

Digital audio broadcasting system — Specification of the DAB command set for receivers (DCSR)

The European Standard EN 50320:2000 has the status of a
British Standard

ICS 33.170

National foreword

This British Standard is the official English language version of EN 50320:2000.

The UK participation in its preparation was entrusted by Technical Committee EPL/100, Audio, video and multimedia systems, to Subcommittee EPL/100/1, Radio receiving equipment, which has the responsibility to:

- aid enquirers to understand the text;
- present to the responsible European committee any enquiries on the interpretation, or proposals for change, and keep the UK interests informed;
- monitor related international and European developments and promulgate them in the UK.

A list of organizations represented on this subcommittee can be obtained on request to its secretary.

Cross-references

The British Standards which implement international or European publications referred to in this document may be found in the BSI Standards Catalogue under the section entitled “International Standards Correspondence Index”, or by using the “Find” facility of the BSI Standards Electronic Catalogue.

A British Standard does not purport to include all the necessary provisions of a contract. Users of British Standards are responsible for their correct application.

Compliance with a British Standard does not of itself confer immunity from legal obligations.

This British Standard, having been prepared under the direction of the Electrotechnical Sector Committee, was published under the authority of the Standards Committee and comes into effect on 15 March 2001

Summary of pages

This document comprises a front cover, an inside front cover, the EN title page, pages 2 to 84, an inside back cover and a back cover.

The BSI copyright date displayed in this document indicates when the document was last issued.

Amendments issued since publication

Amd. No.	Date	Comments

BSI 03-2001

**Digital audio broadcasting system —
Specification of the DAB command set for receivers (DCSR)**

Système de radiodiffusion sonore
numérique — Spécifications du jeu de
commande DAB pour le récepteur
(DCSR)

Digitales Tonrundfunksystem —
Spezifikation des DAB-Befehlssatzes
für Empfänger (DCSR)

This European Standard was approved by CENELEC on 2000-01-01. 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, Czech Republic, 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

Foreword

This European Standard was prepared by the Technical Committee CENELEC TC 206, Consumer equipment for entertainment and information and related sub-systems.

The text of the draft was submitted to the Unique Acceptance Procedure and was approved by CENELEC as EN 50320 on 2000-01-01.

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) 2001-05-01
 - latest date by which the national standards conflicting
with the EN have to be withdrawn (dow) 2003-01-01
-

Contents

1	General.....	5
1.1	Scope.....	5
1.2	Structure of DCSR	5
2	Reference table.....	7
3	Responses	8
3.1	accepted.....	8
3.2	rejected.....	8
3.3	interim	9
3.4	command_not_implemented	9
3.5	busy.....	10
3.6	syntax_error.....	10
4	Commands and notifications.....	11
4.1	get_receiver_capability	11
4.2	notify_receiver_capability.....	12
4.3	tune.....	24
4.4	get_tii.....	25
4.5	notify_tii	27
4.6	select_tii	30
4.7	get_pad.....	32
4.8	notify_pad	33
4.9	select_pad	34
4.10	get_figs	36
4.11	notify_fig	42
4.12	select_figs.....	43
4.13	get_channel	44
4.14	notify_channel	46
4.15	select_channel.....	48
4.16	get_selection_status	51
4.17	notify_selection_status.....	52

4.18	search_for_ensemble.....	54
4.19	notify_search_for_ensemble	59
4.20	set_drc	60
4.21	get_audio_info	62
4.22	notify_audio_info.....	63
4.23	get_dab_status	64
4.24	notify_dab_status.....	64
4.25	set_dab_status_auto_notification	68
4.26	get_active_info	71
4.27	notify_active_info	72
4.28	notify_service_following.....	72
4.29	manufacturer_specific_command.....	73
4.30	manufacturer_specific_notification	74
4.31	notify_error_message	75
5	Example of DCSR coding.....	77
6	Guidelines for data fields, reserved for future use (Rfu's) or addition (Rfa's) and table entries, reserved for future definition (Rfd's)	83
7	Glossary.....	84
8	References	84

1 General

1.1 Scope

This standard describes a command set which should be used to control DAB receivers. The coding of these commands is also described. This command set is intended to be used on different physical bus systems. The coding should be mapped transparently on different physical interfaces.

Selection conflict management, dynamic bandwidth problems and the device addressing are not within the scope of this document.

