



MHEG-5 Profile for South Africa

Publication Date 22/06/2009

Version 1.0

Status Approved

Copyright Notice – MHEG 5 Profile for South Africa

Any unauthorized reproducing, publishing, broadcasting, diffusion and/or making any adaptation of the MHEG-5 document or the related computer programs described therein, is illegal and expressly prohibited. Any person found to have infringed the copyright of this document shall be subject to prosecution and liable for any damages as a result of such infringement.

Disclaimer:

No person or organization shall, except in the proper course of its duties or as required by law, use and divulge to any person, firm or Company and shall use its best endeavours to prevent the use or disclosure of any trade or business secrets or any information concerning the business or finances of the SABC or of any dealings, transactions or affairs of the SABC or of any person with whom the SABC has business dealings with, as contained in this document.

Table of Contents

1	Introduction	3
1.1	Purpose	3
1.2	Scope	3
1.3	Requirements	3
1.4	History	4
1.5	Glossary	4
1.6	References	5
2	The User Experience	7
3	Specifications	8
3.1	Platform identification	8
3.2	User input extensions	9
3.3	Basic Service Information extensions	10
3.4	MHEG-5 Engine Graphics model	11
3.5	Fonts extensions	12
3.6	NVM persistent storage	14
3.7	Presentation of MHEG and subtitles	15
3.8	Language extensions	15
3.9	Stream from Memory	17
3.10	Unique Id	18
3.11	Native timer extensions	19
3.12	Broadcast file system acceleration and store	21

1 Introduction

1.1 Purpose

This document provides the detailed specification of the MHEG-5 engine required in compliant digital TV receivers. This specification defines “application domain” in the terms set out in Annex D of ISO/IEC 13522-5.

This document defines the following “application domain”:

- “SouthAfricaProfile1”

The specification builds on the UK profile, as defined in [MHEG], adding the following facilities:

- Revision of the ‘User Experience’ to accommodate new remote control and new input registers,
- Support for MHEG EPG,
- Support for direct (non-palette) colour presentation,
- Clarification on the simultaneous presentation of MHEG content and subtitles,
- Non volatile persistent storage,
- Extensions to control selection of audio and subtitles languages,
- Support for non-linear stream content delivered over the carousel,
- Support for Unique Identification of receivers,
- A mechanism to selectively store parts of a broadcast file system in system cache.

1.2 Scope

As far as is practical this document does not intend to create new specifications. Where possible existing public standards and specifications are referenced and, if required, profiled.

1.3 Requirements

The SA system will be provided in a ‘horizontal’ market with a conformance regime. The platform will rely on an EPG written in MHEG and will offer the facility to provide other applications including text services, multi-channel offers and games. The User Interface design covers the whole of the system, including the EPG and program related features. The native code parts of the UI will share graphics assets with the MHEG applications where necessary.

The system will have only two multiplexes and therefore bandwidth available for interactive and EPG applications and data will be limited. All receivers will therefore have non-volatile memory devoted to storage of applications and content, including audio/visual content where required.

This document reflects the requirements set out in [SAREQ]

1.4 History

Issue	Date	Descriptions
0.1	20090215	First draft for internal comment.
0.2	20090220	Draft to SABC for comment
0.3	20090327	Modified User Experience and User Input. Removed HD, IC and PVR. Update File System Acceleration. Added Fonts. Reformat
0.4	20090424	Revisions from MHEG Profile Meeting: <ul style="list-style-type: none"> • Single NVM partition, • User Input Register modified, • Fonts modified, • Language added, • Streams from memory added, • Unique id added, • Updated file system acceleration
0.5	20090508	Revisions from review: <ul style="list-style-type: none"> • Additional Engine Events, • Include all Bold font sizes
0.6	20090522	Added Native Timer extensions. Added EPG and Info keys to Input Registers. Editorial review.

Table 1-1 Version History

1.5 Glossary

Term	Definition
AFD	Active Format Descriptor
AC-3	Dolby Digital (5.1 Channel)
Alphabetic	Characters that typically represent a component of a spoken word. For example the Latin derived characters used to represent English or the Cyrillic characters used to represent Russian.
CharacterSet	MHEG term defined as: Identification of the character set, or set of character sets, that shall be used by default for Text rendering. This Integer shall be encoded with a value representing the character set. The application domain shall define a range of CharacterSet and its semantics.
ContentHook	MHEG term defined as: Determine the encoding format of the data included or referenced by the Content attribute.
D-Book	See technical standards listed elsewhere in this document
DTT	Digital Terrestrial Television
DVB	Digital Video Broadcast organisation
DVB-T	DVB-Terrestrial
DTG	Digital Television Group – a UK digital television industry organisation
EBU	European Broadcasting Union

