

APPLICATIONS OF MPEG-4: DIGITAL MULTIMEDIA BROADCASTING

Marco Grube, Peter Siepen, Christian Mittendorf, Marco Boltz and Mayur Srinivasan

Robert Bosch GmbH / FV/SLM
Robert-Bosch-Str. 200, 31132 Hildesheim, Germany
Tel.: +495121 49-3921; Fax.: +495121 49-173921
E-Mail: marco.grube@de.bosch.com

Abstract

The new MPEG-4 based DMB-system offers multimedia information with an excellent audio- and video quality to the mobile society. The combination of MPEG-4 encoding and decoding with the DAB-standard (Digital Audio Broadcasting) enables for the first time a distortion free TV-reception in fast moving vehicles.

New concepts and algorithms which are required for real-time MPEG-4 audio and video encoding were developed, implemented and verified by their integration into the DMB-system. To cope with the demanding real-time requirements of MPEG-4, the corresponding hardware architecture also has been completely re-designed.

Keywords: DAB, DMB, MPEG-4, AAC, audio-coding, video-coding

1 Introduction

The MPEG-4 standard provides new and efficient methods for compression and presentation of multimedia content, which will be transmitted to mobile recipients in an increasing degree

The MPEG-4 based Digital Audio Broadcasting/Digital Multimedia Broadcasting (DAB/DMB) system is currently the only one that allows mobile reception of high quality video and audio streams with data rates up to 1.8 Mbit/s, even if the recipients are moving with high speeds.

In the following section, we will describe the DAB-transmission system and the improved DMB-channel-protection scheme, which is necessary for robust transmission of video data. This is followed by a short overview of the new features offered by MPEG-4.

Then, we will describe the software-architecture of the MPEG-4-based DMB-system. The major work items were the MPEG-4 audio and video codecs, which had to be highly optimised to achieve real-time performance. Also, good coding efficiency was essential due to the limited bandwidth.

For the hardware architecture of the system, we made use of standard PC-platforms along with newly designed FPGA and DSP-based components, which will be described in the following section. Finally, we finish the paper with a short summary.

2 The DAB/DMB-System

The DAB transmission system was developed for the broadcast of audio and accompanying data by the European standardisation activity Eureka 147 (ETS 300 401) [1]. It was designed for the demands of mobile receivers and offers an undisturbed reception even under difficult conditions. These results are due to the application of an OFDM-transmission system with a bandwidth of 1.5 MHz combined with a convolution code for error protection.

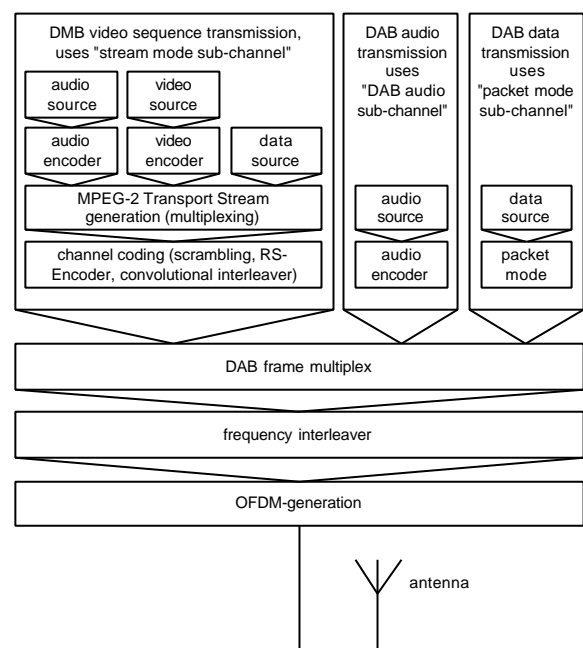


Fig. 1: The DAB/DMB transmission system

The maximum data rate of the DAB system is 1.824 Mbit/s. It is therefore capable of multiplexing different services, which can include several audio channels and additional data. Examples are traffic news, text data, still images and moving pictures. These properties show the suitability of DAB in multimedia broadcasting. This will allow parallel transmission of audio, video, text and additional data as shown in Fig. 1. The elementary streams resulting from the MPEG-4 encoding stage are wrapped into an MPEG-2 Transport Stream (TS) [2]. Compared to the DAB-system [3], DMB improves the error protection with the additional

blocks: scrambling, Reed Solomon encoder and convolutional interleaver. A corresponding system, based on the coding standard MPEG-2 is already available and currently in use for infotainment systems in local public traffic systems.

3 MPEG-4

The common coding standards, e.g. MPEG-1 or MPEG-2, encode the source signal as a complete scene. MPEG-4 follows the trend of composing a scene from different natural and synthetic objects by offering the possibility to encode those objects separately and to compose the resulting scene at the receiver. This approach enables a multitude of possibilities for efficient and signal adapted coding of different objects and allows interactivity on object level [4].

The standardisation process of MPEG-4 took place in several steps, which resulted in different versions of MPEG-4. Version 1 was standardised in February 1999, followed by version 2 in February 2000, which is an extension of Version 1 offering full backward compatibility.

In MPEG-4, *profiles* and *levels* have been introduced in a similar way as in MPEG-2. A profile creates a subset of all coding methods included in MPEG-4. Within a profile, the complexity of the coding methods is again reduced by the introduction of different levels.

In addition to object based functionality, MPEG-4 offers a large number of coding-tools which will highly improve the quality of encoded audio and video. The DMB-application uses the AAC-audio-codec (Advanced Audio Coding) of MPEG-4 version 1 and the Advanced Simple (AS)-profile¹ of MPEG-4 version 2. Both codecs offer excellent coding efficiency at an acceptable expenditure and are therefore well suited for broadcast applications.

