

PUBLIC



IST R&D. FP6-Priority 2.
SPECIFIC TARGETED RESEARCH PROJECT
Project Deliverable

SUIT Doc Number	SUIT_211
Project Number	IST-4-028042
Project Acronym+Title	SUIT- Scalable, Ultra-fast and Interoperable Interactive Television
Deliverable Nature	Report
Deliverable Number	D5.2
Contractual Delivery Date	30 November 2006
Actual Delivery Date	15 January 2007
Title of Deliverable	Real-Time SVC Encoder Specifications
Contributing Workpackage	WP5
Project Starting Date; Duration	01/02/2006; 27 months
Dissemination Level	PU
Author(s)	Olivier Guye (Vitec), Erwan Le Bras (Vitec), Antonio Navarro (IT)

Abstract

This document describes the architecture of a Real Time implementation of a Scalable Video Coding algorithm that can be used to provide Multiple Descriptions used to retrieve information from packet erasures in an error prone communication environment. It relies on the scalable extension of the H.264/AVC algorithm and it defines the different steps allowing the achievement of a real-time implementation that can process up to HDTV formats.

Keyword list: Advanced Video Coding, Scalable Video Coding, Multiple Description Coding, Real-Time Processing, High Definition Television

PUBLIC

Real-Time SVC Encoder Specifications

SUIT_211

15-01-2007

Table of Contents

1	INTRODUCTION.....	4
1.1	SCOPE.....	4
1.2	OBJECTIVE.....	4
2	OVERVIEW OF THE H.264/MPEG-4 AVC VIDEO CODING STANDARD	5
3	OVERVIEW OF THE SCALABLE EXTENSION OF H.264/AVC	8
4	DIFFERENCES BETWEEN OFF-LINE AND REAL-TIME ENCODERS.....	14
5	REAL-TIME CODER DEVELOPMENT TECHNIQUES	16
6	REAL-TIME IMPLEMENTATION OF AN SVC CODER	18
6.1	REAL-TIME IMPLEMENTATION OF A SCALABLE VIDEO CODER	18
6.2	SUIT SVC REFERENCE PLATFORM.....	19
6.3	REAL-TIME ALGORITHM FOR SPATIAL SCALABILITY	20
6.4	REAL-TIME ALGORITHM FOR SNR SCALABILITY	21
6.5	BITSTREAM SPECIFICATION	24
	6.5.1SUIT SVC bitstream structure	24
	6.5.2NAL Header and NAL Unit.....	25
	6.5.3Encoding coefficients in SNR refinement layers.....	27
6.6	BITSTREAM EXTRACTION.....	28
6.7	FILE FORMAT FOR SUIT SVC ELEMENTARY STREAM	30
7	MULTI-DSP ENCODER SPECIFICATIONS.....	31
7.1	VITEC MULTI-DSP PLATFORM FAMILY OVERVIEW.....	31
7.2	SUIT REAL-TIME SVC ENCODER SPECIFICATIONS	31
8	CONCLUSIONS	34
9	ACRONYMS	35
10	REFERENCES.....	37
11	ANNEX: VMC-5400 DATA SHEET	38
12	ANNEX: VP3 DATA SHEET	40

1 Introduction

1.1 Scope

This document is set up inside the framework of SUIT FP6 project. The scope of this document is to describe the specifications of a Real-Time Scalable Video Coder that can be adapted to provide Multiple Description coded video streams and that can deal with a wide range of video formats including High Definition. Starting from the description of the AVC standard and its extension to SVC, it tries to define the techniques to use for developing a real-time version and the computing architecture that can deliver the necessary computation power. In complement, it proposes to provide the project with a SUIT SVC reference software platform that should help SUIT developers and insure that most of their developments will remain available at demonstration time. A first reference platform is delivered simultaneously with this document.

1.2 Objective

The main objective of this document is to present the techniques that are used to implement a real-time version of a video coding algorithm. For implementing a real-time version of a video coding algorithm, there are two possible ways to achieve it: starting from a reference model and optimising it, or starting from a real-time release of a previous encoding scheme which is supporting part of the mechanisms present of the new coding algorithm. As the SVC algorithm is an extension of the AVC one, we are then in a favourable situation for reusing the previous efforts made for building real-time implementations of the AVC algorithm. So a particular attention will be paid on defining the complementary tools to implement in an AVC encoder to transform it in an SVC one.

In order to help SUIT partners in their own developments and before that a real-time version will be available, it is proposed to provide an SVC reference software platform that would fulfil the following requirements:

- it should be a midterm version between the whole JSVM and the real-time release that could be implemented for SUIT demonstrations;
- it should insure that SUIT developments based on it would have a high probability to be included in SUIT demonstrations if they obey to real-time constraints;
- it must be as possible closed to AVC implementation in order to reuse existing real-time encoding platforms or platform currently in development.

Concerning the general objectives of SUIT project, the SVC coding platform will include the following features:

- provide a single framework for encoding, editing bitstreams (for adaptive QoS at intelligent multiplexer or gateway levels) and decoding (at terminal side);
- allow different MD implementations from the simplest one (running SVC several times) up to more advanced ones (EMDSQ implementation in the reference framework);
- take in account HD video formats (preferably in progressive scan);
- and provide a real-time decoder and a renderer.

2 Overview of the H.264/MPEG-4 AVC Video Coding Standard

The H.264/AVC video coding standard has recently emerged from the cooperation of two experts groups: the Video Coding Experts Group (VCEG) from ITU, which has developed all the H.26x video-telephony coding standards and the Moving Picture Expert Group (MPEG) from ISO/IEC, which have built the MPEG video standards used in storage, broadcast or streaming applications.

The first objective of the Joint Video Team (JVT) was to define a new coding scheme that covers both the MPEG-2 and H.263 application domains while overriding existing compression rates by a factor-of-2. The second objective was to accommodate video content delivery to wide variety of bandwidth requirements, picture formats and network environments.

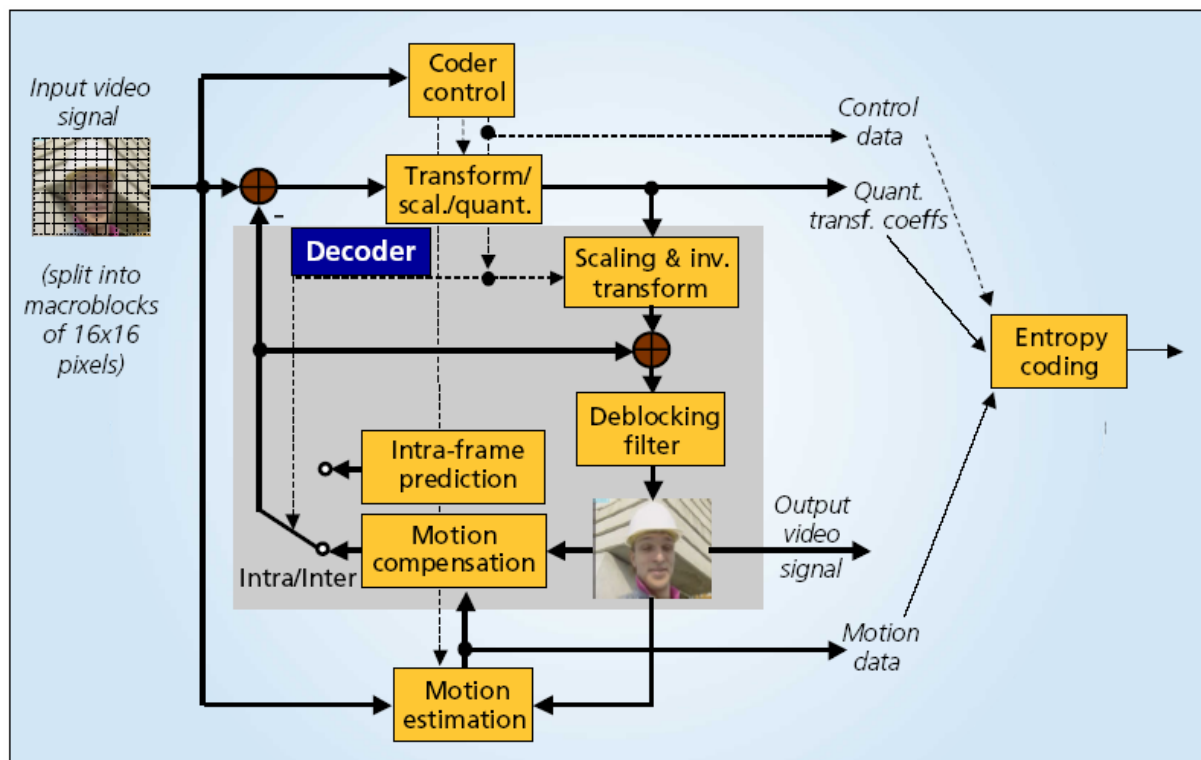


Figure 1: Basic structure of H2.64/AVC for a macro-block

The standard architecture is divided into two layers:

- a Video Coding Layer (VCL) which represents the video content;
- a Network Abstraction Layer (NAL), which formats the VCL representation of the video and provides header information convenient for transport layers or storage media.

A same format is used by NAL units for both packet-oriented transport and bit-stream delivery. The NAL facilitates the ability to map H.264/AVC VCL data to transport layers such as:

- RTP/IP for real-time wire-line and wireless Internet services (conversational and streaming);
- file formats, e.g. ISO MP4 for storage and MMS;
- H.32x for wire-line and wireless conversational services;
- MPEG-2 systems for broadcast services.

The VCL is similar in spirit to previous standards as MPEG-2 Video: it consists of a hybrid of temporal and spatial prediction, in conjunction with transform coding, as shown by the above diagram (excerpt from [1]).

With comparison to previous standards, H.264/AVC provides significant improvements in:

- Inter-Prediction and Motion Estimation;
- Intra-Prediction and Transform coding;
- Entropy Coding;
- Error containment.

Motion estimation with MPEG-2 allows to search for 16x16 fixed size blocks of pixels inside one or two reference pictures, H.264 provides more flexibility with:

- fine-grained motion estimation where motion estimation applies on blocks of variable size inside a macro-block (MB);
- multiple reference frames can be used;
- unrestricted motion search allows displacements outside a reference frame using spatial prediction;
- motion vector prediction for continuous movements.

Intra prediction and transform coding:

- intra prediction is a new feature that allows to efficiently code uniform areas of picture using direction dependent intra modes;
- implementation of a 4x4 integer Discrete Cosine Transform (DCT) that provides a finer grain and an accurate data transform;
- an in-loop de-blocking filter for smoothing edges and avoiding the occurrence of visual artifacts.

Entropy coding takes in account context of data being encoded and provides two alternative methods:

- Context-Adaptive Variable Length Coding (CAVLC) that employs multiple variable length codeword tables to encode data;
- Context-Adaptive Binary Arithmetic Coding (CABAC) that provides an improved encoding scheme with unmatched bit/symbol ratios.

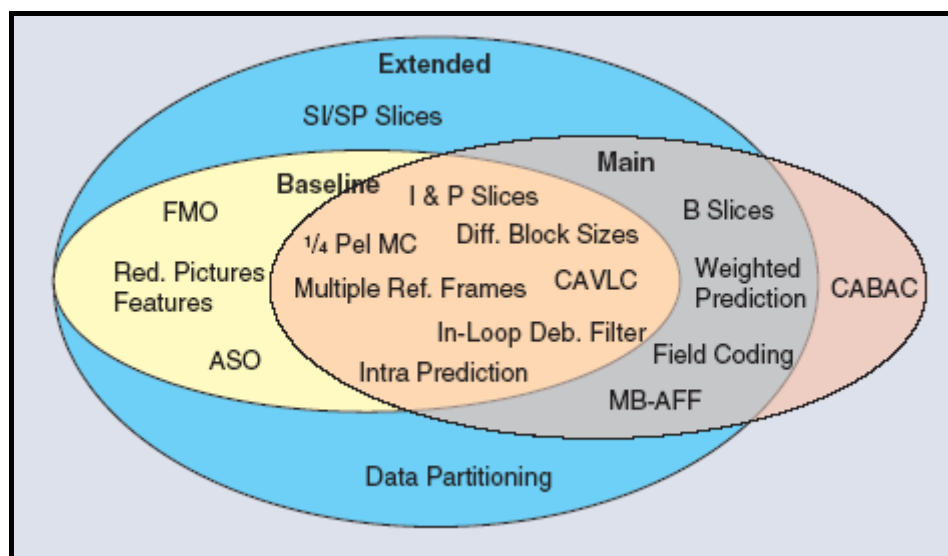


Figure 2: H.264/AVC profiles and features

Three profiles have been defined in order to cover the main application domains:

- a Baseline Profile, which is dedicated to conversational application, like video-telephony and video-conferencing;
- a Main Profile designed for television applications;
- an Extended Profile, more appropriate for streaming and mobile video services.

The Baseline profile supports all features in H.264/AVC except the following two features sets:

- B slices, weighted prediction, CABAC, field encoding and macro-block adaptive frame field coding (MB-AFF);
- SP/SI switching slices and slice data partitioning.

The first set of additional features is supported by the Main profile, without taking in account flexible macro-block ordering (FMO), arbitrary slice ordering (ASO) and redundant pictures.

The Extended profile supports both features of Baseline and Main profiles, except for CABAC.

All decoders conforming to a specific profile must support all features in that profile. Encoders are not required to make use of a particular set of features supported in a profile but have to provide conforming bit-streams. Each profile is itself decomposed into levels describing the complexity and the relevant resources to be used for decoding a bit-stream and displaying it in real-time.

