STB can run without getting any data from a master STB.

3.9 Set PIN Code

Description

This command allows the head-end to change the set-top box PIN code. The operation field identifies the PIN code that has to be modified in case the set-top box manages several PIN codes.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    8+N
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xC8
    operation               8     uimsbf    0x01..0xFF
    data{
      pin_length            8     uimsbf    PIN length
      for(i=0; i<N; i++){
        character           8     uimsbf    PIN character
      }
    }
    checksum                8     bslbf
  }
}
```

Parameters

pin_length	Number of bytes the PIN code is composed of.
character	ASCII code of each character composing the PIN code.

Example

The following command will change the PIN code number 1 to "1234".

```
IRD_command = `640C00000007C801043132333469
```

In this example the 4-byte sequence number is equal to `00000007.

3.10 Force Stand-by

Description

This command allows the head-end to force a set-top box to enter in the stand-by mode. It could be an indirect way to force a set-top box to get a software download. Indeed in most of set-top boxes the download process is triggered by entering the standby mode.

Format

```
IRD_command(){
  EMM_command                8      uimsbf    0x64
  length                     8      uimsbf    7
  command_body(){
    sequence_number          32     uimsbf
    command_id               8      uimsbf    0xC9
    operation                8      uimsbf    0x01
    checksum                 8      bslbf     0x36
  }
}
```

Parameters

None

3.11 Configure Camlock

Description

This command allows the head-end to enable or disable the camlock feature. Setting the operation field to 0 disables the camlock feature, while setting that field to 1 enables this feature.

Format

```
IRD_command(){
  EMM_command                 8     uimsbf    0x64
  length                      8     uimsbf    7
  command_body(){
    sequence_number           32    uimsbf
    command_id                8     uimsbf    0xCA
    operation                 8     uimsbf    0x00..0x01
    checksum                  8     bslbf
  }
}
```

Parameters

operation	0x00 disable camlock 0x01 enable camlock

3.12 Copy Protection

For further enlightenment on the subject of Copy Protection, please refer to document [4].

3.12.1 Validate POD_ID/Host_ID

Description

This command allows the head-end to validate a POD_ID and Host_ID couple, according to their absence into head-end managed CRLs. The operation field identifies that command as a validation command.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    20
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xCB
    operation               8     uimsbf    0x00
    data{
      POD_ID                64    uimsbf    Validated POD ID
      Host_ID               40    uimsbf    Validated Host_ID
    }
    checksum                8     bslbf
  }
}
```

Parameters

`POD_ID`	8 bytes value characterizing a valid POD_ID.
`Host_ID`	5 bytes value characterizing a valid Host_ID.

Example

The following command will validate a POD_ID/Host_ID couple of 0x0102030405060708/0x0102030405.

```
IRD_command = `641400000011CB000102030405060708010203040502
```

In this example the 4-byte sequence number is equal to `` `00000011 ``.

3.12.2 Revoke POD_ID/Host_ID

Description

This command allows the head-end to revoke a POD_ID and Host_ID couple, according to their presence into head-end managed CRLs. The operation field identifies that command as a revocation command.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    20
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xCB
    operation               8     uimsbf    0x01
    data{
      POD_ID                64    uimsbf    Revoked POD ID
      Host_ID               40    uimsbf    Revoked Host_ID
    }
    checksum                8     bslbf
  }
}
```

Parameters

`POD_ID`	64 bits value characterizing a valid POD_ID.
`Host_ID`	40 bits value characterizing a valid Host_ID.

Example

The following command will revoke a POD_ID/Host_ID couple of 0x0102030405060708/0x0102030405.

```
IRD_command = `641400000013CB010102030405060708010203040501
```

In this example the 4-byte sequence number is equal to `` `00000013 ``.

3.12.3 Force Authentication

Description

This command allows the head-end to force a POD_ID and Host_ID couple to restart the copy protection authentication process from beginning, as if inserted for the first time.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    7
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xCB
    operation               8     uimsbf    0x02
    checksum                8     bslbf     0x33
  }
}
```

Parameters

None.

3.12.4 Set Key Session Period

Description

This command allows the head-end to set a key session period for a given POD/SC.

Format

```
IRD_command(){
  EMM_command                8     uimsbf    0x64
  length                     8     uimsbf    9
  command_body(){
    sequence_number          32    uimsbf
    command_id               8     uimsbf    0xCB
    operation                8     uimsbf    0x03
    key_session_period       16    uimsbf
    checksum                 8     bslbf
  }
}
```

Parameters

key_session_period — 16 bits value giving the session key refresh time with a resolution of 10 second. Null means unlimited.

