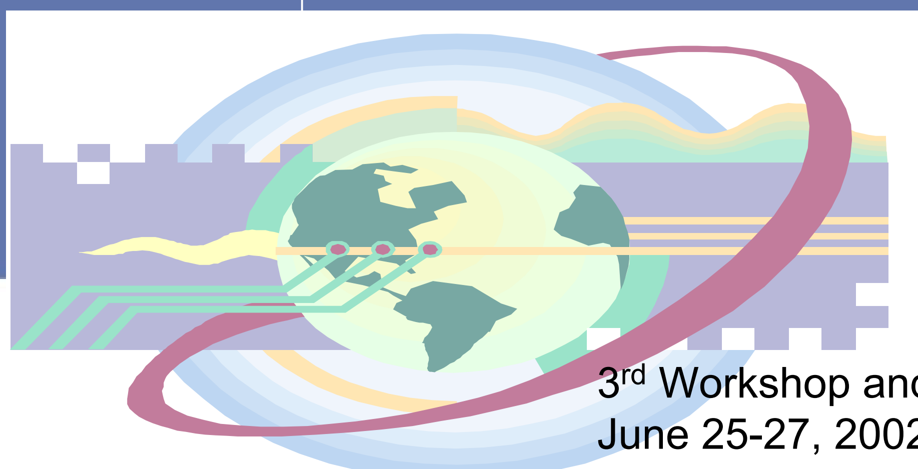
MPEG-4 AVC (a.k.a. JVT)
An Overview



Ajay Luthra
Motorola
Broadband Communications Sector

3rd Workshop and Exhibition on MPEG-4 (WEMP4)
June 25-27, 2002
San Jose, California USA

MOTOROLA

June 25, 2002

intelligence everywhere™

Overview of presentation

- Background
- Intro to MPEG-4 AVC (Part 10)
- Comparison with MPEG-2, MPEG-4 ASP
- Demonstrations

Overview of presentation

- **Standard is still evolving**
- **Some of the information presented here may not be current, accurate or complete**
- **Readers should read the standard specification for accuracy and complete description**

MPEG-4

- **Coding of Audiovisual Objects and Multimedia contents**
  - Video, Graphics, Animations, Text, Audio, Speech
- **Provides standardized technology and framework for the compression and distribution of multiple media types for various applications**

MPEG-4 Parts

- **Systems : ISO/IEC 14496-1**
- ***Visual : ISO/IEC 14496-2 / Part 2***
- **Audio : ISO/IEC 14496-3**
- **Conformance : ISO/IEC 14496-4**
- **Reference Software : ISO/IEC 14496-5**
- **DMIF (Delivery Multimedia Integration Framework) : ISO/IEC 14496-6**
- **Software for MPEG-4 Tools : ISO/IEC 14496-7**
- **MPEG-4 on IP Framework : ISO/IEC 14496-8**
- ***Visual: AVC (Advanced Video Coding), ISO/IEC 14496-10 / Part 10***
  - *MPEG and ITU Joint Video Team (JVT)*
- **Transport of MPEG-4 AVC on MPEG-2 TS: ISO/IEC 13818 -1 /2000 Amendment 3**

MPEG-4 Parts

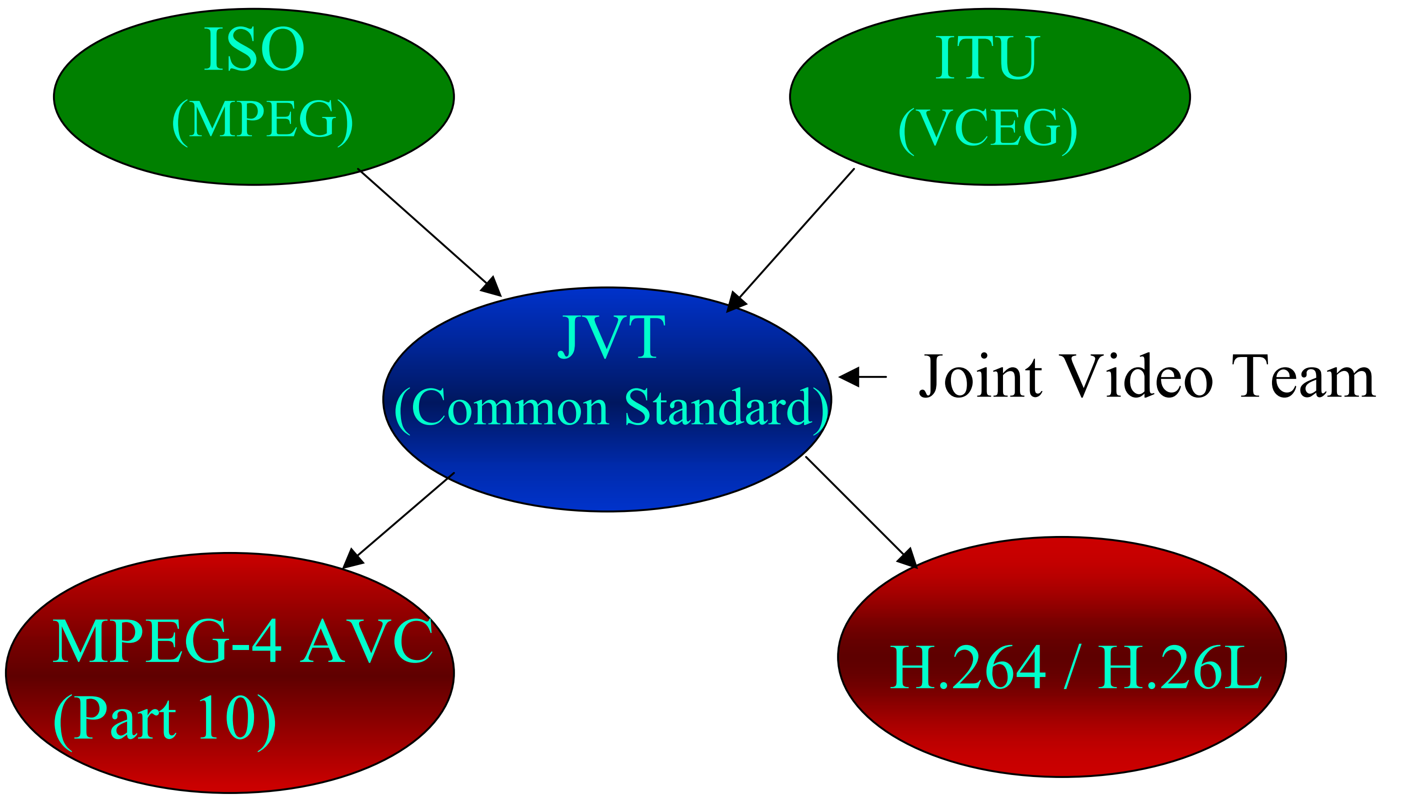
- **Modular**
  - Parts - Visual, Audio, System etc – can be selected and used individually

MPEG-4 Visual Parts

- **Visual - Part 2 (ISO/IEC 14496-2)**
  - – Video
    - Coding of Natural Video
  - – SNHC (Synthetic-Natural Hybrid Coding)
    - Facial & Body Animation
    - Graphics Coding
  - – Texture Coding
  - – Sprite Coding
- **Visual – Part 10 (ISO/IEC 14496 –10)**
  - – AVC (Advanced Video Coding)
  - – JVT (Joint Video Team), ISO + ITU-T
  - – Focused solely on coding of Natural Video
  - – Very high coding efficiency

MPEG-4 Advanced Video Coding (AVC)

MPEG-4 AVC


ISO
(MPEG)
ITU
(VCEG)
JVT
(Common Standard)
Joint Video Team
MPEG-4 AVC
(Part 10)
H.264 / H.26L

MPEG-4 Advanced Video Coding (AVC)

- **Working Draft 2 - January 2002**
- **Committee Draft (CD) - May 2002**
- **Final CD - July 2002**
- **FDIS (Final Draft International Standard) – December 2002**

Video Coding Standards

- **MPEG-2**
  - State of the Art, 1994
- **MPEG-4 Video, Part 2**
  - ASP (Advanced Simple Profile)
  - State of the Art, 1999
  - ~ 1.5x coding gain (on average)
- **MPEG-4 AVC, Part 10**
  - State of the Art, 2002
  - More than 2x coding gain over MPEG-2 (on average)
  - Final Draft Standard in Dec 2002

MPEG-4 AVC

Working Draft 2, Rev 8 based description

MPEG-4 AVC

- NAL (Network Adaptation Layer)
- VCL (Video Coding Layer)

Video Hierarchy

- **Sequence, consisting of**
- **Pictures, consisting of**
- **Slices, consisting of**
- **Macroblocks, consisting of**
- **Blocks, consisting of**
- **Pixels / Pels**

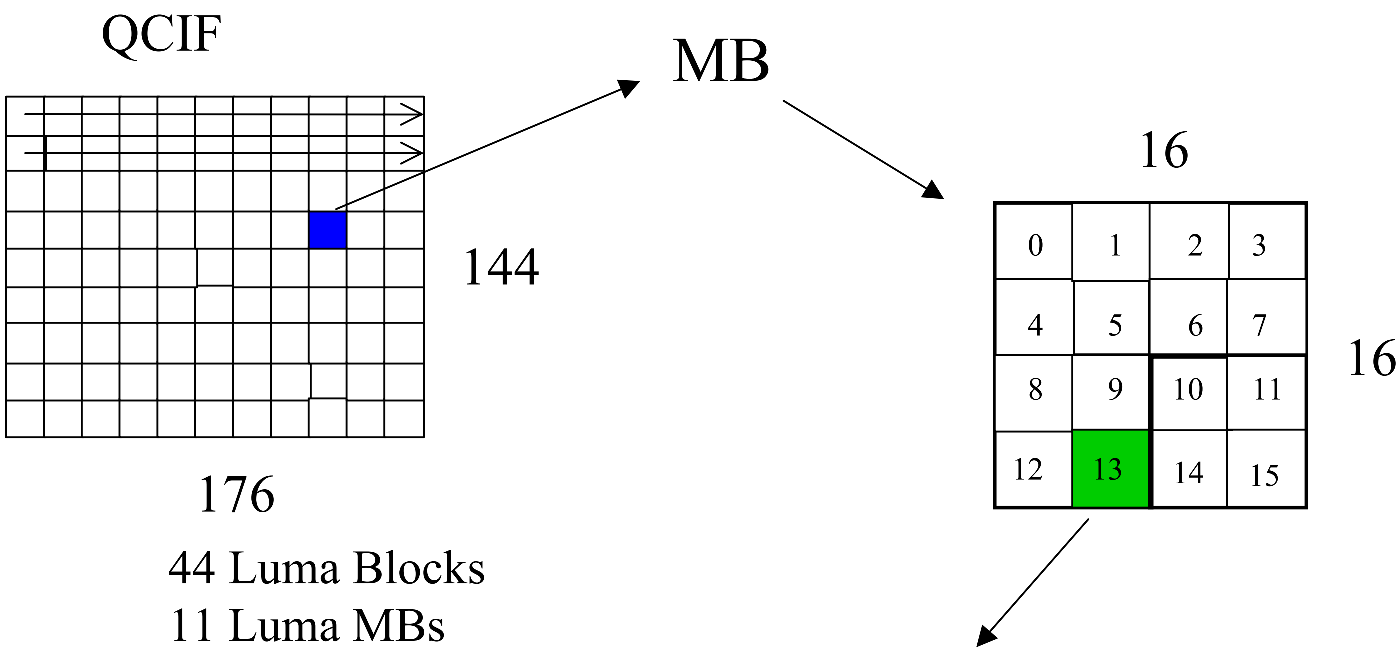
NAL

VCL

Video Hierarchy - Picture

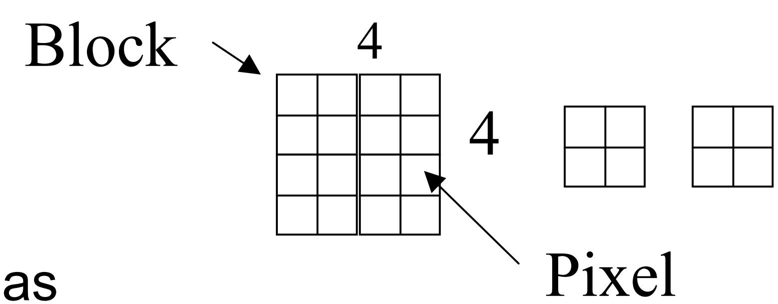
- 4:4:4
- 4:2:2
- **4:2:0**
- **Progressively Scanned**
  - Frame
- **Interlaced Scanned**
  - 1 Frame consists of 2 fields

Video Hierarchy - Picture, MB, Block



No Frame/Field distinction
at MB and Block level (so far)