3.1 MPEG-4 Audio

MPEG-4 audio represents a new kind of audio coding standard. Unlike its predecessors MPEG-1 and MPEG-2, it describes not only a small set of highly efficient compression schemes but also a complete toolbox for a broad range of applications from low bitrate speech coding (down to 2 kbit/s) to high quality audio coding (64 kbit/s per channel and above) or music synthesis. The natural coding part within MPEG-4 audio describes traditional type speech coding, high quality audio coding algorithms and their combination to enable new functionality like scalability (hierarchical coding) so as to overcome the boundaries of coding algorithms. The MPEG-4 Advanced Audio Coder (AAC) does generic high quality audio coding at medium to high bitrates.

The AAC follows the same basic coding paradigm as MPEG-1/2 Layer-3 (MP3) (high frequency resolution filterbank, non-uniform quantisation, Huffman coding, iteration

loop structure using analysis-by-synthesis), but offers improvements over Layer-3 in a lot of details and uses new coding tools for higher quality at low bitrates.

3.2 MPEG-4 Video

The coding method of MPEG-4 video is mainly based on a block based hybrid coding method, which is already used in MPEG-2 and different ITU-video standards. It consists of a motion estimation and compensation unit and a DCT-based coding algorithm for the remaining prediction error. In MPEG-4, the basic concept of a hybrid video coder was extended to allow the coding of arbitrarily shaped objects [8].

Compared to MPEG-2, MPEG-4 increases the coding efficiency by a large number of optimisations. The motion estimation and compensation algorithms of MPEG-4 are highly improved, especially in the ACE-profile and in the newly defined AS-profile (Advanced Simple) of MPEG-4. Therefore, this codec has been chosen for the DMB-system. The available data rate for video coding in DMB is 1.3 Mbit/s. Currently, the resolution of the sequences is CIF (352x288 pixel), but the system is designed in such a way that CCIR (720x576) resolution could be used too, if CPUs with sufficient performance are available.

3.3 MPEG-4 Systems

Compared to MPEG-2, the system part of MPEG-4 [11] offers numerous new possibilities. The most important new parts are the binary language BIFS (binary format for scenes) and the definition of the MPEG-4 File Format.

3.3.1 MPEG-4 BIFS

MPEG-4 BIFS allows the description of the contents of a complete scene including animations and interactions in the scene as a hierarchical scene graph. The corresponding synthetic graphic elements are described by their parameters and the video and audio data encoded with MPEG-4 are included by reference. For the evaluation of this information in the decoder, a compositor is needed, which will create the individual images of the scene by using the scene-graph description.

Applications of MPEG-4 systems are planned for future versions of the DMB-system, which will use MPEG-4 systems for the multiplex and display of additional (travel) information and to use the BIFS-syntax as standardised interface to editing programs.

3.3.2 MPEG-4 File Format

In addition to the "Live" mode of operation wherein audio-visual data is encoded "live" from a real-time source, MPEG-4 Systems [6] also offers some interesting possibilities for storing offline audio-visual data. Audio-visual data could thus be encoded offline with better quality and with a

¹ The AS-profile basically corresponds to the ACE-profile without object based coding tools

maximum bit rate of 1.5 Mbit/s (= the maximum DAB channel data rate).

Instead of storing data at the receiver end, more flexibility could be attained by storing the encoded data at the transmitter end by a standardised file format allowing carriage of MPEG-4 content, since making modifications to the stored data at the receiver end would involve the end-user. Data could then be suitably multiplexed into an MPEG-2 Transport Stream. This constitutes the “Stream” mode of operation. The “streamed” data is transmitted along the same channel as “live” data. Thus, dynamic switching between the two modes could be possible at the transmitter end itself. The file format stores an audio-visual presentation in the form of access units (media data). The format is transmission protocol independent and serves as a basis for a wide range of functionality. For instance, the file could be in interchange format wherein all the media data and the information needed to interact with it are contained in a single file. A single presentation could also be contained in separate files, thereby allowing the reuse of content and the possibility of including media data in files that is not according to a particular standard or specification.

The file is structured as a sequence of objects that may inherently contain other objects. Thus, the file format is object-oriented and can be implemented in an efficient way for interpretation purposes.

The basic unit of the MPEG-4 file format is called an ‘atom’. Access units (audio/visual) can be stored in atoms called ‘media’ atoms. Information describing the media data is called meta data and is stored in a ‘movie’ atom. Hence, for a single presentation, only one movie atom is allowed that contains relevant information for all the media data in the presentation. The ‘movie’ atom usually occurs at the top-level of a file. The movie atom contains several descriptive sub-atoms that store information such as:

- initial object descriptor
- type of media data (audio, visual, etc.)
- offset/location and size of the media samples within the MPEG-4 file
- decoding and composition time stamps in the form of tables
- user data and copyright information

Only certain atoms (for instance, the ‘movie’ atom) are mandatory and the presence of the other atoms/sub-atoms in the MPEG-4 file are left to the discretion of the implementation desired. Thus, the extensible nature of this file format offers innumerable prospects in terms of functionality.

4 Software Architecture

The realisation of the MPEG-4 based DMB transmission system is mainly based on standard PC-architecture with additional PCI/CPCI extension cards designed especially for the DMB-system. To achieve real-time performance for

video and audio encoding, which are both pure software implementations, there is a high demand for speed-optimisations of the encoders. These aspects will be described in the following sections.

4.1 MPEG-4 Audio

The audio codec used in the DMB-system is compliant with the MPEG-4 Advanced Audio Coding (AAC) Low Complexity Object Type and works with a sampling frequency of 44.1 kHz or 48 kHz. It produces an average bitrate of 128 kbit/s for two channels and outputs CD comparable audio quality [5], [6].

The AAC codec is based on the MPEG-4 Verification Model (VM) which has been profiled, debugged and optimised at C-code level. Beyond this, new algorithms have been implemented for the most process intensive parts. The target platform of the encoder is a Pentium III@500MHz with Linux as the operating system, whereas for the decoding system, a compact PCI decoder board has been developed using the Texas Instruments C6701 floating point DSP.

The following paragraphs give a short introduction to the structure of the encoder and describe the algorithmic improvements in more detail.