1.2 Structure of DCSR

The DCSR paper describes the Instruction Set, which consists of three types of messages as follows:

The “Command” is used by a controlling device in order to tell the DAB receiver to perform a certain action, to deliver certain information or to move into a certain state.

The “Response” contains only a quick reaction ($t_1 < 100$ ms, bus transfer time not included) to the “Command”, e.g. accepted, rejected, busy. If the final response cannot be provided within 100 ms an interim response shall be sent within 100 ms. t_{1x} shall be zero if no interim response is sent. This response is mandatory and returned to the sender of the command.

The “Notification” contains the entire answer to the “Command”. The commands are considered from the DAB receiver point of view. The notifications are sent to the controller or to the specified output.

A new command shall not be sent before the final response of the previous command was received ($t_2 \geq 0$).

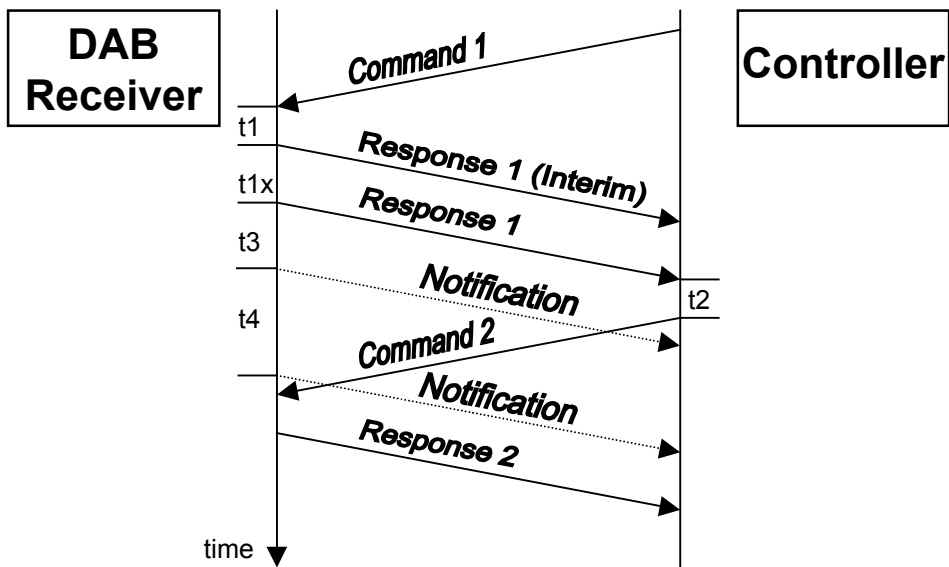
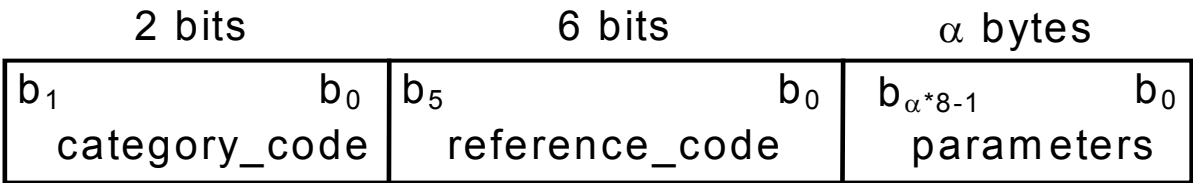


Figure 1 — General command response structure

NOTE: All parameter fields are always present unless otherwise stated.

The DCSR structure is coded as follows:



2 Reference table

	category code	reference code
get_receiver_capability	0x1	0x01
tune	0x1	0x02
get_tii	0x1	0x03
select_tii	0x1	0x04
get_pad	0x1	0x05
select_pad	0x1	0x06
get_figs	0x1	0x07
select_figs	0x1	0x08
get_channel	0x1	0x09
select_channel	0x1	0x0A
get_selection_status	0x1	0x0B
search_for_ensemble	0x1	0x0C
set_drc	0x1	0x0D
get_audio_info	0x1	0x0E
get_dab_status	0x1	0x0F
set_dab_status_auto_notification	0x1	0x10
get_active_info	0x1	0x11
manufacturer_specific_command	0x1	0x20
accepted	0x2	0x01
rejected	0x2	0x02
interim	0x2	0x03
command_not_implemented	0x2	0x04
busy	0x2	0x05
syntax_error	0x2	0x06
notify_receiver_capability	0x3	0x01
notify_tii	0x3	0x03
notify_pad	0x3	0x05
notify_fig	0x3	0x07
notify_channel	0x3	0x09
notify_selection_status	0x3	0x0B
notify_search_for_ensemble	0x3	0x0C
notify_audio_info	0x3	0x0E
notify_dab_status	0x3	0x0F
notify_active_info	0x3	0x11
notify_service_following	0x3	0x12
manufacturer_specific_notification	0x3	0x20
notify_error_message	0x3	0x30