➡ Field Picture has field MBs and Blocks, Frame Picture has Frame MBs and Blocks

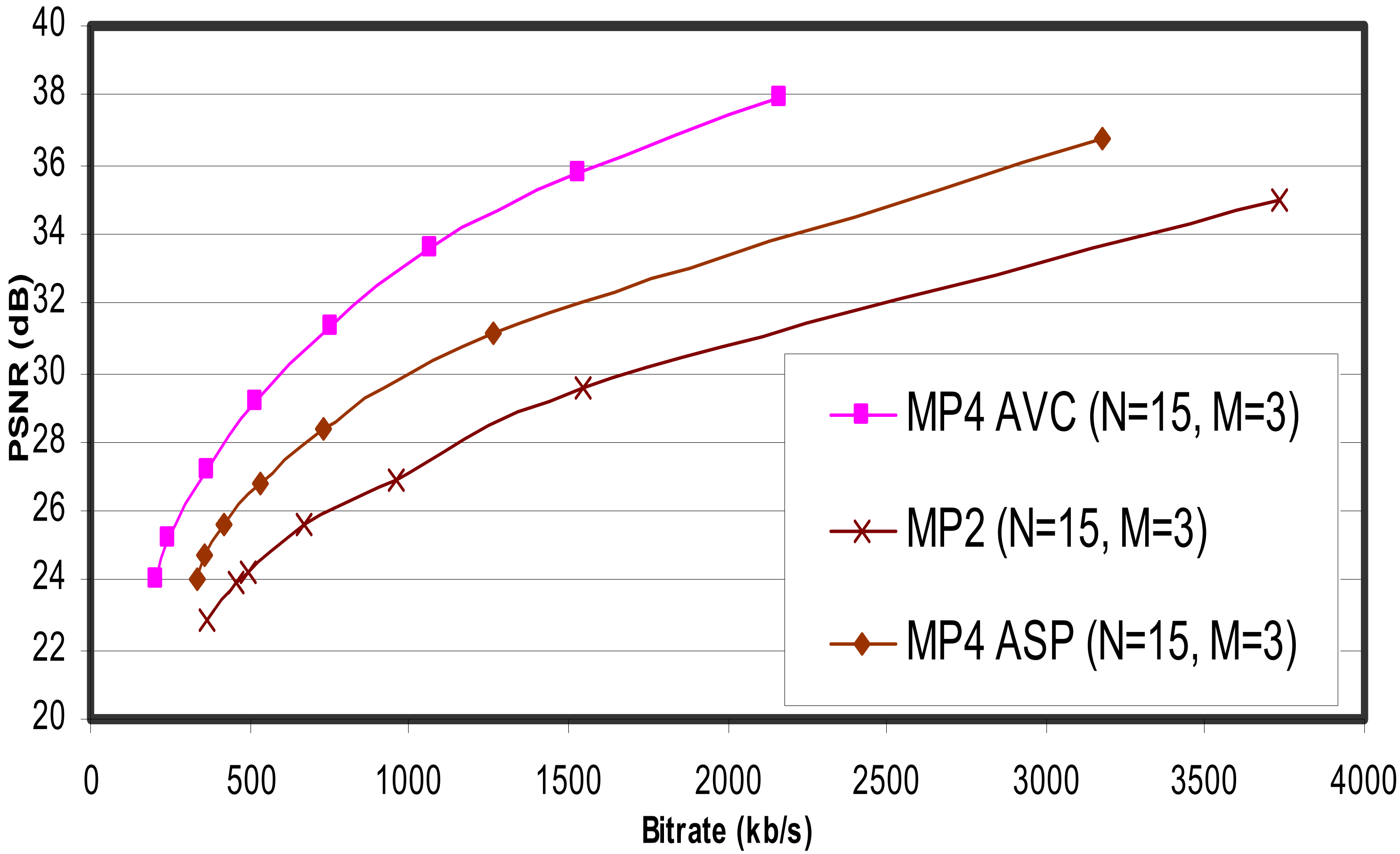


MPEG-4 AVC (WD2 Rev 8)

- **Motion compensated DCT-like transform**
- **I, P, B, SP, SI**
- **Multiple reference frames**
- **Variable block size - 7 Block Patterns**
  - 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4
- **Up to 16 MVs per MB**
- **4x4 Integer DCT like transform**
- **Improved Intra Prediction**
- **Deblocking filter**
- **Interlace Coding Tools**
- **Universal VLC**
- **Arithmetic Coding (CABAC)**
- **1/4th Pel, 1/8th Pel Interpolation**
- **…**

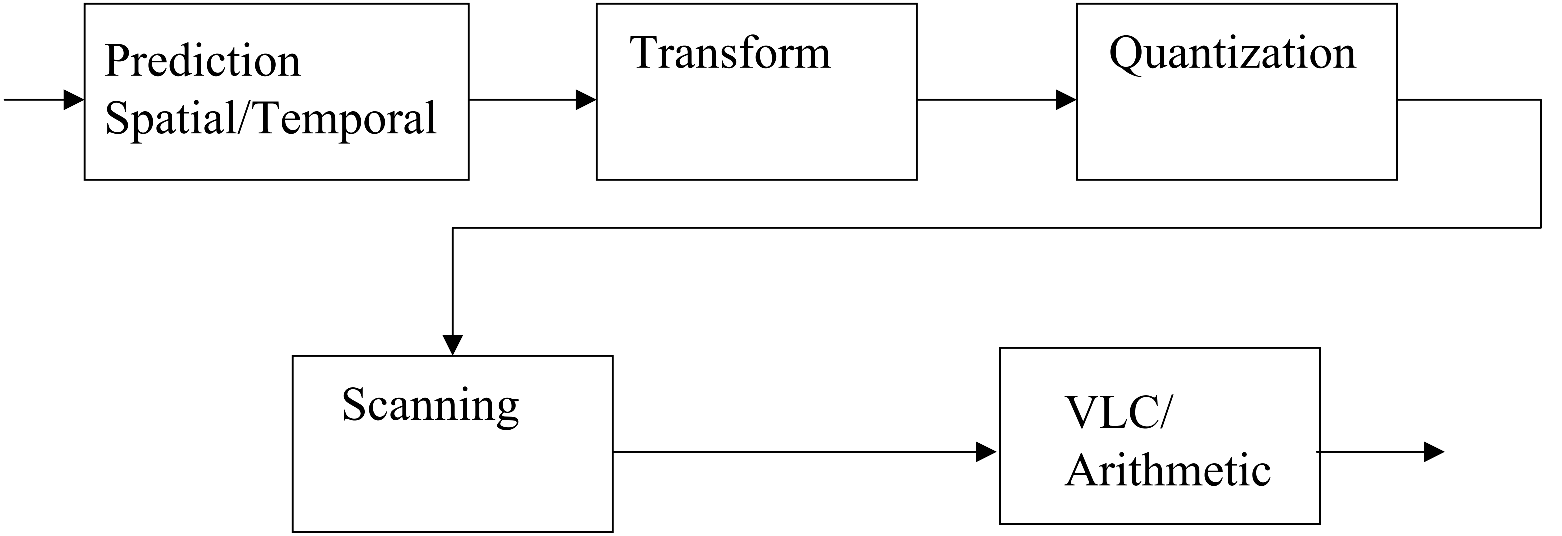
MPEG-2, MPEG-4 ASP, MPEG-4 AVC (WD2)

Mobile & Calendar (CIF)



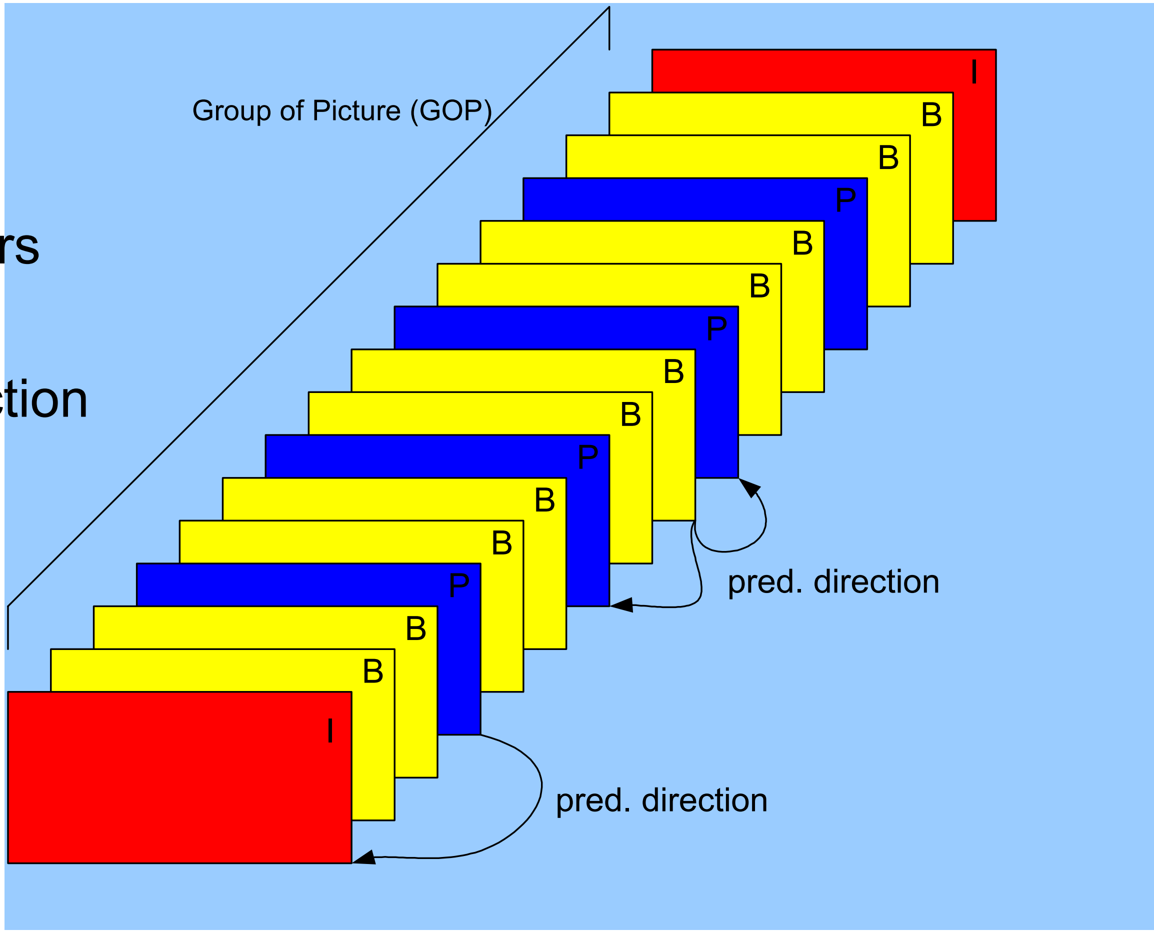
Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another

Video Encoding Block Diagram



Basic Frame Types

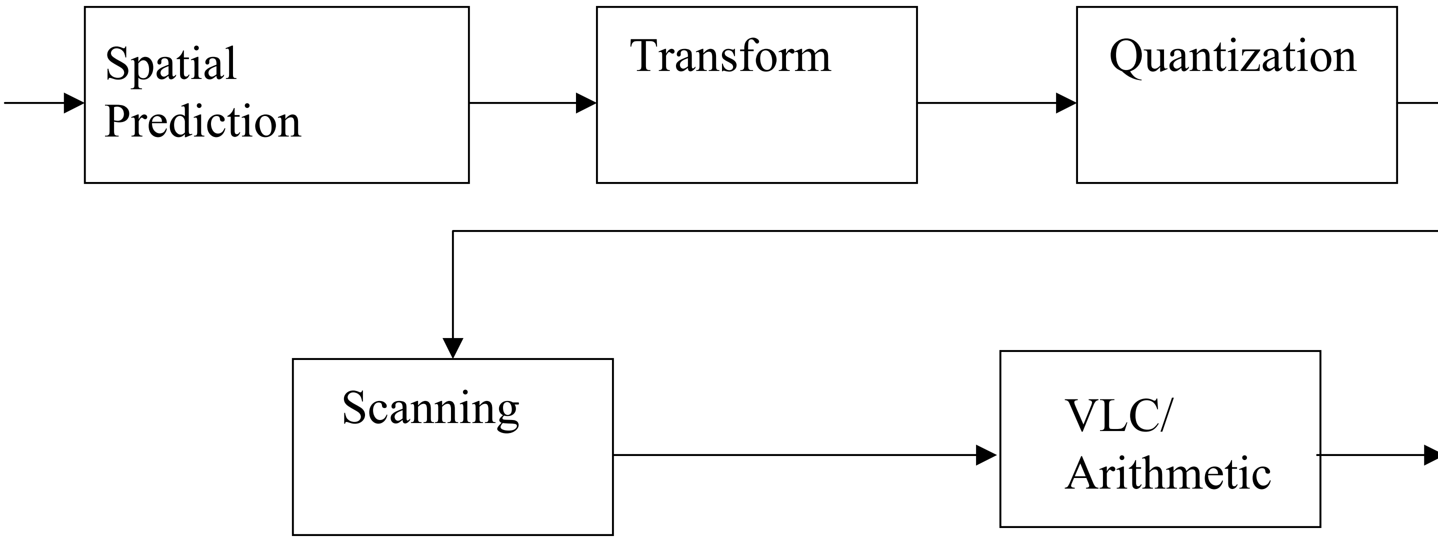
- I: no reference to others
- P: forward prediction.
- B: bi-directional prediction
  - Disposable
  - Non-disposable



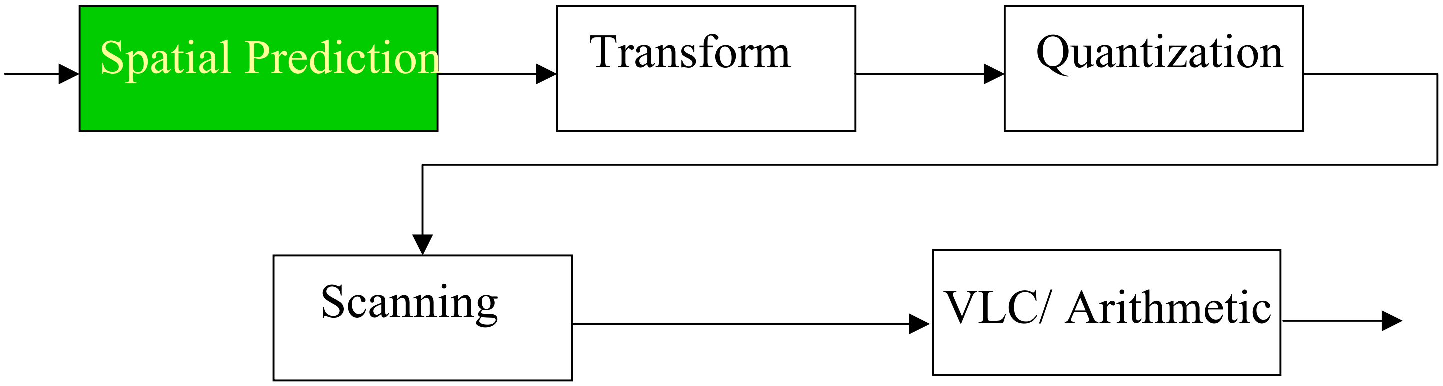
Intra (I) Frame

- Spatial only prediction (No Temporal prediction)
- Segmented into Macro Blocks (MBs) each of size 16x16 pixels
- MB is segmented into 16 blocks each of size 4x4 pixels

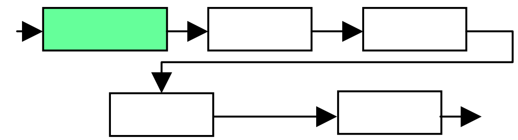
Intra (I) frame coding



I-frame coding: Prediction (Spatial)

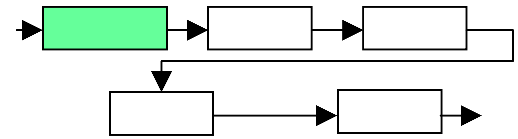


I-frame coding: Prediction (Luma)