EPG	Electronic Programme Guide
EIT	Event Information Table
ETSI	European Telecommunication Standards Institute
FTA	Free to Air
HD	High Definition
HDCP	High-Bandwidth Digital Content Protection
HDMI	High-Definition Multimedia Interface
HDTV	High Definition Television
IRD	Integrated Receiver Decoder
iDTVs	Integrated Digital Televisions
JSON	Javascript Object Notation
MHEG-5	Multimedia and Hypermedia information coding Expert Group
May	Indicates an event or provision which is permitted, but not mandatory
MPEG	Moving Pictures Expert Group
Must	Indicates that a third party must comply to ensure correct operation.
NIT	Network Information Table
NVM	Non-Volatile Memory
OSD	Onscreen Display
RAM	Random Access Memory
SD	Standard Definition
SDTV	Standard Definition Television
SI	Service Information
STB	Set-Top-Box
Shall	Indicates a mandatory provision
Should	Indicates a desirable, but not mandatory, provision
Transport Stream	A stream format defined in [MPEG SYS]
TS	Transport Stream
UTF	Unicode Transformation Format
Will	Indicates an assumption about existing states or future events

1.6 References

Reference	Version	Description
D-Book	Issue 5.0	Digital Television Group: Digital Terrestrial Television, Requirements for Interoperability
[D-Book6]	Issue 6.0	Digital Television Group: Digital Terrestrial Television, Requirements for Interoperability
DVB Coding	ETR 101 154 v1.7.1	Digital Video Broadcasting (DVB); Implementation Guidelines for the use of video and audio coding in Broadcasting Applications based on the MPEG-2 transport stream
DVB SI	EN 300 468 V1.9.1	Digital Video Broadcasting (DVB) Digital Broadcasting Systems for Television, Sound, and Data Services. Specification for service information (SI) in Digital Video Broadcasting (DVB) ETSI
DVB SI Codes	ETSI 162	Digital Broadcasting Systems for Television, sound and data services, allocation of service information (SI) codes for digital Video Broadcasting (DVB) systems. ETSI.
DVB SI Guide	TR 101 211 V1.7.1	Digital Video Broadcasting (DVB); guidelines on implementation and usage of Service information(SI)
DVB Sub	ETSI 300 743	Digital Video Broadcasting (DVB); DVB Subtitling Systems. ETSI.
DVB-T	EN 300 744 V1.5.1	Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television.
ETSI MHEG	ES 202 184 v1.1.1	MHEG-5 Broadcast Profile
HD MHEG	13 March 2008	DTG HD MHEG Group; High Definition and Interactive Services Specification
MHEG Interaction Channel	24 July 2008	DTG Return Path Group – Interaction Channel Working Group MHEG Specification
MHEG	1.06.05	MHEG 1.06 UK Profile as published in [D-Book]
MPEG SYS	ISO/IEC 13818-1	Information Technology - generic coding of moving pictures and associated audio: systems
DTG DTR	1.1	DTG Functional Specification for Digital TV Recorders
ISO 639-2		Codes for the representation of names of languages.
ISO 8859-1		Information technology – 8 bit single byte coded graphic character sets – Part 1: Latin alphabet No. 1
GBS	ETSI TS 102 323 V1.3.1 (2008-04)	Digital Video Broadcasting (DVB); Carriage and signalling of TV-Anytime information in DVB transport streams
UI	5	090225-sabc-specv5b.pdf (Reference only)
SAREQ	TBD	SABC Requirements for Interactive Systems

2 The User Experience

The section of [MHEG] describing The User Experience is deleted and replaced by reference to [UI].

3 Specifications

Unless stated otherwise specifications follow those described in [MHEG].

Note: No features specified in [MHEG] are removed or modified so as to fail conformance tests defined for [MHEG]. This specification defines additions to [MHEG]. Some features (e.g. additional input registers) require invocation by the MHEG application before they become active. For example, additional input registers require attributes to be set in the application. Without such activation the receiver shall conform to [MHEG] and shall pass the conformance tests specified for [MHEG].

3.1 Platform identification

3.1.1 *International profile string*

The following table defines a set of “international profile strings” associated with this specification.

Profile name	Version	International Profile String
UK MHEG	1.0.6	
South Africa Profile	001.000.000	INTZA001000000

Table 3-1 List of profiles supported by receivers compliant with this profile

3.1.2 *UniversalEngineProfile GetEngineSupport feature string*

The UniversalEngineProfile (UEP()) GetEngineSupport feature string is defined in [MHEG]. It allows version information about the platform to be interrogated. This clause extends this behaviour allowing the “international profile string” for each of the international profiles that the receiver supports to be interrogated in addition to those defined by [MHEG].

Receivers conformant to this specification shall support at least the following profile names:

- South Africa Profile

3.1.3 *WhoAml resident program*

The WhoAml (WAI()) resident program is defined in [MHEG]. It returns a string that contains a space separated set of sub-strings. This clause extends this resident program so that the string returned contains additional space separated sub-strings. The additional sub-strings are the “international profile string” for each of the international profiles that the receiver supports.

