CCITT SGXV
Working Party XV/4
Specialists Group on
Coding for Visual Telephony

Document 525
1989

TITLE: Description of Ref. Model 8 (RM8)

SOURCE: Specialist Group On Coding for Visual Telephony

version : June 9, 1989

revision 1 :	May 20	, 1988	doc #339
revision 2 :	July 20	, 1988	
revision 3 :	Sept 5	, 1988	doc #375
revision 4 :	Oct 19	, 1988	doc #396
revision 5 :	Nov 19	, 1988	
revision 6 :	Jan 21	, 1989	doc #446
revision 7 :	may 21	, 1989	
revision 8 :	June 9	, 1989	doc #525

CONTENTS

1 INTRODUCTION

This document gives a comprehensive description of the software in order to simulate REFERENCE MODEL 8 hereafter abbreviated to RM8. This model is used in the course of the research for comparison i.e. to choosing core elements for the flexible hardware specifications.

The given description does NOT intend to substitute the document " Flexible hardware specification for p x 64 kbit/s". This model is a configuration which has the ability to operate at various bitrates p=1,..,30 (see document 445).

The reader should be aware of the fact that:

1. Some adopted techniques described in this document are not a matter of standardization. For the flexible hardware other solutions are therefor allowed. In order to have comparable simulation results the methods described in this document are mandatory.

2. Some implemented techniques in RM8 are debatable but are used for comparison purposes only. One argument is the choice and length of the adopted sequences.

The readers are asked to give comments and corrections to remove ambiguous parts. The reader can send his amendments to:

PTT Research Neher laboratories
Transmission and Coding
Visual Communications Research
Room C159
P.O. BOX 421
Telefax: 31 70 43 6477
g-mail : HLSdn15.bitnet

2 DESCRIPTION OF REFERENCE MODEL 8

In the proceeding text reference model 8 will be described. Starting with the basic format parameter choice referred to as common source input format. The spatial sizes are specified where these are most critical where the temporal frequency could be variable.

3 COMMON SOURCE INPUT FORMAT (C.S.I.F).

The parameters for the C.S.I.F. are:

	Full CSIF	1/4 CSIF
Number of active lines Luminance (Y) Chrominance (U,V)	 288 144	 144 72
Number of active pixels per line Luminance (Y) Chrominance (U,V)	 360 180	 180 90

Table 1 : Source format (full CSIF and 1/4 CSIF)

The number of coded pels per line is reduced, because 360 divided by 16 does not yield in an integer value. The obtained format is called significant pel area (SPA).

3.1 Definition Of The Significant Pel Area.

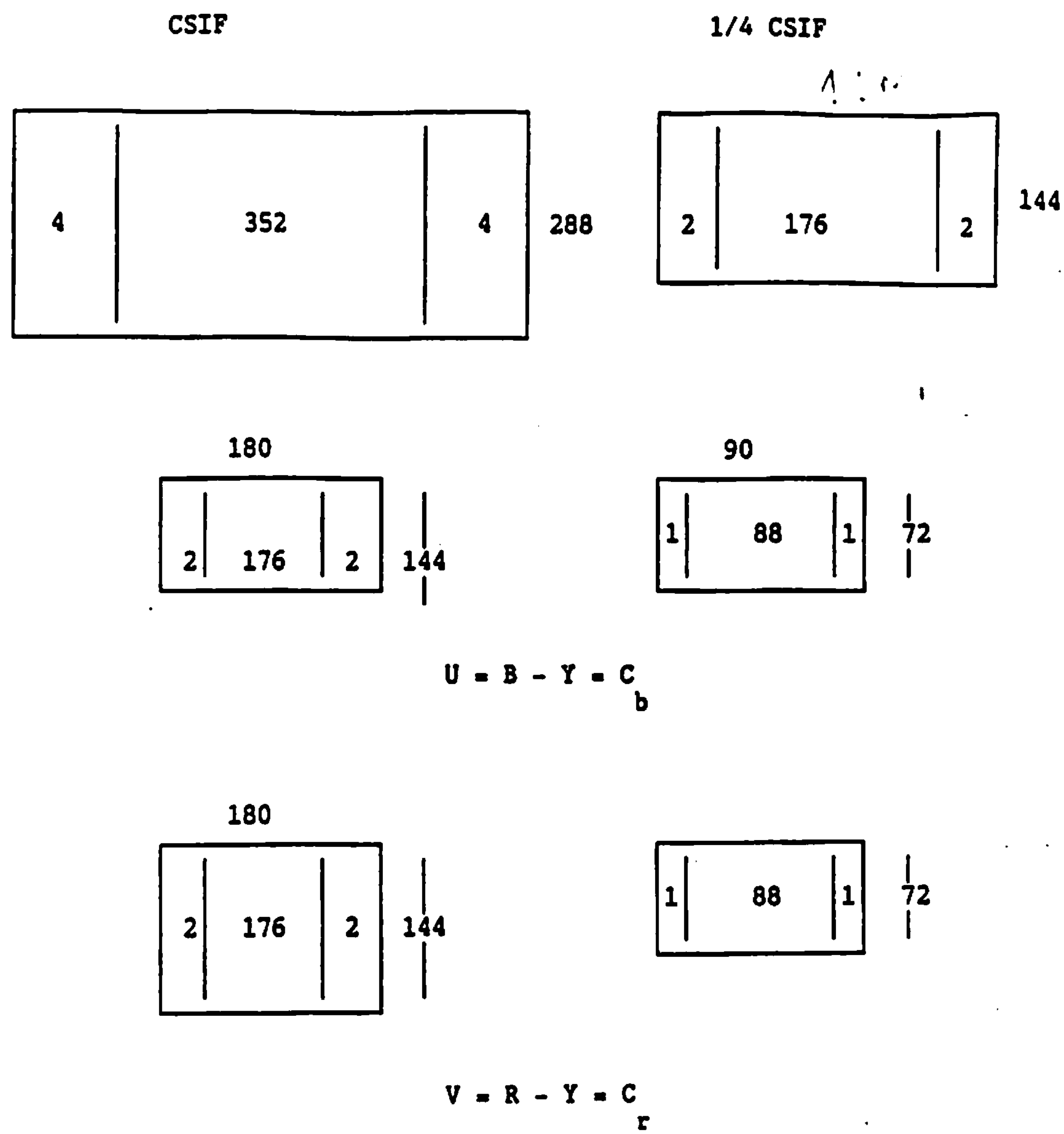


Figure 1 : Definition significant pel area

The number of pixels of the significant pel area (SPA) become:

352 x 288	=	101,376	pixels (Y)
176 x 144 x 2	=	50,688	pixels (U,V)
total	=	152,064	pixels/frame (Y,U,V)

CIF

176 x 144	=	25,344	pixels (Y)
88 x 72 x 2	=	12,672	pixels (U,V)
total	=	38,016	pixels/frame (Y,U,V)

1/4 CIF

In table 2 the influence of the frame rate on the number of pixels per second is given. This figure includes the number of pixels for the chrominance as well.

Frame Rate 30 Hz	Sub-sampling factor	number significant pixels/second	Mbit/s	number active pixels/second 1/4 CSIF	Mbit/s
15 Hz	1 : 2	2,280,960	18.3	570,240	4.6
10Hz	1 : 3	1,520,640	12.2	380,160	3.1
7.5Hz	1 : 4	1,140,480	9.1	285,120	2.3

Table 2 Bitrate versus frame rate

The first column of table 2 gives the frame rate and the third column depicts the number of omitted frames in the coding process for full CSIF and 1/4 CSIF the different values are tabulated. Applying the number of active pixels in one frame the total number of pixels per second are given in the next column with the corresponding bitrates.

4 BAS: 'F REFE NCE MODEL 8

4.1 Introduction

The used coding configuration is known as a hybrid DPCM/transform coder. Hybrid denotes a technique which involves more than one redundancy reduction technique, in this case interframe methods where the calculations are performed in pixel and transform domain. This coding procedure requires two transforms, i.e. a forward transform and an inverse transform, which are both located in the coding loop. Due to the usage of a block transform the incoming image is partitioned in non-overlapping blocks of N x N pixels . At the moment the blocksize of the transform is set to N=8.
A simple differential pulse coding modulation loop (DPCM) can be identified as the generic structure of the configuration. This DPCM-loop works in the temporal dimension i.e. interframe. For this purpose a frame memory is included in the loop containing the previously reconstructed image or frame. The generic structure of the reference model depicted in figure 2 is based on:

1. Macro blocks
2. Discrete Cosine Transform (DCT)
3. Variable length coding applying a semi-uniform quantizer
4. A zig-zag scanning of quantized coefficients
5. Displacement estimation
6. Buffer control

Figure 2 Hybrid transform/DPCM encoder.