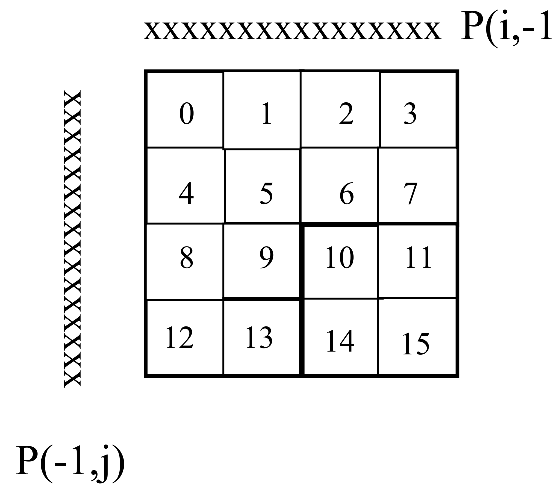


- **MB (16x16) based**
- **Block (4x4) based**

I-frame coding: Prediction (Luma)



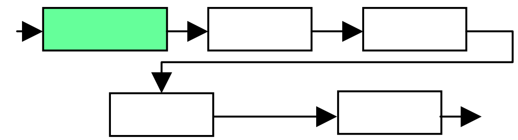
- **MB (16x16) based**
  - Mode 0 (Vertical)
    - Pred (I,j) = P(i,-1)
      I,j = 0, … 15
  - Mode 1 (Horizontal)
    - Pred (I,j) = P(-1,j)
      I,j = 0, … 15
  - Mode 2 (DC)
    - Pred(i,j) = $(\sum_{i=0}^{15}(P(-1,i)+P(i,-1)))/32$
      i,j=0..15,
  - Mode 3 (Plane)



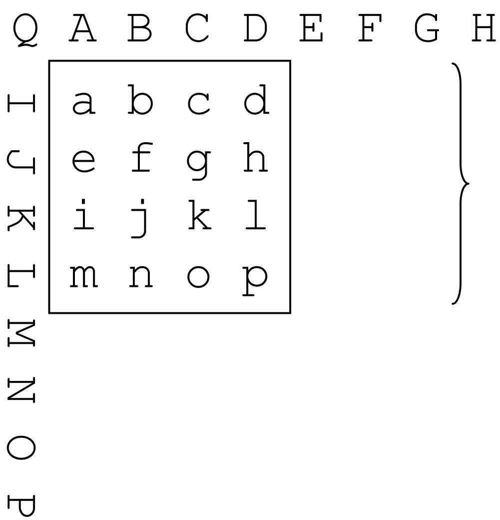
I-frame coding: Prediction (Luma)

- **Signaling the mode**
  - Trio
    - I-mode: 0,1,2,3
    - Coded Block Pattern for Chroma
      (no chroma, non-zero 2x2 but no chroma AC, 2x2+chroma)
    - AC Coeff present flag: 0 => no AC in the 16x16

I-frame Coding: Prediction (Luma)



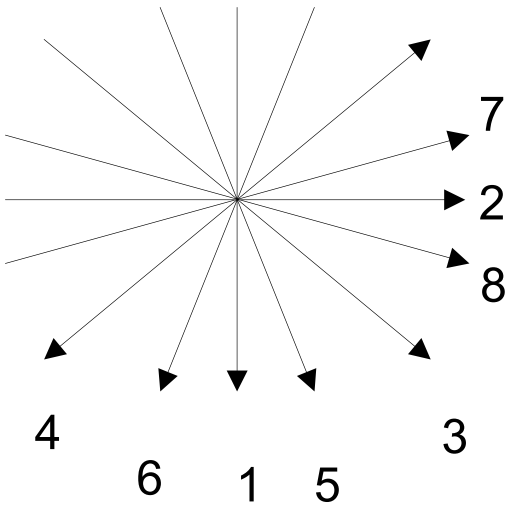
- **Block (4x4) based**



- **Predict (spatially) a,b … p from A,B … Q**

I-frame coding: Prediction (Luma)

- **Block (4x4) based**



Q	A	B	C	D	E	F	G	H
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				
M								
N								
O								
P								

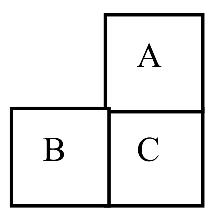
9 Prediction Modes (directions)

- Mode 0 (DC Prediction)
  - Predict all pixels from (A+B+C+D+I+J+K+L)/8
- Mode 1 (Vertical Prediction)
  - a,e,i,m Predicted from A; b,f,j,n from B …
- Mode 2 (Horizontal)
  - a,b,c,d from I; e,f,g,h from J …
- Mode 3 (Diagonal Down-Right)
  - a,f,k,p from (A+2Q+I+2)/4
  - e,j,o from (Q+2I+j+2)/4 …
- …
- Mode 8

I-frame coding: Prediction (Luma)

- **Block (4x4) based**
  - Flagging the prediction mode of each block will need too many bits
  - To save bits, correlation among adjacent blocks is exploited
  - For example, if A and B used mode 2, then C will have the following prediction modes in decreasing probability :

    2 8 7 1 0 6 4 3 5



I-frame coding: Prediction (Luma)

- **Block (4x4) based - Prediction Correlation**

Index	B/A	Outside	0	1	2	3
0a	Outside	0--------	01-------	10-------	---------	---------
1a	0	02-------	021648573	125630487	021876543	021358647
2a	1	---------	102654387	162530487	120657483	102536487
3a	2	20-------	280174365	217683504	287106435	281035764
4a	3	---------	201385476	125368470	208137546	325814670
5a	4	---------	201467835	162045873	204178635	420615837
6a	5	---------	015263847	152638407	201584673	531286407
7a	6	---------	016247583	160245738	206147853	160245837
8a	7	---------	270148635	217608543	278105463	270154863
9a	8	---------	280173456	127834560	287104365	283510764

I-frame coding: Prediction (Luma)

- **Block (4x4) based**
  - Instead of sending information about mode directly, the index number is sent, e.g. number 1 will signify that mode 8 out of

    2 8 7 1 0 6 4 3 5 was selected when A and B used the mode 2
  - To further compress the information, probability numbers of two adjacent blocks are sent together (see table on the next slide) in the following order

0	0	2	2
1	1	3	3
4	4	6	6
5	5	7	7

I-frame Coding: Prediction (Luma)

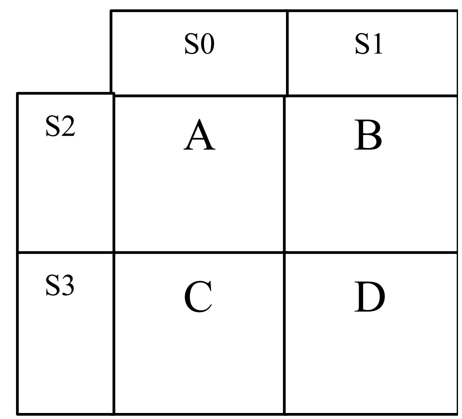
Code	P0,P1		Code	P0,P1		Code	P0,P1		Code	P0,P1
0	0,0		21	2,3		41	2,6		61	6,5
1	0,1		22	3,2		42	6,2		62	4,7
2	1,0		23	1,5		43	3,5		63	7,4
3	1,1		24	5,1		44	5,3		64	3,8
4	0,2		25	2,4		45	1,8		65	8,3
5	2,0		26	4,2		46	8,1		66	4,8
6	0,3		27	3,3		47	2,7		67	8,4
7	3,0		28	0,7		48	7,2		68	5,7
8	1,2		29	7,0		49	4,5		69	7,5
9	2,1		30	1,6		50	5,4		70	6,6
10	0,4		31	6,1		51	3,6		71	6,7
11	4,0		32	2,5		52	6,3		72	6,7
12	3,1		33	5,2		53	2,8		73	5,8
13	1,3		34	3,4		54	8,2		74	8,5
14	0,5		35	4,3		55	4,6		75	6,8
15	5,0		36	0,8		56	6,4		76	8,6
16	2,2		37	8,0		57	5,5		77	7,7
17	1,4		38	1,7		58	3,7		78	7,8
18	4,1		39	7,1		59	7,3		79	8,7
19	0,6		40	4,4		60	5,6		80	8,8
20	6,0									

I-frame coding: Prediction (Chroma)

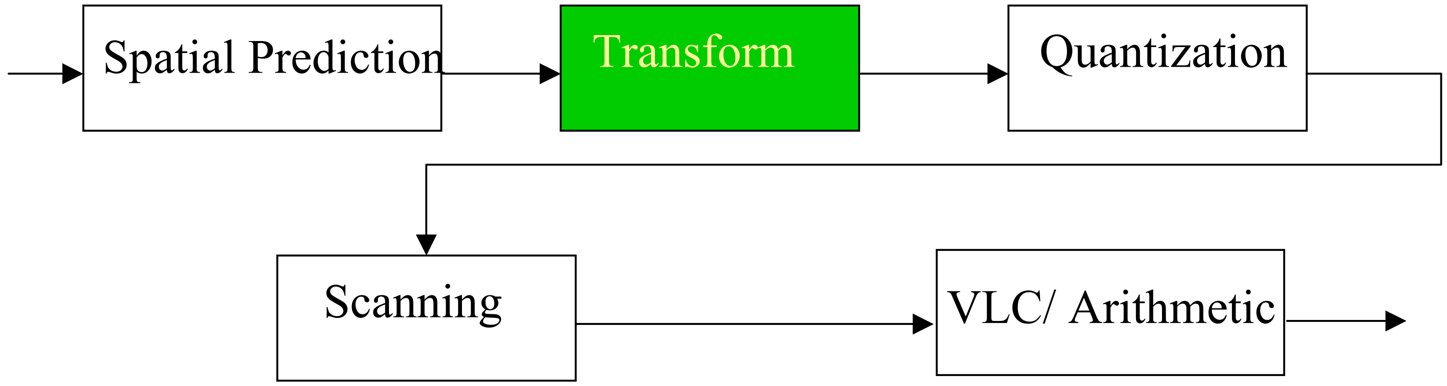
- **Only one mode**
  - 8x8 Chroma macroblock (4:2:0) consisting of
    4, 4x4 blocks ABCD
  - 4 Prediction cases depending upon whether S0, S1, S2 or S3 are inside or outside

    e.g., if all inside
    - A= (S0+S2+4)/8
    - B= (S1+2)/4
    - C= (S3+2)/4
    - D= (S1+S3+4)/8

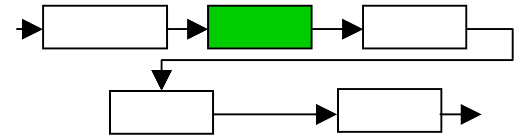
    S0,S1,S2 and S3 are the sums of
    4 neighboring samples



I-frame coding: Transform



I-frame coding: Transform



- **First step for Residue coding**
  - Block (4x4) based

$$Y = AXA^T = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} a & b & a & c \\ a & c & -a & -b \\ a & -c & -a & b \\ a & -b & a & -c \end{bmatrix}$$

DCT Transform

$$a = 1/2$$

$$b = \sqrt{1/2} \times \cos(p/8) \quad = 0.653281482438\ldots$$

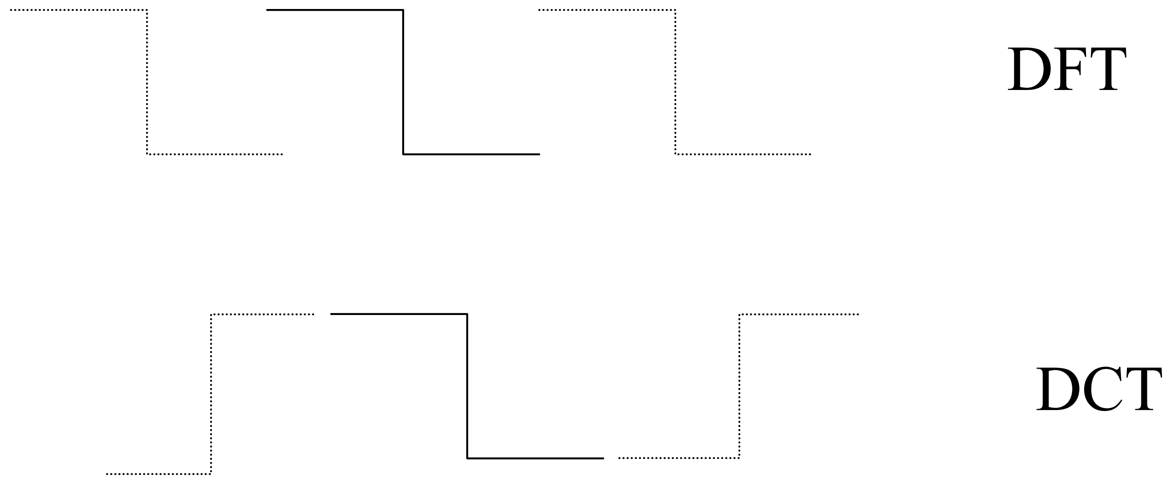
$$c = \sqrt{1/2} \times \cos(3p/8) \quad = 0.382683432365\ldots$$

I-frame coding: Transform

- **As coefficients get quantized after transform, why to maintain such a long precision**
- **Why DCT?**
  - Compact coefficients
  - Orthogonal bases