Receivers conformant to this specification shall support at least the following profile names:

- South Africa Profile

3.2 User input extensions

The base set of user input keys that all receivers shall support is defined in [MHEG]. The remote control defined by [UI] may not include all the keys defined in section “13.6.1 Remote control functions” of [MHEG], however the MHEG-5 engine shall still support the EventData values associated with these keys.

This clause specifies additional packages of UserInputEventData values, User input registers and EngineEvents.

3.2.1 Menu key

The Menu key shall generate UserInputEventData value 301.

3.2.2 Languages key

The Language key shall generate UserInputEventData value 302.

3.2.3 Interactivity key

The Interactive key shall generate UserInputEventData value 304.

3.2.4 UserInput registers

In addition to the input registers defined in [MHEG], receivers shall support the following input registers. All three input registers (23, 24 and 25) must be supported.

UserInput EventData value	Function Name	Register Number		
		23	24	25
1	Up		✓	✓
2	Down		✓	✓
3	Left		✓	✓
4	Right		✓	✓
5-14	0,1,2,3,4,5,6,7,8,9 respectively		✓	
15	Select		✓	✓
16	Cancel / Exit ²	✓	✓	✓
100	Red	✓	✓	
101	Green	✓	✓	
102	Yellow	✓	✓	
103	Blue	✓	✓	
104	Text	✓	✓	✓
105	Info ²	✓	✓	✓
300	EPG ²	✓	✓	✓
301	Menu	✓	✓	

302	Languages	✓	✓	✓
304	Interactivity	✓	✓	✓

Table 3-2 Input registers

3.2.5 Engine Events

Additional engine events are generated when the user activates the following keys and there is an active scene object. The events are raised independently of the InputEvent register selected at the current moment or whether any interactible has the InteractionStatus of True.

If a key press causes both the EngineEvent and the UserInput event then the EngineEvent shall be raised first.

EventData		Notes
Name	Value	
InfoKeyFunction	105	Generated when the user activates the Info key and there is an active scene object.
EPGKeyFunction	300	Generated when the user activates the EPG key and there is an active scene object.
MenuKeyFunction	301	Generated when the user activates the Menu key and there is an active scene object.
LanguageKeyFunction	302	Generated when the user activates the Language key and there is an active scene object.
InteractivityKeyFunction	304	Generated when the user activates the Interactivity key and there is an active scene object.

Table 3-3 UserInput Engine Events

3.3 Basic Service Information extensions

The base set of resident programs that all receivers shall support is defined in [MHEG]. This clause specifies an additional package of resident programs for accessing Service Information.

Receivers shall implement the Service Information package defined in [ETSI MHEG] i.e. SI_GetServiceInfo and SI_GetEventInfo. The SIPackageSupport GetEngineSupport “feature” string, as described in [ETSI MHEG], shall also be supported.

Note that for the Resident Program SI_GetEventInfo the return parameter freeNotCA shall reflect the value of the free_CA_mode field from the Event information section.

3.3.1 Information string format

The string format supported by the DVB Event sections is not compatible with MHEG. Therefore the information strings obtained from these sections must be filtered to make them presentable.

All strings provided by the ResidentPrograms of the Service Information package shall be encoded in UTF-8. Furthermore the control codes (0x80-0x9F and 0xE080-0xE09F) shall not

be present. Where the CR control code (0x8A, 0xE08A) is present in the section it shall be replaced with the value 0x0D. Note that this may result in the string containing characters that are not available in the selected character repertoire.

3.4 MHEG-5 Engine Graphics model

[MHEG] defines a colour model based on a fixed palette of colours, each specified with 8 bits per component with 3 levels of transparency. This specification defines a new colour model, based on direct (non-palette) colour presentation.

3.4.1 Colour range

For this specification the graphics plane shall provide an RGB colour space with at least 16 bits per pixel. At least 4 bits shall be used for each of the red, green, blue and transparency components. The graphics plane shall provide at least 16 evenly spaced levels of transparency.

3.4.2 Direct/absolute colours

As described in [MHEG], colour definitions within MHEG-5 Visibles are encoded with 8 bits per component. Bitmap content may use a variety of different bit depths. Where the graphics plane supports fewer bits per component, colours and transparency values shall be mapped by dropping the least significant bits. Where this contradicts anything stated in section 14.3.4 of [MHEG], the present clause shall take priority.

This colour model may be used also for applications executed by the receiver that are conformant to [MHEG] only.

3.4.3 Identifying direct colour support

In order to identify receivers that can support the colour resolution model described in this specification the following GetEngineSupport feature strings are added:

String		Constraint
Standard	Short	
ColourDepth(N)	CoD(N)	Shall return "true" for N=16 if the receiver supports representation of MHEG colour to at least an accuracy of 4 bits per component.