3 Overview of the Scalable Extension of H.264/AVC

While the H.264/AVC standard provides state-of-the-art compression performance for single layer video coding, it does not support any form of scalability (SVC). In parallel, the JVT group is also working on a scalable version of H.264 which would have close similarities with H.264/AVC video coding standard. The current working draft combines the coding primitives of H.264/AVC with an open-loop coding structure allowing merging spatial, temporal and quality scalabilities [2].

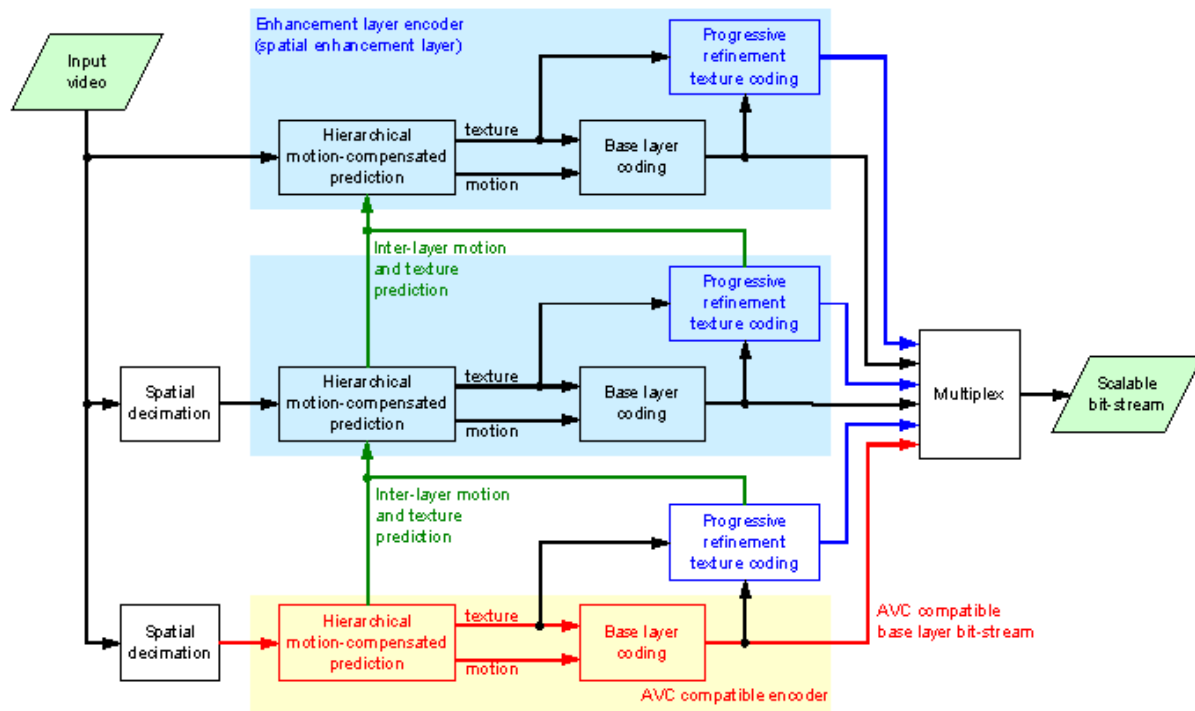


Figure 3: SVC Encoding Architecture

The key features of the scalable extension of H.264 / MPEG-4 AVC are:

- hierarchical prediction structure ;
- layered coding scheme with switchable inter-layer prediction mechanisms ;
- base layer compatibility with H.264 / MPEG-4 AVC ;
- fine granular quality scalability using progressive refinement slices ;
- usage and extension of the NAL unit concept of H.264 / MPEG-4 AVC.

The basic coding scheme for achieving a wide range of spatio-temporal and quality scalability can be classified as layered video codec. The coding structure depends on the scalability space that is required by the application. Just above, a block diagram for a typical scenario with 3 spatial layers is depicted [3].

Spatial scalability is modelled using a Laplacian pyramid to compute the different spatial layers applying successive decimations from the highest resolution down to the lower one [4].

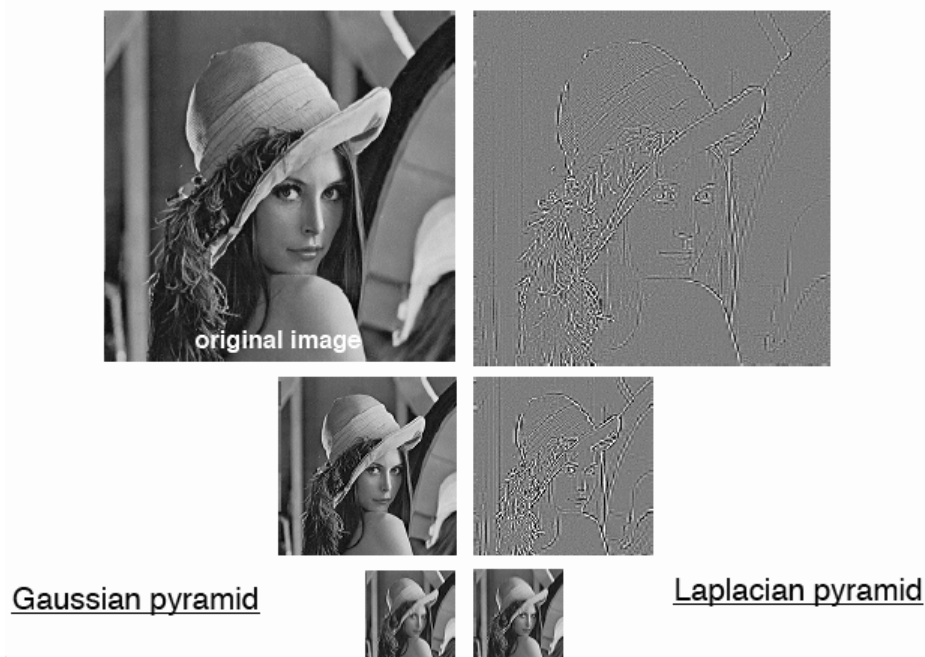


Figure 4: Spatial Decimation

Temporal scalability is implemented through a hierarchical prediction structure that can be realised according two different approaches:

- a simple coding of hierarchical pictures that can be easily implemented using H.264 B pictures (**BH**);
- a more generalised approach using **motion-compensated temporal filtering (MCTF)**.

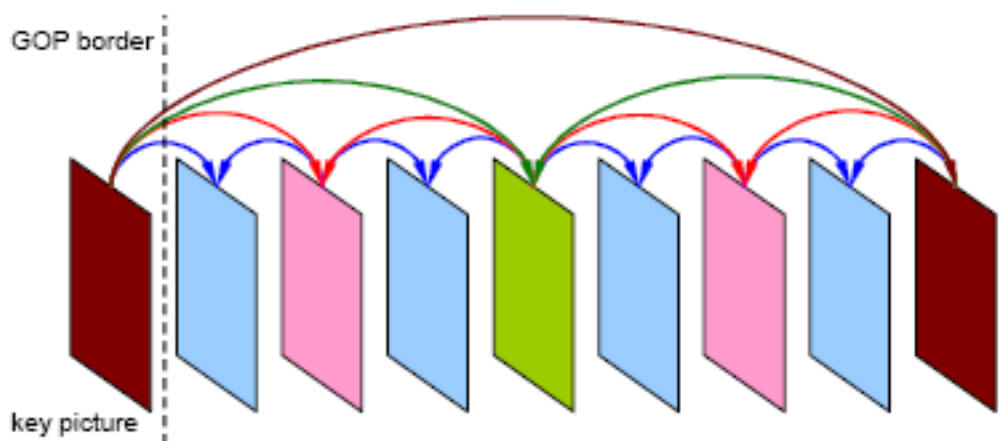


Figure 5: B Hierarchical temporal prediction

In the above figure, an example of the hierarchical prediction structure for a group of 8 pictures with dyadic temporal scalability is depicted. The first picture of a video sequence is intra-coded as a key picture; key pictures are coded in regular (or even irregular) intervals. A key picture and all pictures that are temporally located between the key picture and the previous key picture are considered to build a group of pictures (GOP). The sequence of key pictures is independent from any other pictures of the video sequence, and in general it represents the minimal temporal

resolution that can be decoded. Furthermore, the key pictures can be considered as re-synchronisation points between encoder and decoder. The remaining pictures of a GOP are hierarchically predicted by applying successive predictions between couple of pictures selected from the GOP hierarchical organisation:

- the middle picture is predicted from one or the two key pictures bordering the GOP;
- for each sub-GOP, the same process is applied twice between the middle picture and each key pictures;
- and so on.

The hierarchical picture coding can be extended to motion-compensated filtering by adding an updating step to the motion-prediction one ([5]). The MCTF decomposition process starts at the highest temporal resolution. The group of pictures is partitioned into pictures A and pictures B. The pictures B are predicted using the pictures A and replaced by the motion-compensated prediction residuals. The prediction residuals of the pictures B are then again motion-compensated, but this time towards the pictures A, and the obtained motion-compensated prediction residuals are added to the pictures A, so that the pictures A are replaced by a low-pass version that is effectively obtained by low-pass filtering along the motion trajectories. This process is iteratively applied to the set of low-pass pictures as illustrated in the next figure until a single low-pass picture is obtained as key picture.

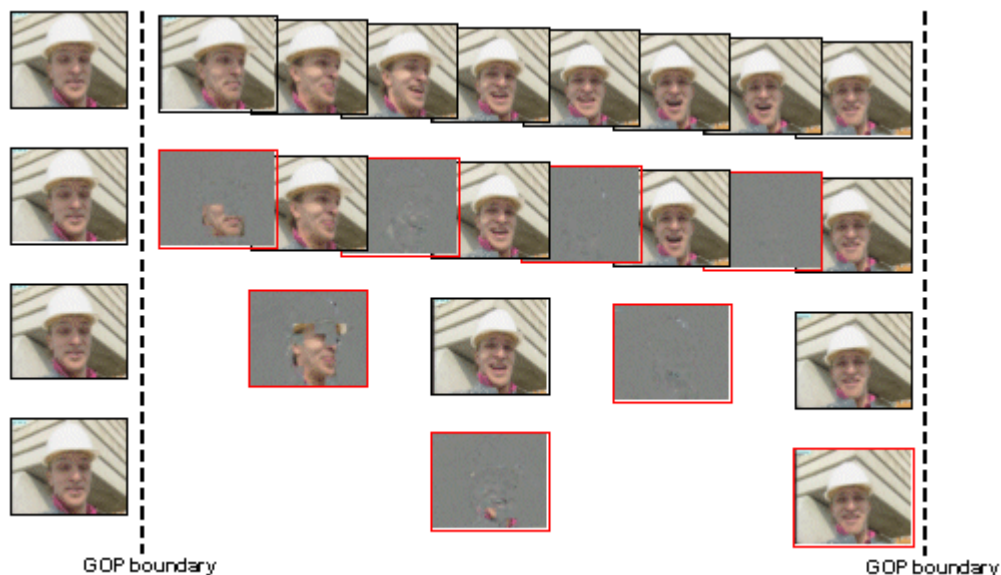


Figure 6: MCTF temporal prediction

As a first interpretation, the pictures for different layers are coded independently with layer-specific motion information. Although, the following techniques turned out to provide gains and were included into the scalable video codec as **Inter-layer Prediction** techniques:

- prediction of intra-macroblocks using up-sampled base layer intra blocks ;
- prediction of motion information using up-sampled base layer motion data ;
- prediction of residual information using up-sampled base layer residual blocks.

Inter-Layer Intra Texture Prediction may be applied according three different manners, using:

- unrestricted prediction, where any block of any lower level is chosen (still not in the draft standard);
- constrained prediction, where prediction can applied only from intra-coded blocks of base layer;

- constrained for single loop decoding, where prediction can be applied only from intra-coded blocks of base layer in any picture of the temporal decomposition as well as the key picture.

Up-sampling techniques are then applied between two successive spatial layers, by using:

- a half-pel interpolation for a ratio-of-2;
- a quarter-pel interpolation for a ratio between 1 and 2.

Inter-Layer Motion Prediction can be applied from a previous layer by using:

- same image reference indices;
- same macroblock partition;
- same motion vectors which are up-sampled;
- a possible quarter-pel refinement.

Inter-Layer Residual Prediction may enhance motion information in conjunction with Inter-Layer Motion Prediction.

Some experiments made in [6] to measure the optimal use of coding modes, lead to the following statistics:

Mode/ MPEG	Intra	Inter	Inter-Lay Texture Pred.	Inter-Lay Motion Pred.	Inter-Lay Residual Pred.
Forman	1%	18%	7%	72%	2%
Football	2%	28%	27%	35%	8%

Table 1: Coding Mode Usage Experiments

The first video sequence presents a single person speaking to a camera like in a news programme, the second one presents two teams playing football like in a sport event: in the last one motion takes more importance than in the former one. It can be seen that an optimal mode selection between two successive spatial layers stands:

- in trying to use inter-layer prediction modes for 70% up to 80% of mode decisions;
- in keeping other single layer coding modes for the remaining.

Inter-layer prediction modes may then usefully apply for encoding inter-layer redundancy. Single-layer coding modes consequently apply for detecting innovation between successive spatial layers.

In this example, inter-layer residual prediction is not greatly used because motion compensation is starting from SNR base layers for avoiding drift effects when a bitstream is truncated.