DCT Transform

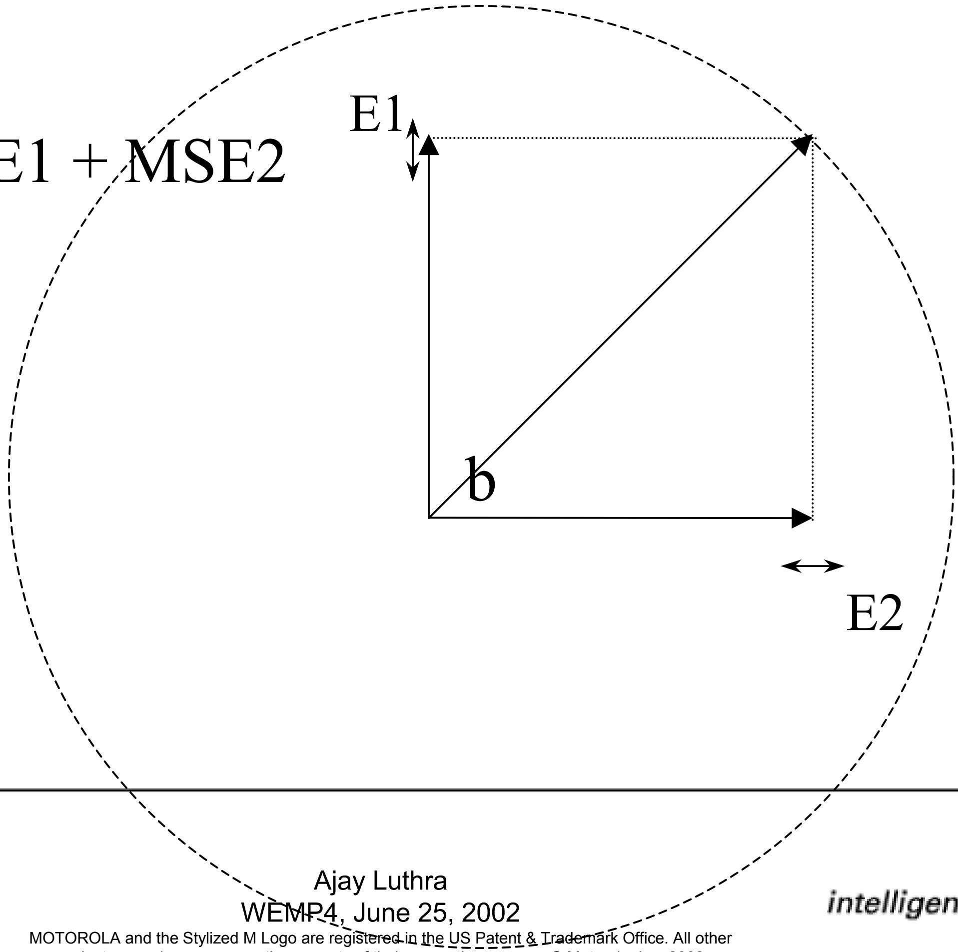
- **Compact Coefficients**



DCT Transform

- **Orthogonal Bases**

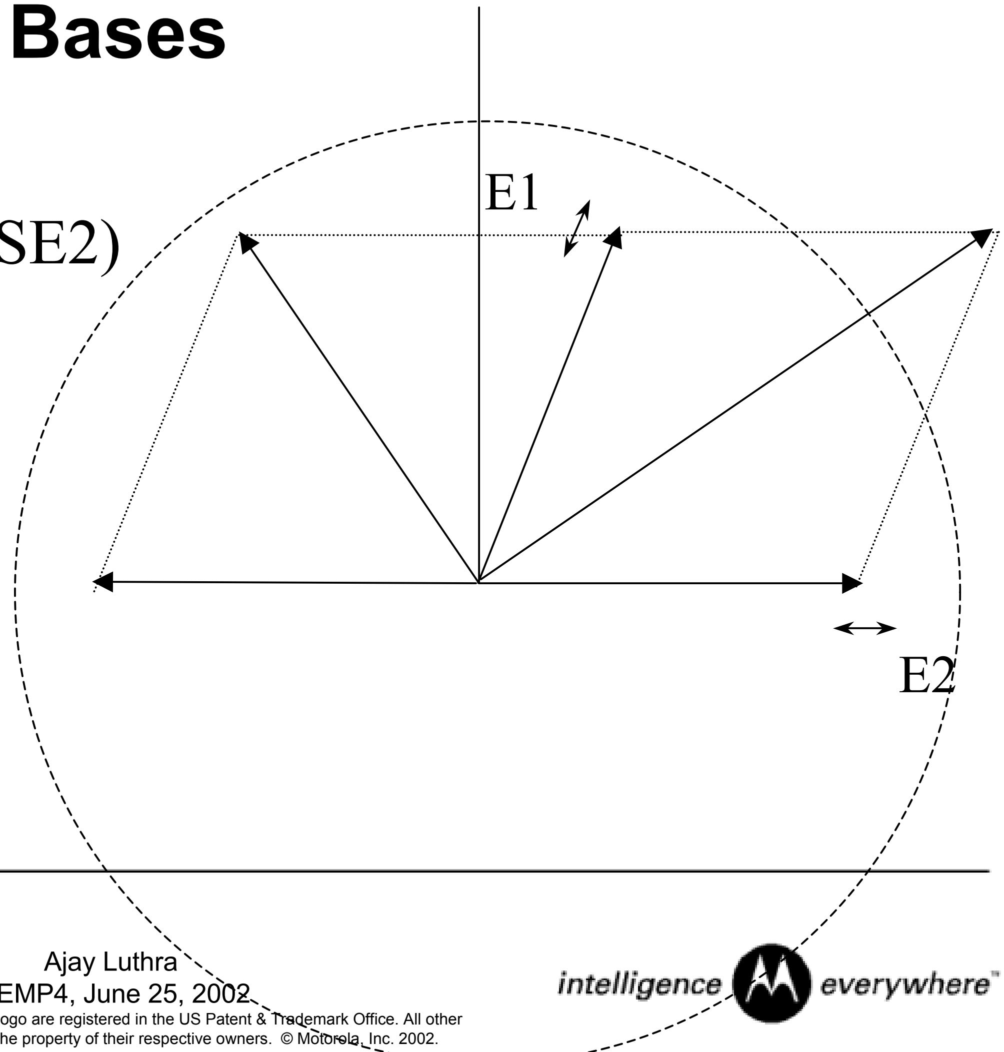
MSE = MSE1 + MSE2



DCT Transform

- **Non Orthogonal Bases**

MSE = a*(MSE1 + MSE2)



I-frame coding: Transform

- **First step for Residue coding**
  - Block (4x4) based

$$Y = AXA^T = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} a & b & a & c \\ a & c & -a & -b \\ a & -c & -a & b \\ a & -b & a & -c \end{bmatrix}$$

DCT Transform

$$a = 1/2$$

$$b = \sqrt{1/2} \times \cos(p/8) \quad = 0.653281482438\ldots$$

$$c = \sqrt{1/2} \times \cos(3p/8) \quad = 0.382683432365\ldots$$

I-frame coding: Transform

$$Y = BCXC^T B =$$

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix}\begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & d \\ 1 & d & -1 & -1 \\ 1 & -d & -1 & 1 \\ 1 & -1 & 1 & -d \end{bmatrix}\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$$

$$d = c/b$$

$\otimes$ ⇒ Element-by- Element multiplication

$$Y = \left(CXC^T\right) \otimes E =$$

$$\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix}\begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & d \\ 1 & d & -1 & -1 \\ 1 & -d & -1 & 1 \\ 1 & -1 & 1 & -d \end{bmatrix}\right) \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

I-frame coding: Transform

$$Y \otimes Q = CXC^T \otimes E \otimes Q$$

$$= CXC^T \otimes Q'$$

- C is now the new Transform
- Element d in Matrix C = $\sqrt{2} - 1$
  = 0.41421356237 …
- Only 1 non-trivial element
- Round it to 7/16, 3/8 or ½
- Simplest choice, ½ , was picked
- Orthogonality of the bases vectors?

I-frame coding: Transform

- To maintain orthogonality of original A Matrix, b now needs to be modified:

$$b = \sqrt{\frac{1}{2(1+d^2)}} = \sqrt{2/5}$$

$$Y = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} & -\frac{1}{2} & -1 \\ 1 & -1 & -1 & 1 \\ \frac{1}{2} & -1 & 1 & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & \frac{1}{2} \\ 1 & \frac{1}{2} & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 1 & -\frac{1}{2} \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

I-frame coding: Transform

- To avoid truncation error due to dividing by 2, every other row of the first matrix and every other row of the second (transposed) matrix is multiplied by 2

$$Y = \left(CXC^T\right) \otimes E_{forw} =$$

$$\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

- This C replaces DCT and is the *Transform* matrix and is used in all (except 16x16 Intra DC) the 4x4 Block transforms

I-frame coding: Transform (Luma DC)

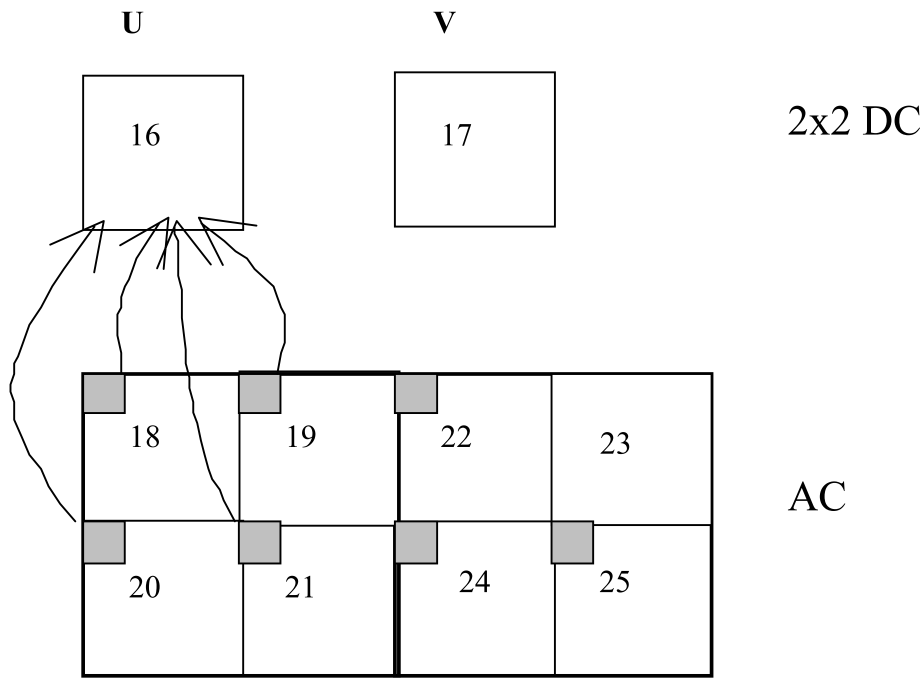
- **Intra 16x16**
  - 16 DC coefficients of 16 blocks are further transformed using Hadamard transform

$$Y_D = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_{D00} & x_{D01} & x_{D02} & x_{D03} \\ x_{D10} & x_{D11} & x_{D12} & x_{D13} \\ x_{D20} & x_{D21} & x_{D22} & x_{D23} \\ x_{D30} & x_{D31} & x_{D32} & x_{D33} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \right) // 2$$

// => rounding to the nearest integer

I-frame coding: Transform (Chroma DC)

- **8x8 MB – 4, 4x4 blocks for each chroma component**



I-frame coding: Transform (Chroma DC)

- **Chroma DC Transform**

$$Y = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} DC_{00} & DC_{01} \\ DC_{10} & DC_{11} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

I-frame coding:Transform (Chroma)

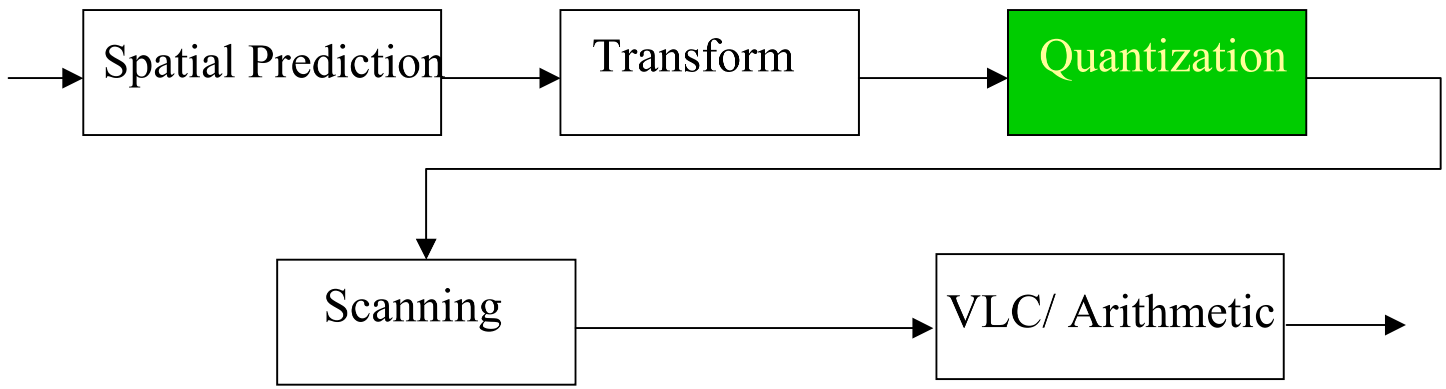
- **Coded Block Pattern (CBP) Information for Chroma**
  - Pattern to indicate which blocks in 8x8 Chroma MB have coeeficients
  - Flag called nc
    - nc = 0; no chrominance coefficients at all
    - nc = 1; non-zero 2x2 transform coefficients and all AC = 0
    - nc = 2; non-zero 2x2 and at least 1 non-zero AC coeff.

I-frame coding: Prediction (Luma)

- **Signaling the mode**
  - Trio
    - I-mode: 0,1,2,3
    - Coded Block Pattern for Chroma

      (no chroma, non-zero 2x2 but no chroma AC, 2x2+chroma)
    - AC Coeff present flag: 0 => no AC in the 16x16

I-frame coding: Quantization



I-frame coding: Quantization (WD2 Rev 8)

- Coefficients except DC Luma and Chroma (see note below)

$$Y_Q(i,j) = \left[Y(i,j) \cdot Q((QP+12)\%6, i, j) + f \right] / 2^{15+(QP+12)/6}, \quad i,j = 0,\ldots,3$$

where,

- a % b implies remainder of a divided by b (both positive integers)
- $|f| = \begin{cases} (2 \cdot 2^{15+(QP+12)/6})/6 \\ \end{cases}$
- Q(m,i,j) = $M_{m,0}$ for (i,j) = {(0,0),(0,2),(2,0),(2,2)}, Q(m,i,j) = $M_{m,1}$ for (i,j) = {(1,1),(1,3),(3,1),(3,3)}, Q(m,i,j) = $M_{m,2}$ otherwise;

$$M = \begin{bmatrix} 13107 & 5243 & 8066 \\ 11916 & 4660 & 7490 \\ 10082 & 4194 & 6554 \\ 9362 & 3647 & 5825 \\ 8192 & 3355 & 5243 \\ 7282 & 2893 & 4559 \end{bmatrix}$$

Note: In CD it was changed: QP+12 was replaced by new QP with appropriate adjustment to the QP range.