Table 3-4 Colour depth feature string

3.5 Fonts extensions

This clause extends sections "15.3 Fonts" and "15.5 Text rendering" of [MHEG].

3.5.1 Downloading

Receivers shall support downloadable fonts using the MHEG-5 Font class. Text objects referencing downloaded fonts shall do so using a reference to an MHEG-5 Font object. Downloaded fonts cannot be referenced by name.

3.5.2 Embedded font

In addition to the font "rec://font/uk1", defined in [MHEG], receivers shall implement the Arial font. The font is referenced in MHEG applications as "rec://font/sa1" for plain style and

“rec://font/sa1-bold” for bold style and shall be available in at least the sizes and styles defined in Table 3-5.

Size (points)	TV Lines over 'Cap-V'	Informative Name	Styles	
			Plain	Bold
14	10		✓	✓
16	11		✓	✓
17	12		✓	✓
18	13		✓	✓
20	14		✓	✓
21	15		✓	✓
24	17		✓	✓
28	19		✓	✓
50	34		✓	✓
52	35		✓	✓
54	37		✓	✓

Table 3-5 Arial sizes and styles

The font is available from TBD.

3.5.3 OpenType fonts

Receivers shall support OpenType® fonts with TrueType™ outlines, as defined by the OpenType specification version 1.4. This specification is published on the following web sites:

<<http://www.microsoft.com/typography/otspec/default.htm>>

<<http://partners.adobe.com/asn/tech/type/opentype/index.jsp>>

OpenType fonts are identified using a CHook value of 10.

3.5.3.1 Profile of OpenType

Receivers are required to support the 'required' tables, the 'tables related to TrueType outlines' and the 'kern' table (format '0' horizontal kerning only). Support for other tables is optional. Application authors are advised to avoid using fonts that make use of optional tables as these may be rendered differently by different receivers.

Receivers are not required to support TrueType Collections; in this Profile, one MHEG-5 Font object provides a single font.

3.5.4 Font parameters

For OpenType fonts, the following table defines the values to be used for the font metrics parameters referenced in section “15.5 Text Rendering” section of [MHEG].

Parameter name	Obtained from
metricsResolution	unitsPerEm field, defined in the Font Header ('head') table
outlineResolution	
advanceWidth, charSetWidth	advanceWidth values, defined in the Horizontal Metrics ('hmtx') table.
xMin, yMin, yMax	xMin, yMin, yMax, defined in the Font Header ('head') table
kern	value, defined in the Kerning ('kern') table

Table 3-6 OpenType font parameters

3.5.4.1 Text styles

When referencing a downloaded font, the 'plain' text style (see [MHEG], “15.4.1 FontAttributes”) shall refer to the single text style present in the referenced font file, whatever that may be.

3.5.5 Presentation

Text objects referencing a downloaded font shall not be rendered until the receiver has downloaded the required font data. Until then, the object shall be presented as defined in [MHEG], “14.10 Appearance of Visible objects during content retrieval”. Where an attempt to download content for an MHEG-5 Font object fails, any Text objects dependant on it shall be rendered in the receiver's in-built font.

3.5.6 Defensive response

Receivers shall implement the following measures to ensure robust behaviour with any applications that attempt to use in-built or downloadable fonts that are not available, or font characteristics that the receiver doesn't recognise:

- If the font requested by a Font object font is not available, the receiver shall use its in-built font.
- If any of the attributes of the Font object are invalid, e.g. an unsupported content hook, the receiver shall use its in-built font.
- If the font style of a Text object is recognised then it shall be used; if not, the style referenced by 'plain' shall be used.
- Where a font has a limited range of available sizes, if the requested font size of a Text object does not match one of those available, the receiver shall substitute the next smaller size available. If the required font is smaller than the smallest available, then the smallest available size shall be used.
- Any character not supported by the engine shall not be considered as part of the input.

3.6 NVM persistent storage

In addition to the RAM-backed persistent store as defined in [MHEG], receivers shall implement a persistent store that is non-volatile.

The data area may be used by platform-wide applications and service or broadcaster-specific applications. The data area shall be able to store a minimum of 512k bytes of data.

3.6.1 Accessing the store

The NVM store shall be accessed with the StorePersistent and ReadPersistent elementary actions. The file name used to access this storage shall be of the form "nvm://<name>". It is the responsibility of the broadcasters to arrange a practise for the use of <name> such that there is no accidental collision of file names. The <name> part shall be at most 8 bytes in length.

The rules for writing to and reading from the NVM store shall be as defined in [MHEG] for the RAM-based store. This includes the policy for deleting older files to make room for new ones.

3.7 Presentation of MHEG and subtitles

Receivers conformant with this specification shall support the simultaneous presentation of MHEG-5 applications and DVB subtitles. The rules for management of subtitle presentation by an MHEG application are described in [MHEG].