Category code:
0x1: Command
0x2: Response
0x3: Notification

3 Responses

3.1 *accepted*

Category:

Response (mandatory)

Purpose:

The receiver indicates to the controller that it is able to start the execution of the command.
No notify_error_message is allowed to follow, but other notifications may follow.

Syntax:

accepted ()

Parameter:

none

3.2 *rejected*

Category:

Response (mandatory)

Purpose:

The receiver indicates to the controller that it cannot process the command (e.g. parameter out of range).

A notify_error_message will follow, or in case of a manufacturer_specific_command a manufacturer_specific_notification (containing the error message) will follow.

Syntax:

rejected ()

Parameter:

none

3.3 *interim*

Category:

Response (mandatory)

Purpose:

The receiver indicates to the controller that it cannot start executing the command within 100 ms. Subsequent to an initial response of INTERIM, the receiver shall not send any additional INTERIM responses for this command. The receiver shall send a final response when the command execution is started. No further command is allowed before the final response is received.

Syntax:

interim ()

Parameter:

none

3.4 *command_not_implemented*

Category:

Response (mandatory)

Purpose:

The receiver indicates to the controller that it cannot process the command, because this command is not implemented in the receiver.

No notification will follow.

Syntax:

command_not_implemented ()

Parameter:

none

3.5 *busy*

Category:

Response (mandatory)

Purpose:

The receiver indicates to the controller that it cannot process the command at this time, because the receiver is busy. The controller should send this command again later.

No notification will follow.

Syntax:

busy ()

Parameter:

none

3.6 *syntax_error*

Category:

Response (mandatory)

Purpose:

The receiver indicates to the controller that he cannot process the command, because there is a syntactical error in the received command (e.g. wrong number of parameters).

No notification will follow.

Syntax:

syntax_error ()

Parameter:

none

4 Commands and notifications

All meaningless command parameters are to be ignored (handled as 'don't care') by the receiver.

4.1 *get_receiver_capability*

Category:

Command (mandatory)

Purpose:

The controller asks the DAB receiver for its capabilities.

Syntax:

`get_receiver_capability ()`

Parameter:

none

Reaction:

`response(accepted)`

`notify_receiver_capability(...)`

The receiver provides the receiver capabilities.

`response(rejected)`

`notify_error_message(...)`

`response(interim)`

`response(busy)`

`response(syntax_error)`

4.2 *notify_receiver_capability*

Category:

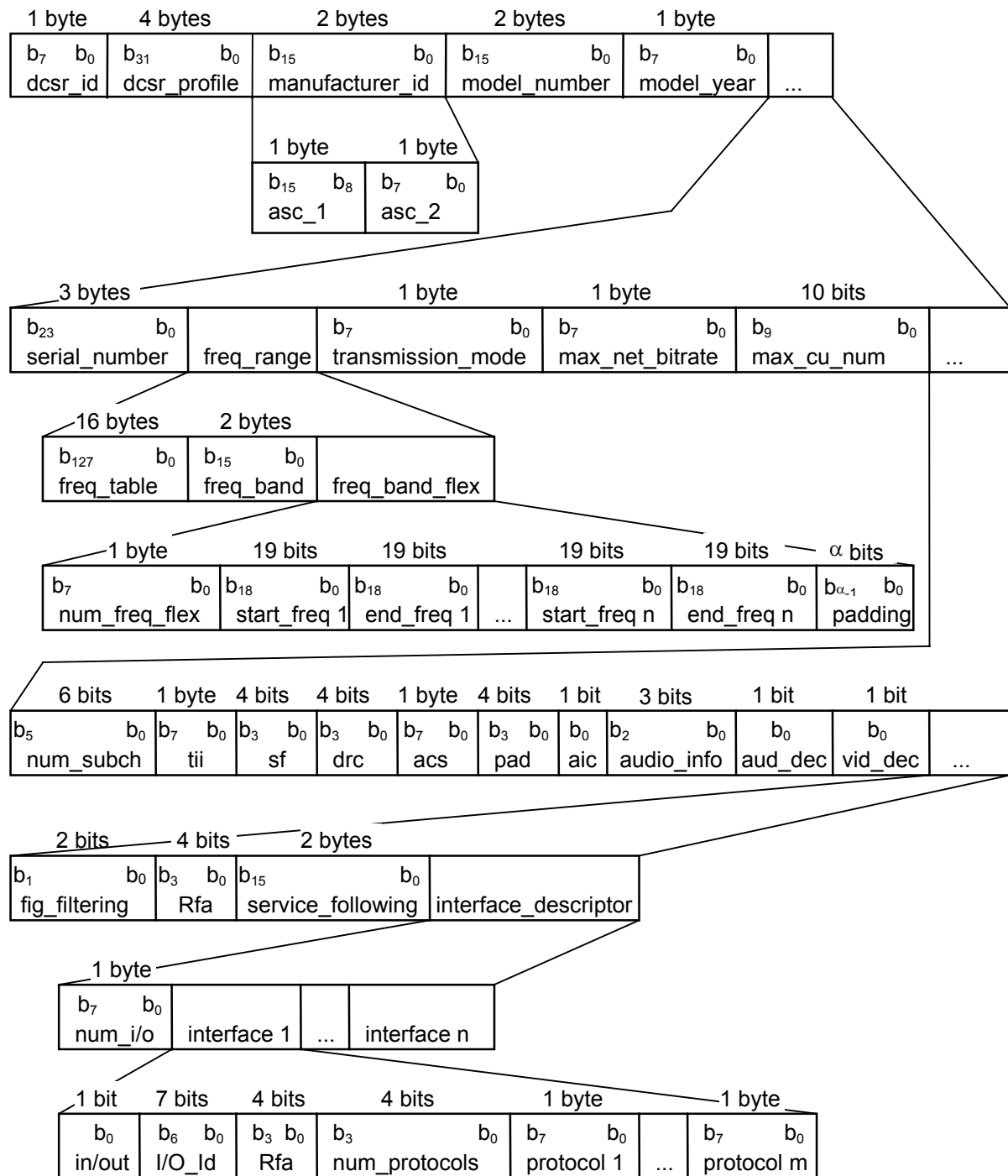
Notification (mandatory)

Purpose:

The DAB receiver provides its capabilities to the controller.

Syntax:

`notify_receiver_capability (dcsr_id, dcsr_profile, manufacturer_id, model_number,
model_year, serial_number, freq_range, transmission_mode, max_net_bitrate,
max_cu_num, num_subch, tii, sf, drc, acs, pad, aic, audio_info, aud_dec, vid_dec,
fig_filtering, Rfa, service_following, interface_descriptor)`



Parameter:

dcsr_id

This 8-bit field, coded as unsigned binary number, shall specify the DCSR Identifier (1...254). The DCSR Identifier determines the supported DCSR version. Since the DCSR versions are backward compatible, a certain *dcsr_id* means all lesser *dcsr_id*'s are also supported.

b7...b0	
0000 0000	signals no information
0000 0001	DCSR version 1
0000 0010	DCSR version 2
...	...
1111 1110	DCSR version 254
1111 1111	Rfd (escape option)

dcsr_profile

This 32-bit field indicates which optional commands and notifications are implemented in the receiver.

Bit flag	
b0	get_tii
b1	get_pad
b2	select_tii
b3	select_pad
b4	select_figs
b5	set_drc
b6	get_audio_info
b7	get_channel
b8	notify_channel
b9	manufacturer specific command
b10..b31	Rfa

The flags shall be coded as follows bi (i=0...31):

0: optional command not implemented;

1: optional command implemented.

manufacturer_id

This 16-bit field, coded as 2 ASCII characters, shall specify the manufacturer identifier based on CRIN (Car Radio Identification Number, see ref.[5]).

ASC_1: This 8-bit field, coded as ASCII character, shall specify the first character of manufacturer identifier (A-Z) according to ref.[5].