Example

The following command will set an key session period of 120 seconds.

```
IRD_command = `640900000017CB03000C26
```

In this example the 4-byte sequence number is equal to `00000017.

3.13 Restore Factory Settings

Description

This command allows the head-end to restore factory settings of the set-top box. Settings affected by this command are set-top box dependent. For instance the favourite channel list and password may be cleared, and tuner settings reset to default values.

Format

```
IRD_command(){
  EMM_command              8     uimsbf    0x64
  length                   8     uimsbf    7
  command_body(){
    sequence_number        32    uimsbf
    command_id             8     uimsbf    0xCC
    operation              8     uimsbf    0x01
    checksum               8     bslbf     0x33
  }
}
```

3.14 Force Tune with Timeout

Description

This command forces the STB to tune to a service defined by the network_id / transport_id / service_id for a defined duration (in seconds). If the STB is able to query the access rights needed for the service, then the tuning should occur only if the subscriber has access to the service. After the defined duration the STB shall tune back to the last previously watched service.

Format

```
IRD_command(){
  EMM_command               8     uimsbf    0x64
  length                    8     uimsbf    15
  command_body(){
    sequence_number         32    uimsbf
    command_id              8     uimsbf    0xCD
    operation               8     uimsbf    0x01
    data{
      network_id            16    uimsbf
      transport_id          16    uimsbf
      service_id            16    uimsbf
      timeout               16    uimsbf
    }
    checksum                8     bslbf
  }
}
```

Parameters

`network_id` corresponds to the network_id as described in the DVB Network Information Table (NIT).

`transport_id` corresponds to the transport_id as described in the DVB Network Information Table (NIT).

`service_id` corresponds to the service_id as described in the DVB Service Description Table (SDT). It may also correspond to the program number found in the MPEG Program Map Table (PMT).

`timeout` In seconds.

3.15 Pop-up

3.15.1 Display Pop-Up

Description

This command allows the set-top box to display pop-up messages.

Format

```
IRD_command(){
  EMM_command              8     uimsbf    0x64
  length                   8     uimsbf    10+N
  command_body(){
    sequence_number        32    uimsbf
    command_id             8     uimsbf    0xCF
    operation              8     uimsbf    0x00
    data{
      popup_id             10    bslbf     Pop-up identifier
      total_segment        6     bslbf     Total number of segments
      persistence          2     bslbf     0 normal
                                           1 timeout
                                           2 user acknowledged
                                           3 reserved
      segment_number       6     bslbf
      for (i=0; i<N; i++){
        message            8     bslbf     Pop-up message body
      }
    }
    checksum               8     bslbf
  }
}
```

Parameters

`popup_id` Unique pop-up identifier

`total_segment` Total number of segments required to carry the whole message. It’s a 6-bit variable covering the range [1..63]. Each segment may carry up to 45 bytes.

`persistence` Gives some information about the pop-up behavior:

0 Pop-up remains displayed until it is replaced by another one or removed by the "Remove Pop-Up" command defined in 3.15.2.

1 Pop-up automatically disappears after a while. The duration of the timeout is free, but should not be shorter than 10s.

2 Pop-up remains displayed until the user's acknowledgement (by pressing any key)

`segment_number` Identifies the current segment. The first segment is equal to 0 and the last segment is equal to total_segment-1.

Notes

1. If the total length of a pop-up message is larger than 45 bytes, then it is split in several segments, each having the same pop-up identifier and consecutive segment numbers. As there is at the most 63 segments of 45 bytes per message, the maximum length of a message is equal to 2835 bytes.
2. The channel change is allowed during the display of a normal or a timeout pop-up, provided it remains displayed over the video stream. On the contrary, a user acknowledged pop-up may be removed by a channel change.
3. If a new pop-up (new popup_id) is received during the display of another pop-up, this latter one shall be replaced at once with the new one, including the persistence parameter. This means for instance that a normal pop-up may be replaced by a timeout pop-up if desired.

Example

1 segment pop-up message with the following parameters:

popup_id : 4
total_segment : 1
persistence : 0 (normal)
message : "Pay your bill!"

command : `` `641800000017CF00010100506179207 96F75722062696C6C2132 ``

3.15.2 Remove Pop-Up

Description

This command removes any kind of pop-up displayed through the "Display Pop-Up" command. It is useful in case the head-end decides to remove a persistent pop-up.