3.8 Language extensions

This clause specifies a package of extensions that may be used to control the presentation of audio and subtitles from within an MHEG-5 application. Audio and subtitle languages are identified by a language code, a 24 bit field as specified by [ISO 639-2] and using a 3 character code as specified by [ISO 8859-1].

3.8.1 LNG_GetPreferredLangs

Returns the current preferred audio and subtitle languages.

Synopsis GPL(audio, subs)

in/out/in-out	type	name	comment
out	GenericOctetString	audio	3 char language code identifying the current preferred audio. If no preferred option is set the string "und" shall be returned.
out	GenericOctetString	subs	3 char language code identifying the current preferred subtitles. If no preferred option is set the string "und" shall be returned.

Table 3-7 LNG_GetPreferredLangs arguments

Description

The resident program retrieves the current user preferences for audio and subtitle languages.

3.8.2 LNG_GetAudioLangs

Returns a list of available audio languages for the specified event.

Synopsis GAL(serviceIndex, eventIndex, audioLangs)

in/out/in-out	type	name	comment
in	GenericInteger	serviceIndex	A receiver specific integer identifying a service. This value shall be consistent with the values returned by SI_GetServiceIndex in[MHEG]
in	GenericInteger	eventIndex	A zero based index to the events on the specified service, 0 being the current event, 1 being the next event.
out	GenericOctetString	audioLangs	A space separated list of 3 char language codes. Each code corresponds to an audio stream that is available to the event as signalled by the EITpf.

Table 3-8 LNG_GetAudioLangs arguments

Description

The resident program retrieves a list of audio languages available to an event as defined by the EITpf.

3.8.3 LNG_SetAudioLang

Sets the preferred audio language.

Synopsis SAL(audio)

in/out/in-out	type	name	comment
in	GenericOctetString	audio	A 3 char language code representing the preferred audio language.

Table 3-9 LNG_SetAudioLang arguments

Description

The resident program indicates the preferred audio language.

3.8.4 LNG_GetSubtitleLangs

Returns a list of available audio languages for the specified event.

Synopsis GSL(serviceIndex, eventIndex, subsLangs)

in/out/in-out	type	name	comment
in	GenericInteger	serviceIndex	A receiver specific integer identifying a service. This value shall be consistent

			with the values returned by SI_GetServiceIndex in[MHEG]
in	GenericInteger	eventIndex	A zero based index to the events on the specified service, 0 being the current event, 1 being the next event.
out	GenericOctetString	subLangs	A space separated list of 3 char language codes. Each code corresponds to a subtitle stream that is available to the event as signalled by the EITpf.

Table 3-10 LNG_GetSubtitleLangs arguments

Description

The resident program retrieves a list of subtitle languages available to an event as defined by the EITpf.

3.8.5 LNG_SetSubtitleLang

Sets the preferred subtitle language.

Synopsis SSL(subs)

in/out/in-out	type	name	comment
out	GenericOctetString	subs	A 3 char language codes representing the preferred subtitle language.

Table 3-11 LNG_SetSubtitleLang arguments

Description

The resident program indicates the preferred subtitle language.

3.9 Stream from Memory

This clause specifies extensions to [MHEG] allowing the MHEG-5 application to play non-linear content acquired from a file system and played from memory.

3.9.1 Content Data Encoding

Non linear stream content shall be identified using stream content hook of 15 as defined in [D-Book6]. In this profile the restriction that the file source may only be "http:" does not apply.

3.9.2 Stream "memory" formats

In this profile the restriction on the use of Storage type of 'memory' for Video and Subtitle components specified in [D-Book] does not apply.

3.9.3 Non linear stream formats.

Receivers shall be able to present transport stream files with data rates up to and including the rates below.

- Full Screen SD 2048kbps, as defined in [D-Book6].

- Quarter Screen SD 512kbps

Support of higher data rates is optional.

The audio, video and subtitle components shall be encapsulated within a single program MPEG-2 transport stream file as defined in [D-Book6].

3.10 Unique Id

This clause specifies a package of extensions that allows the MHEG-5 application access to a receivers' Unique Identifier. The value of the identifier is retrieved with a resident program and validated by extending GetEngineSupport.

3.10.1 Unique STB Id Format

The unique STB Id needs to reflect the market for which it was built and so its format is a matter for the broadcaster/manufacturers however it should be enough to assume that it is in the form of an octet string.

It would be **preferable** though not actually necessary, that the id returned be built up from printable ASCII characters and whilst the unique id data might ultimately represent bit-fields, STB middleware should be able to translate it into a printable equivalent, ASCII hex values for example.

3.10.2 UID_GetUniqueId

Synopsis GUI(isValid, idString)

Input/output	type	name	comment
output	GenericInteger	isValid	1 indicates a valid Id and 0 indicates invalid.
output	GenericOctetString	idString	The contents of the unique id.