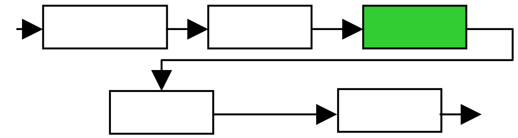
I-frame coding: Quantization

- $Q(m,i,j) = M_{m,0}$ for $(i,j) = \{(0,0),(0,2),(2,0),(2,2)\}$, $Q(m,i,j) = M_{m,1}$ for $(i,j) = \{(1,1),(1,3),(3,1),(3,3)\}$, $Q(m,i,j) = M_{m,2}$ otherwise; where

$$M = \begin{bmatrix} 13107 & 5243 & 8066 \\ 11916 & 4660 & 7490 \\ 10082 & 4194 & 6554 \\ 9362 & 3647 & 5825 \\ 8192 & 3355 & 5243 \\ 7282 & 2893 & 4559 \end{bmatrix}$$

- $$Q(0,i,j) = \begin{pmatrix} 13107 & 8066 & 13107 & 8066 \\ 8066 & 5243 & 8066 & 5243 \\ 13107 & 8066 & 13107 & 8066 \\ 8066 & 5243 & 8066 & 5243 \end{pmatrix}$$

I-frame coding: Quantization



- $$Q(0,i,j) = \begin{pmatrix} 13107 & 8066 & 13107 & 8066 \\ 8066 & 5243 & 8066 & 5243 \\ 13107 & 8066 & 13107 & 8066 \\ 8066 & 5243 & 8066 & 5243 \end{pmatrix}$$

$$Y = \left(CXC^T\right) \otimes E_{forw} =$$

$$\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

$a = 1/2; b = 0.65328$

I-frame coding: Quantization (WD2 Rev8)

- DC Luma

$$Y_Q(i,j) = [Y(i,j)\cdot Q((QP+12)\%6,0,0) + 2f\,]/2^{16+(QP+12)/6}, \quad i,j = 0,\ldots,3$$

- DC Chroma

$$Y_Q(i,j) = [Y(i,j)\cdot Q((QP+12)\%6,0,0) + 2f\,]/2^{16+(QP+12)/6}, \quad i,j = 0,\ldots,3$$

In CD it was changed: QP+12 was replaced by new QP with appropriate adjustment to the QP range.

I-frame coding: Quantization

- **Change in Quantization**
  - $QP_Y = (QP_Y + delta\_qp + 52) \% 52; \quad -26 < delta\_qp < 25$

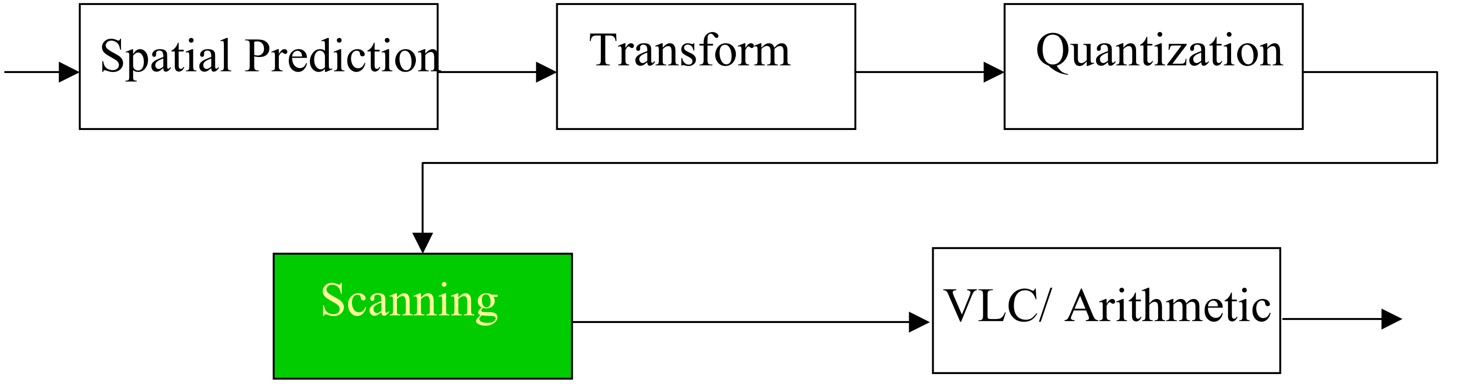
I-frame coding: Quantization (WD2Rev8)

- **Chroma Quantization**
  - QPchroma = QPluma if QPluma < 17
  - QPluma: 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

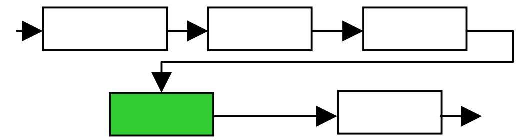
    QPchroma: 17 17 18 19 20 20 21 22 22 23 23 24 24 25 25 25 26 26 26 27 27 27 27

Note: Qp range was changed in CD

I-frame coding: Coefficient Scan

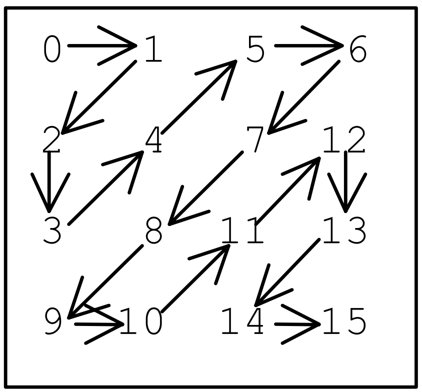


I-frame coding: Coefficient Scanning (WD2Rev8)

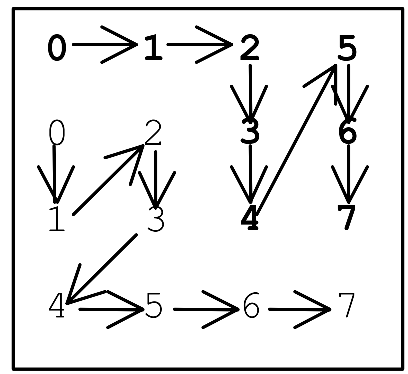


- **Zig-Zag Scan :**

Simple

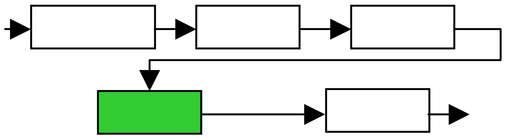


Double



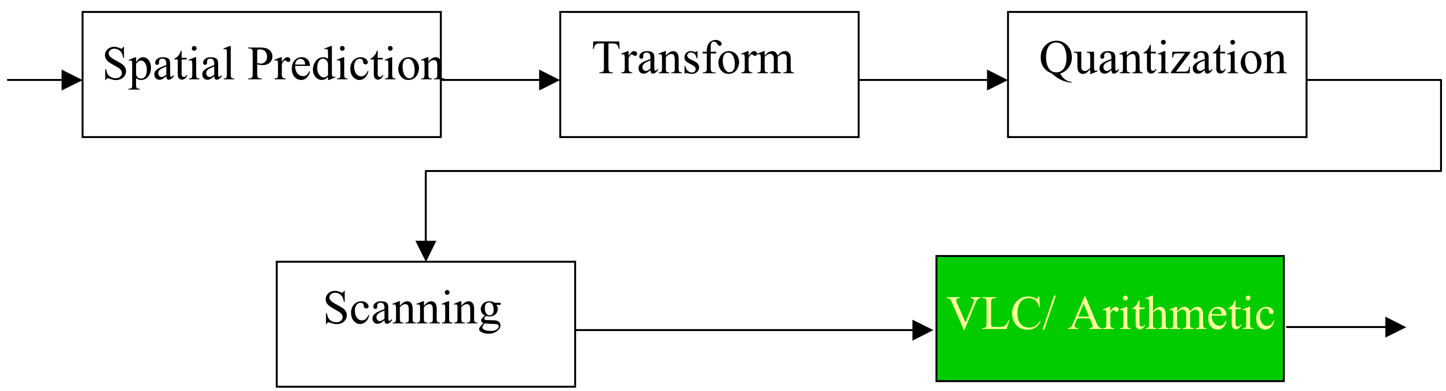
Note: Double scan was dropped in CD

I-frame coding: Coefficient Scan



- **Intra 16x16**
  - First scan through the 16 DC Transform coefficient (No CBP)
  - Then AC coefficients (if present), starting at the second position.

I-frame code: VLC



I-frame coding: UVLC (WD2Rev8)

Code No.	RUN	MB_Type		8x8 mode	MVD D-QUAN	CBP		Tcoeff_chroma_DC		Tcoeff_chroma_AC Tcoeff_luma Simple scan		Tcoeff_luma Double scan	
		Intra	Inter			Intra	Inter	Level	Run	Level	Run	Level	Run
0	0	Intra 4x4	16x16	8x8	0	47	0	EOB	-	EOB	-	EOB	-
1	1	0,0,0[2]	16x8	8x4	1	31	16	1	0	1	0	1	0
2	2	1,0,0	8x16	4x8	-1	15	1	-1	0	-1	0	-1	0
3	3	2,0,0	8x8	4x4	2	0	2	2	0	1	1	1	1
4	4	3,0,0	8x8 (ref=0)	Intra	-2	23	4	-2	0	-1	1	-1	1
5	5	0,1,0	Intra 4x4		3	27	8	1	1	1	2	2	0
6	6	1,1,0	0,0,0[2]		-3	29	32	-1	1	-1	2	-2	0
7	7	2,1,0	1,0,0		4	30	3	3	0	2	0	1	2
8	8	3,1,0	2,0,0		-4	7	5	-3	0	-2	0	-1	2
9	9	0,2,0	3,0,0		5	11	10	2	1	1	3	3	0
10	10	1,2,0	0,1,0		-5	13	12	-2	1	-1	3	-3	0
…	…	…	…		…	…	…	…	…	…	…	…	…

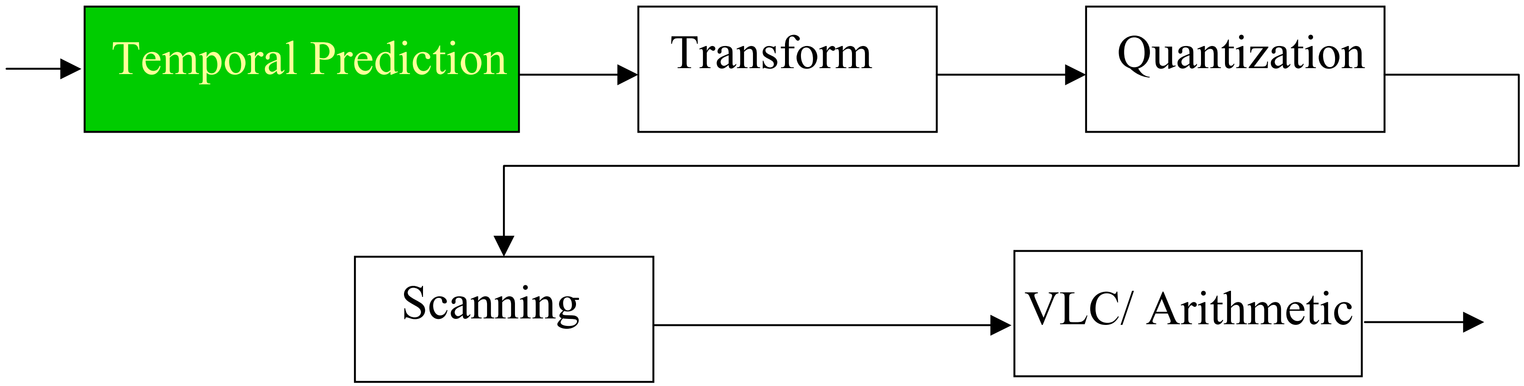
I-frame coding: UVLC

Code_number	Code word
0	1
1	0 1 0
2	0 1 1
3	0 0 1 0 0
4	0 0 1 0 1
5	0 0 1 1 0
6	0 0 1 1 1
7	0 0 0 1 0 0 0
8	0 0 0 1 0 0 1
9	0 0 0 1 0 1 0
10	0 0 0 1 0 1 1
……	. . . .

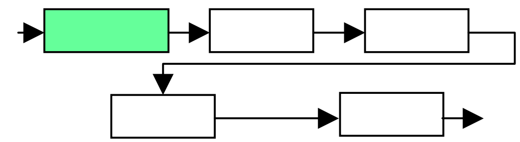
P, B frame coding

P, B-frame Coding

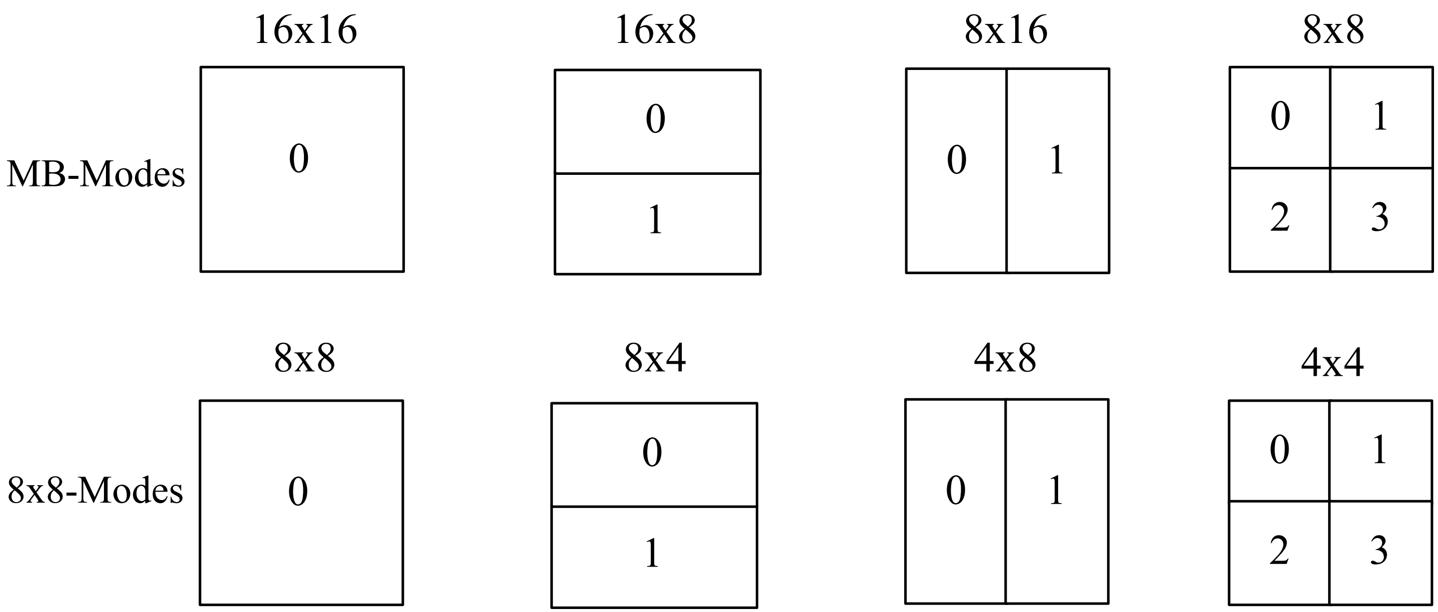
- Temporal Prediction (Motion Estimation)