Let us assume a sequence S of images,

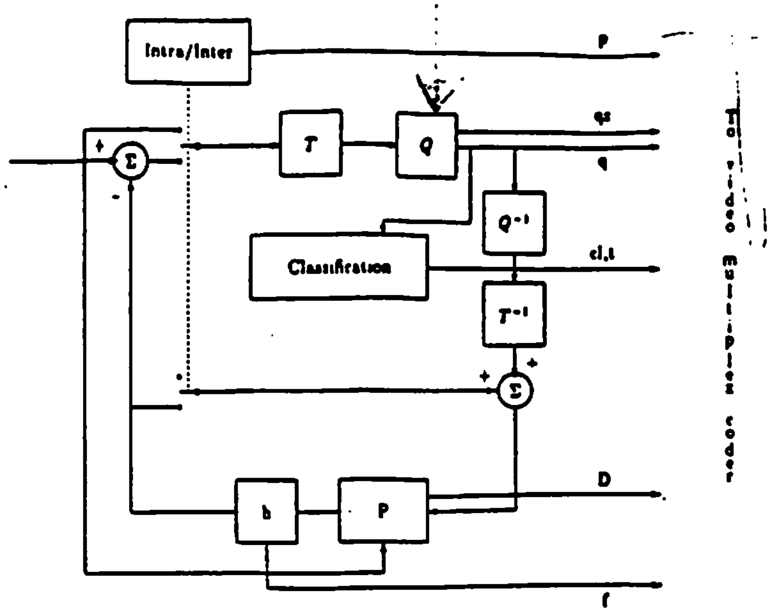
$$S = f(t) \quad \text{with } t = \ldots -3,-2,-1,0,1,2, \ldots$$

where f(t) is a 2-D intensity distribution at time t. Denoting the actual frame by f(t) and the previous frame by f(t-τ), the frame difference fd(t) becomes :

$$fd(t) = f(t) - f(t-\tau)$$

τ = 1 for skip 1 : 1
τ = 2 for skip 1 : 2

The frames are partitioned in blocks of N x N pixels and are numbered



p	=	Flag for intra/inter
P	=	Picture memory
qs	=	Quantisation index
q	=	Quantization index for transform coefficients
cl,t	=	Classification index, threshold
D	=	Motion vector
f	=	Switch loopfilter on/off
T	=	Transform
Q	=	Quantisation
h	=	Loop filter
T^{-1}	=	Inverse transform
Q^{-1}	=	Reconstruction Quantisation

Figure 2 Hybrid transform/DPCM encoder

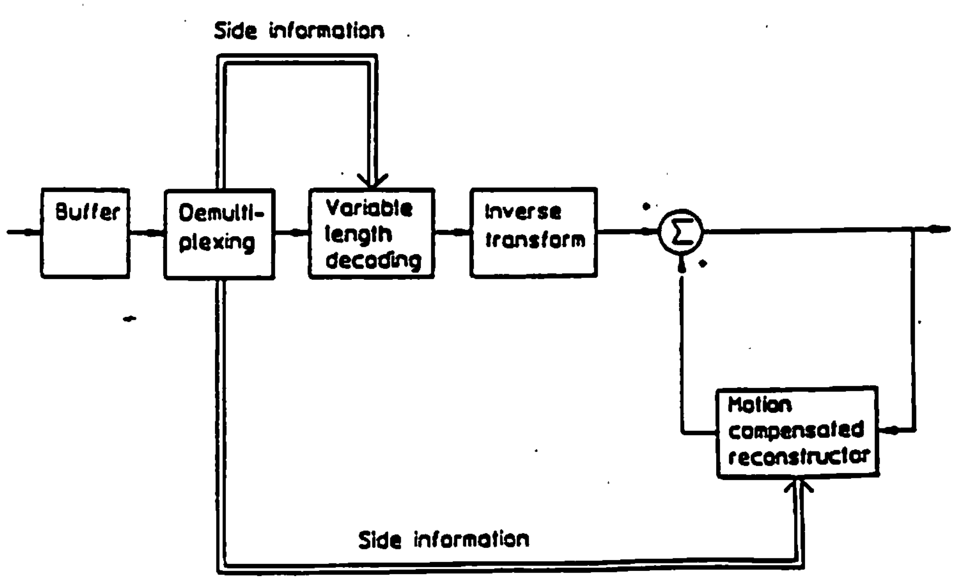


Figure 2 b Hybrid transform/DPCM decoder from left to right along a row of blocks. Let b(s,t) denote the intensities of the pixels in a block s at time t and let B(s,t) denote the coefficients of that block after transformation. The block difference bd(s,t) is obtained by subtracting the previous block $b(s,t-\tau)$ from b(s,t) :

$$bd(s,t) = b(s,t) - b(s,t-\tau)$$

Only blocks which have changed significantly are processed. This procedure is known as Conditional Replenishment (CR). With the change detector a distinction is made been between significant and non-significant changed blocks also called block type discrimination (see section 4.6).

The displacement estimation is achieved by a block matching technique with a search of +/- 7 pixels. The blocksize is 16 x 16.

To obtain the displaced block difference the coder applies a displacement vector $\vec{D}$ which might reduce the block difference bd(s,t).

In the case of a translatory motion, the displaced block difference can be expressed as:

$$dbd(s,t) = b(s,t) - b(s_{\vec{D}},t-\tau)$$

where $\vec{D}$ is the obtained displacement vector for the block under consideration.
Let mb(s,t) be a block of size 2N x 2N in the actual frame f(t), and let SW be a M x M search window in a previous frame $f(t-\tau)$, where $M > N$ and $M = 2N+2D_{max}$.
If a brute force method is used and $D_{max} = N-1$, the number of possible integer displacements within this search window becomes $(2N-1)^2$. The prediction error dbd for all these positions is calculated and the displacement vector $\vec{D}$ which produces the minimum error $dbd(s,t-\tau)$ is stored. Zero displacement can be interpretated as the orthogonal projection. After completion of the calculations the minimum error results in the displaced block difference. In a noiseless case, a pure translation by an integer number of pixels will result in an exact match i.e.

$b(s_{\vec{D}},t-\tau) = b(s,t)$. The motion trajectory is used to obtain the displaced block difference dbd(s,t).

Only integer displacement is considered, the brute force algorithm is optimal but for implementation purposes a coarse-fine 3 step algorithm is used (see appendix A). For each macro-block the displacement vector $\vec{D}$ is calculated indicating a block in the previous frame which results in the smallest prediction error. The displacement calculations are performed outside the coding loop and therefore this vector has to be transmitted as side information. For the transmission of the non-zero displacement vectors a differential

method is adopted using a 1-D prediction of the preceding calculated motion vectors (see section 4.11). The differential values are transmitted applying a VLC.
Next the prediction error (dbd(s,t)) is transformed using a 2-D Discrete Cosine Transform with blocksize N = 8.

4.2 Macro Block Approach

A macro block (MB) consists of a 16 x 16 luminance block and the two corresponding 8 x 8 U and V chrominance blocks. The luminance block is divided into four 8 x 8 sub blocks, i.e. a MB consists of six 8 x 8 sub blocks.
The construction is depicted in figure 3.

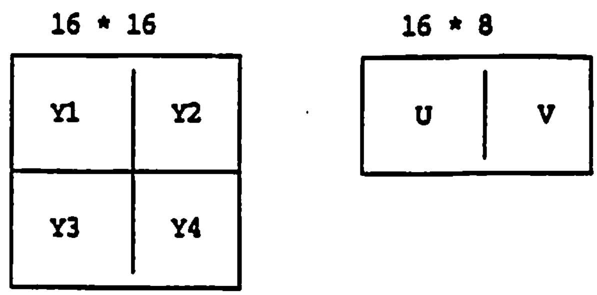


Figure 3 : Construction of a Macro Block (MB)

NOTE : A 16 x 16 Luminance block and the two corresponding 8 x 8 U and V chrominance blocks have the <u>same physical size</u>.

In table 3 the number of macro blocks per frame and the number of group of blocks per frame are shown:

Format	number of GOB in a frame	number of MB in a GOB	total number of MB in a frame
CSIF	12	33	396
1/4 CSIF	3	33	99

Table 3 Relationship between number of Macro blocks and picture format

4.3 Group Of Block Structure

With the introduction of a second picture format the intercommunication of codecs with different formats has to be considered. The smaller format is especially intended for low cost videophones, but realization of a second mode is a burden to all Full CIF codecs. A full CIF codec must be able to receive and transmit Quarter CIF pictures.

To get a good balance between 'wasted' bits for GOB headers and a fast error recovery the number of GOBs for CIF should be about 8 to 18. The block scan in figure 4 respects a high correlation of the motion vectors of succeeding macro blocks.

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22
23	24	25	26	27	28	29	30	31	32	33

width : 176 pels
height: 48 lines

Figure 4 GOB with Macro Block Addresses.

CIF

1	2
3	4
5	6
7	8
9	10
11	12

QCIF

1
3
5

Figure 5 Group Number for CIF Picture and QCIF Picture

For the 12 GOBs of a full CIF picture group numbers with four bits are sufficient. The remaining four numbers can be used for other purposes.

4.4 Discrete Cosine Transform

The block-differences bd(s[x,y],t) are transformed with the Discrete Cosine Transform (DCT).
The 2-D DCT is defined as :

$$BD(s[u,v],t) = \frac{1}{4} C(u)\, C(v) \sum_{x=0}^{7} \sum_{y=0}^{7} bd(s[x,y],t) \cos\left[\frac{\pi u(2x+1)}{16}\right] \cos\left[\frac{\pi v(2y+1)}{16}\right]$$

with u = 0,1,2, ... 7
v = 0,1,2, ... 7

$$bd(s[x,y],t) = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C(u)\, C(v)\, BD(s[u,v],t) \cos\left[\frac{\pi u(2x+1)}{16}\right] \cos\left[\frac{\pi v(2y+1)}{16}\right]$$

with x = 0,1,2, ... 7
y = 0,1,2, ... 7

where x,y = spatial coordinates in the pixel domain
u,v = coordinates in the transform domain

$C(u),C(v) = 1/\sqrt{2}$ for u,v = 0
1 otherwise

The luminance blocks and the chrominance blocks are transformed with a blocksize of 8 x 8 pixels. To assure that the simulation results at the different laboratories are similar it is advisable to exchange the software of the DCT.
The proposed specification for the IDCT Chips can be found in appendix C.