Table 3-12 UID_GetUniqueId arguments

Description

This resident program will retrieve the STB's unique id and to deal with the possibility that the id has somehow been tampered with or incorrectly programmed in the box, the RP is able to report the validity of the id and thus enable an MHEG application to act on this information.

3.10.3 GetEngineSupport String

By making the unique id retrievable through GetEngineSupport a simple application can be authored for targeting particular STBs.

To enable GetEngineSupport to identify the query being made requires the following format.

- **UIDXXXXXXXXXXXX**

X should ideally match the value returned by GUI for the given STB.

3.11 Native timer extensions

Native timers are primarily a function of the native application and allow a receiver to set a timed event to record a programme on an external, connected device. This clause specifies a package of extensions that may be used to control and review native timers.

This package of extensions allows an MHEG application to:

- Set a native timer,
- Delete a native timer,
- Obtain a list of native timers.

NOTE: It is the role of the native application to handle native timer events and MHEG applications shall not receive notification when a native timer fires. The handling of native timer events by the native application may result in the current service being tuned away from and any active MHEG application being killed. The receiver may indicate that this is about to occur before switching services, for example by a notification on a separate graphics plane or by some receiver front panel indicator. This profile does not specify signalling between the receiver and the external recording device.

3.11.1 TIM_SetNativeTimer

Sets a native timer for a specified service, date, time and duration.

Synopsis SNT(serviceIndex, info, result)

in/out/in-out	type	name	comment
in	GenericInteger	serviceIndex	Receiver specific integer, as returned by SI_GetServiceIndex in [MHEG], identifying the service to be recorded.
in	GenericInteger	date	The Modified Julian date on which the timer is to be scheduled. Encoded as the number of days since Midnight on November 17, 1858.
in	GenericInteger	startTime	The time at which the timer is to be scheduled, encoded as the number of seconds since midnight.
in	GenericInteger	duration	The duration of the event to be recorded encoded in number of seconds.
out	GenericInteger	result	The result of the operation: <ul style="list-style-type: none"> • 0 = timer set OK, • -1 = conflict with other timer, • -2 = insufficient receiver resource, • -3 = failed for other reason.

Table 3-13 TIM_SetNativeTimer arguments

Description

The SNT resident program sets a native timer to enable an event to be recorded. The serviceIndex identifies the service to be recorded, the date and startTime identifies when the event starts and the duration identifies the events duration.

3.11.2TIM_ResetNativeTimer

Resets (disables) a native timer with the specified index

Synopsis RNT(timerIndex, result)

in/out/in-out	type	name	comment
in	GenericInteger	timerIndex	A zero based index to all native timers.
out	GenericInteger	result	The result of the operation: <ul style="list-style-type: none"> • 0 = timer reset OK, • -1 = index out of range.

Table 3-14 TIM_ResetNativeTimer arguments

Description

The RNT resident program resets a native timer. Resetting a native timer un-sets any previous date and time and renders it unused.

3.11.3TIM_GetNativeTimers

Gets a list of native timers.

Synopsis GNT(list)

in/out/in-out	type	name	comment
out	GenericOctetString	list	A comma separated list of native timers.

Table 3-15 TIM_GetNativeTimers arguments

Description

The GNT resident program returns an octet string identifying each native timers' current setting. The list is a comma separated series of strings in the following format:

"<serviceIndex>--<date>--<startTime>--<duration>"

Where:

- serviceIndex is a receiver specific integer, as returned by SI_GetServiceIndex in [MHEG], identifying the service to be recorded,
- date is the Modified Julian date of the event, encoded as the number of days since Midnight on November 17, 1858,
- startTime is the start time of the event encoded as the number of seconds since midnight,
- duration is the duration of the event encoded as the number of seconds.

The position of each entry identifies the timerIndex, the first being index zero. If a Timer is not set the entry will be empty. The string shall include all timers set but need not contain any

unused entries after the last used entry. White space may present between entries and can be ignored.

For example:

"1--54974--68400--1800, 3--54975--70200--3600 , , , 5--54977--43200--3600,"

Represents the following list of timers:

Index	Service	Date	Start Time	Duration
0	1	23 rd May 2009	19:00	30 mins
1	3	24 th May 2009	19:30	1 hour
2				
3				
4				
5	5	26 th May 2009	12:00 (midday)	1 hour

3.12 Broadcast file system acceleration and store

This section describes a mechanism whereby a receiver shall provide a storage area that stores a selected part of the broadcast file system so that subsequent file access is faster than may be achieved by accessing the files from the object carousel directly.

The storage area is maintained when the receiver is not tuned to the service from which the files were originally loaded and shall survive receiver standby-cycles.

The stored files shall be made accessible to any service that signals the presence of the same file group identifiers within an object carousel, thus improving the application launch time on those services. To ensure file-system consistency each group of files in the store has an identifier and a version.