4.1.1 Overview: The AAC-Encoder

To gain a better understanding of the encoder structure, Fig. 2 gives a rough overview of the main modules within the MPEG-4 AAC Low Complexity Object Type:

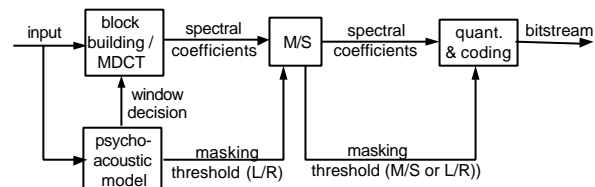


Fig. 2: MPEG-4 AAC audio-encoder

The encoder expects time domain PCM audio samples with a resolution of 16 bits per sample as input. The data stream is windowed with a block length of 2048 or eight times 256 samples per channel (50% overlapping with the foregoing block). From this follows a time equivalent of 21.3 ms (48 kHz) for one stereo “frame”.

One outstanding feature of the AAC is the block building process. This process uses “long” windows in the case of stationary signals and a sequence of eight “short” windows for transient signals. The introduction of “short” windows is necessary, since “long” windows would spread the quantization noise into the time period before the sound event. This leads to the problem of the quantisation noise being audible after decoding (“pre-echos”).

The psychoacoustic model decides which window type has to be used and describes the masking characteristics of the audio signal (masking threshold). This threshold marks the

maximum inaudible error energy in the time and frequency domain and, therefore, defines an upper border for the allowed quantisation noise introduced by the quantisers.

The MDCT (Modified Discrete Cosinus Transformation) transfers each data block to the frequency domain, where the main part of the encoding process takes place: quantisation & coding. The quantisation process adapts the quantisation error in the frequency domain to the masking threshold. The following coding unit uses Huffman codebooks to code the quantised spectral values.

One optional feature is the usage of mid/side coding and/or intensity stereo for further reduction of redundancy.

4.1.2 Quantisation & Coding Module

The profiling process shows that the quantisation & coding module is the most time consuming part of the encoder. The main reason for this is the nested loop approach used by the original VM implementation. The goal of the concept, as displayed in Fig. 3 is the adaptation of the quantisation noise to the masking threshold considering the maximum number of bits available to encode the current frame. Since the codec works with a quasi-variable bitrate, it offers a slightly higher or lower bitrate than the average bitrate over a limited time period so as to encode critical sequences more accurately. Depending on the number of bits used in the foregoing frames, a “bit-reservoir control unit” determines the “allowed” number of bits for the current frame.

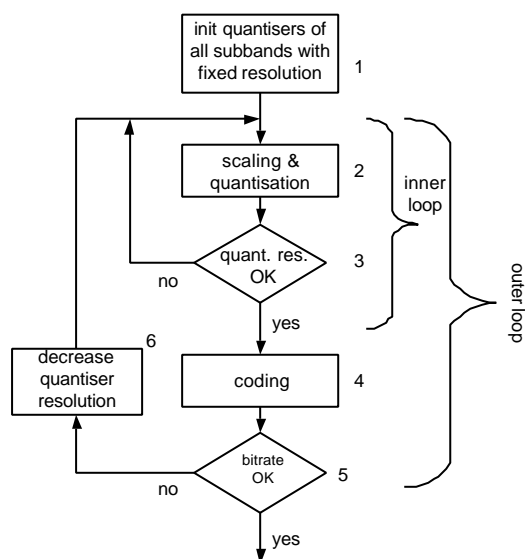


Fig. 3: Nested loop concept for quantisation & coding

The first block initialises the quantisers of all frequency bands (subbands) with a low resolution, which results in a large quantisation error. Within the inner loop, the resolution of each quantiser is increased (block 2) until the error energy of every subband is smaller than the error energy calculated in the psychoacoustic model (masking threshold).

From this follows the need to re-calculate the error energy of every subband in every loop. The loop terminates, if an optimal resolution vector is found (block 3). Block 4 of the outer loop determines the number of bits needed to encode the spectrum with the resolution vector found by the inner loop. Block 5 compares this value to the bitrate given by the bit-reservoir control unit. If the bitrate is smaller, then the quantisation & coding process terminates. Otherwise, the resolution of all quantisers is decreased and a repeated processing of the inner loop is started.

This approach has the disadvantage that the initial quantiser resolution is not correlated to the masking threshold. Therefore, the quantisation & coding module needs several iterations until an acceptable combination of bitrate and quantisation noise level is found. To reduce the amount of loops drastically, two estimation algorithms have been introduced. They take known characteristics of audio signals and requirements from the bit-reservoir control unit into account and consequently allow the initialisation of the quantisers with an optimal resolution vector. The first algorithm estimates the bitrate from the perceptual entropy (PE, information content: audible parts of the audio signal) and then adapts the masking threshold accordingly. The second one estimates the quantiser resolutions from the signal energy and the adjusted masking threshold.

This results in the inner iteration loop becoming obsolete. Also, the outer loop could be reduced to a maximum of five iterations without loss of audio quality, even if the sequences are very critical. The basic ideas of the estimation algorithms can be found in the block diagram of Fig. 4.

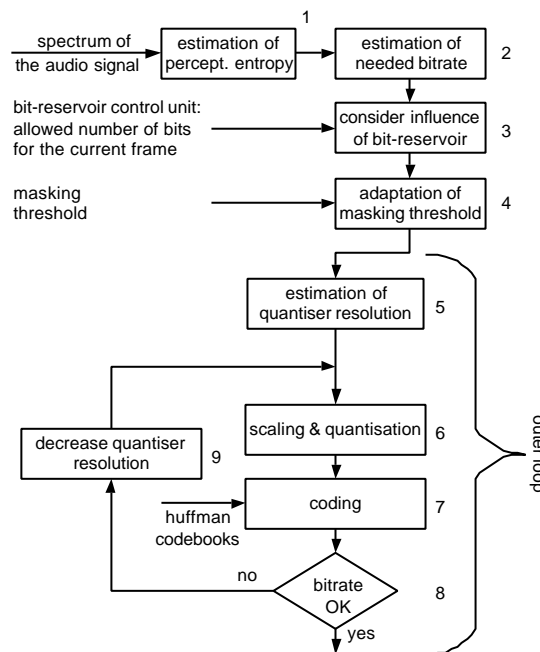


Fig. 4: Improved quantisation & coding process

4.1.3 Estimation of the bitrate from the perceptual entropy and adaptation of the masking threshold

Fig. 4 shows that the estimation process can be regarded as a kind of pre-processing (blocks 1 to 4) for the following outer iteration loop (blocks 5 to 9) [7]. The first block calculates the perceptual entropy from the error energy (masking threshold) and the signal energy. The second step estimates the bitrate from the perceptual entropy, which is needed to encode the frame optimally. The correlation between the number of bits per frame, channel and the perceptual entropy was determined with help of simulations and is depicted in Fig. 5.