A picture is generally represented by a non-scalable base representation, which includes all corresponding motion data as well as a “coarse” approximation of the intra and residual data, and zero or more quality scalable enhancement representations, which represent the residual between the original pictures (or prediction residuals and intra blocks) and their reconstructed base representation (or the subordinate enhancement representation). **Quality scalability** can be simply introduced by applying the same prediction techniques between successive quality layers at a same spatial resolution, starting from the minimum quality provided by an AVC compatible texture encoding for the base quality layer and improving it along each following enhancement layers. Staying at the same resolution, coded information does not need then to be up-sampled. This approach provides the possibility to efficiently incorporate **coarse grain SNR (CGS)** scalability.

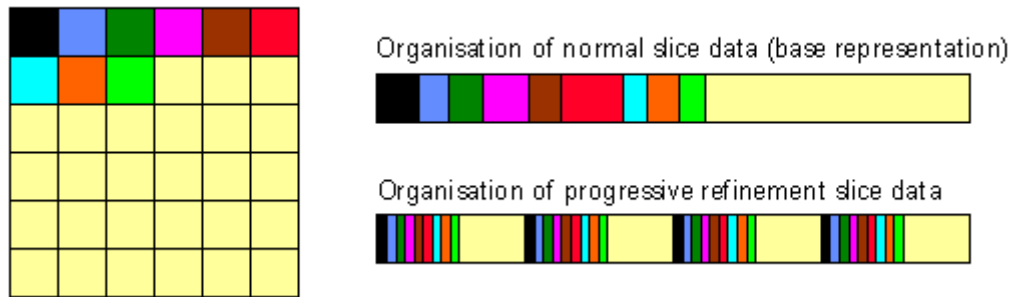


Figure 7: FGS data organization

Progressive refinement slices have been introduced in order to support fine granular quality scalability. Within each spatial resolution **fine grain SNR (FGS)** scalability is achieved by encoding successive refinements of the transform coefficients, starting with the minimum quality provided by AVC compatible intra / residual coding. This is done by repeatedly decreasing the quantization step size and applying a modified CABAC entropy coding process akin to sub-bitplane coding. The base representation can be improved in a fine granular way while the enhancement representation can be truncated at any arbitrary point. The organisation of progressive refinement slices is shown in the figure above.

The coded video data of H.264/AVC is organized into NAL units, each of which is effectively a packet that contains an integer number of bytes. In its scalable extension, an 1-byte extension of the NAL unit header is used to specify the decodability dependency information, as shown underneath, which must be present for insuring a correct decoding.

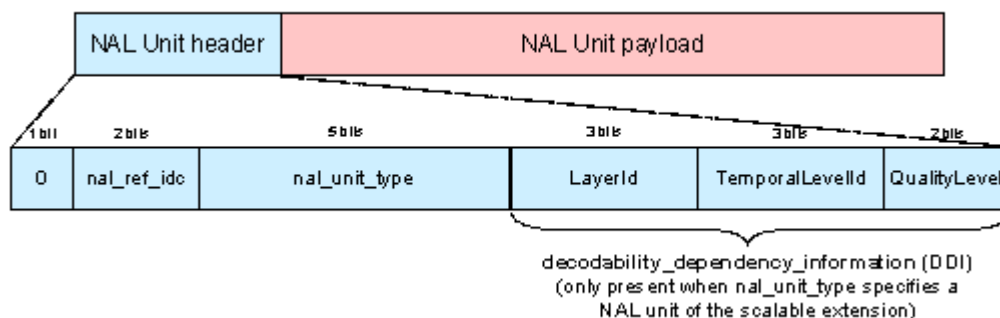


Figure 8: NAL scalable extension

Recently, the SVC extended NAL unit header has been amended in order to indicate a priority order to drop NAL units when bitstream rate must be adapted on the fly [9]. This new interface between video coding layer and network is described underneath.

The Extended Bit (E) is indicating the presence of the 3rd byte in which can be retrieved the decodability dependency information where logically temporal scalability moved in 1st position.

Bit:	0	1	2	3	4	5	6	7
1. Byte	F	NRI		NAL Unit Type				
2. Byte	Simple Priority ID					D	E	
3. Byte	Temporal Level			Dependency ID			Quality Level	

Figure 9: SVC Extended NAL Unit Header

It allows to visit and to truncate a bitstream in an increased flexibility, as shown underneath, by combining the three different kinds of scalability present in the scalable extension of H.264/AVC ([7]).

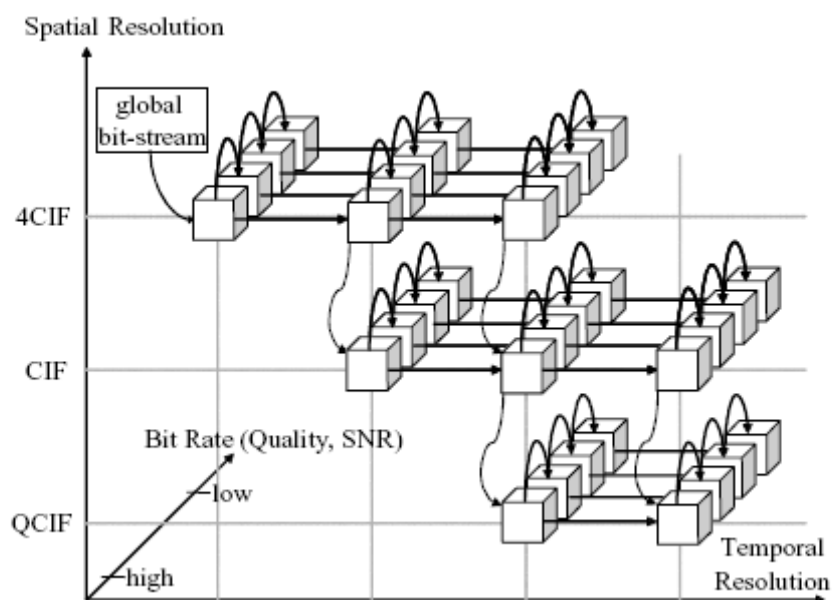


Figure 10: Visiting a Scalable Bitstream

4 Differences between Off-line and Real-Time Encoders

Off-line and real-time encoders are developed in order to meet different objectives:

- off-line encoders are mainly used in the aim to record audiovisual programmes that will be displayed later;
- real-time encoders are used in order to broadcast directly live events.

During off-line encoding, delay is generally not a constraint and is tried to achieve the best quality before storing the encoded content: the full set of available tools offered by the encoding standard can then be used. The result is encoded with a variable bit rate (VBR) before its storage and often trans-rated using a statistical multiplexer at broadcasting time.

At the contrary, delay is a main constraint to satisfy in real-time encoding because the encoded content is simultaneously broadcasted within a channel of a given bandwidth. The result is rather encoded for satisfying a constant bit rate (CBR) delivery and the real-time constraint leads to select the most powerful subset of tools provided by the encoding standard.

All these aspects are summarised in the following table.

Coders	Real-Time	Offline
Audiovisual programs	Broadcasted TV	Recorded TV
Coding toolset	Most efficient subset	Full set
Bit-rate range	Medium	High
Bit-rate regulation	CBR	VBR
Bit-stream quality	Sub-optimal	Optimal

Table 2: Coders characteristics

The SVC draft standard is a document defining the decoding process that an implementation should follow when the standard will be in the Final Draft state. The standard body provides an implementation for encoding and decoding video streams as well: its purpose is to test coding algorithms quality and proves the improvement they provide over previous algorithms. This implementation has not been designed for CPU optimization. Nevertheless it is interesting to show the differences between a CPU efficient implementation of AVC encoding standard and the SVC implementation, which is mainly based on AVC.

The encoded video content has a duration of 6 seconds which will be the upper bound for satisfying to the real-time constraint. The comparison has been set up by using the free software releases *x264* for AVC encoding [10] and *ffmpeg* from *ffmpeg* [11] as decoder and renderer.

Coding context	(SVC) JSVM	(AVC) x264
Single spatial layer 176x144 (QCIF)	20 s	1.1 s
Single spatial layer 352x288 (CIF)	85 s	3.9 s
Single spatial layer 704x576 (4CIF)	366 s	13.0 s
3 spatial layers from QCIF up to 4CIF	958 s	N/A
Single spatial layer 4CIF with 3 temporal ones	1596 s	10.5 s
Single spatial layer 4CIF with 3 quality (FGS) ones	482 s	N/A

Table 3: Coding Performance Comparison

Decoding context	(SVC) JSVM	(AVC) ffmpeg
Single spatial layer 176x144 (QCIF)	1.3 s	0.25 s
Single spatial layer 352x288 (CIF)	3.8 s	0.67 s
Single spatial layer 704x576 (4CIF)	13.0 s	1.97 s
3 spatial layers from QCIF up to 4CIF	37.0 s	N/A
Single spatial layer 4CIF with 3 temporal ones	17.0 s	N/A
Single spatial layer 4CIF with 3 quality (FGS) ones	65.0 s	N/A

Table 4: Decoding Performance Comparison

HD formats to decode	(SVC) JSVM	(AVC) ffmpeg
720p25	26.0 s	3.6 s
720p50	52.0 s	7.2 s
1080i25	47.0 s	6.2 s

Table 5: High Definition Decoding Performances

The previous tables show that the JSVM cannot be used as is for encoding as well as decoding bigger format than CIF.

5 Real-Time Coder Development Techniques

The real-time implementation of a video coder is asking a lot of efforts to a research and development team:

- making decision techniques proposed by software reference models cannot be used as it and must be redesigned for real-time application;
- among all coding tools proposed by a reference model, a compromise must be found between their own efficiency and their impact on software complexity ;
- computation-demanding loops must be revisited and optimized according to the architecture of the processor on which the software is ported.

Usually the development of a real-time video coder follows the flowchart shown below.

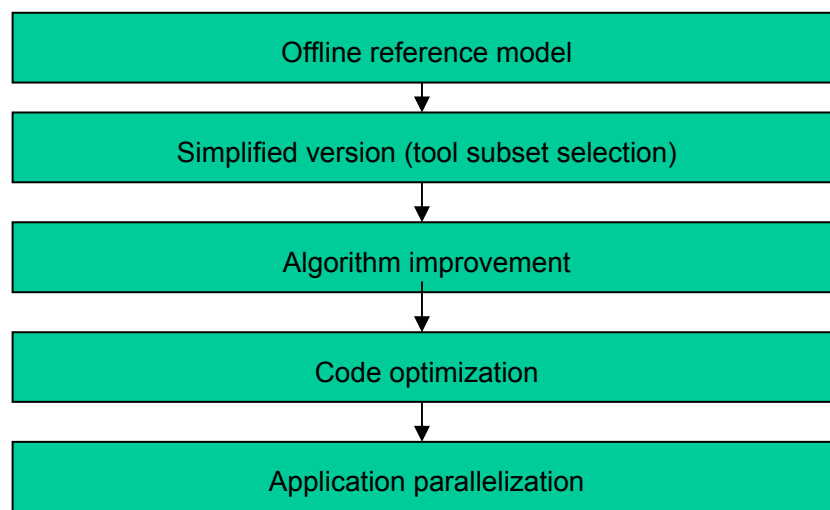


Figure 11: R-T Coder Development Flow-Chart

Considering the architecture of video encoders, most of real-time SDTV H.264/AVC implementation for live broadcast applications have been developed using a multi-DSP architecture, because it allows:

- to get a higher throughput than with a conventional general purpose processor;
- to provide a wide range of computing performances due to processor parallelism and embedded signal processing capabilities;
- to offer rather flexible development platforms before starting a silicon implementation on a FPGA or an ASIC.

Development step	Processor	Language	Market
Reference model -> Algo. improv.	GPP	C/C++	Professional
Code optimization	GPP+ co-proc	C/ASM	
Application parallelization	Multi-GPP/DSP	C/ASM	
Code porting	DSPs/FPGAs	C/ASM/VHDL	
Code porting	ASIC	VHDL	Consumer

Table 6: R-T Coder Platform Architecture

VITEC Multimedia has developed SDTV H.264/AVC real-time encoding components based on multiprocessor electronic boards (for instance, cf. VMC-5400 data sheet in Annex) and is currently building a new one that will afford sufficient computing power to encode HDTV formats. It is planned to use such an environment to implement SVC real-time encoding facilities, but by using most of developments already available for H.264/AVC encoding and adapting them to provide an SVC implementation.

6 Real-Time Implementation of an SVC Coder

6.1 Real-Time Implementation of a Scalable Video Coder

In this spirit, VITEC Multimedia proposes to set up a first encoding solution made from a stack of AVC coders modified according to the following way:

- at least one AVC encoder is used per spatial layer;
- only intra-bloc prediction mode is used in spatial inter-layer prediction (frames are fully reconstructed before spatial up-sampling);
- B-Hierarchical prediction is used to implement temporal scalability;
- FGS is implemented to enable SNR scalability;
- AVC syntax is adapted for taking in account scalability information into an SVC compliant bitstream.