3.12.1 File groups

The parts of the DSMCC broadcast file system to be stored are identified as file groups. Signalling within the object carousel defines which groups are required to be stored. File groups have a priority attribute that helps the receiver to limit the number of groups stored when resources are limited. File groups include a receiver-profile as an aid to targeting subsets of the total receiver population with particular file-groups.

File groups make no implications on the way in which those files and directories are transmitted within the object carousel. In particular there is no mapping between groups and DSMCC modules, or between group-version and DSMCC module version. Neither is there a relationship between the ordering of files listed in the file-group and the order in which those files shall be transmitted in the broadcast carousel.

When the file group changes, due to files being added, removed, moved or updated the file-group version shall also be required to change. This enables the storage controller to identify that the stored copy is now stale and that it should start the acquisition process for that file-group. Each file-group is versioned as a single unit and shall be re-acquired in its entirety on each signalled version change.

3.12.2 Scope of file groups

Each file group has associated with it an Owner Id and a Group Id. The Group Id is unique within the scope of an Owner Id. The same file group may be broadcast on multiple broadcast carousels in order to make files visible across a number of services. When this is the case the receiver shall store a single copy of the file group and make it visible to all services where that Owner Id, Group Id pair is signalled as being present. When a single file-group is broadcast on multiple object carousels it is required that the file-group version and the content of the file-group is synchronised across carousels.

3.12.3 Stored groups descriptor

The stored_groups_descriptor is located in the DownloadServerInitiate messages of the broadcast object carousel, in the DSI::ServiceGatewayInfo::UserInfo bytes. Each instance of the stored_groups_descriptor may describe multiple groups. There may be multiple instances of the descriptor to describe all the transmitted file groups within the current object carousel.

The structure of the stored_groups_descriptor is described in Table 2-11.

Syntax	bits	Type
stored_groups_descriptor {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (j=0;j<N;j++) {		
owner_id	16	uimsbf
group_id	16	uimsbf
group_priority	8	uimsbf
use_from_carousel	1	bslbf
reserved	7	bslbf
receiver_profile	8	uimsbf
group_version	8	uimsbf
private_data_length	8	uimsbf
for (k=0;k<M;k++) {		
private_data_bytes		
}		
}		
}		

Table 2-11 Stored Groups Descriptor

Where:

- **descriptor_tag:** this 8-bit field identifies the descriptor. In the stored groups descriptor it is set to 0x80.
- **descriptor_length:** this 8-bit field specifies the number of bytes of the descriptor immediately following this field.
- **owner_id:** this 16-bit field defines the owner of the group used to identify the broadcast corporation or platform identifier.
- **group_id:** this 16-bit field defines the identifier for this group, which shall be unique within the scope of the owner_id.
- **group_priority:** this 8-bit field defines the relative storage priority of the file group, as described in Table 2-12.

Note: The priority used for the storage of individual applications shall be profiled and managed by the managing authority so that a consistent behaviour is maintained across regions and across multiplexes within regions.

Group_priority	Description
0x00 0x01-0x09 0x0b-0xff	System group (platform wide) User application priorities Reserved for future use

Table 2-12 Group priority values

- **use_from_carousel:** this 1-bit flag shall be set to '1' if the file group is accessible from the object carousel directly. It shall be set to '0' if the group may be accessed from store only and may not therefore be accessed directly from the object carousel.

Note: When the use_from_carousel flag is set to '0' this is intended to signal that the group shall be acquired and stored and that only after the group has been stored in its entirety shall the files it defines be made available to higher level processes. Any references to files within the DSMCC files system shall fail until such time that the group has been acquired and stored in its entirety

- **receiver_profile:** this 8-bit field defines a receiver profile identifier that may be used to target receivers that conform to various minimum receiver profiles as described in Table 2-13. **Note:** A receiver may be part of multiple profiles and may therefore return True to multiple ProfileId requests.

Receiver_profile	Description
0x00 0x01-0xff	All receivers Reserved for future use

Table 2-13 Receiver profile values

- **group_version:** this 8-bit field defines the version of the file group identified by the owner_id and group_id.
- **private_data_length:** this 8-bit field defines the length of the private data bytes.
- **private_data_bytes:** this field is not described in the current document.

3.12.4 Group manifest

The group manifest describes the file entries that are present in the current object carousel that shall be stored. Each manifest file is transmitted in the root of the DSMCC file-system as a file named after the following convention:

"group.<owner_id>.<group_id>"

where <owner_id> and <group_id> are the values from the stored_groups_descriptor, which are 16 bit values each represented as 4 hexadecimal digits with lower case characters. If this file is not present or is not correctly structured the broadcast carousel file group shall not be stored.

The group manifest file shall use [JSON] format; a simple, open data structure, the format of which encourages concise documents. The structured format enables a schema to be created for the purpose of document validation prior to injection into the object carousel.