Format

```
IRD_command(){
  EMM_command            8     uimsbf    0x64
  length                 8     uimsbf    7
  command_body(){
    sequence_number      32    uimsbf
    command_id           8     uimsbf    0xCF
    operation            8     uimsbf    0x01
    checksum             8     bslbf     0x30
  }
}
```

3.16 MovieKey

Description

This command permits to send the MovieKey needed that allows decoding an asset (i.e. watch movie), identified by its `asset_id`. **This command is intended to the CDE only.** The MovieKey will be exported by the CDE to any decoder application that has previously registered to `caCdeRegisterIrdMovieKeyExportation()`, but the IRD-command itself is NOT exported to the decoder application

Format

```
IRD_command(){
  EMM_command                  8    uimsbf    0x64
  length                       8    uimsbf    17+N (max length = 71)
   command_body(){
    sequence_number           32    uimsbf
    command_id                 8    uimsbf    0xD0
    operation                  8    uimsbf    0x00
    data {
      moviekey_id             32    uimsbf
      end_of_validity         32    uimsbf    Validity date of the MovieKey
      total_segments           8    bslbf
      segment_index            8    bslbf
      for(i=0; i<N; i++) {                    (Nmax=54)
        asset_MKey             8    bslbf
      }
      checksum                 8    bslbf
    }
   }
}
```

Parameters

`moviekey_id`	Unique identification of the MovieKey.
`end_of_validity`	Date indicating the end of validity of the MovieKey, in UTC. The date is coded in unix date (number of seconds since 1st of January 1970, at 00:00:00). Maximum value is 7th of February 2106, 06:28:15
`total_segments`	Total amount of segments composing the MovieKey
`segment_index`	Identifies the current segment. The first segment is equal to 0 and the last segment is equal to `total_segments-1`.
`asset_MKey`	Bytes composing the MovieKey.

Example

The following commands will send a MovieKey related to an asset. The parameters are:

- `asset_id = '12345678`
- `end_of_validity` = 11th of June 2004, 14:20:00 = 1'086'963'600 = `'40C9BF90`
- MovieKey =
  `'04650101820098F5A0AB56D70242F8BB694B3B8724DE65D745F5AD7A13A405F37473CFE`
  `915A4DC6B3237D45F738001DA4403AF9918E8C6000D87DCF9122EE1FC03F90C02F0AC206`
  `DC986A66801DAE10542D6491FB75E081D5BA35D98C55347A8BBA8BE08EA5858`

- Total size of the MovieKey = 103 bytes ➔ will be split into two (`02) IRD commands (63+40).
- The first segment_index = `00
- The second segment_index = `01
- First 4-byte sequence number = `00000017
- Second 4-byte sequence number = `00000018

1. IRD_command =
```
`644700000017D0001234567840C9BF900200046501018200 98F5A0AB56D70242F8BB694
B3B8724DE65D745F5AD7A13A405F37473CFE915A4DC6B3237D45F738001DA4403AF9918E
8C6000D8761
```
2. IRD_command =
```
`643A00000018D0001234567840C9BF900201CDF9122EE1FC03F90C02F0AC206DC986A66
801DAE10542D6491FB75E081D5BA35D98C55347A8BBA8BE08EA585860
```

3.17 Push-VOD

3.17.1 Content Download and Playback Configuration

Description

This command allows the Head-End to enable or disable Push-VOD content download and/or playback features in a Set-Top Box. Each feature is associated to a single bit set to 1 when enabled and to 0 when disabled.
All features are enabled by default. The features configuration has to be stored in NVM so that no information is lost after power-cycling the set-top box.

Format

```
IRD_command(){
  EMM_command             8     uimsbf    0x64
  length                  8     uimsbf    8
  command_body(){
    sequence_number       32    uimsbf
    command_id            8     uimsbf    0xD1
    operation             8               0x00
    data{
      content_download    1     bslbf     1 enable content download
                                          0 disable content download
      content playback    1     bslbf     1 enable content playback
                                          0 disable content playback
      reserved            6     bslbf     Always set to '11 1111'
    }
    checksum              8     bslbf
  }
}
```

Parameters

content_download

This bit enables (1) or disables (0) the Push-VOD content download. If set to 1, the decoder application may open DIL download sessions (refer to [5]). If set to 0, the decoder application shall immediately close all DIL download sessions. Moreover, the decoder application shall not open any new download session before the Head-End enables it through a new command.

content_playback

This bit enables (1) or disables (0) the Push-VOD content playback. If set to 1, the decoder application is allowed to playback any Push-VOD content already on the HDD. If set to 0, the decoder application shall immediately stop any Push-VOD content playback and shall not start any new playback before the Head-End enables it through a new command.