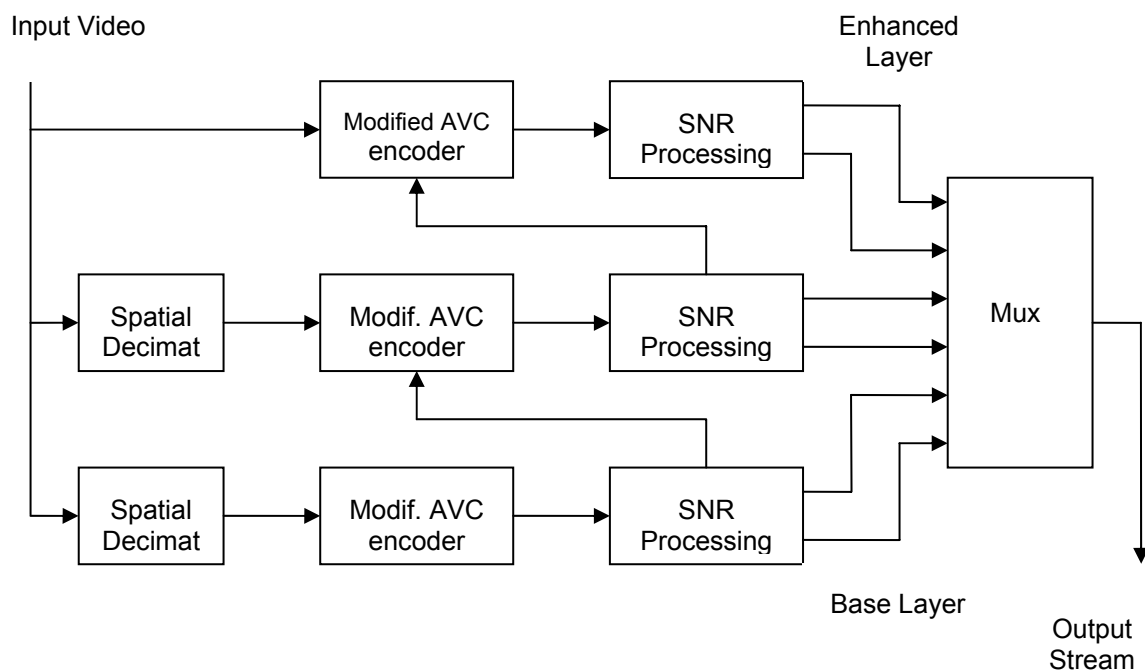


Figure 12: Real-Time SVC Implementation

A real-time decoder will be provided for displaying encoded streams into a PC environment.

One must be aware that real-time H.264/AVC encoders and decoders currently can hardly support TV formats above SDTV without the help of customized chips. Concerning SVC streams, SDTV format may bring an upper limit for live feeds at project timeline.

In a second step, new encoding features will be added to real-time encoding and decoding tools, as for instance FGS scalability. Concerning the FGS tool, another issue is the current state of the reference software. Indeed, there exists many implementations of FGS-like tools, and no decision have been made about which scheme will be kept.

6.2 SUIT SVC Reference Platform

All R-T H.264/AVC coding implementations nearly follow the rules and lead to the same kind of algorithm architecture. They are mainly relying on the study made by the IMEC which has tried to measure the impact of every tool provided by the standard on the encoded result [12]. It enables to select the tools that must be kept in a simplified release of the reference model without losing too much performance.

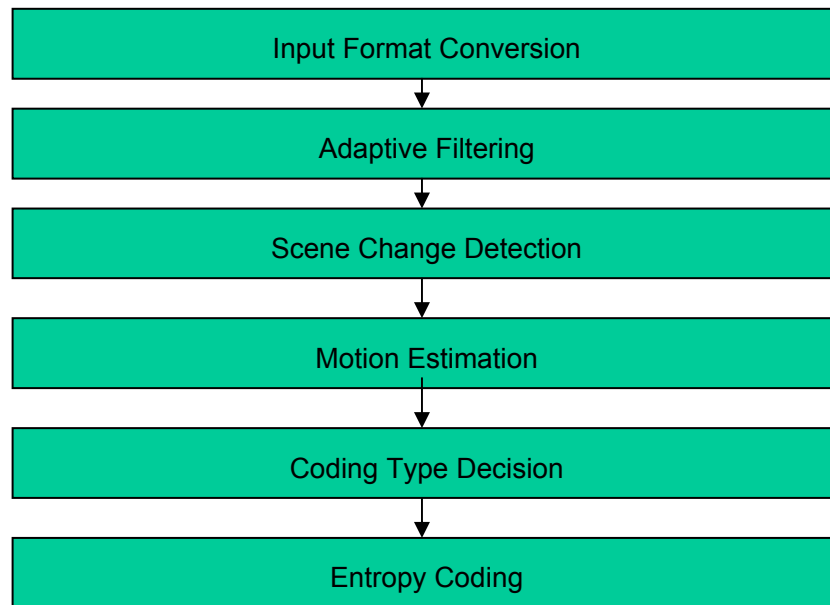


Figure 13: R-T Coder Architecture

Usually, an R-T video coder architecture is composed of the following steps:

- a pre-processing phase including a input format conversion where video frames are rescaled and where the colour reference space of input signal is transformed, adaptive noise filtering and scene change detection;
- a processing phase where motion estimation is done first regardless to mode coding and coding mode selection is applied in a two-pass way including a look-ahead pass in the objective to measure local complexity of the input to be coded rather an R-D optimisation loop [14], and then mode coding as it is the case in off-line implementations ;
- a post-processing phase where entropy coding is performed and the bitstream is produced.

According to the coding platform architecture at one's disposal:

- it is tried to implement pre-processing and post-processing tasks in FPGA when ever it is possible;
- motion estimation is done apart from mode coding and it is tried to use sufficiently regular algorithms as possible in order to afford an FPGA implementation;
- but motion estimation can also be spread between scene detection and inter prediction mode coding, in this case only motion vector refinement is applied for inter prediction coding.

When a multiprocessor architecture is used to achieve real-time performances, two paths can be followed:

- a coarse granularity organisation driven by a task allocation organisation, input streams are then encoded on a frame basis;
- a finer granularity organisation driven by incoming data, data is partitioned into video slices.

The first approach allows to allocate tasks only on a small number of processors. The second one allow to deal with more powerful architectures, but window search size must be shortened during motion estimation to avoid to lower data moves between processors and memories at the boundary of slices.

So in the objective to insure that SUIT developments could have a high chance to be included in SUIT demonstrations, Vitec Multimedia has proposed to afford SUIT partners with an intermediate version between the full JSVM and an R-T release of the next SVC encoding standard. The SUIT SVC Reference encoding and decoding Platform should run on a conventional PC platform and will stay as close as possible from existing R-T AVC implementations and the R-T SVC implementation will derive from them by introducing the SVC extensions designed by JVT to the AVC standard.

As B-H temporal scalability is already existing in the AVC standard, we will propose to the reader to examine now how to implement progressively spatial and quality scalability starting from an AVC platform. It will be first studied how inter-layer prediction modes can be added to conventional intra and inter prediction modes in a single layer coder and then it will be described what are the new syntactic elements to be inserted in an AVC bitstream to produce an SVC compliant one.

6.3 Real-time Algorithm for Spatial Scalability

The SUIT project proposes to process progressive video streams and scalable resolutions with a power of 2. In this aim, it has been chosen to deal with the following scalable formats that may approach a 16:9 aspect ratio:

- CIF with a picture resolution 320 pixels by 176 rows at 25Hz and a bitrate of 0.5 Mbps;
- SD with a 640x352 resolution still at 25Hz and at 1.5 Mbps;
- HD with a 1280x704 resolution at 25Hz (4.25Mbps) and 50Hz (8Mbps).

According to table1, it has been planned to implement successively the texture and then the motion inter-layer prediction modes in complement to existing inter and intra AVC prediction modes.

The implementation of the first inter-layer prediction allows to validate the AVC stack encoding (and decoding) architecture proposed at the beginning of the chapter. Using texture inter-layer prediction, AVC encoders will mainly be used to encode the innovation found in the up-sampled scale, inter-layer redundancy being managed by texture inter-layer prediction.

Such a processing could be implemented using a stack of AVC encoders and a set of a few complementary operators for satisfying real-time constraints:

- +/+, realising the addition of an up-sampled signal and the reconstructed innovation coded by an encoder from the stack;
- +/-, realising the subtraction of a decimated input signal and the up-sampled signal enriched with the encoded layer innovation;
- /2, spatial decimation of input signal;
- x2, up-sampling of reconstructed signal.

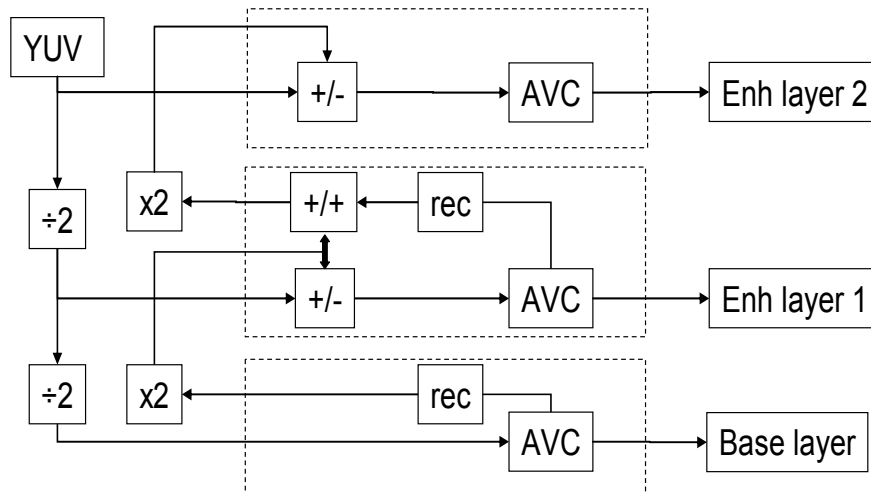


Figure 14: Spatially scalable encoder

The corresponding decoding architecture can be also defined using the same set of operators.

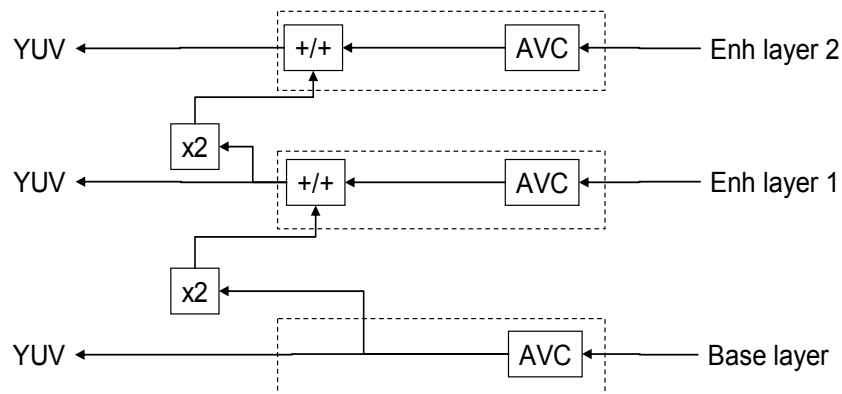


Figure 15: Spatially scalable decoder

6.4 Real-Time Algorithm for SNR Scalability

In the SVC draft standard, two schemes of SNR scalability are proposed: CGS and FGS (FGS has many flavours as well).

The following scheme is an FGS scheme with layers equivalent to the Progressive Refinement of the SVC draft standard. This scheme has been designed to follow the FGS requirements as well as real-time implementability. Indeed, each refinement involves only a quantization /de-quantization /difference (which can be done all in the same function for CPU optimization), and an entropy encoder.

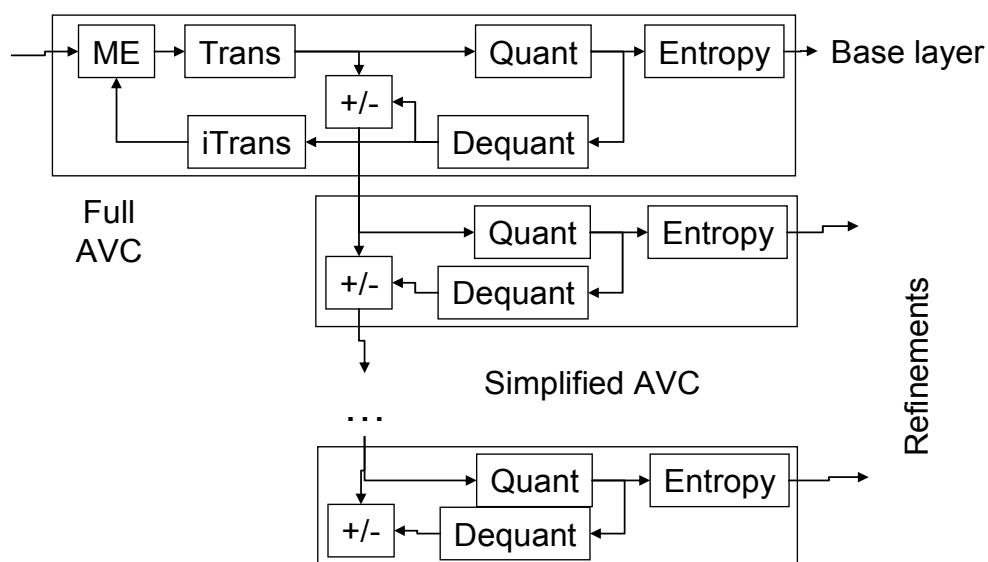


Figure 16: SNR scalable encoder

Moreover, if few bits are generated, the CPU usage of the added entropy coding should be kept low. The scheme for decoding is simpler than the encoding, though somehow equivalent. The added entropy decoder should as well have a low CPU usage for the same reason as in the encoder.