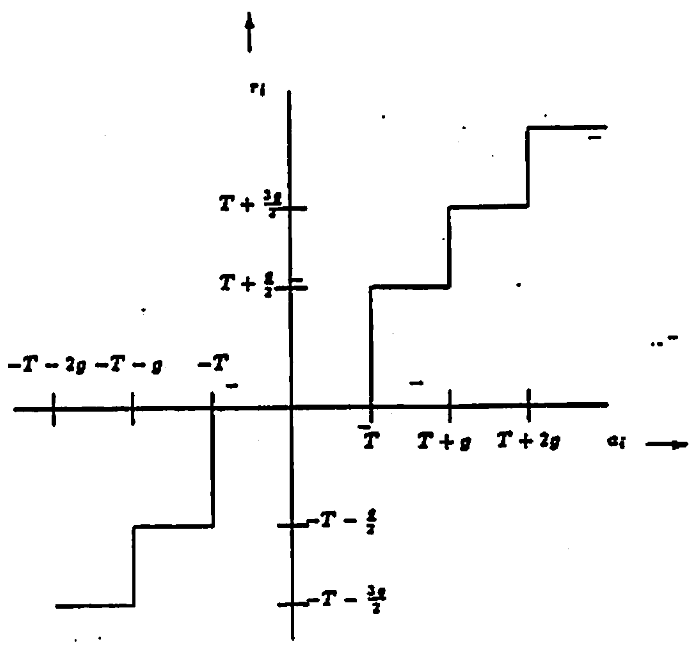
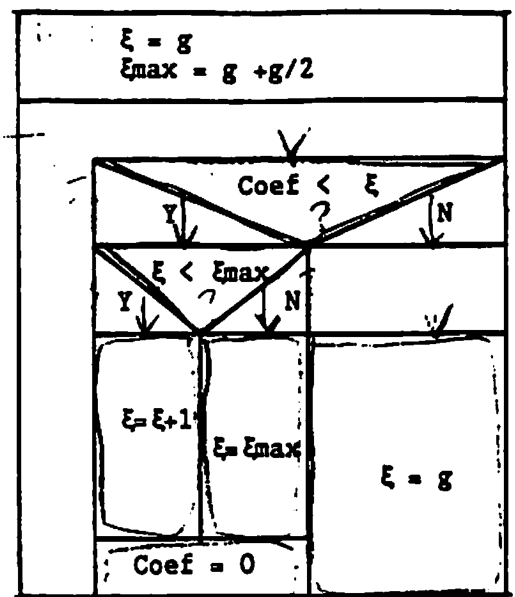


Figure 6 Quantization characteristic

4.5 The Quantization

4.5.1 Variable Threshold -

A variable threshold is applied independently of the quantization strategy to increase the number of zero coefficients. In the case of the variable threshold the threshold and its value depends on the length of string of zeroes. It is assumed that the transformed components have been zigzag scanned to form a one dimensional set of coefficients, before the quantization process. The accuracy of the coefficients is 12 bits. Referring to this scale the threshold ξ is modified within the block according to the variable thresholding algorithm as described below.



Example for g=32:

Coefficients	50	0	0	0	33	34	0	40	33	34	10	32
Threshold ξ *	32	32	33	34	35	36	37	38	32	32	32	33
New Coeff.	50	0	0	0	0	0	0	40	33	34	0	0
Quantized value	48	0	0	0	0	0	0	48	48	48	0	0

* The threshold is valid for the actual coefficient
New Coeff. denotes new coefficients after thresholding and before quantization.

4.5.2 The Quantization Strategy - The result after the transformation and the variable thresholding technique is quantized with an uniform quantizer. The uniform quantizer is defined by a step g and controlled by the buffer state. For RM8 the quantizer threshold has a value $T = g$.

$$q_{dec}(n) = T + (n-1)\,g \quad , \; n = 1,2,\ldots$$

$$q_{dec}(0) = 0$$

Taking into account the negative values the expression becomes:

$$q_{dec}(n) = \frac{n}{|n|}\,\{ T + (|n|-1)\,g \} \quad , \; |n| = 1, 2, 3, \ldots$$

$$q_{rep}(n) = \frac{q_{dec}(n) + q_{dec}(n + n/|n|)}{2} \quad \text{for } |n| = 1, 2, \ldots$$

$$q_{rep}(0) = 0$$

with q_{dec} the decision level

q_{rep} the representation level

g the quantizer stepsize

T threshold.

Example : A transform coefficient c with :

$$1.0\,g \leq c < 2.0g$$

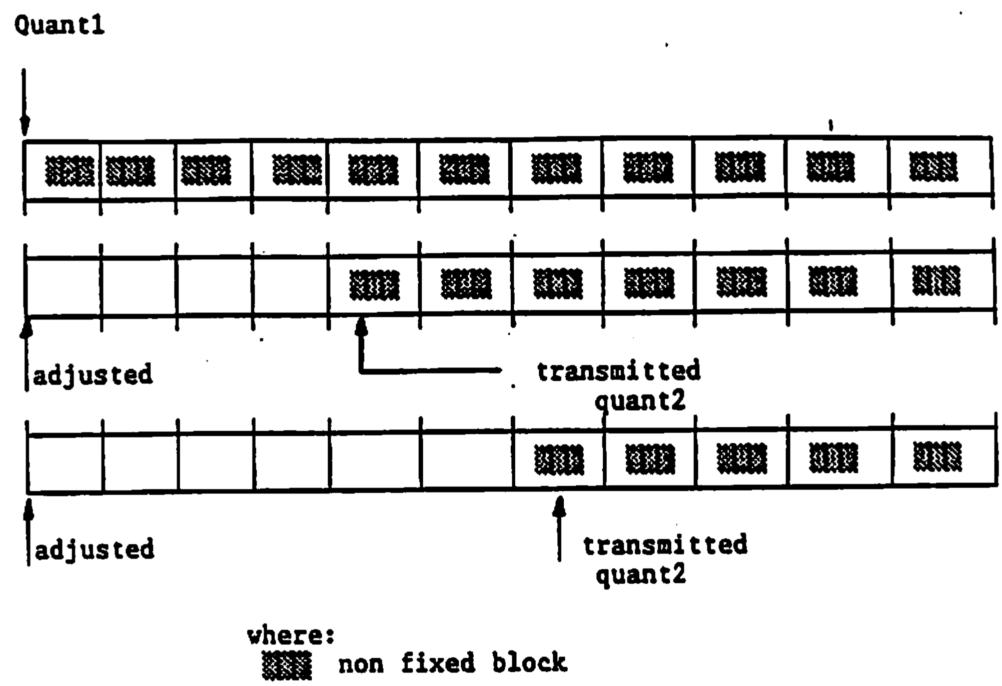
is quantized to the value of 1.5 g. The characteristic of the quantizer is depicted in figure 6.

Figure 6 . Characteristic of the quantizer

The dynamic range of the coefficients in the case of a blocksize of 8 x 8 is [-2048,2047]. The same quantizer is used both for luminance and chrominance coding.

The stepsize of the quantizer is adjusted , when different stepsizes are necessary, every 11th macroblock(at the start of each row of blocks in a GOB). The quantizer stepsize is Transmitted with the FIRST non fixed block (no MC coded, MC coded or intra, see figure 8) in this row of blocks using the TYPE 3 VLC for the 2nd and 3rd row of blocks of a GOB .

The number of bits for the stepsize as well as for the TYPE 3 VLC must be taken into account.



Example adjustment and transmitting in an GOB

4.6 Coding Of Coefficients

4.6.1 Scanning Technique -

In order to increase the efficiency of capturing the non zero components a zig-zag scanning class has been adopted:

ZIG - ZAG SCANNING :

1	2	6	7	15	16	28	29
3	5	8	14	17	27	30	43
4	9	13	18	26	31	42	44
10	12	19	25	32	41	45	54
11	20	24	33	40	46	53	55
21	23	34	39	47	52	56	61
22	35	38	48	51	57	60	62
36	37	49	50	58	59	63	64

The transmission of the coefficients must stop when the last non zero coefficient has been reached.

4.6.2 Coding Of The Scanned Coefficients With A Two-Dimensional VLC.

To increase coding efficiency a two dimensional variable length code has been adopted. This means that "events" are coded. "Event" is defined as :

event : a combination of a magnitude (non-zero quantization index) and a RUN (Number of zero indexes preceding the current non-zero index)

Coefficients unequal to zero defining the end of the run-length are considered as composite rather than separate statistical event.

The run-length and the magnitude of composite events define the entries of the 2-D VLC table which contains the code words for the composite events. Events are coded with Huffman's algorithm. However, events with low probabilities are coded using fixed length codes. These codes consists of the following three parts.

1. Escape (6 bits) for indicating the use of fixed length codes.
2. Run (6 bits)
3. Level (8 bits; See Note 1).

Note 1: Note that clipping must be introduced for the quantized coefficients F : $-128\,g \leq F < 128\,g$! The maximum range for the non-zero coefficients is now $+127g$ and $-128g$..

Note 2: $0 \leq run < 64$ (for blocksize 8)
After the last non-zero coefficient an End-Of-Block (EOB) marker is sent indicating that all other coefficients are zero. The length of the EOB word is two bits.

An example of the two dimensional VLC is given in figure 7 and the table is annexed.

EVENT = (RUN,LEVEL)

Example: (0,3) (1,2) (7,1) EOB

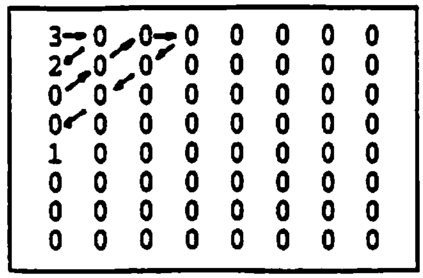


Figure 7 Example 2-D VLC

That means:

* (0,3) The DC component which has the value +3
* (1,2) is next non-zero component according to the zig-zag scanning the number of zeroes is 1.
* The next component is 1 preceded by 7 zeroes, result (7,1)
* EOB is an End of Block marker which indicates that there are no more non zero components.

4.7 Coding Strategy And Block Type Discrimination

In RM8 five different block types can be distinguished :

- Inter coded
- MC coded
- MC not coded
- Intra

- Inter coded + Q
- MC coded + Q
- Intra + Q

The order in which the block type is determined is depicted in figure 8.