P-frame coding: Motion Estimation

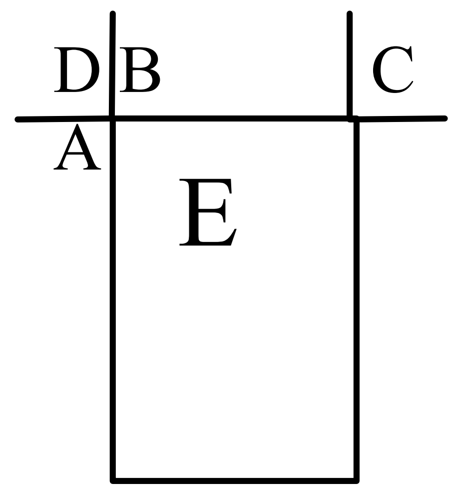


- **Macro Block - 16x16**
  - Prediction block structure
    - 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4



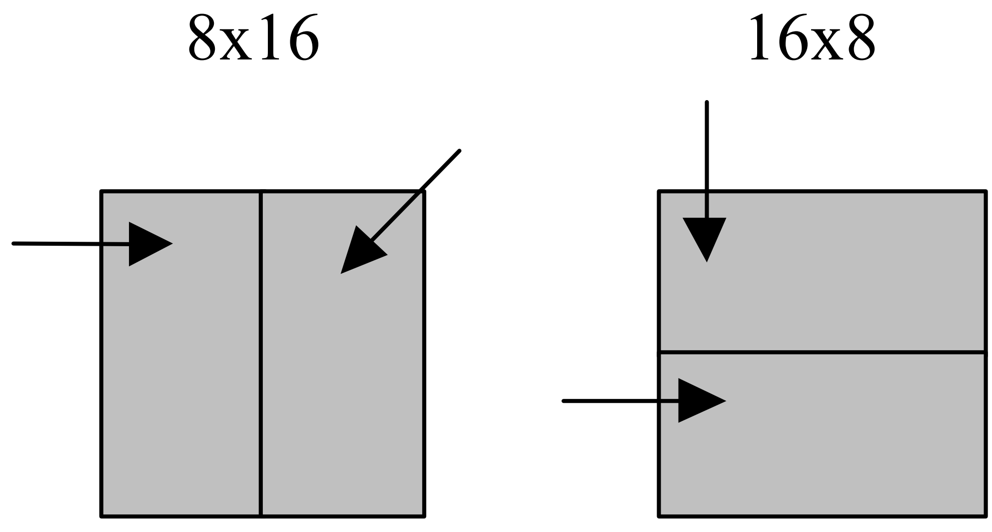
P-frame coding: Motion Vector Coding

- **Median Prediction (except for 16x8 and 8x16 modes)**
- **Median of A, B and C (normally)**
  - Exceptions
- **A,B,C and D can be from different reference frames**



P-frame coding: Motion Vector Coding

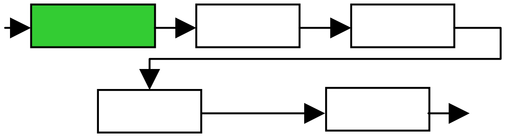
- **Directional Segmentation Prediction**
  - 16x8 and 8x16



P-frame coding: Chroma

- **Chroma vectors half of Luma vectors (4:2:0 resolution)**
  – Luma vector corresponding to 8x16 Luma block will apply to 4x8 Chroma block, so on

P-frame coding: Motion Vector Coding



- **Pixel Resolution**
  - 1/4th Pel

Quarter Pixel Interpolation: Luma

A	d	$\mathbf{b}^{\mathbf{h}}$	d	**A**
e	h	f	h	
$\mathbf{b}^{\mathbf{v}}$	g	$\mathbf{c}^{\mathbf{m}}$	g	$\mathbf{b}^{\mathbf{v}}$
e	h	f	i	
A		$\mathbf{b}^{\mathbf{h}}$		**A**

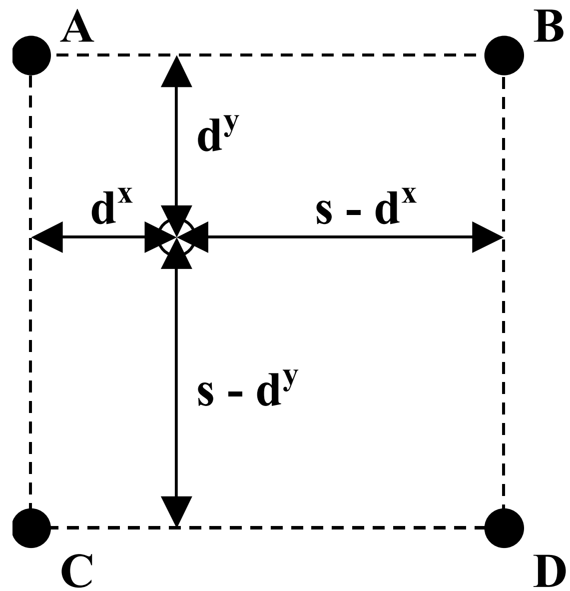
A ➡ Integer Pixel Locations

Quarter Pel Interpolation: Luma

- **Calculate Half Pel values**
  - use 6 tap filter {1, -5, 20, 20, -5, 1} to get b
  - b^h = clip1(((*b*+16)>>5))
  - c from b values using the 6 tap filter
  - c^m = clip1(((*c*+512)>>10))
- **Average of Integer and Half Pel values to find d,e,f,g**
  - e.g. d = (A+b^h)>>1
- **h = (b^h+b^v)>>1 (diagonal direction averaging)**
- **i = (A_1+A_2+A_3+A_4+2)>>2**

clip3(a, b, c) = a if c<a, b if c>b, otherwise c
clip1(c) = clip3(0, 255, c) = 0 if c<0, 255 if c>255, otherwise c

Quarter Pixel Interpolation: Chroma



$$v = ((s - d^x)(s - d^y)A + d^x(s - d^y)B + (s - d^x)d^y C + d^x d^y D + s^2/2)/s^2$$

A,B,C,D are the integer position values
s = 8, d^x and d^y are fractional positions in units of 1/8th sample

P-frame coding: CBP

- **CBPY (Coded Block Pattern for Y)**
  - Flags which 4 8x8 blocks contain coefficients
- **CBP (Coded Block Pattern)**
  - CBP = CBPY + 16.nc

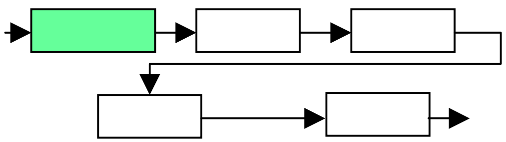
    nc = 0,1 and 2

    nc = 0 : No Chrominance Coefficient

    nc = 1 : non-zero 2x2 Chroma Tx coefficients, All Chroma AC coefficients are 0 (no EOB for Chroma AC sent)

    nc = 2 : other
  - 4 LSBs of CBP contain information about which of 4 8x8 Luma blocks contain non-zero coefficients

B-frame coding: Motion Vector Coding



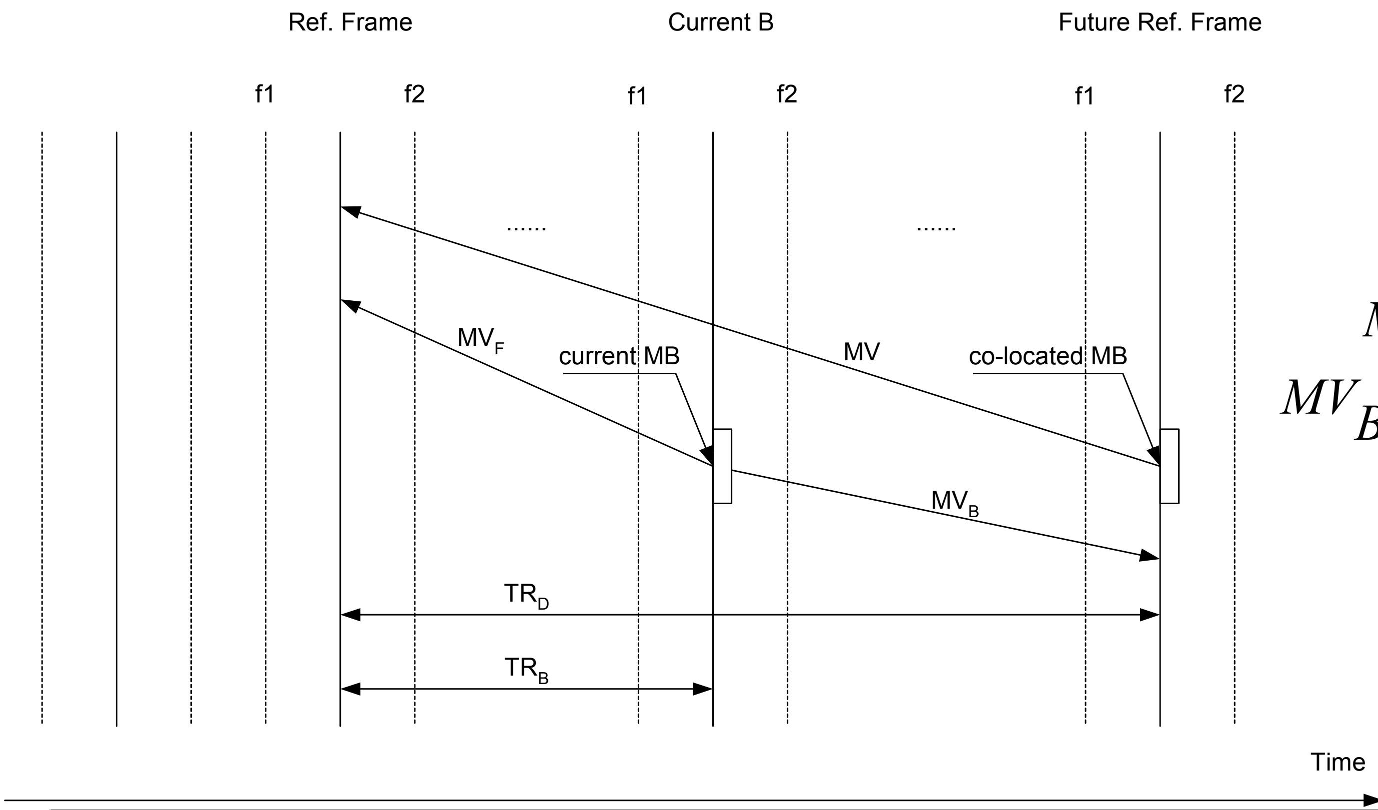
- **Modes**
  - Direct, forward, backward, bi-directional and intra
  - 16x16, 8x16, 16x8, 8x8, 4x8, 8x4, 4x4

B-frame coding: Motion Vector Coding

- **Direct Mode**
  - No MV data is transmitted
  - Same block structure as co-located MB in temporally subsequent picture
  - MVs are computed as scaled version of corresponding MV of the co-located MB