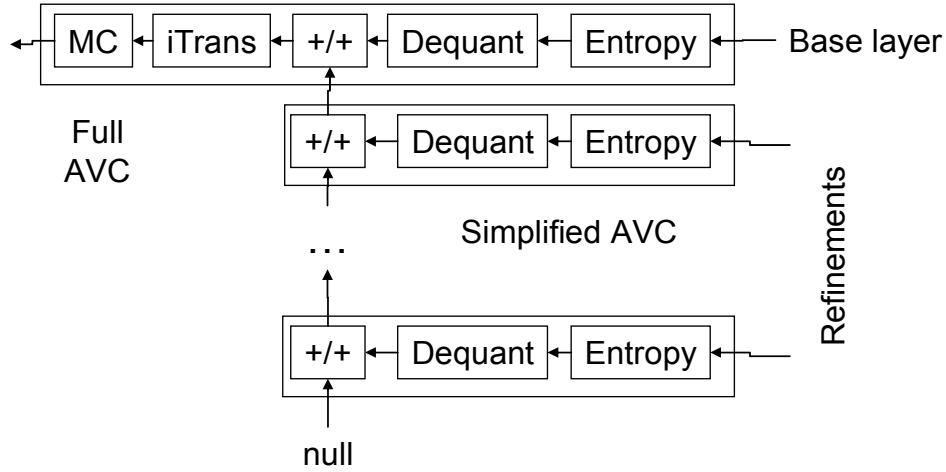


Figure 17: SNR scalable decoder

Quantification relies on the following set of equations for providing the refinement layers:

$$r = X - P_{H.264}(X)$$

$$c = T_{H.264}(r)$$

$$\varepsilon = c - Q_{H.264}^{-1}[Q_{H.264}(c)]$$

Where:

- X is a 4x4 pixel block,
- r is the residual, difference between the pixels and the prediction (either intra or inter prediction),
- c is the coefficients of the residual,
- $Q_{H.264}$ is the base layer quantization,
- ε is the SNR coefficients block.

The ε block is then separated in SNR_i blocks. ε_i blocks are intermediate blocks.

The parameter of this separation is q , specific to the SNR coding scheme. q shall not be less than i , for the i^{th} layer to exist.

So:

$$SNR_0 = \left\lfloor \frac{\varepsilon}{2^q} \right\rfloor, \varepsilon_0 = \varepsilon - SNR_0 \times 2^q,$$

$$SNR_1 = \left\lfloor \frac{\varepsilon_0}{2^{q-1}} \right\rfloor, \varepsilon_1 = \varepsilon_0 - SNR_1 \times 2^{q-1},$$

$$SNR_i = \left\lfloor \frac{\varepsilon_{i-1}}{2^{q-i}} \right\rfloor = \frac{\varepsilon - \sum_{j=0}^{i-1} SNR_j \times 2^{q-j}}{2^{q-i}}$$

This decomposition can be seen as a bit-plan extraction, SNR_i being the plans.

6.5 Bitstream Specification

6.5.1 SUIT SVC bitstream structure

The bitstream structure is organised around access units (cf. [15]).

It starts with a Sequence Parameter Set (SPS) NAL unit in which main information for the decoder are specified and optionally a Video Usability Information (VUI) field can be append to define more precisely sequence parameters as the aspect ration, the video format (PAL, NTSC, ...), colour information, timing information and Hypothetic Reference Decoder (HRD) parameters.

It carries on with the Picture Parameter Set (PPS) in which are specified encoding information as the picture decomposition in slices, the initial QP parameter, the weighted prediction method for B slices, the use of the de-blocking filter, the entropy coding method and the presence of redundant slices.

This information can be refreshed at a chosen time period and can be repeated before each Access Unit. DVB advises to start a GOP with an IDR picture and to send an SPS NALU before each GOP and a PPS NALU before each picture for enabling programme zapping inside a multiplex of channels.

Before an Access Unit, complementary information can be set using a Supplemental Enhancement Information (SEI) NALU in which can be mainly found picture time stamps.

Then an Access Unit will stand as an I or B picture defined at a given time inside a GOP. It will be decomposed into a Base coded picture with all its SNR progressive refinements at the coarser space scale and its series of spatial enhancement including their own SNR refinements, as shown in the following diagram.

A decoder is expecting to have a full Access Unit that is composed of all the NALUs having the same timestamp or the same frame number. An Access Unit consists of:

- a base coded picture;
- all its SNR refinement layers;
- a spatial enhancement layer;
- all its SNR refinement layers;
- and so on.

The encoder outputs by following this incremental order without skipping any frame. To know which NALUs have the same timestamp, the following method can be used:

- the beginning of an AU will stand with the arrival of base coded picture which *nal_unit_type* value will be below or equal to 5;

- the end will be detected with the arrival of a new base coded picture or the end of the stream.

The *nal_unit_type* value of SNR refinement layers and spatial enhancement is 20.

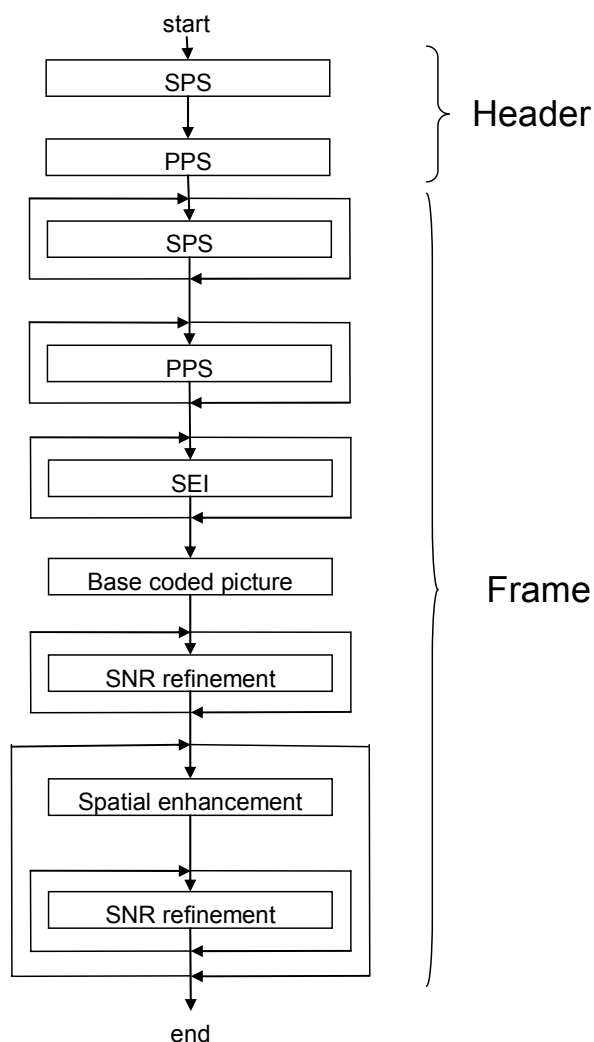


Figure 18: SUIT SVC bitstream structure

6.5.2 NAL Header and NAL Unit

The standard H.264 NAL header structure is shown in the table below. The following one describes the extended structure provided for NAL unit 20 and 21 added for managing scalable VCL information in the emerging SVC standard. It corresponds to the last available recommendations made in a JVT meeting [18].

nal_unit(NumBytesInNALunit) {	C	Descriptor
forbidden_zero_bit	All	f(1)
nal_ref_idc	All	u(2)
nal_unit_type	All	u(5)
NumBytesInRBSP = 0		
for(i = 1; i < NumBytesInNALunit; i++) {		
if(i + 2 < NumBytesInNALunit && next_bits(24) == 0x000003) {		
Rbsp_byte[NumBytesInRBSP++]	All	b(8)
Rbsp_byte[NumBytesInRBSP++]	All	b(8)
i += 2		
emulation_prevention_three_byte /* equal to 0x03 */	All	f(8)
} else		
Rbsp_byte[NumBytesInRBSP++]	All	b(8)
}		
}		

Table 7: Standard AVC NAL header

nal_unit(NumBytesInNALunit) {	C	Descriptor
forbidden_zero_bit	All	f(1)
nal_ref_idc	All	u(2)
nal_unit_type	All	u(5)
nalUnitHeaderBytes = 1		
if(nal_unit_type == 20 nal_unit_type == 21) {		
nal_unit_header_svc_extension()		
}		
NumBytesInRBSP = 0		
for(i = nalUnitHeaderBytes; i < NumBytesInNALunit; i++) {		
if(i + 2 < NumBytesInNALunit && next_bits(24) == 0x000003) {		
rbsp_byte[NumBytesInRBSP++]	All	b(8)
rbsp_byte[NumBytesInRBSP++]	All	b(8)
i += 2		
emulation_prevention_three_byte /* equal to 0x03 */	All	f(8)
} else		
rbsp_byte[NumBytesInRBSP++]	All	b(8)
}		
}		

Table 8: Future SVC NAL header

nal_unit_header_svc_extension() {	C	Descriptor
simple_priority_id	All	u(6)
discardable_flag	All	u(1)
reserved_zero_bit	All	u(1)
temporal_level	All	u(3)
dependency_id	All	u(3)
quality_level	All	u(2)
nalUnitHeaderBytes += 2		
}		

Table 9: SVC Extension of NAL header

6.5.3 Encoding coefficients in SNR refinement layers

The progressive refinements layers are encoded using the following scheme described in the two successive tables: one for parsing coefficients in a macroblock and the other one applying entropy coding over them. SNR coefficients are CABAC coded using I_Slice context type with a qstep set to 1 and the model 0.

residual_macroblock_SNR_cabac(coeffLevel, maxNumCoeff) {	C	Descriptor
coded_mb_flag	3 4	ae(v)
if(coded_mb_flag) {		
for(i=0;i<16;i++) {		
residual_block_SNR_cabac()		
}		
}		
}		

Table 10: Coding SNR residual macroblocks

residual_block_SNR_cabac(coeffLevel, maxNumCoeff) {	C	Descriptor
coded_block_flag	3 4	ae(v)
if(coded_block_flag) {		
numCoeff = maxNumCoeff		
i = 0		
do {		
significant_coeff_flag[i]	3 4	ae(v)
if(significant_coeff_flag[i]) {		
last_significant_coeff_flag[i]	3 4	ae(v)
if(last_significant_coeff_flag[i]) {		
numCoeff = i + 1		
for(j = numCoeff; j < maxNumCoeff; j++)		
coeffLevel[j] = 0		
}		
}		
i++		
} while(i < numCoeff-1)		
coeff_abs_level_minus1[numCoeff-1]	3 4	ae(v)
coeff_sign_flag[numCoeff-1]	3 4	ae(v)
coeffLevel[numCoeff-1] =		
(coeff_abs_level_minus1[numCoeff - 1] + 1) *		
(1 - 2 * coeff_sign_flag[numCoeff - 1])		
for(i = numCoeff-2; i >= 0; i--) {		
if(significant_coeff_flag[i]) {		
coeff_abs_level_minus1[i]	3 4	ae(v)
coeff_sign_flag[i]	3 4	ae(v)
coeffLevel[i] = (coeff_abs_level_minus1[i] + 1) *		
(1 - 2 * coeff_sign_flag[i])		
} else		
coeffLevel[i] = 0		
}		
} else		
for(i = 0; i < maxNumCoeff; i++)		
coeffLevel[i] = 0		
}		

Table 11: Entropy coding of SNR residual coefficients

6.6 Bitstream Extraction

Using the scalability information that stands in SVC extended NAL headers, it is possible to truncate the bistream in order to satisfy to current bandwidth constraints. But it is uneasy to know the impact on the reconstructed video stream and it may be then difficult to multiplex several different streams into only one respecting a given bitrate.

In order to solve this issue, it will be provided information about the quality of information that stands in every spatial and SNR layers of a coded bitstream. According to the encoding algorithm used for base layers, PSNR information can be easily retrieved from residual computation stage. At the contrary, when progressive refinement layers are produced, PSNR should be computed in addition, but it becomes a very computation intensive task. To avoid this drawback, we have

chosen to build an estimate of this measure. In [19], an innovative quality estimator is proposed. The algorithm is based on the counting of the zeros that are generated by a dct+quantization scheme. This scheme has been proven particularly efficient in the context of real-time rate-control algorithm. We propose to implement it to provide quality information for the base layer as well as the scalability layers.

The following graph has been extracted from the original paper. On the right side, the ρ -domain and the distortion are correlated. It has been computed on the MPEG “Foreman” sequence using QCIF video format.