Example

The following command disables Push-VOD content download, but enables playback of already downloaded content:

IRD-Command: `'640800000004D1007FB0`

with: `7F` = `'0111 1111'`

- o 0 disables new content download
- o 1 enables playback of already downloaded content

3.17.2 Push-VOD Partition Formatting

Description

This command forces the STB to format the HDD partition(s) containing the Push-VOD content (assets and metadata files).

Format

```
IRD_command(){
  EMM_command                 8     uimsbf    0x64
  length                      8     uimsbf    7
  command_body(){
    sequence_number           32    uimsbf
    command_id                8     uimsbf    0xD1
    operation                 8               0x01
    checksum                  8     bslbf     0x2E
  }
}
```

Parameters

None

3.17.3 Erase Asset

Description

This command forces the STB to erase a Push-VOD asset identified by its unique `asset_id`, if it has already been downloaded by the DIL.
In addition, the decoder application shall set the lowest download priority for that asset through the DIL. It shall therefore call:

```
dilSetAssetPriority(asset_id, DIL_PRIORITY_IGNORE_ASSET);
```

Format

```
IRD_command(){
  EMM_command              8     uimsbf    0x64
  length                   8     uimsbf    11
  command_body(){
    sequence_number        32    uimsbf
    command_id             8     uimsbf    0xD1
    operation              8               0x02
    asset_id               32    uimsbf    Unique asset identifier
    checksum               8     bslbf
  }
}
```

Parameter

`asset_id`	This is the asset identifier, which is unique over the complete CAS.

3.17.4 Erase Metadata File

Description

This command forces the STB to erase a particular file on the HDD, if it has already been downloaded by the DIL.

Format

```
IRD_command(){
  EMM_command              8     uimsbf    0x64
  length                   8     uimsbf    7 + N
  command_body(){
    sequence_number        32    uimsbf
    command_id             8     uimsbf    0xD1
    operation              8               0x03
    for(i=0; i<N; i++){
      filename_char        8     uimsbf    Characters composing the filename
                                           to erase
    }
    checksum               8     bslbf
  }
}
```

Parameters

`filename_char`	Characters composing the filename of the file to erase on the STB HDD, relative to the `pxMetadataPath` the decoder application gave as initialisation parameter to the DIL.

Example

The following command requests the decoder application to erase a file named "`A/0000022F/0000022F.pmt-01`".

Ird-Command =

`'641E00000005D103412f30303030303232462f30303030303232462e706d742d30314C`

3.17.5 Set Downloads Wake-Up

Description

This command defines a certain number of time slots (maximum 17) during which the decoder shall be awake (and downloading).
The decoder shall store the time slots settings in NVM so that no information is lost after power-cycling the set-top box.
Each command resets the settings of the previous command. If the operator wants to clear all time slots (the STB shall never awake itself), it can send a command without time slot.
The default setting is no time slot.

Format

```
IRD_command(){
  EMM_command              8     uimsbf    0x64
  length                   8     uimsbf    7 + 4*N
  command_body(){
    sequence_number        32    uimsbf
    command_id             8     uimsbf    0xD1
    operation              8               0x04
    for(i=0; i<N; i++){                    (Nmax = 17)
      reserved             2               Always 0x00
      start_day_of_week    3     uimsbf    Weekdays
      start_minutes        11    uimsbf    Max value = 0x05A0 (1440 min/day)
      reserved             2               Always 0x00
      stop_day_of_week     3     uimsbf    Weekdays
      stop_minutes         11    uimsbf    Max value = 0x05A0 (1440 min/day)
    }
    checksum               8     bslbf
  }
}
```

Parameters

start_day_of_week
This value defines the day of the week the Nth time slot begins. If its value is 0x00, the time slot is valid for all weekdays. Else, it corresponds to the weekday number (0x01 being Monday and 0x07 being Sunday).

start_minutes
Beginning of the Nth time slot, defined as the number of minutes since midnight (00:00).

stop_day_of_week
This value defines the day of the week the Nth time slot ends. If its value is 0x00, the time slot is valid for all weekdays. Else, it corresponds to the weekday number (0x01 being Monday and 0x07 being Sunday).

stop_minutes
End of the Nth time slot, defined as the number of minutes since midnight (00:00).