ASC_2: This 8-bit field, coded as ASCII character, shall specify the second character of manufacturer identifier (A-Z) according to ref.[5].

model_number

This 16-bit field, coded as an unsigned binary number, shall specify the model number based on CRIN (ref. [5]). This 16-bit binary field represents the 4 digits (decimal) model number of CRIN.

b₁₅...b₀	decimal model_number
0000 0000 0000 0000...	0000
0000 0000 0000 0001	0001
...	...
0010 0111 0000 1111	9999
0010 0111 0001 0000	not to be used
...	...
1111 1111 1111 1110	not to be used
1111 1111 1111 1111	not to be used

model_year

This 8-bit field, coded as ASCII character, shall specify the model year based on CRIN (ref. [5]).

serial_number

This 24-bit field, coded as an unsigned binary number, shall specify the serial number based on CRIN (ref. [5]). This 24-bit field is the binary representation of the 7 digits (decimal) serial number of CRIN.

b₂₃...b₀	decimal serial_number
0000 0000 0000 0000 0000 0000	000 0000
...	...
1001 1000 1001 0110 0111 1111	999 9999
1001 1000 1001 0110 1000 0000	not to be used
...	...
1111 1111 1111 1111 1111 1111	not to be used

freq_range (freq_table, freq_band, freq_band_flex)

The structure *freq_range* provides information about which tune/search strategies are provided by the receiver and which frequencies/frequency bands are supported.

freq_table

This 128-bit field indicates the predefined frequency tables used for the command *search_for_ensemble*.

If the frequency band a frequency table is dedicated to is not supported completely by the receiver, then only those frequency table entries that are inside the supported band are used (the complete ensemble must be inside the supported band).

Bit flag	
b0	16 kHz steps
b1	CEPT frequency table for Band III, ref.[7]
b2	CEPT frequency table for L-Band, ref.[7]
b3	Canada frequency table for L-Band
b4-b127	Rfa

The flags shall be coded as follows bi (i=0...127):

0: Frequency table not supported;

1: Frequency table supported.

freq_band

This 16-bit field indicates the supported predefined DAB frequency bands.

Bit flag	Frequency band (freq_band)
b0	Band III (174,000 MHz...240,000 MHz)
b1	L-Band (1452,000 MHz...1492,000 MHz)
b2-b15	Rfa

The flags shall be coded as follows bi (i=0...15):

0: Frequency band not supported;

1: Frequency band supported.

freq_band_flex (num_freq_flex, start_freq x, end_freq x, padding)

The structure *freq_band_flex* provides the possibility for the receiver to signal additionally provided DAB frequency bands, which cannot be indicated by the parameter *freq_table* (see above). If no additional frequency bands are provided, *num_freq_flex* shall be set to zero. *Freq_band_flex* consists of the following fields:

num_freq_flex: This 8-bit field, coded as an unsigned binary number, shall specify the number of additional bands defined by a list of start and stop frequencies (maximum 255). The value '0' means that there is no additional frequency band supported. The *start_freq x* and the *end_freq x* fields are not present in this case.

start_freq x: This 19-bit field, coded as an unsigned binary number, shall specify the start frequency (in multiples of 16 kHz) of an additional receiver defined band x; see ref.[1].

end_freq x: This 19-bit field, coded as an unsigned binary number, shall specify the end frequency (in multiples of 16 kHz) of an additional receiver defined band x; see ref.[1].

padding: This variable field of padding bits (set to zero) is used to guarantee a byte aligned coding. The length of this field is calculated as follows:

$$\text{number of padding bits} = q \cdot 8 \text{bits} - \text{num_freq_flex} \cdot 2 \cdot 19 \text{bits}$$

$$0 \leq \text{number of padding bits} < 8$$

q is an integer value

transmission_mode

This 8-bit field indicates the supported modes. It is possible that not every receiver will support each transmission mode in each frequency band.

Bit flag	
b0	transmission Mode I
b1	transmission Mode II
b2	transmission Mode III
b3	transmission Mode IV
b4-b6	Rfa
b7	automatic mode detection

The flags shall be coded as follows bi (i=0...7):

0: Mode not supported;

1: Mode supported.

max_net_bitrate

This 8-bit field, coded as an unsigned binary number, shall specify the maximum bit rate (excluding FIC) at the output of the Viterbi decoder in multiples of 8 kbit/s.

max_cu_num

This 10-bit field, coded as an unsigned binary number, shall specify the maximum number of CUs per CIF which can be handled by the receiver.

num_subch

This 6-bit field, coded as an unsigned binary number, shall specify the maximum number of MSC sub-channels which can be decoded simultaneously.

tii

This 8-bit field indicates the TII support.