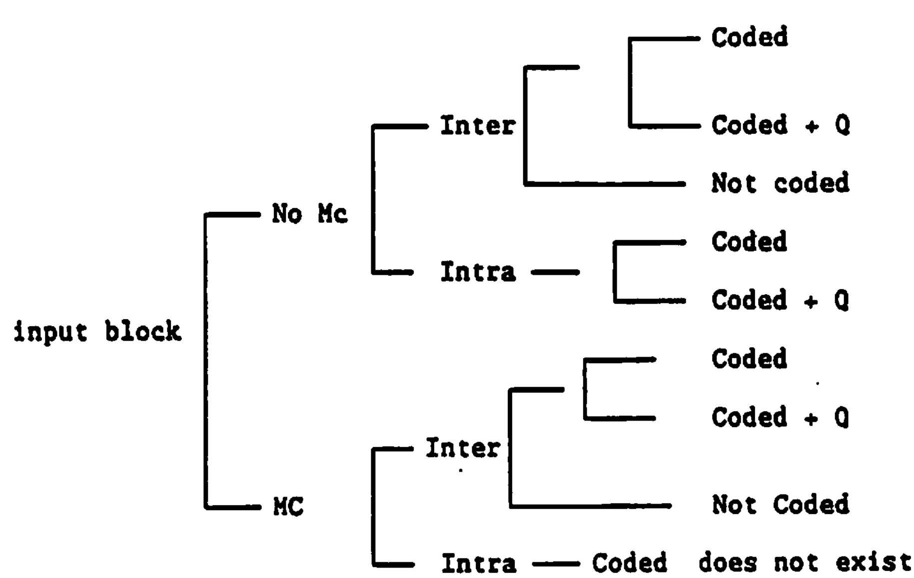


Figure 8 Decision Tree RM8

	Macro Block type	code with rel. addr.
0	No MC not coded	-
1	Inter coded	1
2	MC coded	01
3	MC not coded	001
4	Intra	0001
5	Inter coded + Q	00001
6	MC coded + Q	000001
7	Intra + Q	0000001

Table 4 : Adopted VLC for macro block types

If after quantization, all the quantized components of a sub block are zero, the sub block is declared to be not coded (blocktype 0,3). If all six sub blocks in a MB are not coded, the MB is declared to be not coded. In all other cases the MB is declared to be coded.

Y1	Y2	Y3	Y4	U	V

NOTE : The ~~Data Per Block (DPB)~~ is only transmitted if the macro block is coded.

All modified MB's (blocktype 2,3,4) are addressed with relative addressing, similar to the relative addressing used in RM4 (table 5) . The other block-types are coded according to the VLC in table 4.

N.B. : The last string of fixed blocks in a GOB is not encoded.

number of fixed MB's	codevord length	codevord
0	1	0
1	3	100
2	3	101
3	4	1100
4	4	1101
5	5	11100
6	5	11101
7	6	111100
8	6	111101
9	7	1111100
10	7	1111101
11	8	11111100
12	8	11111101
13	9	111111100
14	9	111111101
15	11	11111111000
16	11	11111111001
17	11	11111111010
18	11	11111111011
19	11	11111111100
20	11	11111111101
21	13	1111111111000
22	13	1111111111001
23	13	1111111111010
24	13	1111111111011
25	13	1111111111100
26	13	1111111111101
27	15	111111111111000
28	15	111111111111001
29	15	111111111111010
30	15	111111111111011
31	15	111111111111100
32	15	111111111111101
33	0	

Table 5 : Adopted VLC for relative addressing of non-fixed MB's

NOTE : More simulations have to be done to ensure that relative addressing causes visible gain.

4.8 Block Addressing For Macro Block Attribute

By the introduction of the Macro Block scheme in RM8, the side information could be reduced further with the introduction of pattern information. This pattern information consists of a set of 63 pattern indicating codec/non-coded blocks within the macro block. The patterns are depicted in figure 9.

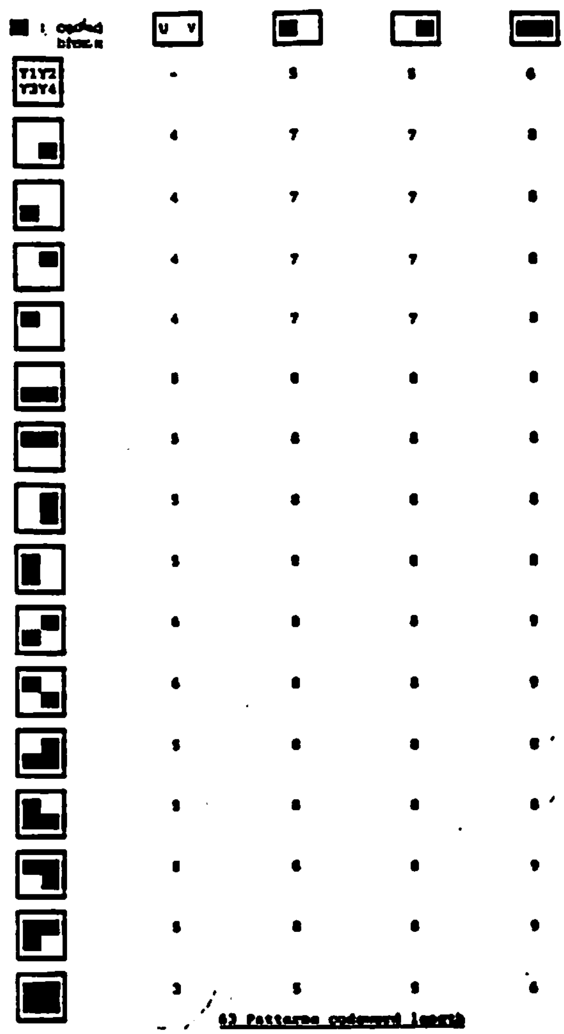


Figure 9 Pattern information Macro block

The pattern number = 32 Y1 + 16Y2 + 8Y4 + 4Y4 + 2Cb + Cr

The VLC for the pattern is given in table 6.

Pattern	Length	Codeword
60	3	000
4	4	0010
8	4	0011
16	4	0100
32	4	0101
12	5	01100
48	5	01101
20	5	01110
40	5	01111
28	5	10000
44	5	10001
52	5	10010
56	5	10011
1	5	10100
61	5	10101
2	5	10110
62	5	10111
24	6	110000
36	6	110001
3	6	110010
63	6	110011
5	7	1101000
9	7	1101001
17	7	1101010
33	7	1101011
6	7	1101100
10	7	1101101
18	7	1101110
34	7	1101111
7	8	11100000
11	8	11100001
19	8	11100010
35	8	11100011
13	8	11100100
49	8	11100101
21	8	11100110
41	8	11100111
14	8	11101000
50	8	11101001
22	8	11101010
42	8	11101011
15	8	11101100
51	8	11101101
23	8	11101110
43	8	11101111
25	8	11110000
37	8	11110001
26	8	11110010
38	8	11110011
29	8	11110100
45	8	11110101
53	8	11110110
57	8	11110111
30	8	11111000
46	8	11111001
54	8	11111010
58	8	11111011
31	8	11111100
47	8	11111101
27	9	111111100
39	9	111111101
55	9	111111110
59	9	111111111

Table 6 Code word length Pattern information

NOTE:

If a macro block type is "intra", its pattern information is <u>not</u> transmitted.

Example:
Y1 block is coded, Y2 ,Y3 ,Y4 ,U and V blocks not coded the pattern will be pattern 32.

4.9 Relative Addressing For Blocks Within A Group Of Blocks Structure

The relative addresses is adopted as depicted in figure 10. Run lengths are generated for the number of blocks to the next active blocks.

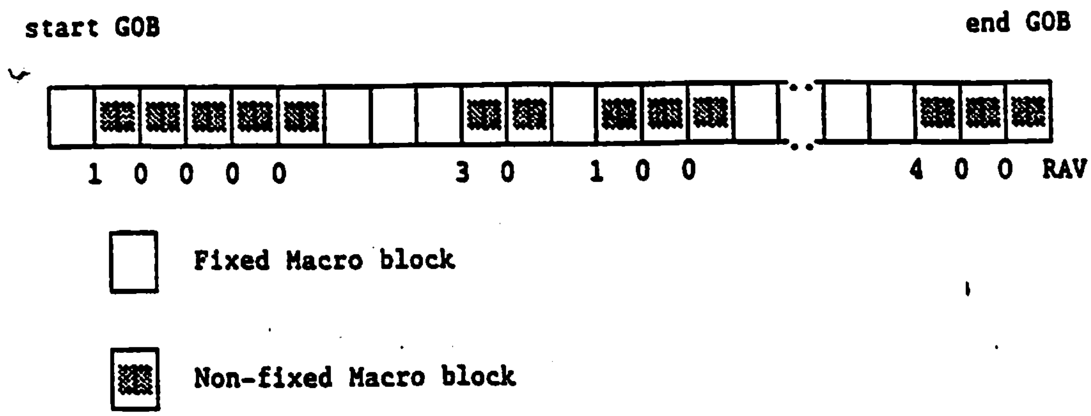


Figure 10 Relative addressing

Where RAV is the relative addressing value (i.e. the number of fixed blocks preceding a non-fixed block).
The last string of fixed blocks in a group of blocks is not encoded.
The GOB start code (see video multiplex) indicate the beginning of the next GOB. The table used for the runs is given in table 5.

4.10 Filter In The Loop

The introduction of a low pass filter after motion compensation (MC) could have the following advantages:

i. A reduction of high frequency artifacts introduced by MC.

ii. A reduction of quantization noise in the feedback loop.

The filter could be controlled with:

1. Displacement vector
2. Prediction error

A filter with impulse response as depicted in (1) is applied on a block of 8 x 8 pixels. The filter is applied both on luminance and the chrominance.

$$h(k,l) = \begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix} \cdot 1/16 \qquad (1)$$

4.10.1 Filtering Inside The Block Boundaries. -

Simplified processing at the block boundaries proposed for the 121 loop filter which yields in similar results was adopted but needs 20 - 40 % fewer operations. At the block boundaries the filter coefficients need to be adjusted in the case of adopting filtering inside the block. For the three cases the coefficients are depicted in the figure below.

i. $1/16 \begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}$ for pixels inside the block edges

ii. $1/16 \begin{vmatrix} 4 \\ 8 \\ 4 \end{vmatrix}$ for pixels on the block edges

iii. 1/16 | 16 | for pixels on the block corner positions

The processing of a block is revealed in figure 11.