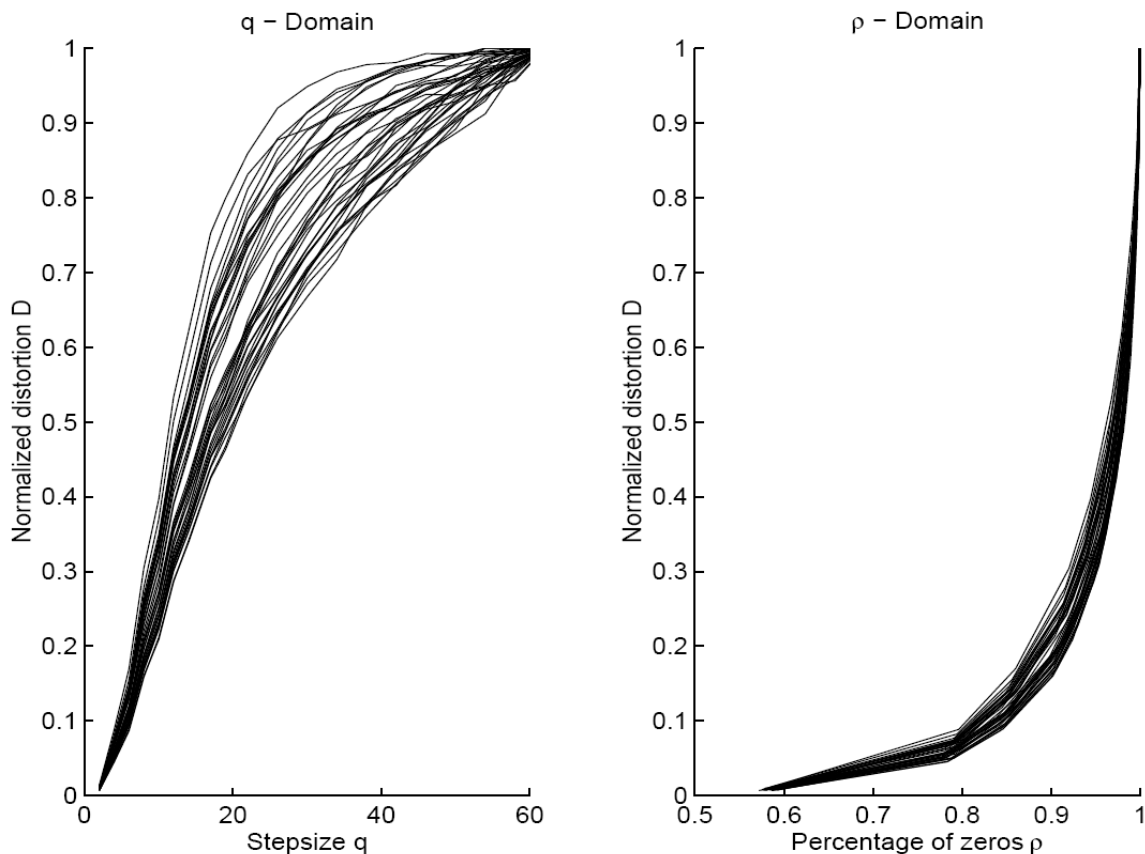


Figure 19: Distortion curves a video sequence displayed in q -domain (left) and ρ -domain (right)

To give as much information as used in an internal rate-control, we propose to give the quantization step and the ρ -domain value of the frame coded. This will enable the user to select among the layers, according to the quality, as the stream size only is not sufficient, to fit to the network constraints. This process could be applied to provide a post-encoding rate control.

After each spatial layer (base coded picture or spatial enhancement with their respective SNR refinements) will appear an SEI of *payloadType* 25 that includes quality information concerning the current spatial layer.

6.7 File format for SUIT SVC elementary stream

We propose a simple file format specification to handle scalable video streams generated for the SUIT project. As SVC standard is developed inside the AVC standard (Annex G), it borrows the same Network Abstraction Layer as the MPEG-4-part 10 standard. This specification does not provide muxing of Access Unit which has to be defined by transport layers. These specifications are not stated in the working draft, so we propose to implement a SUIT project format file that will help to the RTP encapsulation of SVC bitstreams. In addition, this solution is usually adopted by H.264 vendors to simplify the handling of supplemental information and Access Unit streaming.

SUIT file format		
	Size	Value
SUIT File header	4 bytes	« SUIT »
Header Size	4 bytes	header_size
Header	header_size bytes	
While() {		
FrameNum	4 bytes	
SliceNum	4 bytes	
Timestamp	4 bytes	
Scalability.spatial	1 byte	0=base, 1= 1rst layer, ...
Scalability.SNR	1 byte	0=base, 1=1rst layer, ...
Quality.quantstep	1 byte	Quantization step (0 to 51)
Quality.pdomain	1 byte	% of zero coefficients
Access Unit Size	4 bytes	au_size
Access Unit	au_size bytes	
}		

Table 12: SUIT SVC file format

7 Multi-DSP Encoder Specifications

7.1 VITEC Multi-DSP platform family overview

Since mid-2003, VITEC Multimedia has started the development of a DSP-based family of video multiprocessor platforms. They are designed to afford OEM customers with advanced means for developing real-time video applications based on MPEG technology (MPEG-1/2/4 up to SD/HD levels). VITEC Multimedia is trying to set up a software component library and provide an integrated development environment with these platforms. It will help designers to prototype quickly MPEG-compliant audio and video applications. VITEC provides the following services to OEM customers:

- SDK and source code for building OS drivers and DSP sample codes;
- Technical support with purchase of development kit;
- Production facilities to deliver boards in volume at reasonable cost;
- Board customisation and FPGA implementation.

The current platforms cover a large range of professional and industrial video application:

- professional MPEG encoders for Broadcast and Telco;
- multi-channel video streaming over IP;
- multi-channel video surveillance;
- digital camera over IP.

VITEC Multimedia can currently afford developers coprocessor or stand-alone electronic boards enabling to encode in real-time MPEG-4 AVC SDTV or MPEG-2 HDTV streams with its VP3 product family. For instance, it could be found in Annex the data sheets of two different electronic boards:

- VMC 5400, which is an additional board that can be installed in a PC in order to provide real-time performance for encoding SDTV video streams according to MPEG-4 AVC standard;
- VP3 platform, which is a stand-alone solution for encoding HDTV video streams according to MPEG-2.

A more powerful platform is under development in VITEC Multimedia. It should afford sufficient computing power to encode in real-time HDTV video streams up to 1080p according to MPEG-4 AVC. It is expecting to use the new platform at project end in order reach real-time performance to encode 720p video streams according to future MPEG-4 SVC. SUIT real-time SVC encoder will get back the same specifications.

7.2 SUIT Real-Time SVC Encoder Specifications

The SUIT real-time SVC encoder will be built using a new VP3-70 extremely powerful platform made from a mother board connecting up to 15 processing clusters that can work in parallel. Each cluster is VP3-PMC board implementing 5 x DSP TMS320DM642-720 MHz from Texas Instruments providing 28.8 GIPs, leading to a maximum of 115.2 GOPs per processor.

Each DSP has a private local memory of 64MB with a throughput of 1328 MB/s and DSPs can communicate information to each others in different ways through the DMA controller services.

It also includes a number of user dedicated FPGAs allowing to implement any kind of specific processing including video conversion, de-interlace, scaling, adaptive filtering, video complexity analysis and entropy coding.



Figure 20: VP3-70 System

VP3-70 is a complete system integrated in a 19" rack. It supports 2 HD-SDI inputs and 2 DVB-ASI outputs as well as 4 AES/EBU digital audio inputs. In addition a 1 GBit/s Ethernet interface is provided for data output.

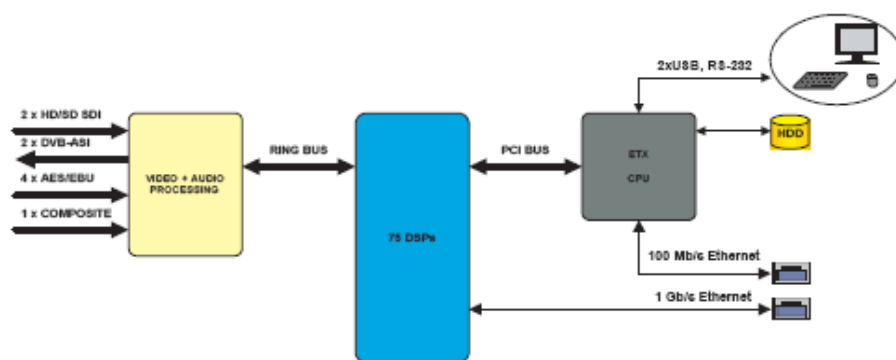


Figure 21: VP3-70 Overview

An embedded CPU controls the overall platform and provides a set of interfaces including an IDE interface for hard disk, 2xUSB, 1xRS-232 and a 100 MBit/s Ethernet for remote control.

The signal processing core includes up to 75 DSPs in 15 clusters of 5 DSPs each.

TECHNICAL SPECIFICATIONS

VP³-70™ offers :

Inputs / outputs	Video Inputs	- 2 HDSDI input through 2 BNC - 1 composite input through DB15 connector
	Video Outputs	- 2 DVB-ASI outputs in dual encoder configuration - 1 DVB-ASI output and 1 HDSDI loop output in single encoder configuration. - 1 composite output through DB15 connector
	Audio Inputs	- 4 AES/EBU through DB15 connector - Audio de-embedding from HDSDI-SDI

Other Inter- faces	- 100Mbps Ethernet through RJ45 connector
	- 1Gbps Ethernet through a second RJ45 connector
	- RS232 communication port
	- I2C interface
	- Compact flash interface
	- 2 USB
	- VGA interface
	- 14-pin JTAG for external emulation hardware support. This is used to connect VP ³ -70 to the TI emulator software.

Rear panel :

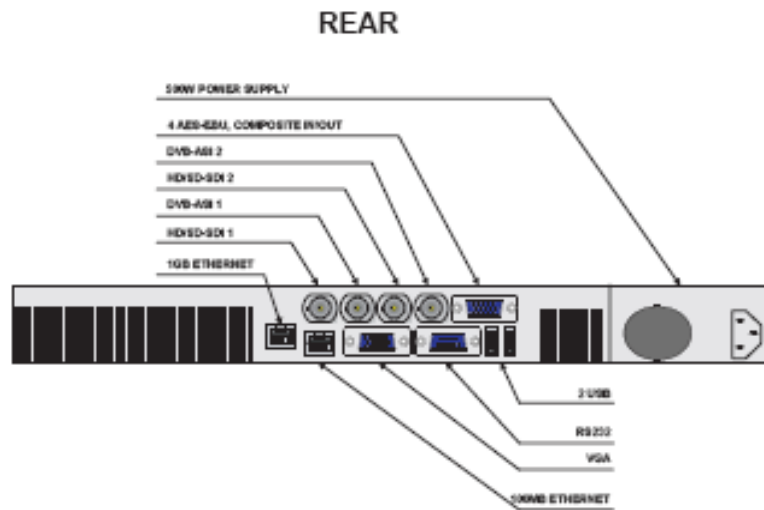


Figure 22: VP3-70 Technical Specifications

8 Conclusions

This document presents the future SVC video coding standard starting from the former one AVC from which it is derived and carrying on with its scalable extension SVC that can afford the three different types of scalabilities: spatial, temporal and SNR scalabilities.

To describe how real-time implementations of video coders are done, first a comparison has been made between off-line and real-time encoders and then real-time development techniques are presented.

Next, the specifications of a real-time implementation dealing up with HD video formats are listed, describing the tools that will be used to implement the different kinds of scalabilities, the syntax of the out coming bitstream and the way to edit it.

For waiting a real-time release, it is provided a SUIT SVC reference platform that should help SUIT developers to start the implementation of their own tools or to start the encoder integration in SUIT demonstrators.

Concerning Multiple Descriptions, it would allow implementing several different approaches like:

1. Unbalanced MD
2. MD based on redundant slices,
3. MD based on EMDSQ.

Priority information inserted in the bitstream will enable the SUIT gateway to directly adapt video content to the available channel bandwidth and the terminal ability. Quality information will enable to perform statistical multiplexing at the SUIT playout side. Furthermore, auxiliary information is provided with NAL units to ease the RTP encapsulation of scalable bitstreams.

9 Acronyms

ASIC	Application Specific Integrated Circuit
ASO	Arbitrary Slice Ordering
AU	Access Unit
AVC	Advanced Video Coder
BH	Bidirectional interpolated Hierarchical pictures
CABAC	Context-Adaptive Binary Arithmetic Coding
CAVLC	Context-Adaptive Variable Length Coding
CBR	Constant Bit Rate
CGS	Coarse Grain SNR scalability
CPU	Central Processing Unit
DCT	Discrete Cosine Transform
DSP	Digital Signal Processor
DVB	Digital Video Broadcast
FGS	Fine Grain SNR scalability
FMO	Flexible Macro-block Ordering
FPGA	Flexible Programmable Gate Array
GOP	Group Of Pictures
GPP	General Purpose Processor
HRD	Hypothetical Reference Decoder
IDR	Instantaneous Decoder Refresh
JVT	Joint Video Team
MB	Macro-Block
MB-AFF	Macro-Block Adaptive Frame Field coding
MCTF	Motion-Compensated Temporal Filtering
MDC	Multiple Description Coding
MMS	Multimedia Messaging Service
MP4	MPEG-4 multimedia container format
MPEG	Moving Picture Expert Group
NAL	Network Abstraction Layer
NALU	Network Abstraction Layer Unit
PPS	Picture Parameter Set
RTP	Real-time Transport Protocol
SEI	Supplemental Enhancement Information
SNR	Signal-to-Noise Ratio

SPS	Sequence Parameter Set
SVC	Scalable Video Coding
VBR	Variable Bit Rate
VCEG	Video Coding Experts Group
VCL	Video Coding Layer
VHDL	Very High-level Design Language
VUI	Video Usability Information