Notes

1. If the operator wants to reset all time-slots, it can send a command with N=0.
2. The default settings is no time slot (by default, the decoder application never awakes itself)

Example

Let's consider following time slots:

Mon 02:00am – Mon 05:00am
Wed 01:00am – Wed 05:00am
Thu 10:30pm – Fri 03:00am
Everyday from 06:00am to 08:00am

These time-slots are to be translated like this:

Slot		Weekday		Time				Value
• Slot 1	start	Mon	001	02:00am	120min	000 0111 1000	→	0x0878
	stop	Mon	001	05:00am	300min	001 0010 1100	→	0x092C
• Slot 2	start	Wed	011	01:00am	60min	000 0011 1100	→	0x183C
	stop	Wed	011	05:00am	300min	001 0010 1100	→	0x192C
• Slot 3	start	Thu	100	10:30pm	1'350min	101 0100 0110	→	0x1D46
	stop	Fri	101	03:00am	180min	000 1011 0100	→	0x28B4
• Slot 4	start	All	000	06:00am	360min	001 0110 1000	→	0x0168
	stop	All	000	08:00am	480min	001 1110 0000	→	0x01E0

So the resulting IRD-Command would be:

'641B00000007D1040878092C183C192C1D4628B4016801E054

3.18 Force Software Download

Description

This command allows the head-end to ask the set-top box to check whether a download stream is available and performs the software update if necessary.

Format

```
IRD_command(){
  EMM_command                  8     uimsbf    0x64
  length                       8     uimsbf    7+N
  command_body(){
    sequence_number            32    uimsbf
    command_id                 8     uimsbf    0xD2
    operation                  8     uimsbf    0x00
    for(i=0; i<N; i++){
      version_number           8     uimsbf    Version
    }
    checksum                   8     bslbf
  }
}
```

Parameter

`version_number`	String containing a version number. This version could be used by the application to know if a software update is necessary. This string is optional and its format is manufacturer dependent.

Example

The following command forces a software download without specifying any version number:

```
IRD_command = `640700000018D2002E
```

In this example the 4-byte sequence number is equal to `` `00000018``.

3.19 Change Usage ID

Description

This command allows the head-end to change the set-top box usage ID. All set-top boxes programmed with the same usage ID are related to the same download stream. For instance, field test set-top boxes may be assigned a usage ID that differs from production set-top boxes in order to be upgraded independently.

Format

```
IRD_command(){
  EMM_command                8     uimsbf    0x64
  length                     8     uimsbf    8
  command_body(){
    sequence_number          32    uimsbf
    command_id               8     uimsbf    0xD3
    operation                8     uimsbf    0x00
    usage_id                 8     uimsbf    Usage ID
    checksum                 8     bslbf
  }
}
```

Parameter

usage_id	8-bit identifier used to create groups of set-top boxes.

3.20 Set Community Type

Description

This command allows the head-end to change the set-top box community type that is used to customize the behavior of the set-top box application.

Format

```
IRD_command(){
  EMM_command                 8     uimsbf    0x64
  length                      8     uimsbf    8
  command_body(){
    sequence_number           32    uimsbf
    command_id                8     uimsbf    0xD4
    operation                 8     uimsbf    0x00
    community_type            8     uimsbf    Community type
    checksum                  8     bslbf
  }
}
```

Parameter

`community_type`

8-bit identifier corresponding to the set-top box community type. A community type of 0 means that the set-top box is not community specific and has the standard behavior. Set-top boxes belonging to a community are associated a non-null community type.

4 Specific IRD Commands

For any specific commands required by a manufacturer that doesn't belongs to the set of generic commands defined in §3, the procedure here after has to be followed:

- The manufacturer has to issue a formal document specifying the format and the behavior of the specific command. The command must comply with the general format defined in §2, but is restricted to the definition of the *operation* and *data* fields:

```
IRD_command(){
  EMM_command            8
  length                 8    uimsbf
  command_body{
    sequence_number      32   uimsbf
    command_id           8    uimsbf
    operation            8    uimsbf
    for(i=0; i < N; i++){
      data               8    bslbf
    }
    checksum             8    bslbf
  }
}
```

- The specification is provided to NagraVision for approval by sending an email to the following address:

  cak@nagra.com

- NagraVision evaluates the specification to know whether it is acceptable and assign a value to the *command_id* field. This allows to guarantee a global consistency all over the networks and allows to avoid conflicts between different commands.
  NagraVision reserves the right to modify the command and move it in the set of generic commands if its usage suits a wider scope.
- In case the command remains a specific command, the manufacturer updates the specification with the *command_id* assigned by NagraVision and publishes a new version of the document.
- In case the command becomes a generic command, NagraVision updates the present document with the new command and publishes a new version.

If the request for a specific commands comes from an operator instead of a manufacturer, the procedure here above remains the same, except that the specification is written by the operator. It is then provided to manufacturers providing set-top boxes over the operator network for implementation.

—— END OF DOCUMENT ——