Bit flag	
b0	main_id and sub_id
b1	field strength information
b2	carrier information
b3	delay information derived from the TII signal
b4	rough position defined by region Id derived from TII signal and the FIC information
b5	receiver position defined by longitude and latitude derived from TII signal and the FIC information
b6	Rfa
b7	Rfa

The flags shall be coded as follows bi (i=0...5):

0: TII processing not supported;

1: TII processing supported.

sf

This 4-bit field indicates the support of particular sampling frequencies for audio.

The support of 48 kHz sampling frequency is mandatory if an audio decoder is present.

Bit flag	
b0	24 kHz (LSF)
b1	Rfa
b2	Rfa
b3	Rfa

The flags shall be coded as follows bi (i=0,1):

0: Sampling frequency not supported;

1: Sampling frequency supported.

drc

This 4-bit field indicates the support of dynamic range control.

Bit flag	
b0	DRC (using F-PAD, see ref.[1])
b1	DRC (receiver generated)
b2	Rfa
b3	Rfa

The flags shall be coded as follows bi (i=0..3):

0: not supported;

1: supported.

acs

This 8-bit field indicates the supported Access Control Systems (ACS).

Bit flag	
b0	NR-MSK [3]
b1	Eurocrypt EN 50094 [4]
b2	Rfa
b3	Rfa
b4	Rfa
b5	Rfa
b6	Rfa
b7	Rfa

The flags shall be coded as follows bi (i=0..7):

0: Access Control System not supported;

1: Access Control System supported.

pad

This 4-bit field indicates the support of PAD output.

Bit flag	
b0	PAD output of the selected audio sub-channel which is sent to the on board audio decoder
b1	PAD output of any selected audio sub-channel
b2	PAD output of a not selected audio sub-channel
b3	PAD output streams of more than one audio sub-channel in parallel

The flags shall be coded as follows b_i ($i=0\dots3$):

0: not supported;

1: supported.

aic

This flag indicates the support of the AIC:

$b0 = 0$: receiver is not able to provide the FIGs from the AIC;

$b0 = 1$: receiver is able to provide the FIGs from the AIC as long as selected sub-channels do not fully occupy available decoding capacity.

audio_info

This 3-bit field indicates the support of audio_info.

Bit flag	
b0	audio_info of the selected audio sub-channel which is sent to the on board audio decoder
b1	audio_info of any selected audio sub-channel
b2	audio_info of a not selected audio sub-channel

The flags shall be coded as follows b_i ($i=0\dots2$):

0: not supported;

1: supported.

aud_dec

This flag indicates the presence of an audio source decoder:

$b0=0$: no audio source decoder present;

$b0=1$: at least one audio source decoder present.

vid_dec

This flag indicates the presence of a video source decoder:

b0=0: no video source decoder present;

b0=1: at least one video source decoder present.

fig_filtering

This 2-bit field indicates the level of FIG filtering support.

b₁ b₀	
00	receiver is not able to perform FIG filtering at all
01	receiver is able to perform basic FIG filtering by type/extension
10	receiver is able to perform basic FIG filtering by type/extension and FIG data field header bits (C/N, OE, P/D, D1, D2, TCId)
11	receiver is able to perform advanced FIG filtering by type/extension, FIG data field header bits (C/N, OE, P/D, D1, D2, TCId) and by evaluation of the Service Information Version (SIV, see in ref.[1])

service_following

This 16-bit field indicates the service following to different broadcast systems. If the receiver supports the service following to FM or AM then another command set is needed to control the FM or AM part of the receiver.

Bit flag	
b0	DAB ensemble, no local windows
b1	DAB ensemble, with local windows
b2	Rfa
b3	Rfa
b4	FM with RDS
b5	FM without RDS
b6	AM (MW in 9 kHz steps & LW)
b7	AM (MW in 5 kHz steps & SW)
b8..b15	Rfa

The flags shall be coded as follows b_i (i=0...15):

0: service following not supported;

1: service following supported.

interface_descriptor (num_i/o, interface x)

The *interface_descriptor* list provides information about which interfaces are provided by the DAB receiver and which protocols are supported on this interface.

num_i/o: This 8-bit field indicates the total number of interfaces (n) which are supported by the receiver (bi-directional interfaces shall be treated as two separate interfaces). The value zero is not allowed.

interface x (in/out, I/O_Id, num_protocols, protocol x)

The *interface x* field provides information about one particular interface.

in/out: This 1-bit flag indicates the direction of the interface. For bi-directional interfaces the type (I/O_Id) shall be indicated twice for in and for out:

0: output of the receiver;

1: input to the receiver.