10 References

- [1] R.Schäfer, T.Wiegand and H. Schwarz, The emerging H.264/AVC standard – EBU Technical Review – January 2003
- [2] H. Schwarz, D. Marpe and T. Wiegand, Overview of the Scalable H.264/MPEG4AVC Extension – ICIP, Atlanta, GA, USA, October 2006
- [3] H. Schwarz, D. Marpe and T. Wiegand, Basic Concepts for Supporting Spatial and SNR Scalability in the Scalable H.264/AVC Extension - Proc. IWSSIP 2005, Chalkida, Greece, September 22-24, 2005
- [4] P.J. Burt and E.H. Adelson, The Laplacian Pyramid as a Compact Image Code - IEEE Trans. Communications, 31(4):532--540, Apr. 1983
- [5] H. Schwarz, D. Marpe and T. Wiegand, MCTF and Scalability Extension of H.264/AVC – Proc. Picture Coding Symposium – San Francisco, USA, December 2004
- [6] H. Schwarz, T. Hinz, D. Marpe and T. Wiegand, Constrained Inter-Layer Prediction for Single-Loop Decoding in Spatial Scalability – Proc. ICIP, Genova, Italy, September 2005
- [7] H. Schwarz, D. Marpe, T. Schierl and T. Wiegand, Combined Scalability Support for the Scalable Extension of H.264/AVC – Proc. VCIP, Beijing, China, July 2005
- [8] T. Schierl, H. Schwarz, D. Marpe and T. Wiegand, Wireless Broadcasting using the Scalable Extension of H.264/AVC - Proc. ICME 2005, Amsterdam, The Netherlands, July 6-8, 2005
- [9] G. Liebl, T. Schierl, T. Wiegand and T. Stockhammer, Advanced Wireless Multiuser Video Streaming using the Scalable Video Coding Extensions of H.264/MPEG4-AVC - ICME 2006, Toronto, Canada, July 2006
- [10] <http://developers.videolan.org/x264.html>
- [11] <http://ffmpeg.mplayerhq.hu/>
- [12] S. Saponara, C. Blanch, K. Denolf, J. Bormans, The JVT advanced Video Coding Standard: Complexity and Performance Analysis on a Tool-by-Tool Basis – Packet Video 2003, Nantes, France, April 2003
- [13] Z. He and D. Wu, Linear Rate Control and Optimal Statistical Multiplexing for the JVT Video Encoding – Submitted to Journal of Visual Communication and Image Representation, Special Issue on Emerging H.264/AVC video coding standard – 2004
- [14] M. Militzer, M. Suchomski, K. Meyer-Wegener, Improved p-Domain Rate Control and Perceived Quality Optimization for MPEG-4 Real-Time Video Applications – Proc. of ACM MM'03, pp. 402-411, November 2-8, 2003, Berkeley, California, USA
- [15] JVT-G050, Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC) – 7th Meeting: Pattaya, Thailand, 7-14 March, 2003
- [16] JVT-P202, Joint Scalable Video Model JSVM-3 – 16th Meeting: Poznan, Poland, July, 2005
- [17] JVT-T202, Joint Scalable Video Model JSVM-7 – 20th Meeting: Klagenfurt, Austria, 15-21 July, 2006
- [18] JVT-T202-AnnexG, Joint Draft 7 of SVC Amendment– 20th Meeting: Klagenfurt, Austria, 15-21 July, 2006
- [19] Z. He, P-domain rate-distortion analysis and rate control for visual coding and communication. PhD Dissertation, University of California, Santa Barbara, 2001.

11 Annex: VMC-5400 Data Sheet

**DIGITAL VIDEO EXPERTS**

VMC-5400TM

A d v a n c e d A V C / H . 2 6 4 E n c o d e r

SDI and Composite capture and compression board for developers

Specifically designed for developers

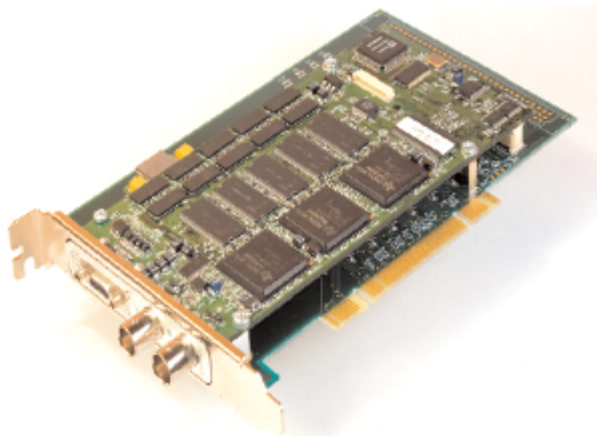
The VMC-5400TM capture and compression board makes very high quality H.264 / AVC hardware encoding (low CPU consumption) a reality for video ingest and professional applications.

The VMC-5400TM board takes advantage of VITEC's multi-DSP architecture enabling a real-time flexible video compression.

Designed for mission critical and video broadcast applications, the VMC-5400TM is ideally suited for advanced video integration ranging from airborne military video encoders to rack mounted TELCO IP-TV solutions.

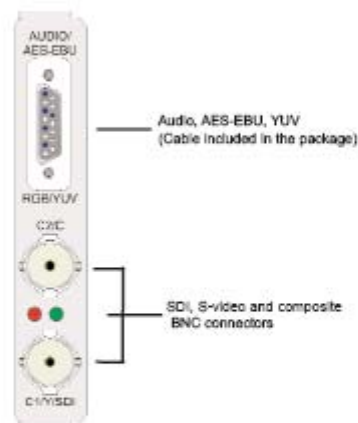
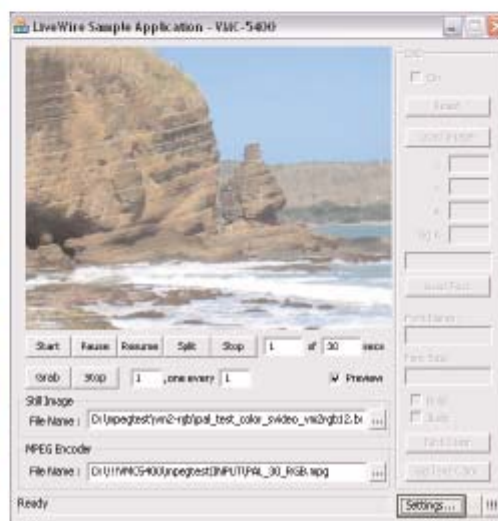
Key benefits of the multi-DSP architecture:

- **Easy and Economic Upgrades**
Unlike solutions based on ASIC or FPGA, VITEC's DSP based solutions are software upgradeable thus enabling economic upgrade by offering continuous feature and performance enhancements without hardware replacement.
- **Consistent Installed Base**
A VAR or SI can upgrade its installed base of equipment with a simple software upgrade and keep a constant hardware base, which is much easier to maintain.
- **Cost Effective**
Very powerful and easy-to-use software development tools (SDK), called LiveWireTM, dramatically reduce the time-to-market when implementing a VITEC product.



TECHNICAL SPECIFICATIONS

Inputs / Outputs	Video Formats	NTSC/PAL
	Video Inputs : Analog	YUV, RGB, S-Video, composite
	Video Inputs : Digital	SDI
	Audio Inputs : Analog	Analog balanced and unbalanced stereo
	Audio Inputs : Digital	Digital Audio AES/EBU, embedded SDI
	Preview on VGA	yes
Video Encoding	MPEG-4 AVC/H.264	FD1, 3/4D1, 3/3D1, HD1, CIF, QCIF
	AVC profile	MP @ L3
	MUX Encapsulated in MPEG-2 TS	yes
	Frame rate	29.97 (NTSC) 25 (PAL)
	Bitrate MPEG-4	64 Kbit/s to 10 Mbit/s
	Bitrate regulation mode MPEG-4 AVC/H.264	CBR, VBR
	VBR with Fixed Quantizer	yes
	GOP definition	yes
	IBP distance settings	yes
	Scene change detection	yes
Audio Encoding	AAC Low Complexity	yes
	Bitrate AAC-LC	6 to 448 Kbit/s (96 Kbit/s for CD quality)
	Audio Mode	Stereo
Signal calibration	Brightness, Contrast, Saturation, Hue Adjustments	yes
	Audio Level Adjustment	yes
Still Image	Resolution	720x576, 352x288 (PAL/NTSC) 720x480, 352x240 (NTSC)
	Field or Frame (2 fields) Capture	yes
Developers Resources	Operating Systems	Windows XP, 2000
	Development Kits	Low level SDK/API LiveWire framework Demo application source code
Advanced features	Audio deembedding from SDI	yes
	Still Image capture while encoding	yes
	Audiometer overlaid on preview	yes
	Status overlay on preview window	yes
	Pause/Resume mode	yes
	Split mode (back to back files)	yes
	Deblocking filter	yes
	CABAC	yes
	Adaptive noise filter	yes



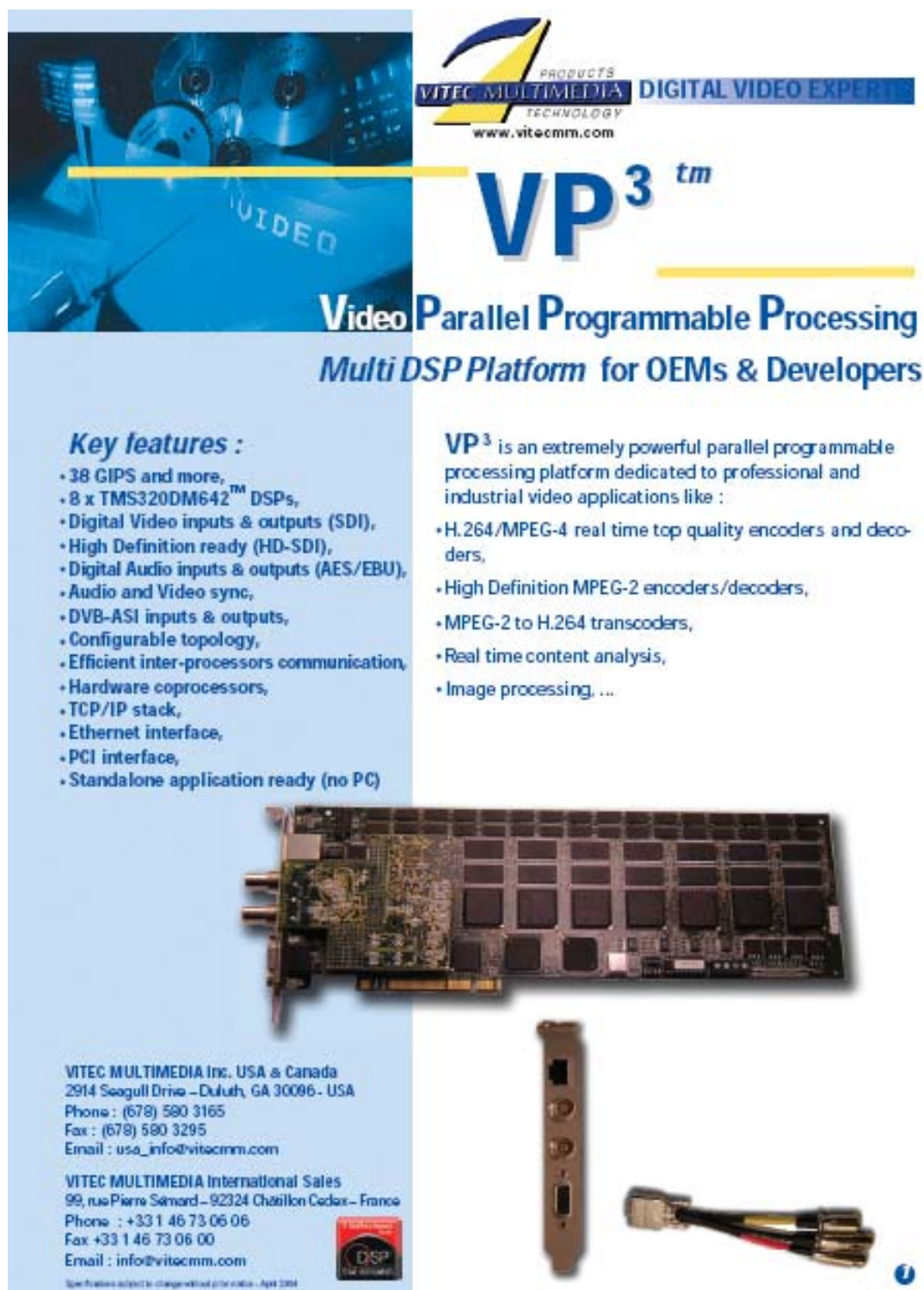
VITEC MULTIMEDIA Inc. USA & Canada
2914 Seagull Drive – Duluth, GA 30096 - USA
Phone : (678) 580 3165
Fax : (678) 580 3295
Email : usa_info@vitecmm.com



VITEC MULTIMEDIA International Sales
99, rue Pierre Sévère – 92324 Châtillon – France
Phone : +33 1 46 73 06 06
Fax +33 1 46 73 06 00
Email : info@vitecmm.com

Specifications subject to change without prior notice - March 2006

12 Annex: VP3 Data Sheet



The graphic is a promotional data sheet for the VP3 product. It features a blue and yellow color scheme. At the top left, there is a collage of digital media including CDs, DVDs, and a video camera. To the right of this is the VITEC MULTIMEDIA TECHNOLOGY logo, which includes a stylized 'V' and the text 'PRODUCTS DIGITAL VIDEO EXPERTS' and 'www.vitecmm.com'. Below the logo, the product name 'VP3tm' is written in large, bold, blue letters. Underneath this, the text 'Video Parallel Programmable Processing' and 'Multi DSP Platform for OEMs & Developers' is displayed in a smaller blue font. The central part of the graphic is divided into two columns. The left column, titled 'Key features :', lists 15 bullet points detailing the product's capabilities, such as '38 GIPS and more', '8 x TMS320DM642TM DSPs', and various input/output options. The right column, titled 'VP³ is an extremely powerful parallel programmable processing platform...', describes the product's applications, including 'H.264/MPEG-4 real time top quality encoders and decoders' and 'Image processing, ...'. Below the text, there are three images: a large circuit board (the VP3 module) with a grid of chips, a smaller vertical module, and a cable with multiple connectors. At the bottom left, contact information for VITEC MULTIMEDIA Inc. (USA & Canada) and VITEC MULTIMEDIA International Sales (France) is provided, including phone, fax, and email addresses. A small 'DSP' logo is also present at the bottom center.