Fig. 5: Bits per frame = $f(PE)$

The algorithmic correlation can be given in a good approximation by

$$br = a * pe + b * \sqrt{pe} + c.$$

Appropriate values for a , b and c adapted to the transformation length of the MDCT (for signals with transient or stationary characteristics) can be found with the help of simulations.

The estimation process assumes the calculated entropy and consequently a masking threshold, which results in a quantisation process without audible distortions. If the estimated bitrate is not available from the bit-reservoir control unit (block 3), then the quantiser resolution must be decreased and therefore, a larger amount of quantisation noise must be allowed (block 4).

4.1.4 Estimation of the quantiser resolution from signal and error energy

With the results described above, the preparatory work which was essential is now complete, since the allowed error energy threshold is now adapted to the bitrate available for the current frame. Therefore, the “new” threshold is used as a starting point for the following estimation of the quantiser resolution needed in each subband (block 5) [7].

To determine the quantiser resolution (scalefactors in terms of MPEG-4) effectively, an estimator was developed based on simulations by quantising and re-quantising a series of spectral values with different scalefactors. The results can be found in Fig. 6, where the quantiser error energy level is shown vs. the scalefactors used.



Fig. 6: Error signal level vs. scalefactors under assumption of Gaussian White Noise (GWN)

The figure shows a scalefactor that increases linearly with the error energy level, until the error energy runs into saturation. This condition arises if the quantisation error energy equals the signal energy, i.e. the signal is quantised to “zero”. The correlation found in the simulations can be described as follows ($error_dB$ is error energy level, A can be determined in simulations):

$$error_dB = A * scf + offset.$$

A further simulation for the “offset”, i.e. the error energy level for the scalefactor $scf = 0$ shows a close relation between the error signal level and the signal energy level. This “offset” depends on the probability distribution of the spectral values to be quantised. The assumption of Gaussian White Noise (GWN) in this scenario is a very good approximation, whereas, the procedure itself could be used for other signal types and probability distributions. From these simulations, a rule to calculate the scalefactors (equivalent to the quantiser resolutions) from the error energy level and the signal energy level has been determined.

Block 6 of the outer iteration loop processes the quantisation without iterations. Block 7 encodes the quantised data and Block 8 compares the number of bits needed to encode the signal with the number of bits available. If this value is small enough, then the quantisation & coding module terminates. Otherwise, the quantiser resolution is decreased in all subbands and a second iteration loop is started.

The objective of estimating the scalefactor directly from the signal energy and the error energy for each subband is thereby achieved. Hence, the quantisation noise is shaped without iterations (and the inner loop is obsolete).

4.1.5 Performance

The encoder modifications described above reduce the algorithmic complexity compared to the faultless VM code on both hardware platforms, CPU and TI C6701 DSP, drastically. The measurement showed that the encoder runs on a Pentium III@500MHz platform with a factor of 2.5 faster than real-time, compared to a factor of seven slower than real-time at the beginning of the optimisations. For both processors, the performance (computationally complexity) compared to the VM-code decreased by a factor of 17. A factor of eight comes from the algorithmic optimisations within the quantization & coding routine, whereas a factor of two could be achieved with optimisations within the psychoacoustic model.

4.1.6 Quality

The audio quality was judged in listening tests and, additionally, with the help of a measurement tool for the objective and perceptual based evaluation of compressed audio signals. The used algorithm is based on the original reference implementation of the “Perceived Audio Quality” Measure (PEAQ), which was the source for ITU-R Recommendation BS.1387. The result of the measurement is the “diffgrade” value. The “diffgrade” describes the loss of subjective audio quality in the range from -0.0 (original) to -4.0 (very bad quality). A value of -0.95 corresponds to CD comparable quality. The following table shows the results for the speed-optimised encoder:

Codec	Test 1	Test 2	Test 3	Speech
DMB, 96 kbit/s	-2.70	-3.06	-2.88	-1.43
DMB, 128 kbit/s	-1.04	-1.47	-1.34	-0.13

Table 1: Results of the “PEAQ” quality measurement

From this table, it can be seen that the 96 kbit/s mode (stereo) is still not fully optimised. This could constitute an important work package for the future. The 128 kbit/s stereo mode reaches very good results with an average value of -1.28 for music signals. Coded speech excerpts are not distinguishable from the original source (CD-comparable quality).

4.1.7 Decoder

The target platform for the decoder is a Texas Instruments C6701 floating point DSP. The decoder software was based on the original VM code, which was debugged, profiled and optimised under consideration of DSP specific characteristics: memory usage, code size, etc. The optimised software is able to decode MPEG-4 audio five to seven times faster than real-time.

4.2 MPEG-4 Video

The MPEG-4 video encoder developed for the DMB project is a completely new design, which is optimised for true real-

time performance on standard PC-hardware. It is able to encode sequences with CIF resolution according to the MPEG-4 ACE-profile in real-time on a 500 MHz Dual-Pentium III processor PC-system. The most computational intensive part of a video-coding algorithm is usually motion estimation. Therefore, a fast motion estimation algorithm [9] has been adapted to the requirements of an MPEG-4 encoder so as to reduce the high computational power needed by standard (block matching) motion estimators. This fast motion estimation is based on the principle of optical flow and supports motion vectors with 1/4-pel and 1/2-pel resolutions. Furthermore, an optimised algorithm for the Global Motion Estimation along with a quality optimised rate control is implemented. To achieve real time performance (especially for the 1/4-pel motion estimation), a large number of performance optimisations has been carried out by using the MMX and SSE capabilities of the Pentium III processor. The encoder also supports multithreading, so that real time performance is achieved on a dual processor board. The MPEG-4 video decoder is also able to handle the ACE profile. The functions which consume the most computational power (e.g. motion compensation) are also optimised with assembler code. Here, the decoder only uses the MMX instruction set extension so as to be compatible with a broader range of CPUs. Another feature of the decoder is its ability to decode MPEG-2 bitstreams.