I/O_Id: This 7-bit field, coded as an unsigned binary number, shall specify the type of interface (I/O Identifier), as described by the following table:

b₆...b₀	Physical Interface
000 0000	Not specified (default)
000 0001	Audio speaker
000 0010	Headphone
000 0011	Audio line fixed level
000 0100	Audio line variable level
000 0101	Audio digital (optical)
000 0110	Audio digital (electrical)
000 0111	IEEE1394
000 1000	RF-input
000 1001	Video RGB
000 1010	Video FBAS
000 1011	S-Video
000 1100	RS 232
000 1101	D2B
000 1110	IEEE 1284
000 1111	IEC958
001 0000	SPI
001 0001	I ² C
001 0010	USB

(continued)

b₆...b₀	Physical Interface
001 0011	IrDA
001 0100	Rfd
...	...
111 1110	Rfd
111 1111	Rfd (escape option)

The position n of the appropriate interface in the interface_descriptor list is used for further referencing. For the first interface in this list n shall be 1, for the last interface n shall be num_i/o . To address the x^{th} interface within the command to be used as the input or output interface, x is used as the input_interface_reference or output_interface_reference respectively.

num_protocols: This 4-bit field indicates the total number of protocols (m) which are supported by the receiver on the particular interface. If this value is set to zero, then no protocol is signalled. A default protocol should be used.

protocol x : This 8-bit field, coded as an unsigned binary number, indicates a protocol which is supported by the receiver on the particular interface.

b₇...b₀	protocol
0000 0000	not specified (default); this code should be used if the physical interface implies the protocol
0000 0001	RDI low capacity mode
0000 0010	RDI high capacity mode (normal frame)
0000 0011	RDI high capacity mode (extended frame)
0000 0100	SPDIF
0000 0101	AES-EBU
0000 0110	I ² S
0000 0111	DCSR universal protocol for byte-oriented interfaces
0000 1000	Rfd
...	...
1111 1110	Rfd
1111 1111	Rfd (escape option)

4.3 tune

Category:

Command (mandatory)

Purpose:

The controller asks the DAB receiver to use a certain input interface, to tune directly to a particular DAB frequency with the given transmission mode.

Syntax:

tune (input_interface_reference, Rfa, keep_decoding, transmission_mode, tune_freq)

1 byte	1 bit	1 bit	3 bits	19 bits
b ₇ b ₀ input_interface_reference	b ₀ Rfa	b ₀ keep_decoding	b ₂ b ₀ transmission_mode	b ₁₈ b ₀ tune_freq

Parameter:

input_interface_reference

This 8-bit field, derived from the interface descriptor of the receiver capability notification, shall address the input interface (RF-input) for the selection. If the input_interface_reference is set to zero the input interface does not change. If no input interface was specified before, the default input is used.

keep_decoding

This flag indicates whether the receiver should try to continue decoding without interruption while performing this command or not:

b₀ = 0: do not try to continue decoding;

b₀ = 1: try to continue decoding.

transmission_mode

This 3-bit field specifies the DAB transmission mode.

b ₂ ...b ₀	
000	keep the current transmission mode
001	transmission mode I
010	transmission mode II
011	transmission mode III
100	transmission mode IV
101	Rfd
110	Rfd
111	detect the appropriate transmission mode automatically

If the transmission mode is not supported for the given frequency, the command will be rejected.

The transmission_mode is set to “000” for example if the controller asks the receiver to change just the antenna input or change to another frequency in the same MFN.

tune_freq

This 19-bit field, coded as an unsigned binary number, shall specify the DAB centre frequency, see ref.[1].

tune_freq = 0 means keep the currently tuned frequency.

Reaction:

response(accepted)

 notify_dab_status()

 The receiver provides the DAB status.

response(rejected)

 notify_error_message(...)

response(interim)

response(busy)

response(syntax_error)

4.4 get_tii

Category:

Command (optional)

Purpose:

The controller asks the DAB receiver for output of the transmitter identification information (TII).

This command, with its associated notification (notify_tii), allows the controller to monitor the number and identification of the transmitters contributing to the received DAB signal. This data can be represented in many ways, the receiver signals which representations it supports in its receiver capabilities and any of these can be selected in the tii_select parameter. Use of these representations:

- The main_id and sub_id can be used to identify the region in which the DAB signal is currently received.
- Carrier information will allow the controller to estimate delays from the transmitters or perform proprietary TII processing.