1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2 4 2	2
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1
1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1 2 1	1

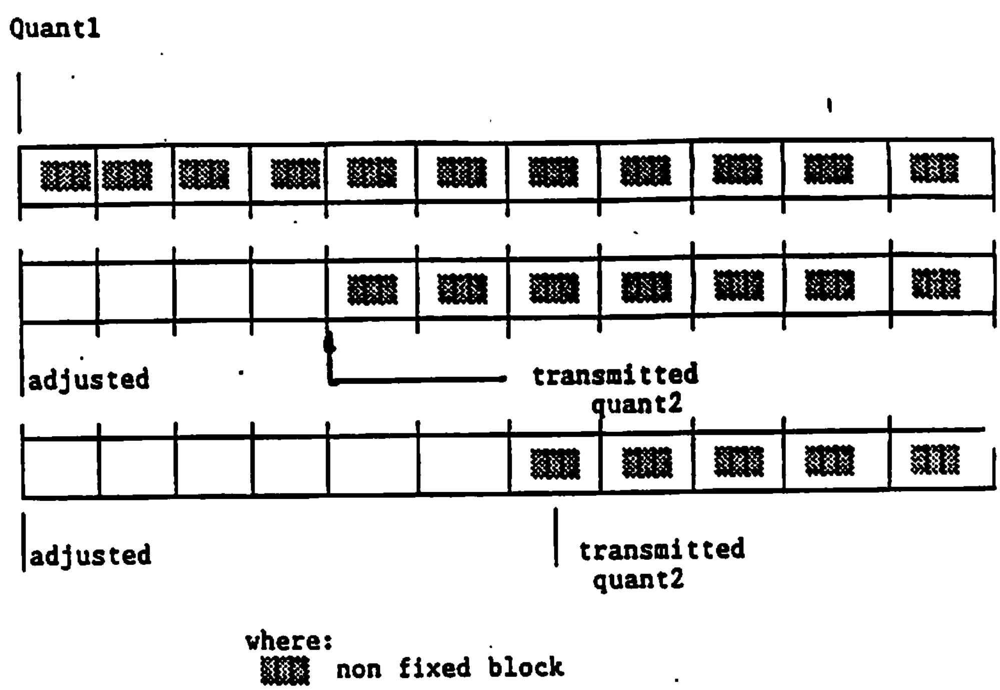
Figure 11 Filtering of an block with the 121 filter

In reference model 8 the filter is controlled with the motion vector i.e. if the motion vector is non zero the filter is on. Luminance blocks as well as chrominance blocks are filtered.

4.11 Buffer

4.11.1 Buffer Control

For RM8 the stepsize varies from 4 to 64 with step 2. The dependence on the buffer fullness is depicted in table 7 (the bitrate is q*64 kbit/s). The stepsize of the quantizer is adjusted every 11th macroblock when different stepsizes are necessary, (at the start of each row of blocks in a GOB). The quantizer stepsize is Transmitted with the FIRST non fixed block (no MC coded, MC coded or intra, see figure 8) in this row of blocks using the TYPE 3 VLC for the 2nd and 3rd row of blocks of a GOB .



Example adjustment and transmitting in an GOB

NOTE: If all blocks in a row of the GOB (see figure 5) either are fixed or MC not coded, the quantizer stepsize is not transmitted. Each GOB the stepsize is adapted according to table 6. This means that 5 bit/GOB are spent because of the 31 different stepsize values.

buffer content [kbit]	stepsize quantizer
< 400*q	———> 4
< 600*q	———> 6
< 800*q	———> 8
.	.
.	.
.	.
< 6000*q	———>60
< 6200*q	———>62
≥ 6200*q	———>64

or : step = 2 * INT (buf_cont / [200*q]) + 2

Table 7 : Quantizer stepsize as a function of the buffer fullness

Where INT denotes the truncation of the fraction, i.e. 1.5 -> 1 , 1.3 -> 1 and 1.6 -> 1.

The buffer size is only related to the bitrate. In annex F a table is provided giving the number of bits per macro block for each combination of frame and bit rates for full and QCIF. Buffersize= q x 6.4 kbit

4.11.2 Buffer Overflow -

A buffer size of q*6.4 kbit is intended. After each MB the buffercontent can be calculated (mean 15 bit/MB for q = 1; 10 Hz). When the buffer fullness exceeds q*6.4 kbits, the coefficients and the motion vector are set to zero in the next macro block (however resulting in a small buffer overflow).

N.B. : When the frame rate is not equal 10 Hz or the bitrate is not equal to 64 kbit/s, the mean number of bits per MB are revealed in appendix F.

4.12 Motion Estimation

The prediction error can be minimized with motion compensation.

At the moment the 3-step method is adopted in RM8 with blocksize 16 x 16 i.e. macro block based. The method can be found in appendix A.

The 3-step method is applied on luminance only. The motion vector for the chrominance is derived from the luminance by dividing the luminance vector by two and truncate the result to integer value.

example :

for luminance	-->	for chrominance
(3, 2)		(1, 1)
(-5,-6)		(-2,-3)

4.12.1 Differential Motion Vector Coding -

The coding of the displacement vector with fixed codevord length (FLC) as used in RM5 vastes a lot of bits when the moving area in the scene is large. Instead of transmitting the value of the calculated vector itself, the differential vector is transmitted applying a variable length code.
The differential technique is employed based on a 1-D prediction: the prediction is the motion vector of the previous macro block.
In case of the first macro block in the GOB, the previous vector is zero. The adopted VLC is depicted in table 8.

Amplitude	Code vord	Code vord length
-14	11111110000	11
-13	11111101111	11
-12	11111101110	11
-11	1111110110	10
-10	1111110101	10
-9	1111110100	10
-8	111111001	9
-7	111111000	9
-6	11111011	8
-5	11111010	8
-4	1111100	7
-3	11110	5
-2	1110	4
-1	110	3
0	0	1
1	100	3
2	1010	4
3	10110	5
4	1011100	7
5	10111010	8
6	10111011	8
7	101111000	9
8	101111001	9
9	1011110100	10
10	1011110101	10
11	1011110110	10
12	10111101110	11
13	10111101111	11
14	10111110000	11

Table 8 Adopted VLC for Differential motion vector

4.13 MC/No MC Decision

For the moment we use the characteristic as defined in RM4 (adapted to blocksize 16 x 16). The evaluation function for displacement estimation is ~~a sum of absolute differences concerning to all of the pels in a block.~~ The characteristic whether to suppress the displacement vector is depicted in figure 11. The characteristic is determined experimentally.

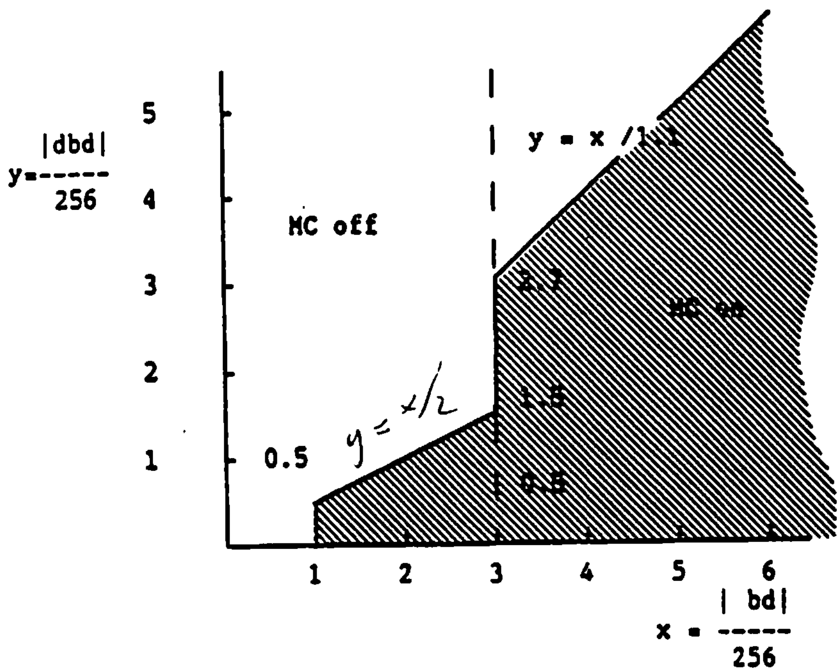


Figure 11. Characteristic MC/ No MC

Where dbd denotes the displaced block difference and bd the block difference, see also page 9.

NOTE: MC off includes the solid line. This characteristic resolves partly the sticking noise in the uncovered background (#107).
The absolute values are teken pel by pel

4.14 Intra Mode Decision

The most important reason for the re-introduction of the inter/intra switch are:

1. much better performance at scene cuts, too fast movement and in areas of decovered background (e.g. when Trevor raises his hands)

2. better error resilience and very simple implementation of (necessary) forced update

The implementation of the decision can be described with:

```
k = 1, 16
  l = 1, 16
    OR = O(k,l)
    Dif   = OR - S(k,l)
    VAR   = VAR+ Dif*Dif
    VAROR=VAROR+OR*OR
    MWOR =MWOR + OR
VAR = VAR/256
VAROR=VAROR/256-MWOR/256*MWOR/256
```

Where: O(k,l) denotes the pixels in the original macro block
S(k,l) denotes the pixels of the <u>motion</u> <u>compensated</u> estimated macro block.

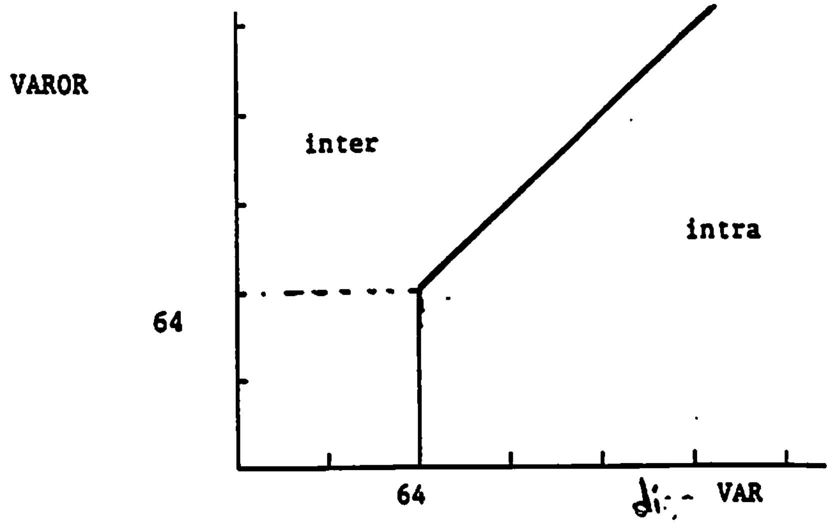


Figure 13. Characteristic intra / inter

NOTE1:
The parameters are calculated as integer values.
The lower threshold of 64 for the decision is empirically optimized.
The decision is depicted in figure 12.