4.2.1 Multi-thread concept

The encoder can use up to three threads allocating the most time-intensive procedures of the encoding process. These deal with frame grabbing, motion estimation and texture encoding.

The texture encoding thread is present at the highest hierarchy level. This one is always present and based on specific options in the parameter file, decides upon creating another thread and on directing and sharing certain tasks with it. The second one is the motion estimation thread, which is initiated at the encoder level.

At the motion estimation level, the decision about using the third thread, which reads from the frame grabber, is made based on a certain option flag in the parameter file.

It is also possible to choose the usage of two threads instead of three. In that case, reading from the frame grabber and the motion estimation is done by one single thread.

The communication between these threads is realised by the usage of critical sections, which are realised by special variables. These sections can be locked by either thread, assuring that only one thread can be at the corresponding critical section at the same time. The other synchronisation mechanism is the use of events, forcing one thread to wait until it receives a signal from another thread to proceed.

Assuming the usage of three threads, the structure in Fig. 7 is valid for the encoding process:

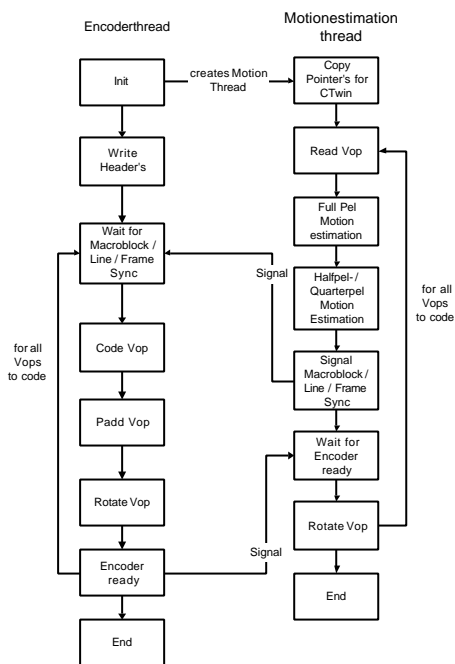


Fig. 7: Two-threaded working flow

Fig. 8 shows the case of using three separate threads.

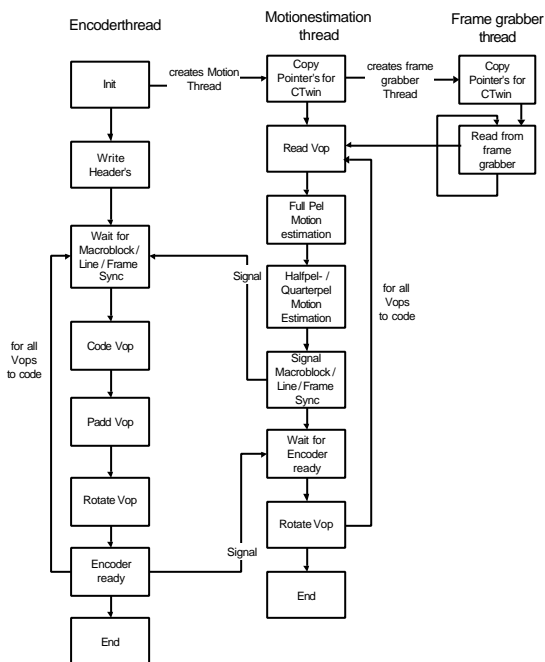


Fig. 8: Three-threaded working flow

It is also possible to use only one thread, but this will result in the lowest performance of the encoding process.

4.2.2 Automatic Resolution Modification (ARM)

The DMB-system requires a constant bitrate (CBR) for the transmission of the video bitstream and therefore requires the use of a rate control to stay within the limits of the given rate. To adjust the bitrate, the rate control modifies the quantization parameter (QP), which controls the quantization of the DCT-coefficients. Video sequences, which are complex in structure and movement, have to be quantised coarser than simpler scenes.

In some situations, it may occur that the required bitrate can not be guaranteed although a very high QP is already chosen. In this case, single frames of a sequence have to be skipped, which means, they will neither be coded nor transmitted. The saving of bits can be immense but the skipping of one or several frames can result in a very annoying visual impression of the decoded sequence.

To find a solution to this problem, it is possible to choose a resolution for the source sequence, which is low enough to avoid frame skipping from the very beginning. But this also leads to a low visual quality on the other hand. The problem requires a compromise between two contradictory parameters. The first one is: increasing the visual quality by choosing a high resolution of the source sequences. The second one is: avoiding frame skipping at complex scene situations by choosing a low resolution for the source sequence.

The approach of ARM to cope with the problems explained above is to adapt the resolution of the source sequence during the coding process. In the case described here, two different resolutions are used. The switching criterion implemented is derived from the fill level of a bit buffer and the QP-value. Based on this criterion, the coding resolution is varied automatically and is adapted to the content of the sequence.

If a high complexity of the video sequence is detected by these criterions (high buffer level, rough quantization), skipping of frames is avoided by reducing the resolution of the source format by filtering and subsampling. With sequences of medium or low complexity, the coding process uses the highest resolution and provides the highest visual quality.

Since ARM is not an inherent part of the encoder, the switching of the coding format is realised by partly reinitialising the coding process. In the context of MPEG-4, a new Video Object Layer (VOL) header and an INTRA-coded frame is sent to the bitstream in the case of resolution switching.