B-frame coding: Motion Vector Coding

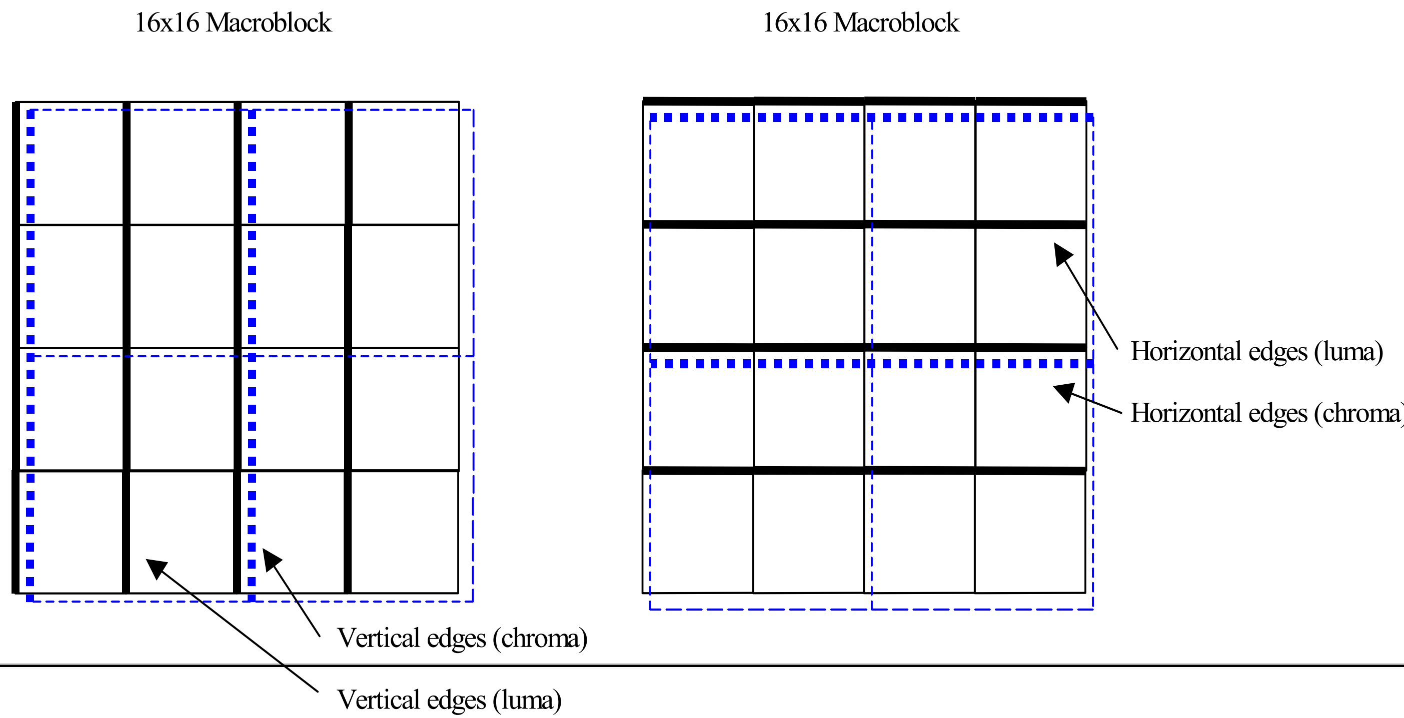
- **Direct Mode**



$$MV_F = TR_B \cdot MV / TR_D$$
$$MV_B = (TR_B - TR_D) \cdot MV / TR_D$$

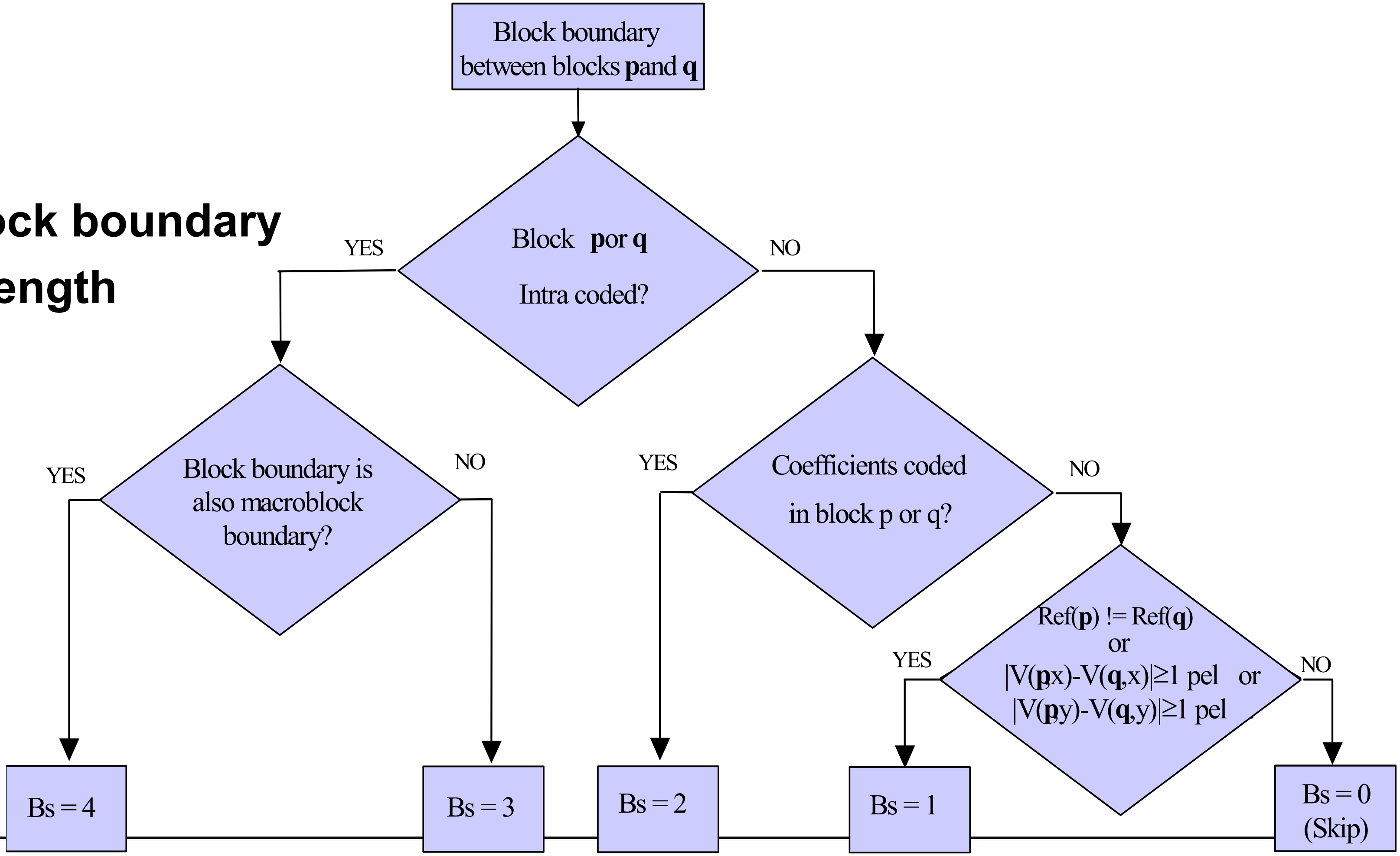
Deblocking Filter

- **Filtering done on an MB basis**
  - 3 internal vertical edges then 3 internal horizontal edges
  - Left edge, Top edge of MB after corresponding left and top MBs are reconstructed



Content dependent filtering

- **Block boundary strength**



Ref(p) is the reference frame or field of block p and V(p) is the motion vector of block p

Content dependent filtering

p_3	p_2	p_1	p_0	q_0	q_1	q_2	q_3

8 samples across the block boundary

Filtering if

$$Bs \neq 0 \;\; \&\& \;\; |p_0 - q_0| < \alpha \;\; \&\& \;\; |p_1 - p_0| < \beta \;\; \&\& \;\; |q_1 - q_0| < \beta$$

	QP_{av}																									
	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13
α	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	5	6	7	9
β	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	3	4	4	4

Content dependent filtering

QP_{av}	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
α	10	12	14	17	20	24	28	33	39	46	55	65	76	90	106	126	148	175	207	245	255	255	255	255	255	255
β	6	6	7	7	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15	16	16	17	17	18	18

Content Dependent Filtering

- **Filtering of Edges with Bs < 4**
- **Filtering of Edges with Bs = 4**

Content dependent filtering: Bs < 4

Δ = clip3($-C, C, (((q_0 - p_0) << 2 + (p_1 - q_1) + 4) >> 3)$)
P_0 = clip1($p_0 + \Delta$)
Q_0 = clip1($q_0 - \Delta$)

$a_p = |p_2 - p_0|$
$a_q = |q_2 - q_0|$

If $a_p < \beta$ for a luminance edge, a filtered sample P_1 *shall be produced by*

$P_1 = p_1$ + clip3($-C0, C0, (p_2 + P_0 - 2*p_1) >> 1$)

If $a_q < \beta$ for a luminance edge, a filtered sample Q_1 *shall be produced by*

$Q_1 = q_1$ + clip3($-C0, C0, (q_2 + Q_0 - 2*q_1) >> 1$)

Content dependent filtering

Value of Clipping Parameter

	QP_{av}																									
	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Bs = 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
Bs = 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
Bs = 3 or 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

	QP_{av}																									
	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
Bs = 1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13
Bs = 2	1	1	1	1	1	2	2	2	2	3	3	3	4	4	5	5	6	7	8	8	10	11	12	13	15	17
Bs = 3 or 4	1	2	2	2	2	3	3	3	4	4	4	5	6	6	7	8	9	10	11	13	14	16	18	20	23	25

Content Dependent Filtering: Bs=4
(WD2 Rev8)

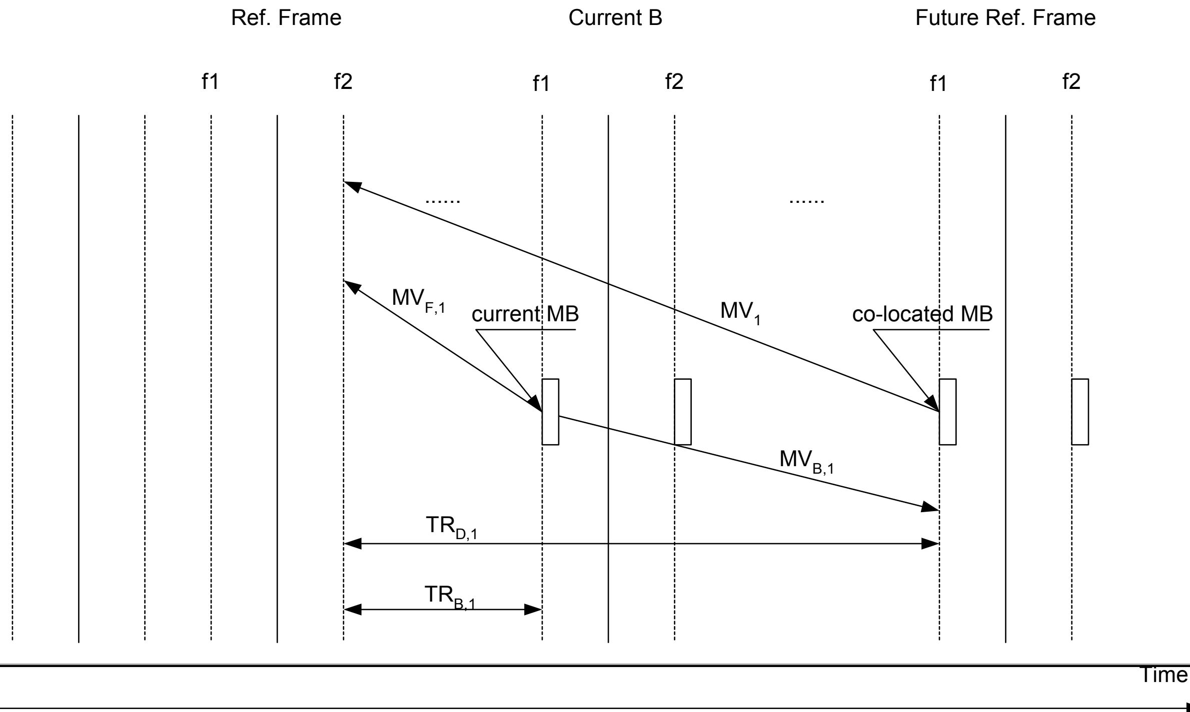
- **If (Bs == 4) && ($a_p < \beta$) && ($a_q < \beta$) && (1 < $|p_0 - q_0|$ < (QP_{av} >> 2))**
  - $P_0 = (p_2 + 2*p_1 + 2*p_0 + 2*q_0 + q_1 + 4) >> 3$
  - $P_1 = (p_3 + 2*p_2 + 2*p_1 + 2*p_0 + q_0 + 4) >> 3$
  - $P_2 = (2*p_3 + 3*p_2 + p_1 + p_0 + q_0 + 4) >> 3$ (Luma)
  - $Q_0 = (p_1 + 2*p_0 + 2*q_0 + 2*q_1 + q_2 + 4) >> 3$
  - $Q_1 = (p_0 + 2*q_0 + 2*q_1 + 2*q_2 + q_3 + 4) >> 3$
  - $Q_2 = (2*q_3 + 3*q_2 + q_1 + q_0 + p_0 + 4) >> 3$ (Luma)

Note: Some changes were made in CD

Interlaced Video

Interlaced Video

- **Adaptive Field Frame Switching**
  - Picture layer
- **E.g. Direct Mode**



Arithmetic Coding

- Context-based Adaptive Binary Arithmetic Coding (CABAC)

MPEG-4 AVC – Committee Draft (CD)

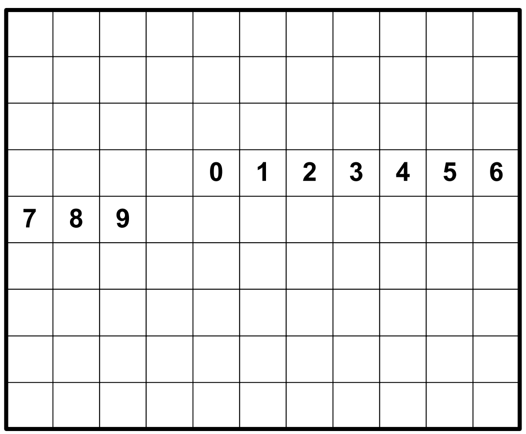
Highlights of some of the changes from WD-2 to CD

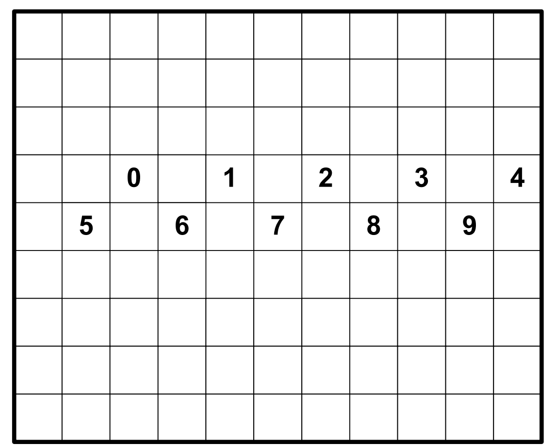
MPEG-4 CD changes highlights

- **Qp values were shifted up by 12**
- **Bi-directional pictures were renamed as Bi-predictive and they can be used as reference**
- **Modification to de-blocking filter**
- **Flexible Macroblock Ordering (FMO)**
- **Adaptive Block Transform (ABT)**
- **Context Adaptive VLC (CAVLC) was added**
- **Improved CABAC**
- **Definition of Profiles and Levels**
  - Baseline and Main Profiles

Flexible Macroblock Ordering (FMO)

- **MBs not necessarily in raster scan order**
- **mb_allocation_map**
  - assigns MBs to a slice group
  - maximum of 8 slice groups





MOTOROLA

Ajay Luthra
WEMP4, June 25, 2002

intelligence everywhere™

Adaptive Block Transform (ABT)

- **Inter Mode: transform block size adapts to the block size used for motion compensation**
- **Intra Mode**
  - 8x8, 8x4, 4x8, 4,4
- **Transform Matrix**
- **Coefficient Scanning**
- **De-blocking Filter**
- **VLC Tables**
- **CABAC**

CAVLC

- **Coding and decoding of transform coefficients**
- **String of coefficients at the high frequencies at +- 1**
- **Parameter Num-Trail – number of coefficients and Trailing 1s (T1s); T1 = 0,1,2,3**
- **Coefficients other than T1s Level information is coded**
- **For decoding of Coefficient data (both Level and Run) scanning is done in reverse order**
  - In Level information, signs of T1s are decoded first (in reverse order) then the Level Information
  - Similarly Run information

CAVLC

- **Coding Flow**
  - Num of Coeff + T1 -> Sign T1 -> Level -> Total Zeros -> Run