NOTE2:
Inter mode includes the solid line

For INTRA blocks the first coefficient is the Transform DC value linearly quantized with a stepsize of 8 and no dead-zone. The resulting values are represented with 8 bits. A nominally black block will give 0001 0000 and a nominally white block 1110 1011. The code 1000 0000 is replaced by 1111 1111.

Coefficients after the last non-zero one are not tranmitted. EOB is always the last item in blocks for which coeffients are transmitted. After the blocks have been declared intra the coefficients are transmitted as depicted below:

FOR LUMINANCE (Y) AND FOR CHROMINANCE (U,V)

Mean value =C00 coefficient of sub-block SB	AC	EOB
8 bits FLC	VLC	2bits

where SB = Sub block
L = Luminance
U,V= Chrominance

4.15 First Picture

In order to have a simple method of starting the simulations it was agreed to adopt following method:

- Disregarding the number of bits for the first frame
- Using the stepsize depicted in table 9 for the various bitrates.
- Start the second picture with half full buffer

Video bitrate channels	Bitrate	Stepsize
q = 1	59.4	g = 32
q = 5	297	g = 16
q = 23	1472	g = 12
q = 29	1856	g = 8

Table 9 Bitrate versus stepsize first picture

The temporal reference remains as depicted in table 10.

original sequence	1	2	3	4	5	6	7	8	(10 Hz)
coded sequence	1	2	3	4	5	6	7	8	(10 Hz)

Table 10 : Temporal reference

Note:

For comparison purposes the number of bits for the first picture is counted. For the statistics the first frame is omitted.

5 QUARTER CIF SIMULATIONS (UNDER STUDY)

The QCIF simulations are carried out using the GOB structure outlined in section 4.3. The relation of the number of bits per MB is depicted in annex F.

* for 10 Hz use 60 bits per MB with q = 1
* for 15 Hz use 39 bits per MB with q = 1
* for 30 Hz use 18 bits per MB with q = 1
* for a first approach a prefilter one to one for the down conversion (see figure 13) is used
* for the up conversion a linear interpolation filter is used.
* the result will be displayed by using a full screen
* the SNR will be calculated at the CIF and QCIF level
* the use of other filters must be studied.

Down conversion CIF --> QCIF

```
____X____0____X__________X____0____X_____
    x1   p1   x2         x3   p2   x4
```

$$p1 = \frac{x1 + x2}{2} \qquad p2 = \frac{x3 + x4}{2}$$

Up conversion QCIF --> CIF

```
__________0____X__________X____0__________
          p1   x2r        x3r  p2
```

$$x2r = 3/4\ p1 + 1/4\ p2$$
$$x3r = 1/4\ p1 + 3/4\ p2$$

Figure 14 Down and up conversion CIF --> QCIF

6 VIDEO MULTIPLEX ARRANGEMENT

The Video multiplex is constructed in a hierarchical structure;

1. Picture layer
2. Group of block layer
3. Macro Block layer
4. Block layer

Picture Layer (PL)

PSC	TR	TYPE1	PEI1	PARITY	PEI2	PSPARE	GOB Data

Group Of Block Layer (GOBL)

GBSC	GN	TYPE2	QUANT1	GEI	GSPARE	MB Data

Macro Block Layer (MBL)

MBA	TYPE3	QUANT2	MVD	CBP	Block Data

Block Layer (BL)

TCOEFF	EOB

6.1 Picture Layer

PICTURE HEADER :

PSC	TR	TYPE1	PEI1	PARITY	PEI2	PSPARE	
20	5	15	1	0 8	1	0 16	bits

The picture header constist of:

1.	picture start code	(PSC)	20
2.	Temporal reference	(TR)	5
3.	Type information	(TYPE1)	13
4.	Extra insertion information	(PEI1)	1
5.	Parity information	(PARITY)	0 or 8
6.	Extra insertion information	(PEI2)	1
7.	Spare information	(PSPARE)	0 or 16

6.1.1 The Group Of Blocks Layer -

GOB-HEADER :

GBSC	GN	TYPE2	QUANT1	GEI	GSPARE	
16	4	6	5	1	0 16	bits

The Group of blocks header consists of:

1.	Group of Block Start Code	(GBSC)	16
2.	Group number	(GN)	4

3.	Type informatirn	(TYPE2)	6
4.	Quantizer information	(QUANT1)	5
5.	Extra insertion information	(GEI)	1
6.	Spare information	(GSPARE)	0 or 16

6.1.2 Macro Block Layer -

Macro BLOCK-DATA:

MBA	TYPE3	QUANT2	MVD	CBP

For the Macro block data on can distinguish:

1.	Macro Block address	(MBA)	
2.	Block type information	(TYPE3)	
3.	Quantizer type	(QUANT2)	5 bit
4.	Motion vector data	(MVD)	
5.	Coded Block Pattern	(CBP)	

6.1.3 Block Layer -

BLOCK-DATA :

TCOEFF	EOB

For the Block Data on can distinguish:

1. **Transform coefficients** (TCOEFF)

2. **End of Block Marker** (EOB)

Presentation of results

Sequence :
Modification :
Number of tracks :

Institute :
Date :
Temporal resolution :

	Item			15th pict.	mean seq
1.	RMS for luminance				
2.	SNR for luminance SNR for chrominance(U) SNR for chrominance(V)				
3.	Mean value of step size				
4.	Mean value of the number of non-zero coefficients				
5.	Mean value of the number of zero-coefficients				
6.	Block type of Macro	Fixed Coded Coded MC Fixed MC Intra Coded + Q Coded MC + Q Intra + Q			
7.	Block type of Y	Fixed Coded Coded MC Fixed MC Intra			
8.	Block type of UV	Fixed Coded Coded MC Fixed MC Intra			
9.	Number of bits	Macro attributes MVLB MVCB			
		End of block			
		Motion vectors			
		Coefficients	Y U V		
			Total		
		Total			

Bits for first frame Number of forced to fixed mb's	

COMMENTS TO TABLE : PRESENTATION OF RESULTS

The statistics exclude the first frame. Because the second picture to code is skipped, the number of tracks for the statistics is equal to the number of coded tracks - 2.

ad 1 :

$$P_{err}(t) = \sum_{i=1}^{352} \sum_{j=1}^{288} [\, \dot{f}(i,j) - f(i,j)]^2$$

$$P_{norm} = \frac{1}{\text{number pixels in frame}} P_{err}(t)$$

$$RMS = \sqrt{P_{norm}}$$

with:

$P_{err}(t)$ = frame based energy error

$f(i,j)$ = original pixel value

$\dot{f}(i,j)$ = reconstructed pixel value

P_{norm} = normalized error

RMS = root mean square error

ad 2 : The SNR is calculated for luminance as well as chrominance.

$$SNR = 20 \log \frac{255}{RMS}$$

ad 3 : The quantizer can be adapted after each MB Therefore the mean value of the stepsize becomes :

$$E(g) = \frac{\sum^{pic} \text{stepsize of non fixed MB}}{\text{no. of non fixed MB}}$$

ad 4 and 5 :

For the calculation of the mean value of the number of non-zero coefficients (MVNZC) and zero-coefficients (MVZC) the DC-component is included.

ad 7 : The number of fixed Y-blocks must be at least 4 x the number of fixed MB's, because

- a fixed MB consists of 4 fixed Y blocks
- a coded MB consist of 0 or more (but not all) fixed Y blocks

ad 8 : A similar story as in 7 holds for the number of fixed UV-blocks (at least 2 x the number of fixed MB's)

ad 9 : The number of bits for the MB attributes also includes the bits for relative addressing.

number of bits for 'Motion vectors' must be equal to (excluding rounding errors) :

8 x (#macro coded MC + #macro fixed MC)

mvLMB denotes bits for Mean Value of Luminance Macro Block (see intra mode, section 4.13), mvCMB are bits for Mean Value of Chrominance Macro Block.

ad 10: For comparison the number of bits for the first frame is given.

In order to check buffer overflow in the codec the number of forced to fixed Macro block is counted.

INDEX

APPENDIX A

3-STEP-ALGORITHM

Assuming a maximum displacement of 7 pixels the step algorithm iterates in three steps to the resulting minimum absolute error.

Step 1 : The actual block B is matched with 9 blocks in the previous window SW at the following positions :

(-4-4)	(0-4)	(4-4)
(-4 0)	(0 0)	(4 0)
(-4 4)	(0 4)	(4 4)

The position of the non-shifted prediction is (0 0). The order of the search has been defined as:

2	3	4
5	1	6
7	8	9

If the calculated error in position 1 has the same value as on position 3 than position 1 is chosen. The error is calculated by the Boolean less than i.e truncation.

Step 2 : For the second step a new search pattern is used. The best match of step 1 is the center of this pattern :

(-2-2)	(0-2)	(2-2)
(-2 0)	(0 0)	(2 0)
(-2 2)	(0 2)	(2 2)

Step 3 : The position of the best match in step 2 is the central position of the third and final search pattern :

```
(-1-1)(0-1)(1-1)
(-1 0)(0 0)(1 0)
(-1 1)(0 1)(1 1)
```

The best match of step 3 is the resulting minimum match error.