Comparing the various methods of providing a constant bitrate, it can be seen that ARM causes the least negative side effects. Choosing a very high QP results in a very low subjective visual quality of the video. Frame skipping makes the sequence to stagnate when the missing frames are replaced by "freezing" the previous one. Therefore frame skipping should be avoided whenever possible.

Using ARM results in a vague or blurred visual effect, which still looks subjectively better than the other side effects. Also, the resolution switching only occurs in situations of heavy movements, pans, blends or cuts in the scene. In these scenes, the human eye is usually unable to notice these side effects due to the blur resulting from the movement.

In the following sections, the different steps of ARM are described in detail:

- a) Before coding a single frame of the sequence, the decision of reducing the resolution is made by querying the level of a bit buffer whose content is explained later on. The content of the buffer is re-calculated after a frame is coded and therefore, is valid for the next frame of the sequence. If the level rises beyond a certain limit, a reduction is initiated.
- b) If the buffer is at an uncritical level (below a certain limit) the following coding process uses the full resolution of the source video.
- c) If the decision on the reduction is positive, the grabbing of the source frame is done by using a decimation process, which reduces the frame to a quarter of its original size. The coding process will now use much less CPU-power and will use a finer QP, which results in a better quality of the reconstructed frame.
- d) Based on an indication within the bitstream, the previous encoding session is cancelled and a new one is started. This new session begins again with the initialisation of the relevant parts of the encoder with the new values. A new VOL header is written and the first frame of this new session is encoded as an I-frame.
- e) For decoding these kind of frames, the decoder software has to “know” how to handle frames with a smaller resolution than the frame size of the sequence. An interpolation to the display resolution is necessary to present the frame correctly. Except for some blurry or vague visual effects, frame skipping and block effects are minimised or completely avoided.
- f) Before coding the next frame, the amount of used bits for this frame is added to the mentioned bit buffer. At the same time the amount of bits that could have been required is subtracted.
- g) If the level of the buffer goes below a certain limit, the encoder will decide on a switch back to the normal, full resolution. Again, the current encoding session will be cancelled and a new one will be started with the full size values. A new VOL header is written and the first frame after the decision has been made will be encoded as intra frame.
- h) There is one restriction on the frequency of the switches. A definable value describing the amount of frames that have to be coded in the same size avoids a flickering in the sequence that could occur due to frequent changes of the resolution values.

The decoder receives the bitstream in the size it has been encoded and decides when to resize the reduced frames. It is also possible to signalise those reduced frames on system level. But the method of managing the recognising and resizing of switched frames directly in the decoder has finally been chosen [10].

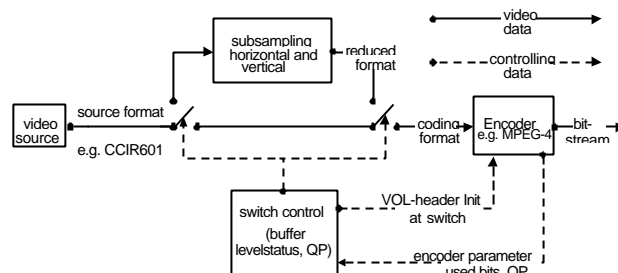


Fig. 9: Encoder with ARM

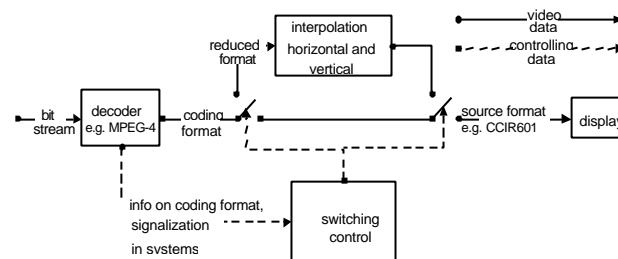


Fig. 10: Decoder with ARM

The following figures show some results regarding the visual quality of a decoded test sequence encoded with ARM:

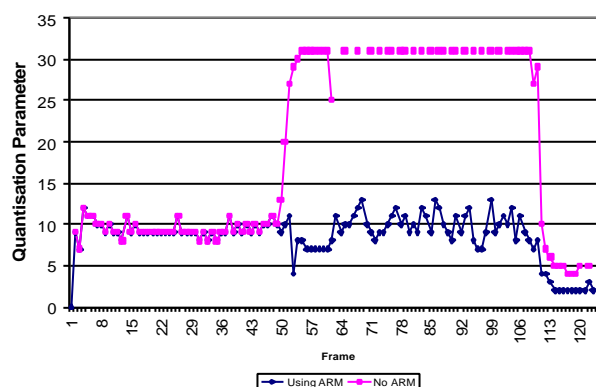


Fig. 11: Quantization parameter used by the encoder with and without ARM

The gaps in the graph without ARM (Fig. 11) show that several frame skipings occurred during the encoding pro-c

ess. The higher QP in this graph indicates a lower visual quality of the decoded sequence.

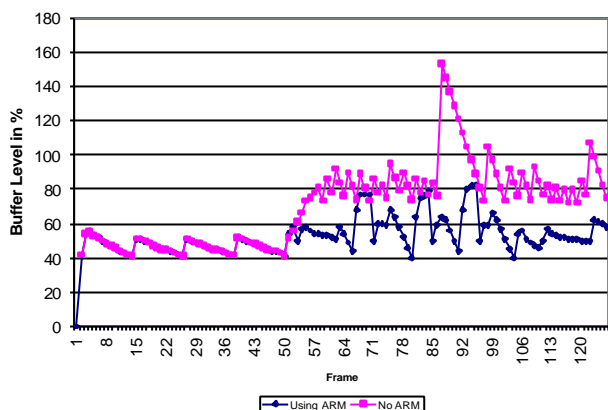


Fig. 12: Buffer level of the encoding process with and without ARM

Using ARM, the buffer level is always lower compared to that without ARM. Encoding without ARM makes the buffer level overflow which has to be compensated (notice the peak in Fig. 12) by skipping frames and coding at a higher QP. That results in a poorer visual quality.

4.3 MPEG-4 File-Interpretation

The MPEG-4 file interpretation concept has been realised in software so as to create, interpret and transmit MPEG-4 audio/visual data.