Table 2.14 defines a [JSON] schema describing the format of the manifest file.

```

{
  "description": "Stored Group Manifest Schema",
  "type": "object",
  "properties": {
    "version": { "type": "integer" },
    "size": { "type": "integer" },
    "nodes": { "type": { "items": {
      "type": "object",
      "properties": {
        "node": { "type": "string" },
        "count": { "type": "integer", "optional": true }
      }
    } } }
  }
}

```

Table 2.14 **JSON Schema for manifest**

Table 2.15 provides an example of a manifest file that conforms to the previously described JSON schema. White space has been added to the example to aid readability only. White space is not required in conformant JSON documents.

```

{
  "version": 0,
  "size": 2500000,
  "nodes": [
    { "node": "/path/to/files/file001" },
    { "node": "/path/to/files/file002" },
    { "node": "/path/to/lots_of_files/file", "count": 99 }
  ]
}

```

Table 2-15 **Example JSON manifest**

Where:

- **version:** is a version indicator for the file group. The version shall be changed when any files with the group change or files are added to or removed from the group. This value shall agree with the version field carried in the DSI stored_groups_descriptor.
- **size:** indicates the total size in bytes of the files that are contained in the group prior to DSMCC encoding. Note that this does *not* include the size of directory objects.
- **nodes:** A container for node objects.
- **node:** defines the absolute path to the file within the current object carousel that shall be stored as part of the current group. Each file required to be stored as part of the group shall be defined explicitly. Directory entries that comprise paths to files shall not be described.
- **count:** is an optional field that if set describes that a number of files of given filename may be located at the current directory level. Each file has a numeric suffix in the range 000 to count. The extension to the filename shall be in the format "nnn" and shall include leading zeros. This may be used as a short-cut to describe a group that contains many, similarly named files and is intended to reduce the size of the manifest file only. The storage manager shall resolve and store the absolute path to each file at the time of storage in the same manner as when each file is referenced individually.

3.12.5 Group manifest management

The group update and acquisition process relies on the group version signalled in the stored_groups_descriptor and in the manifest file to be of equal value. Changes to the descriptor and manifest file shall therefore be synchronised at the head-end by the object carousel encoder. The client shall attempt to reacquire the manifest file pertinent to a group for a period of five minutes after change in group version is signalled within the stored_groups_descriptor. If the group manifest file is not updated within the defined period this is an error state and the client may halt the acquisition of the group until either of the following is true.

1. Until a new version is signalled in the stored_groups_descriptor.

OR

2. Until the current object carousel is un-mounted and an object-carousel is mounted that makes reference to the group.

The manifest file pertinent to a group shall be present in the object carousel for the same period over which the file(s) it describes are present in the object carousel.

The manifest file shall itself not be listed as an entry in the current or any other manifest and shall therefore not be stored. Manifest files shall always be acquired directly from the object carousel.

3.12.6 Stored File Access

3.12.6.1 Read access

Read access to stored files shall be transparent to any process that accesses the DSMCC file-system in that file references into DSMCC shall instead reference the stored equivalent if the following condition is true:

1. The file exists in the store and the group to which it belongs is not listed as stale.

AND

2. The stored group identifier and version that reference the file agree with the identifier and version in the DSI Stored Group Descriptor.

This enables the stored file mechanism to be accessed by any system that uses DSMCC object carousel for the carriage of files.

The use of the stored file-system by an MHEG application shall not interfere with other data delivery mechanisms such as Stream-Events, Interaction Channel and Hybrid File-System MHEG applications that rely on data specified with content-cache-priority zero should ensure that this data is not stored and shall therefore always be obtained from object carousel directly.

3.12.6.2 Write access

During the period over which a group is initially acquired from the object carousel or during the period over which an existing group is reacquired it shall be marked as stale by the cache manager. Individual files in the group shall not be available from store until the entire group has been acquired. Over this period references to DSMCC for the group currently being stored shall resolve to the object carousel instead of the store only if the use_from_carousel flag is set to "1" in the stored_groups_descriptor.

The client that implements the defined file acceleration and storage mechanism may employ methods of managing NVRAM to reduce the number of write-cycles and therefore prolong the life of NVRAM. Techniques may include keeping all updates to groups in RAM until the

receiver is placed into standby or until RAM usage reaches a manufacturer defined level before updating NVRAM.

3.12.7 Stored Group Deletion

Groups should not be required to be deleted during normal operation as they shall be removed either

1. When an update to the group is signalled and the new version of the group has been acquired.

OR

2. In the case where receiver resources are limited and a request to store a different group of higher priority is made. In this instance the memory used by the lower priority group(s) shall be relinquished and the associated reference to the stored group(s) shall be removed from cache.

It shall also be possible to relinquish memory allocated to stored groups by explicit methods. This may be achieved by signalling a standard group version change in the stored_groups_descriptor and associated manifest file. The manifest shall signal that the group size is zero bytes and shall describe an empty nodes section. Any previously used memory by the files in the group shall be relinquished.