An example of the search process for the 3-step algorithm :

```
1       1       1

            2   2   2

1       1   2   1   2
                  3 3 3
            2   2 3 2 3
                  3 3 3
1       1       1
```

APPENDIX B

ADOPTED VARIABLE LENGTH CODES

Note:

The EOB marker needs 2 bits. In the case of a coded block and in that coded block only one coefficient exists, the code word length could be 2 bits instead of three bits.

see appendix H section codes for coeffient data page H-5

Word Length of VLC for Two-dimensional Coding

LEVEL (absolute value)

RUN	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	. . 128
0	3	5	6	8	9	9	11	13	13	13	13	14	14	14	14	20	20	20	20	20	
1	4	7	8	11	13	14	14	20	20	20	20	20	20	20	20	20	20	20	20	20	
2.	5	8	11	13	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
3	6	9	13	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
4	6	11	13	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
5	7	11	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
6	7	13	[illegible]	20	20	[illegible]	[illegible]	20	20	20	20	20	20	20	20	20	20	20	20	20	
7	7	13	[illegible]	20	20	[illegible]	[illegible]	20	20	20	20	20	20	20	20	20	20	20	20	20	
R 8	8	13	20	20	20	20	[illegible]	20	20	20	20	20	20	20	20	20	20	20	20	20	
U 9	8	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
N 10	9	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
11	9	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
12	9	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
13	9	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
14	11	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
15	11	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
16	11	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
17	13	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
18	13	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
19	13	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
20	13	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
21	13	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
22	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
23	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
24	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
25	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
26	14	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
27	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
28	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
.																					
.																					
63																					

** Word lengths of all other EVENTs (combination of RUN and LEVEL) are 20: ESCAPE CODE(6bits)+RUN(6BITS)+LEVEL(8bits).

** Word length of EOB is 2.

Code Set for Two-dimensional Coding of Coefficient Quantization Index (RM4)

No	RUN	LEVEL	CODE LENGTH	CODE	CODE STRUCTURE
1	0	-1	3	111	VLC
2	0	1	3	110	
3	-1	-1	4	1011	VLC
4	1	1	4	1010	
5	2	-1	5	10011	VLC
6	2	1	5	10010	
7	0	-2	5	10001	
8	0	2	5	10000	
9	3	-1	6	001111	
10	3	1	6	001110	
11	4	-1	6	001101	VLC
2	4	1	6	001100	
3	0	-3	6	001011	
4	0	3	6	001010	
5	5	-1	7	0001111	
6	5	1	7	.	VLC (4bits)
7	1	-2	7	.	
8	1	2	7	.	+
9	6	-1	7	.	
20	6	1	7	.	FLC (3bits)
1	7	-1	7	.	
2	7	1	7	0001000	
3	8	-1	8	00001111	
4	8	1	8	.	VLC (5bits)
5	0	-4	8	.	
6	0	4	8	.	+
7	9	-1	8	.	
8	9	1	8	.	FLC (3bits)
9	2	-2	8	.	
30	2	2	8	00001000	
31	10	-1	9	001001111	VLC (6bits)
2	10	1	9	.	
3	0	-5	9	.	+
4	0	5	9	.	
5	1	-3	9	.	
6	1	3	9	.	FLC (3bits)
7	3	-2	9	.	
8	3	2	9	001001000	
9	11	-1	9	000001111	
40	11	1	9	.	VLC (6bits)
1	12	-1	9	.	
2	12	1	9	.	+
3	0	-6	9	.	
4	0	6	9	.	FLC (3bits)
5	13	-1	9	.	
6	13	1	9	0000010000	

No	RUN	LEVEL	CODE LENGTH	CODE	CODE STRUCT
47	4	-2	11	00000011111	
8	4	2	11	.	
9	14	-1	11	.	
50	14	1	11	.	
1	15	-1	11	.	
2	15	1	11	.	VLC (7bits)
3	1	-4	11	.	
4	1	4	11	.	+
5	2	-3	11	.	
6	2	3	11	.	FLC (4bits)
7	0	-7	11	.	
8	0	7	11	.	
9	5	-2	11	.	
60	5	2	11	.	
1	16	-1	11	.	
2	16	1	11	00000010000	
3	17	-1	13	0000000111111	
4	17	1	13	.	
5	6	-2	13	.	
6	6	2	13	.	
7	0	-8	13	.	
8	0	8	13	.	
9	3	-3	13	.	
70	3	3	13	.	
1	1	-5	13	.	
2	1	5	13	.	
3	18	-1	13	.	VLC (8bits)
4	18	1	13	.	
5	19	-1	13	.	+
6	19	1	13	.	
7	0	-9	13	.	FLC (5bits)
8	0	9	13	.	
9	20	-1	13	.	
80	20	1	13	.	
1	21	-1	13	.	
2	21	1	13	.	
3	7	-2	13	.	
4	7	2	13	.	
5	2	-4	13	.	
6	2	4	13	.	
7	0	-10	13	.	
8	0	10	13	.	
9	4	-3	13	.	
90	4	3	13	.	
1	8	-2	13	.	
2	8	2	13	.	
3	0	-11	13	.	
4	0	11	13	0000000100000	

No	RUN	LEVEL	CODE LENGTH	CODE	CODE STRUCTURE
95	22	-1	14	00000000111111	
6	22	1	14	.	
7	23	-1	14	.	
8	23	1	14	.	
9	24	-1	14	.	
100	24	1	14	.	
1	25	-1	14	.	
2	25	1	14	.	
3	26	-1	14	.	
4	26	1	14	.	
5	0	-12	14	.	
6	0	12	14	.	
7	0	-13	14	.	VLC (9bits)
8	0	13	14	.	
9	0	-14	14	.	+
110	0	14	14	.	
1	0	-15	14	.	FLC (5bits)
2	0	15	14	.	
3	1	-6	14	.	
4	1	6	14	.	
5	1	-7	14	.	
6	1	7	14	.	
7	2	-5	14	.	
8	2	5	14	.	
9	3	-4	14	.	
120	3	4	14	.	
1	5	-3	14	.	
2	5	3	14	.	
3	9	-2	14	.	
4	9	2	14	.	
5	10	-2	14	.	
6	10	2	14	00000000100000	
127	EOB WORD		2	01	VLC
128	ESCAPE CODE		6	001000	VLC

** Other EVENTs (combination of RUN and LEVEL) are coded to:
ESCAPE CODE(6bits)+RUN(6bits)+LEVEL(8bits)

APPENDIX C

PROPOSED SPECIFICATION FOR INVERSE DCT CHIPS

Specification for Inverse DCT

1. Generate random integer pixel data values in the range -L to +H according to the attached random number generator (C version). Arrange into 8x8 blocks by allocating each set of consecutive 8 numbers in a row. Data sets of 10000 blocks each should be generated for (L=256, H=255), (L=H=5) and(L=H=300).
2. For each 8x8 block, perform a separable, orthonormal, matrix multiply, Forward Discrete Cosine Transform (FDCT) using at least 64-bit floating point accuracy.
3. For each block, round the 64 resulting transformed coefficients to the nearest integer values. Then clip them to the range -2048 to +2047. This is the 12-bit input data to the inverse transform.
4. For each 8x8 block of 12-bit data produced by step 3, perform a separable, orthonormal, matrix multiply, Inverse Discrete Cosine (IDCT) using at least 64-bit floating point accuracy. Round the resulting pixels to the nearest integer, and clip to the range -256 to +255. These blocks of 8x8 pixels are the "reference" IDCT output data.
5. For each 8x8 block of 12-bit data produced by step 3, use the proposed IDCT chip or an exact-bit simulation thereof to perform an Inverse Discrete Cosine Transform. Clip the output to the range -256 to +255. These blocks of 8x8 pixels are the "test" IDCT output data.
6. For each of the 64 IDCT output pixels, and for each of the 10,000 block data sets generated above, measure the peak, mean and mean square error between the "reference" and "test" data.
7. For any pixel, the peak error should not exceed 1 in magnitude.
   For any pixel, the mean square error should not exceed 0.06.
   Overall, the mean square error should not exceed 0.02.
   For any pixel, the mean error should not exceed 0.015 in magnitude.
   Overall, the mean error should not exceed 0.0015 in magnitude.
8. All-zeros in must produce all-zeros out.
9. Rerun the measurements using exactly the same data values of step 1, but change the sign on each pixel.

```
                         /*L and H must be long, ie, 32 bits*/
long rand(L,H)
long      L,H;
{
   static long   randx = 1;   /*long is 32 bits*/
   static double    z = (double)0x7fffffff;

   long   i,j;
   double x;                       /*double is 64 bits*/

   randx = (randx * 1103515245) + 12345;
   i = randx & 0x7ffffffe;                    /*keep 30 bits*/
   x = ( (double)i ) / z;                     /* range 0 to 0.99999... */
   x *= (L+H+1);                              /* range 0 to  < L+H+1 */
   j = x ;                                    /*truncate to integer*/
   return( j - L );                   /*range  -L  to  H  */
}
```

APPENDIX D

FIGURES

APPENDIX E

TABLES

APPENDIX F

RELATION BITS PER MACROBLOCK AND CHANNEL RATE

This annex presents a new formula to compute the main number of available bits per MacroBlock (MB).

In order to be more clear and to prevent ambiguous interpretations of the bitrate a parameter q is introduced. The parameter q denotes the bitrate for video only i.e. q = 1 the bitrate is 64 kbit/s. In the case of p=1 this could be interpreted as 64 kbit/s for video and 64 kbit/s for audio.

The formula in section 4.11 of doc. 396 (RM6) was computed starting from the particular case of q= 1 (64 kbit/s) and k = 3 (frame rate 10 Hz), giving 460 overhead bits per frame to be used for the Picture Header (PH) and for the group of MacroBlocks Headers (GMBH).

```
bitrate     : q * 64 kbit/s  ( p = 1..29)
frame rate  : 30 / k Hz      ( k = 1..4)
```

--> mean number of bits per MB :

```
    for CSIF     :  5 * k * q  bits/MB
    for 1/4 CSIF : 20 * k * q  bits/MB
```

Assuming the value of 460 bits (as suggested by the formula with q = 1 and k = 3) enough to take into account all the overhead bits which are to be used for PH and GOBs, the formula, when applied with other values of q and k, yields a number of bits for PH and GOBs very far from 460.