The “Stream Mode” scenario explained previously is as shown in Fig. 13.

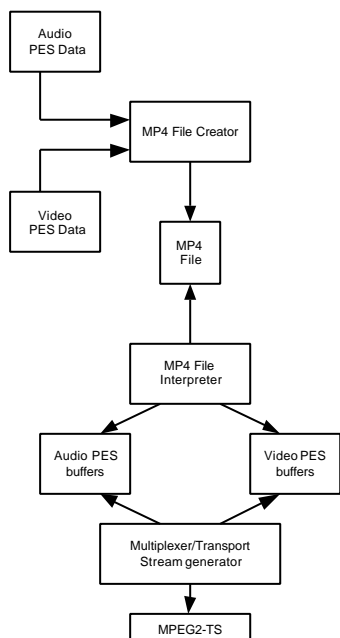


Fig. 13: Stream Mode

The software developed for the MPEG-4 file creation and interpretation process is based on the Syntactic Description Language (SDL). This language directly extends the typing system of the C/C++ programming language, thereby providing an excellent framework for the organisation of the various atoms/sub-atoms. Each atom is identified by a header field that gives both the size and type of information contained. Since all information is encapsulated within pertinent atoms, parsing of the file for the required atoms becomes simplified.

4.3.1 MPEG-4 File Creator:

Audio-visual data encoded offline with very low bit-rates are packetised into individual PES streams which are then composed into an MPEG-4 file by the MPEG-4 File Creator. Synchronisation is achieved by storing the Presentation Time Stamp (PTS) and the Decoding Time Stamp information (DTS) in the form of tables.

The DTS table (stored in the ‘dttts’ atom) contains decoding time deltas according to the relation

$$DT(n+1) = DT(n) + STTS(n) \text{ where,}$$

$STTS(n)$	table entry(offset) for sample n
$DT(n+1)$	DTS entry for sample n+1
n	number of samples.

The PTS table (stored in the ‘ctts’ atom) contains composition time deltas according to the relation

$$CT(n) = DT(n) + CTTS(n) \text{ where,}$$

$CTTS(n)$	table entry(offset) for sample n
$CT(n)$	PTS entry for sample n.

This MPEG-4 file is then stored on a hard disk local to the transmitter side and “interpreted” as and when desired.

4.3.2 MPEG-4 File Interpreter:

Interpretation involves extracting the access units (audio/visual) and creating PES streams along with relevant time-stamping information. Due to the structured nature of the file format, it is possible to concurrently interpret audio/video data by the usage of threads.

These streams are then multiplexed based on the DTS time stamp values of the PES streams into an MPEG-2 compliant transport stream and transmitted over the DAB system.

5 Hardware architecture

The entire system concept comprises of all components necessary for transmitting and receiving audio-visual data. In this context, the requirements of the stationary transmission station and the mobile receiver have to be distinguished.

The transmission station consists of a content creation and a transmission unit. In the content creation unit, the preparation of the content takes place by the use of special author-

ing tools and is encoded using a real-time MPEG-4 encoder. In the offline-case, the resulting bitstreams are transferred to the file-server in the transmission unit and will be transmitted later, whereas, in the live-mode, the output of the MPEG-4 encoder is directly fed to the data and ensemble-multiplexer.

5.1 Encoder architecture

The outstanding features of the encoder system are its modularity and expandability. The processing of the audio and video signals is done in two different subsystems. Both are based on a dual CPU-board. The demanding MPEG-4 algorithms are processed on three Pentium III processors with 800MHz. A fourth processor is used for multiplexing and for additional tasks. Both dual Pentium III systems are connected via an Ethernet connection.

For the connection of external sources, a universal interface board was developed. This board is able to deal with analog and digital audio and video sources. Moreover, this board delivers timestamp information for both the video and audio signals, which are necessary to synchronise audio and video signals reliably. These timestamps are derived from a 10 MHz reference clock and are used by the multiplex unit to create an MPEG-2 Transport Stream [2] from the audio and video streams. This transport stream is then fed to the TSG-board (transport stream generator), which applies the standard DAB channel coding and adds an additional channel code (Reed Salomon and Interleaving) for a safe transmission over the DMB-channel.

To cope with the real-time processing demands, the operating system on both encoder subsystems is SMP-Linux with a real-time extension.

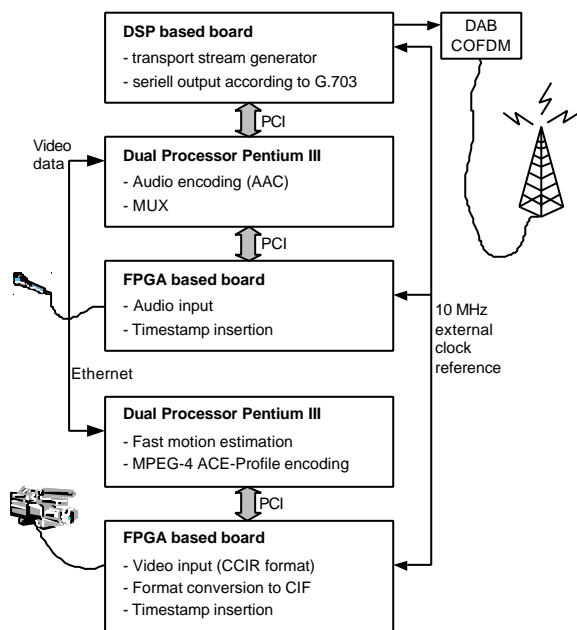


Fig. 14: Encoder system

5.2 Decoder architecture

With respect to mobile use, the important demands of the decoder are small size and an increased reliability of the bus-connectors. For these reasons, the decoder system uses Compact-PCI technology.