Key features :

- 38 GIPS and more,
- 8 x TMS320DM642TM DSPs,
- Digital Video inputs & outputs (SDI),
- High Definition ready (HD-SDI),
- Digital Audio inputs & outputs (AES/EBU),
- Audio and Video sync,
- DVB-ASI inputs & outputs,
- Configurable topology,
- Efficient inter-processors communication,
- Hardware coprocessors,
- TCP/IP stack,
- Ethernet interface,
- PCI interface,
- Standalone application ready (no PC)

VP³ is an extremely powerful parallel programmable processing platform dedicated to professional and industrial video applications like :

- H.264/MPEG-4 real time top quality encoders and decoders,
- High Definition MPEG-2 encoders/decoders,
- MPEG-2 to H.264 transcoders,
- Real time content analysis,
- Image processing, ...

VITEC MULTIMEDIA Inc. USA & Canada
2914 Seagull Drive - Duluth, GA 30096 - USA
Phone : (678) 580 3165
Fax : (678) 580 3295
Email : usa_info@vitecmm.com

VITEC MULTIMEDIA International Sales
99, rue Pierre Sémard - 92324 Châtillon Cedex - France
Phone : +33 1 46 73 06 06
Fax : +33 1 46 73 06 00
Email : info@vitecmm.com

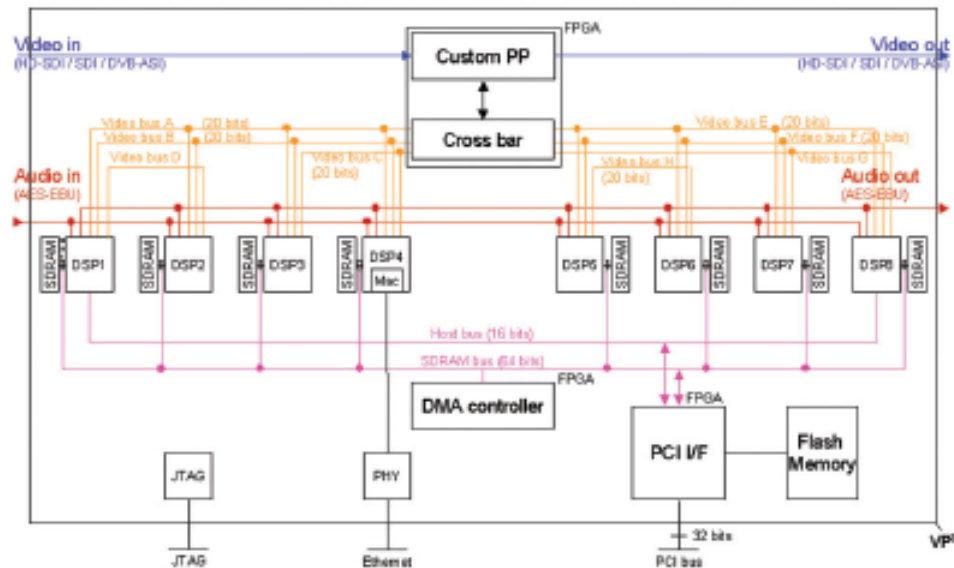
Specifications subject to change without prior notice - April 2004

ARCHITECTURE OF THE BOARD

VP³ implements 8 x TMS320DM642™ DSPs from Texas Instruments running at 600 MHz (and soon 720 MHz, 1GHz, ...) thus providing up to

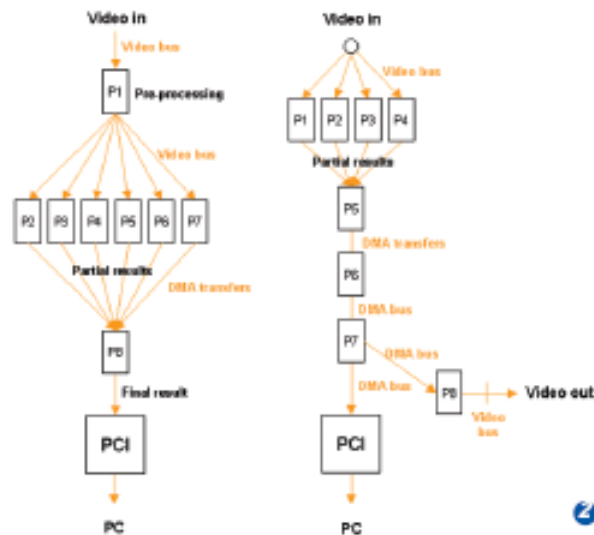
38,4 GIPS with a maximum of 4 operations per instruction (4 operations of 8 bits, 2 operations of 16 bits and 1 operation of 32 bits) which is a maximum

of 153,6 GOPS. Each DSP has a private local memory of 128 MB (SDRAM running at 100 MHz and 64 bits, which provides a throughput of 800 MB/s).



CONFIGURABLE TOPOLOGY

VP³ architecture is highly flexible and can be configured to fit to the specific needs of the developer's applications. Each DSP has 3 powerful and configurable video ports which are used as one of the communication ways between them. The topology of the array of 8 processors can be defined as a simple pipeline of eight processors, a fully parallel scheme or a mix of both. A cross-bar implemented in an FPGA interconnects the video ports of the DSPs.



INTER-DSPs COMMUNICATION

DSPs can communicate information to each others in several ways :

- Direct memory to memory block exchanges through the DMA controller services. High performance DMA inter-processor's communication channels have been optimized to allow 1 to 1, 1 to n or 1 to all data exchanges.
- Video data or raw data through their video ports and the cross-bar.
- The host busses of the DSPs are linked to PCI interface and can send/receive messages to/from the host through the PCI interface.

DMA CONTROLLER PRINCIPLE

One DSP initiates a Memory Block Transfer by sending a request to the DMA Controller specifying the list of destination DSPs which shall receive the message. The hardware controller puts the source and destination DSPs in hold state and takes control of their local memories to read the source memory and write into all the destination memories simultaneously. Once the transfer is achieved the DMA controller sends an interrupt to the source DSP to warn it that its message has been sent and also to the destination DSPs to warn them that they received a message.

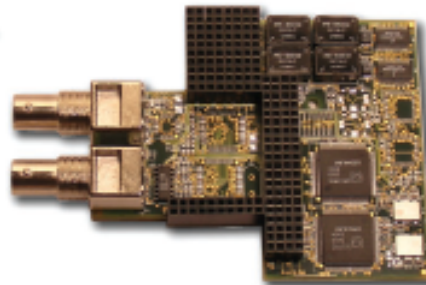


HARDWARE CO-PROCESSORS

- The FPGA implementing the cross-bar service is also the one interconnecting the digital video inputs and outputs to the processors array. It is large enough (up to 600,000 gates : Xilinx's XC2S600E) to host a hardware coprocessor implementing your own preprocessing algorithms acting on the video data itself (for instance : filters, scaler, ...). The content of the FPGA can be downloaded at the initialization of the board from the flash memory or by software.
- The FPGA implementing the DMA controller is also large enough (up to 600,000 gates : Xilinx's XC2S600E) to host another hardware coprocessor implementing your own computation algorithms too heavy or complex to be implemented in software (for instance : CABAC, ...). The content of the FPGA can be downloaded at the initialization of the board from the flash memory or by software.

DAUGHTER BOARD

VP³ includes a main board and a daughter board. The main board includes the PCI and Ethernet interfaces and the daughter board includes the video and audio inputs and outputs. To develop a specific daughter board with other kind of video and audio formats (DVI, analog,...) please contact a Vitec representative.



SOFTWARE TOOLS

VP³ comes with a complete software environment :

- Windows WDM driver (.SYS) able to support multi-board applications in the same PC,
- The source code of a sample application running under Windows XP/2000 and addressing directly the WDM driver,
- Source code of DSP sample applications like :
 - video pass-through, audio pass-through,
 - Memory BIST,
 - DMA transfers.
- Vitec recommends the usage of the Texas Instruments development tools to develop software for the DSPs themselves,
 - C/C++ compiler,
 - simulator,
 - emulator via the RTDS protocol and standard JTAG connector,
- Multi-DSP applications can be emulated by instancing several times the TI emulator software under Windows and the JTAG connector on the VP³ board.

SOFTWARE TOOLS

A complete framework, called LiveWire™, for developers who want to use VP³ hardware to develop a product running under Windows. LiveWire™ provides a set of ready to use connectable components leading to a drastic cut of the development time. LiveWire™ has many advantages. It :

- ensures highly flexible, scalable, truly customizable solutions,
- is designed to allow well-structured parallel development,
- allows to concentrate on solution specific tasks,
- overcomes the limitations of existing technologies such as DirectShow and COM in general,
- allows live reconnection of functional components without interruption of active processes,
- is compatible with Win32, COM, scriptable languages (Visual Basic, Java Script,...),
- takes advantages of XML based technologies and uses the Apache Xerces XML parser,
- provides different levels of SDK abstraction, from high level API for scripting languages through Win32 API for limited backward compatibility to the low level set of COM Interfaces for advanced development in C++.

LiveWire™ parts :

- LiveWire Core
- LiveWire Components
- LiveWire XML-based Profiles
- LiveWire Custom Components Wizard for MS Visual Studio C++
- LiveWire Multiplatform Shell
- LiveWire SDK
- LiveWire Tutorial and Samples

To start using LiveWire™ based products, all you have to do is to create an instance of Assembly Container, initialize it with XML-based Configuration Profile and run. Different sophisticated profiles can be created without extensive programming, using Integrated Property Page or directly by editing the XML file in the text editor of your choice. Components parameters persistence comes then automatically.

Very little programming is needed to use advanced features, such as Command Scheduling and Atomic Command Blocks. With a few extra lines of code you can complete an application capable of running execution scripts with frame accurate precision.

Custom LiveWire™ components creation is simplified by Wizard and they can be easily integrated into existing Assemblies.

The most important advantage of the SDK is the layered structure of the LiveWire™ framework which allows a quick development cycle.

TECHNICAL SPECIFICATIONS

Inputs / Outputs	Video Inputs	SDI, HD-SDI
	Audio Inputs	AES/EBU, Audio de-embedding from SDI
	Video Output	SDI, HD-SDI
	Audio Output	AES/EBU, Audio embedded in SDI
Other Interfaces	DVB-AS input and output	
	PCI Interface	
	10 Base-T or 100 Base-TX Ethernet using single RJ-45 connector	
	16-pin JTAG for external emulation hardware support. This is used to connect VP ³ to the TI emulator software	
Other specifications	Usage of the board	• PCI plug-in card • a stand-alone equipment with an external power supply (+5V and +3V) and a large Flash memory to store the program and the FPGA contents.
	Size	SD4 mm x 937 mm (12.76" inch x 42.1" inch)
	Weight	330 g
	Flash capacity	6 MB
	Compatibility	The PCI Interface is a 32 bit 33MHz PCI rev2.3 compatible with 5V and 3V3 with an auto-detection device

DSPs SPECIFICATIONS

VP³ uses 8 TMS320C642™ :

- High-Performance Digital Media Processor :
 - 600-MHz Clock Rate (and soon 720 MHz, 1 GHz, ...),
 - Eight 32-Bit Instructions/Cycle,
 - 4800 MIPS,
 - Fully Software-Compatible With C64x.
- Velocity™.2. Extensions to Velocity™. Advanced Very-Long-Instruction-Word (VLW) TMS320C64x DSP Core :
 - Eight Highly Independent Functional Units With Velocity™.2. Extensions:
 - Six ALUs (32-/40-Bit), Each Supports Single 32-Bit, Dual 16-Bit, or Quad 8-Bit Arithmetic per Clock Cycle,
 - Two Multipliers Support Four 16 x 16-Bit Multiplies (32-Bit Results) per Clock Cycle or Eight 8 x 8-Bit Multiplies (16-Bit Results) per Clock Cycle,
 - Load-Store Architecture With Non-Aligned Support,
 - 64 32-Bit General-Purpose Registers,
 - Instruction Packing Reduces Code Size,
 - All Instructions Conditional.
- Instruction Set Features
 - Byte-Addressable (8-/16-/32-/64-Bit Data),
 - 8-Bit Overflow Protection,
 - Bit-Field Extract, Set, Clear,
 - Normalization, Saturation, Bit-Counting,
 - Velocity™.2. Increased Orthogonality.
- L1/L2 Memory Architecture
 - 128K-Byte (16K-Byte) L1P Program Cache (Direct Mapped),
 - 128K-Byte (16K-Byte) L1D Data Cache (2-Way Set-Associative),
 - 2M-Byte (256K-Byte) L2 Unified Mapped RAM/Cache (Flexible RAM/Cache Allocation).
- Endianness: Little Endian
- Enhanced Direct-Memory-Access (EDMA) Controller (64 Independent Channels)
- 10/100 Mb/s Ethernet MAC (EMAC)
- Three Configurable Video Ports : supports Multiple Resolutions and Video Standards.
- Three 32-Bit General-Purpose Timers
- IEEE-1149.1 (JTAG)

C64x, Velocity™.2, Velocity™ and TMS320C64x are trademarks of Texas Instruments.
All trademarks are the property of their respective owners.
† IEEE Standard 1149.1-1990 Standard-Test-Access Port and Boundary Scan Architecture.