Taking into account the outlined considerations, we suggest a new formula to be used in simulations, which allows the computation of the mean number of bits per MB for every value of p and k with a defined number of overhead bits per frame to code the PH and the GOBs:

$$MBIT = \frac{64.000 * k * q - 30 * NOV}{30 * NMBL}$$

with:

bitrate : q * 64 kbit/s (q = 1..30)
frame rate : 30 / k Hz (k = 1..4)

NMBL = number of macroblocks in a picture= | -> 396 for CSIF
| -> 99 for 1/4 CSIF

NOV = number of overhead bits reserved for each frame

MNBIT = mean number of available bits per Macroblock
For practicle reasons the nearest integer is taken.

Table I. and II. give some values of MBIT (for both the CSIF and 1/4 CSIF resolution) computed by the formula in doc. 396 and the newly given formula for different values of NOV. The values of NOV for 1/4 CSIF resolution was chosen smaller than those for CSIF resolution, as there are less GMBs in 1/4 CSIF. The right values of NOV must be defined in accordance with the number of GOBs in a frame, depending on the resolution (CSIF or 1/4 CSIF) and on the definition of the macroblock structure (12 GOBs for CSIF and 3 GOBs for 1/4 CSIF). Per frame the overhead bits are constant i.e. increasing the frame rate means that more overhead bits need to be transmitted.

	Full CIF	QCIF
PH	42 - 66	42 - 66
GOBH	32 - 48	32 - 48
nGOBs	12	3

Number overhead bits equals PH + nGOBs x GOBH i.e. CIF= 66 + 1248642 and QCIF = 66 + 348 = 210.

RELATION BITS PER MACROBLOCK AND CHANNEL RATE

	k = 1 (30 Hz)	k = 2 (15 Hz)	k = 3 (10 Hz)	k = 4 (7.5 Hz)	
q = 1	20	40	60	80	doc #396
64kbit/s	19	41	62	84	NOV = 210
q = 2	40	80	120	160	doc #396
128kbit/s	41	84	127	170	NOV = 210
q = 6	120	240	360	480	doc #396
384kbit/s	127	256	386	515	NOV = 210

TABLE I: Mean number of bits per MacroBlock using different formulas (1/4 CSIF resolution)

RELATION BITS PER MACROBLOCK AND CHANNEL RATE

	k = 1 (30 Hz)	k = 2 (15 Hz)	k = 3 (10 Hz)	k = 4 (7.5 Hz)	
q = 1	5	10	15	20	doc #396
	4	9	15	20	NOV = 642
q = 2	10	20	30	40	doc #396
	9	20	31	42	NOV = 642
q = 6	30	60	90	120	doc #396
	31	63	95	128	NOV = 642
q = 12	60	120	180	240	doc #396
	63	128	192	257	NOV = 642
q = 24	120	240	360	480	doc #396
	128	257	386	516	NOV = 642
q = 30	150	300	450	600	doc #396
	160	322	483	644	NOV = 642

TABLE II: Mean number of bits per MacroBlock using different formulas (CSIF resolution)

APPENDIX G

ACTION POINTS

1. Optimization of type 3 VLC
2. Buffer regulation
3. Impact of the modification at q x 64 kbit/s where q 1
4. QCIF simulation work
5. Up and down converters CIF <-- QCIF
6. Refresh mode: minimum stepsize of the quantizer
7. Scene change
8. Simulation at CIF and QCIF taking into account the exact number of bits as defined in the hardware specification document.

APPENDIX H

VLCS FOR P64 KBIT/S INITIAL COMPATIBILITY CHECK

This Appendix replaces document 449 mentioned on page 12 of the Flexible Hardware specification dated 10 March 1989. The modifications have already been incorporated here.

H.1 ;MACRO BLOCK ADDRESSING

Codes for Macroblock Addressing

1	1
2	011
3	010
4	0011
5	0010
6	0001 1
7	0001 0
8	0000 111
9	0000 110
10	0000 1011
11	0000 1010
12	0000 1001
13	0000 1000
14	0000 0111
15	0000 0110
16	0000 0101 11
17	0000 0101 10
18	0000 0101 01
19	0000 0101 00
20	0000 0100 11
21	0000 0100 10
22	0000 0100 011
23	0000 0100 010
24	0000 0100 001
25	0000 0100 000
26	0000 0011 111
27	0000 0011 110
28	0000 0011 101
29	0000 0011 100
30	0000 0011 011
31	0000 0011 010
32	0000 0011 001
33	0000 0011 000
Stuffing	0000 0001 111
Start code	0000 0000 0000 0001

H.2 ;CODES FOR MOTION VECTOR COMPONENT DIFFERENTIALS

Codes for Motion Vector Component Differentials

-16 & 16	0000 0011 001
-15 & 17	0000 0011 011
-14 & 18	0000 0011 101
-13 & 19	0000 0011 111
-12 & 20	0000 0100 001
-11 & 21	0000 0100 011
-10 & 22	0000 0100 11
-9 & 23	0000 0101 01
-8 & 24	0000 0101 11
-7 & 25	0000 0111
-6 & 26	0000 1001
-5 & 27	0000 1011
-4 & 28	0000 111
-3 & 29	0001 1
-2 & 30	0011
-1	011
0	1
1	010
2 & -30	0010
3 & -29	0001 0
4 & -28	0000 110
5 & -27	0000 1010
6 & -26	0000 1000
7 & -25	0000 0110
8 & -24	0000 0101 10
9 & -23	0000 0101 00
10 & -22	0000 0100 10
11 & -21	0000 0100 010
12 & -20	0000 0100 000
13 & -19	0000 0011 110
14 & -18	0000 0011 100
15 & -17	0000 0011 010

H.3 ;BLOCK PATTERNS

Codes for Coded Block Pattern

1	1010 0	33	1101 011
2	1011 0	34	1101 111
3	1100 10	35	1110 0011
4	0010	36	1100 01
5	1101 000	37	1111 0001
6	1101 100	38	1111 0011
7	1110 0000	39	1111 1110 1
8	0011	40	0111 1
9	1101 001	41	1110 0111
10	1101 101	42	1110 1011
11	1110 0001	43	1110 1111
12	0110 0	44	1000 1
13	1110 0100	45	1111 0101
14	1110 1000	46	1111 1001
15	1110 1100	47	1111 1101
16	0100	48	0110 1
17	1101 010	49	1110 0101
18	1101 110	50	1110 1001
19	1110 0010	51	1110 1101
20	0111 0	52	1001 0
21	1110 0110	53	1111 0110
22	1110 1010	54	1111 1010
23	1110 1110	55	1111 1111 0
24	1100 00	56	1001 1
25	1111 0000	57	1111 0111
26	1111 0010	58	1111 1011
27	1111 11100	59	1111 1111 1
28	1000 0	60	000
29	1111 0100	61	1010 1
30	1111 1000	62	1011 1
31	1111 1100	63	1100 11
32	0101		

H.4 ;CODES FOR COEFFICIENT DATA

The most commonly occuring combinations of zero-run and the following value are encoded with variable length codes as listed in table 5. End of Block (EOB) is in this set. Because CBP indicates those blocks with no coefficient data, EOB cannot occur as the first coefficient. Hence EOB can be removed from the VLC table for the first coefficient.

The remaining combinations of (RUN, LEVEL) are encoded with a 20 bit word consisting of 6 bits ESCAPE, 6 bits RUN and 8 bits LEVEL.

EOB	10
ESCAPE	0000 01

RUN is a 6 bit fixed length code

0	0000 00
1	0000 01
2	0000 10
.	.
.	.
63	1111 11

LEVEL is an 8 bit fixed length code

-127	1000 0001	
.	.	
.	.	
-2	1111 1110	
-1	1111 1111	
0	0000 0000	FORBIDDEN
1	0000 0001	
2	0000 0010	
.	.	
127	0111 1111	

The last bit 's' denotes the sign of the level, '0' for +ve and '1' for -ve.

RUN	| LEVEL|	CODE	
EOB		10	
0	1	1s	IF FIRST COEFFICIENT
0	1	11s	NOT FIRST COEFFICIENT
0	2	0100 s	

VLCS FOR P64 KBIT/S INITIAL COMPATIBILITY CHECK

0	3	0010 1s
0	4	0000 110s
0	5	0010 0110 s
0	6	0010 0001 s
0	7	0000 0010 10s
0	8	0000 0001 1101 s
0	9	0000 0001 1000 s
0	10	0000 0001 0011 s
0	11	0000 0001 0000 s
0	12	0000 0000 1101 0s
0	13	0000 0000 1100 1s
0	14	0000 0000 1100 0s
0	15	0000 0000 1011 1s
1	1	011s
1	2	0001 10s
1	3	0010 0101 s
1	4	0000 0011 00s
1	5	0000 0001 1011 s
1	6	0000 0000 1011 0s
1	7	0000 0000 1010 1s
2	1	0101 s
2	2	0000 100s
2	3	0000 0010 11s
2	4	0000 0001 0100 s
2	5	0000 0000 1010 0s
3	1	0011 1s
3	2	0010 0100 s
3	3	0000 0001 1100 s
3	4	0000 0000 1001 1s
4	1	0011 0s
4	2	0000 0011 11s
4	3	0000 0001 0010 s
5	1	0001 11s
5	2	0000 0010 01s
5	3	0000 0000 1001 0s
6	1	0001 01s
6	2	0000 0001 1110 s
7	1	0001 00s
7	2	0000 0001 0101 s
8	1	0000 111s
8	2	0000 0001 0001 s
9	1	0000 101s
9	2	0000 0000 1000 1s

VLCS FOR P64 KBIT/S INITIAL COMPATIBILITY CHECK

10	1	0010 0111 s
10	2	0000 0000 1000 0s
11	1	0010 0011 s
12	1	0010 0010 s
13	1	0010 0000 s
14	1	0000 0011 10s
15	1	0000 0011 01s
16	1	0000 0010 00s
17	1	0000 0001 1111 s
18	1	0000 0001 1010 s
19	1	0000 0001 1001 s
20	1	0000 0001 0111 s
21	1	0000 0001 0110 s
22	1	0000 0000 1111 1s
23	1	0000 0000 1111 0s
24	1	0000 0000 1110 1s
25	1	0000 0000 1110 0s
26	1	0000 0000 1101 1s