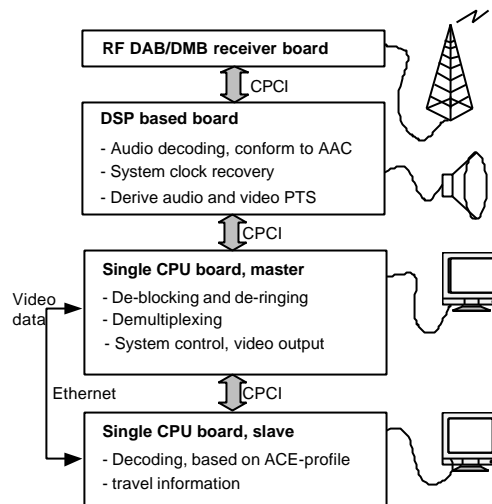


Fig. 15: The decoder system

The decoder system consists of two separate CPUs, which are connected via an Ethernet connection. On both CPUs of the decoder-system, the operating system is Windows NT. The antenna signal is connected to the RF-DMB-module. The master-CPU reads the transport stream from the RF-module and de-multiplexes the audio and video elementary streams. The audio stream is delivered to the DSP-based audio-board, which decodes and presents the AAC-encoded audio data. The video stream is transferred to the slave-CPU, which decodes the video data according to the ACE-profile. The postprocessing of the decoded video stream is done on the slave-CPU. The compositor task on the master CPU finally presents the decoded video stream. An additional task on the slave-CPU is the presentation of information such as travel information on a second display. To achieve synchronised presentation of the video and audio data, there is a central clock recovery unit included on the DSP-board, which delivers synchronisation information to the master and slave CPU.

6 Conclusion

The MPEG-4 based DMB-system, which has been presented in this paper, allows the mobile reception of audio and video information.

We gave a detailed description of the optimisation steps to achieve real-time encoding and decoding according to the MPEG-4 standard. This resulted in new concepts like ARM or the new estimation algorithms for the AAC-Encoder, which improves the audible and visible quality without increasing the bitrate. Their implementation and integration into the system have verified the robustness and perform-

ance of the new algorithms. Additionally we explained a modular and extensible hardware architecture, which is able to cope with our requirements and allows easier adaptation to the system for future applications.

The work described here will be the basis for further DMB-related standardisation activities to make DMB available for public use.

7 Acknowledgement

The work described in this paper was carried out in a project funded by the Bundesministerium für Wirtschaft und Technologie (BMWi).

8 Authors



Marco Grube received his Dipl.-Ing. degree in electrical engineering from the University Dortmund, Germany in 1994. He joined Bosch Advanced Development Multimedia Systems, Hildesheim in January 1995. Currently he is the project leader for the MPEG-4 based DMB system. He has worked in

the R&D design area on digital communication interfaces, MPEG-2 TPS multiplexer, digital television transmission system (DVB-T), VLSI multimedia chip design, FPGA based rapid prototyping and was in charge of leading the workpackage "Chip and system test" of the MEDEA project A116.



Peter Siepen received his Dipl.-Ing. degree in electrical engineering from the University of Duisburg in 1995. He worked at the faculty of communication engineering of the same university as scientific collaborator. His activities included research on digital image compression and fractal coding for data reduction.

He joined Bosch Advanced Development Multimedia Systems, Hildesheim in August 1999. Thus far he worked as software development engineer in the domain of MPEG-4 and for the DMB project.



Marco Boltz graduated from the University of Applied Sciences in Dresden, Germany, and received his Dipl.-Inf. degree in media computer science in 2000. His dissertation work comprised the optimisation of the coding efficiency of a MPEG-4 video encoder for the Heinrich-Hertz-Institut in Berlin. He joined Bosch Advanced

Development Multimedia Systems, Hildesheim in December 2000. Currently he is a software development engineer for the MPEG-4-DMB project.



Christian Mittendorf received his Dipl.-Ing. degree in electrical engineering from the University Hanover, Germany in 1998. He joined Bosch Advanced Development Multimedia Systems, Hildesheim in Mai 1998. Currently he is an software development engineer for the MPEG-4-DMB project.

He has mainly worked on the implementation and optimisation of high-quality real-time MPEG-4 audio and speech codecs and actively contributed to the standardisation of MPEG-4 Audio.



Mayur Srinivasan received his Bachelor's Degree in Electronics and Communication Engineering from B.M.S College of Engineering, Bangalore University, India in 1999. He joined the Department of Communication Applications, Robert Bosch India in October 1999. Currently,

he is working for Advanced Development Multimedia Systems, Hildesheim for the MPEG-4 based DMB system. He has mainly worked on MPEG-2 and MPEG-4 systems.

References

- [1] "Radio broadcast systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers", ETS 300 401, Sophia Antipolis, 1994
- [2] "Information Technology – Generic coding of moving pictures and audio information: Systems", ISO/IEC 13818-1:1996(E)
- [3] T. Lauterbach: „Digital Audio Broadcasting“, Franzis-Verlag, 1996
- [4] S. Bauer, T. Mlasko, B. Schmale and J. Vollmer, „The MPEG-4 Compression and Presentation Scheme and its Future Multimedia Applications“, EMMSEC 99, Stockholm
- [5] "Information technology - Coding of audio-visual objects - Part 3: Audio", ISO/IEC 13818-7: 1997(E)
- [6] "Information technology - Coding of audio-visual objects - Part 3: Audio", ISO/IEC 14496-3: 1999(E)
- [7] Andree Buschmann, "Efficient estimation algorithms for the use with the MPEG-4 audio encoder.", Robert Bosch GmbH, Department FV/SLM, August 2000
- [8] "Information technology-Coding of audio-visual objects-Part 2:Video",ISO/IEC 14496-2/Amd 1,Jan. 2000
- [9] J.-R. Ohm and K. Rummeler, "Variable-Raster Multiresolution Video Processing with Motion Compensation Techniques", Proc. of IEEE Int. Conf. on Image Processing, ICIP-97, Oct. 1997
- [10] M. Boltz: "Optimierung der Kodiereffizienz eines MPEG-4-Videoencoders", HHI Berlin, HTW Dresden, Oct. 2000
- [11] "Information technology - Generic coding of audio-visual objects: Systems", ISO/IEC 14496-1: 1999(E)