CAVLC

- **3 + 1 (Chroma DC) tables; e.g. VLC0**

NumCoef\T1s	0	1	2	3
0	1	-	-	-
1	000011	01	-	-
2	00000111	0001001	001	-
3	000001001	00000110	0001000	00011
4	000001000	000001011	000000101	000010
5	0000000111	000001010	000000100	0001011
6	00000000111	0000000110	0000001101	00010101
7	000000001001	00000000110	0000001100	00010100
8	000000001000	00000001001	000000001010	000000111
9	0000000000111	000000001011	000000000101	0000000101
10	0000000000110	0000000001101	0000000001111	00000001000
11	00000000000011	0000000001100	0000000001110	000000000100
12	00000000000010	00000000000100	00000000000110	0000000000101
13	00000000000101	00000000000111	000000000010001	00000000001001
14	000000000000011	000000000000010	000000000010000	0000000000000011
15	00000000000000001	00000000000000011	00000000000000010	00000000000000101
16	00000000000000000	000000000000001001	0000000000000010001	0000000000000010000

CAVLC

- **Pick 1 of the 3 table based on:**

Upper block (N_U)	Left block (N_L)	N
X	X	$(N_L+N_U)/2$
X		N_U
	X	N_L
		0

N_L = No. of Coeff in block on left
N_U = No. of Coeff in block above the current block
X => Block is available in the same slice

0 <= N < 2 : Num-VLC0
2 <= N < 4 : Num-VLC1
5 <= N < 8 : Num-VLC2
N >= 8 : 6 bit FLC xxxxyy :
xxxx = NumCoeff-1, yy = T1. Exception: 000011 => NumCoeff = 0

CAVLC

- **4 Level Tables (Lev-VLCn)**

Lev-VLC0

Code no	Code	Level (±1, ±2..)	Level (±2, ±3..)
0	1	1	2
1	01	-1	-2
2	001	2	3
3	000`	-2	-3
..	..	..	..
13	00000000000001	-7	-8
14-29	0000000000000001xxxx	±8 to ±15	±9 to ±16
30->	00000000000000001xxxxxxxxxxxx	±16 ->	±17 ->

CAVLC

- **Table Selection (Adaptive)**
- **Adaptation based on Qp, Num of Coeff. and previously coded value**
  - Inter and Intra with Qp >= 21
    - Rules
  - Intra with Qp < 21
    - If Num Coeff > 10 Rule1, else Rule 2
    - If VLC1 and |Level| > 3 Rule 3
    - If VLC >= VLC2 Rule 5

CAVLC

- **Adaptation Rules**

Inter and intra with QP >= 21

First coefficient with VLC0. Next VLC1.

Increase VLC by one (up to 2) if |Level|> 3

Intra with QP < 21

if (number of coefficients > 10)

    First coefficient with VLC1. Next VLC2.

else

    First coefficient with VLC0. Next VLC1.

if (vlc == VLC1) change to VLC2 if |Level|> 3.

if (vlc >=VLC2) increase vlc by one (up to 4) if |Level| > 5

The same procedure is used for chroma AC and DC coefficient levels.

CAVLC

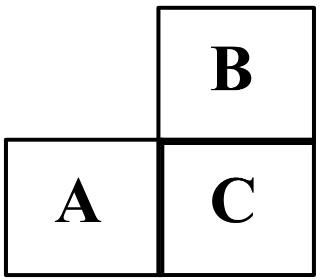
- **Run Decoding**
  - Total Zeros
    - Sum of all zeros located before the last non-zero coeff.
    - E.g. 0030040000201000, Total Zeros is 2+2+4+1 = 9
    - Num Coeff + Total Zeros tables
  - Run before each coeff
    - VLC Table

CABAC (Context-based Adaptive Binary Arithmetic Coding)

- Estimates of conditional Probability of symbols
- Arithmetic Codes (non-integer bits / symbol, hence more efficient)
- Adaptive (Adapt to non-stationary symbol statistics)

CABAC

- **Context modeling**



CABAC

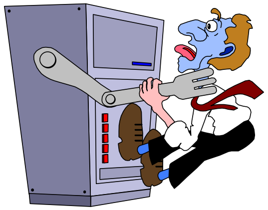
- **Binarization**
  - E.g. Macroblock modes in P-slices

Code Number	Macroblock Mode	Binarization
0		
1	SKIP	0
2	16x16	1 0 0 0
3	16x8	1 0 1 1
4	8x16	1 0 1 0
5 (VLC only)	8x8 (split)	1 0 0 1
6	8x8 (split, all ref=0)	
7	Intra4x4	1 1 0
	Intra16x16 bin	1 1 1
		1 2 3 4

CABAC

- **Table-based Arithmetic Coding**
  - Arithmetic coding
    - Recursive interval subdivision
    - p(0) and 1 - p(0) ratio
    - Most Probable Symbol (MPS) and Least Probable Symbol (LPS)
    - After encoding a decision, prob. estimate is updated
    - Table for Prob. transition and MPS/LPS switch
- **Initialization**
  - R set to 0x8000

MPEG-4 AVC - Profiles & Levels



- **Large number of tools**
- **Wide range of applications**
- **All tools are not needed for a given application**
- **Implementing all the tools will make codecs unnecessarily very expensive**

MPEG-4

- **Profiles**
  - Based on needs of different classes of applications, within each part a subset of tools is defined to form a profile
- **Levels**
  - Within each profile a level of performance is defined to constrain the processing power and memory requirement on real time codecs
  - Defines maximum picture size, compressed bit rate, frame rate etc.

MPEG-4 AVC Profiles
60,000 ft view

MPEG-4 AVC - Profiles and Levels
(Under Consideration)

- **Profiles**
  - Baseline
  - Main
- **Levels**
  - Major: QCIF, CIF, ITU-R 601 (SDTV), HDTV

MPEG-4 AVC - Profiles
(Under Consideration)

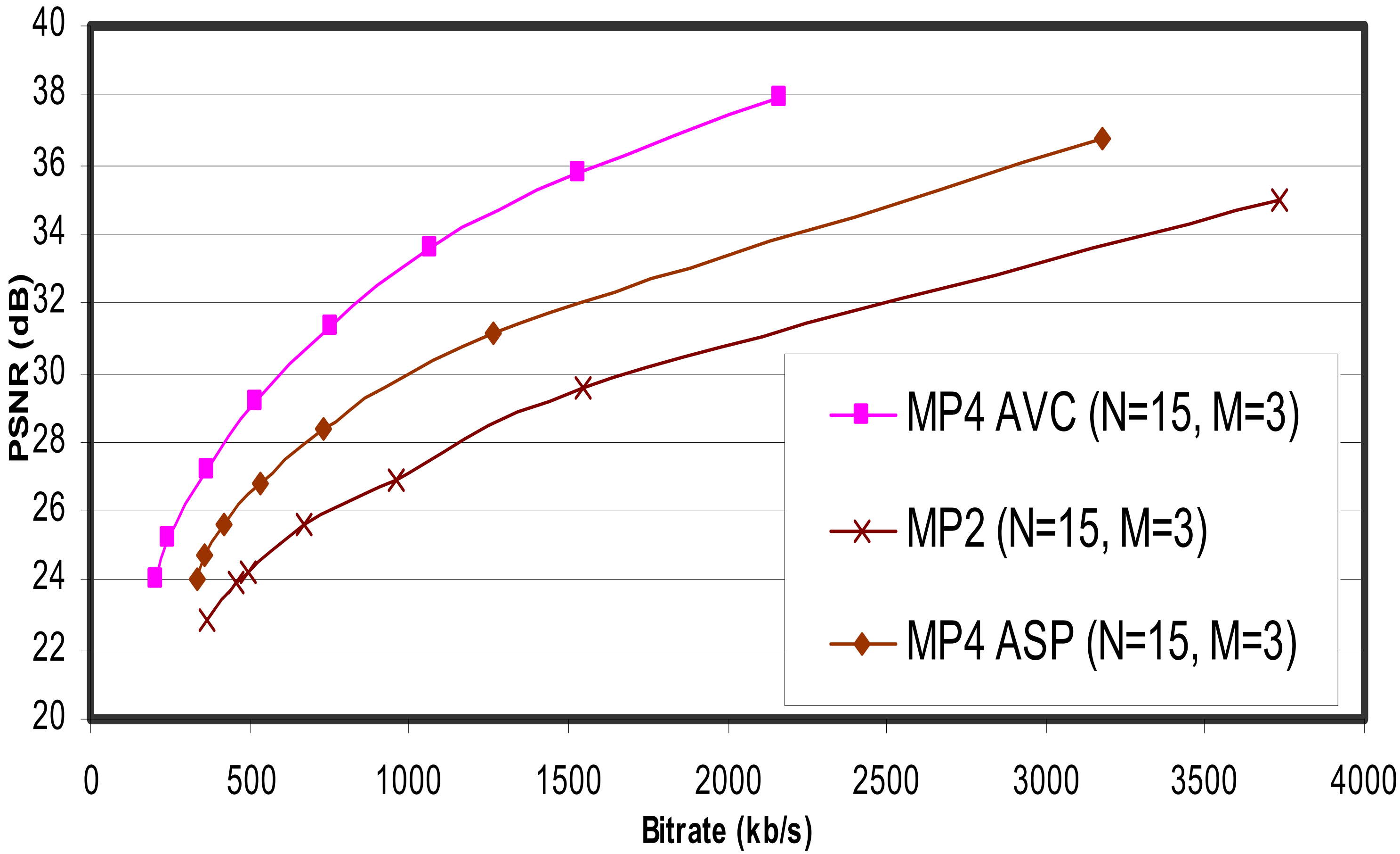
	Baseline	Main
I and P Pictures	X	X
De-blocking filter	X	X
1/4 Pel Motion Comp	X	X
Var. Block size (16x16 to 4x4)	X	X
Interlaced Coding	X	X
Entropy based VLD	X	X
Flex. Macroblock Ordering	X	X
B - Picture		X
CABAC		X
Adaptive Block Transform		X

MPEG-4 AVC - Levels
(Under Consideration)

- **Level 1: QCIF @ 15frames/sec**
  - + Level 1.1: QCIF @ 30 frames/sec, CIF @7.5 frames/sec
  - + Level 1.2: CIF @ 15 frames/sec
- **Level 2: CIF @ 30 frames/sec**
  - + Level 2.1: HHR @ 25 or 30 frames/sec
  - + Level 2.2: SDTV @ 15 frames/sec
- **Level 3: SDTV**
  - + Level 3.1: 1280x720x30p, SVGA @ 50+ frames/sec
  - + Level 3.2: 1280x720x60p,
- **Level 4: HDTV: 1920x1080x60i, 1280x720x60p, 2kx1k @ 30 frames/sec**
- **Level 5: SHDTV ?: 1920x1080x60p, 2kx1k @ 60fps**

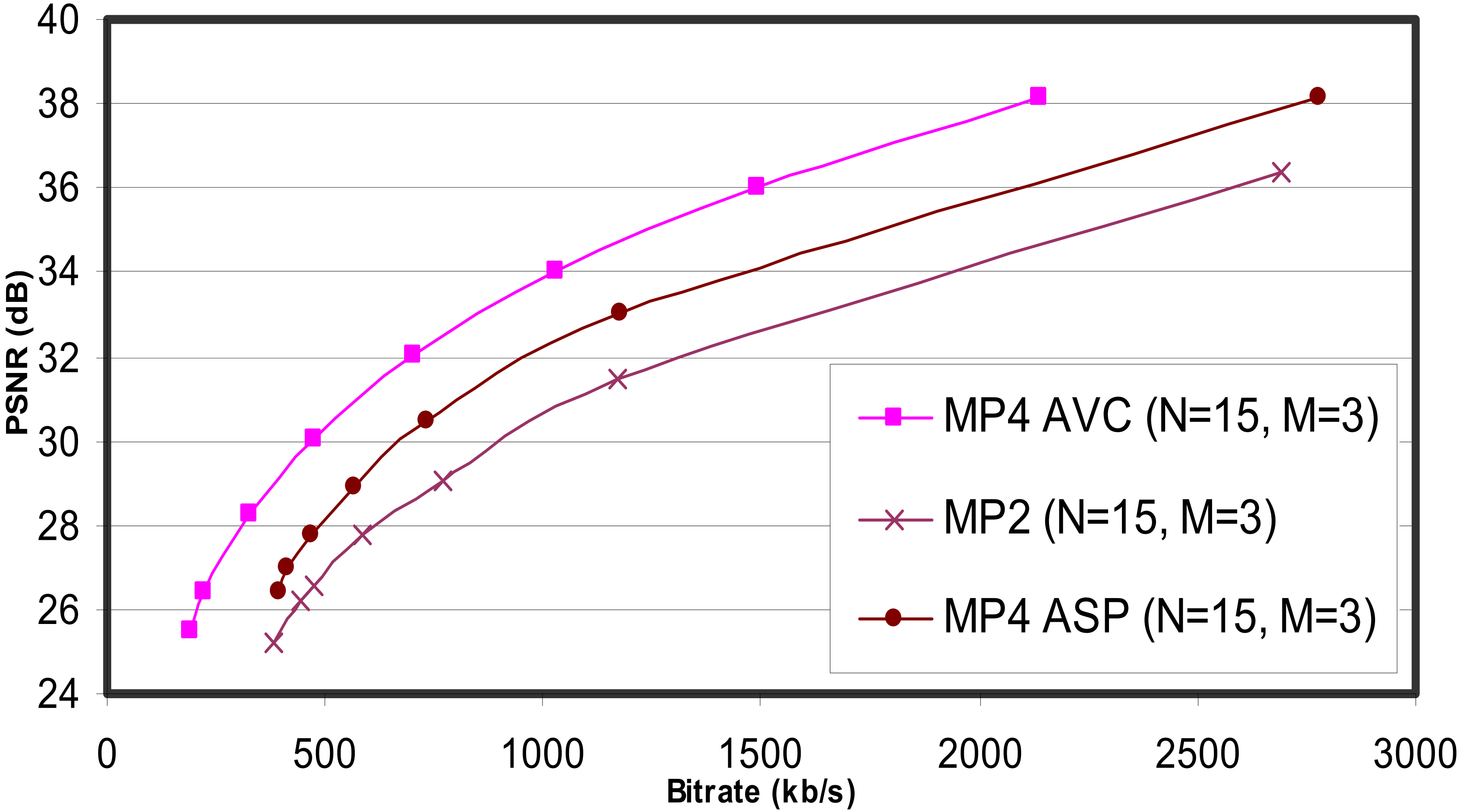
MPEG-2, MPEG-4 ASP, MPEG-4 AVC (WD2)

Mobile & Calendar (CIF)



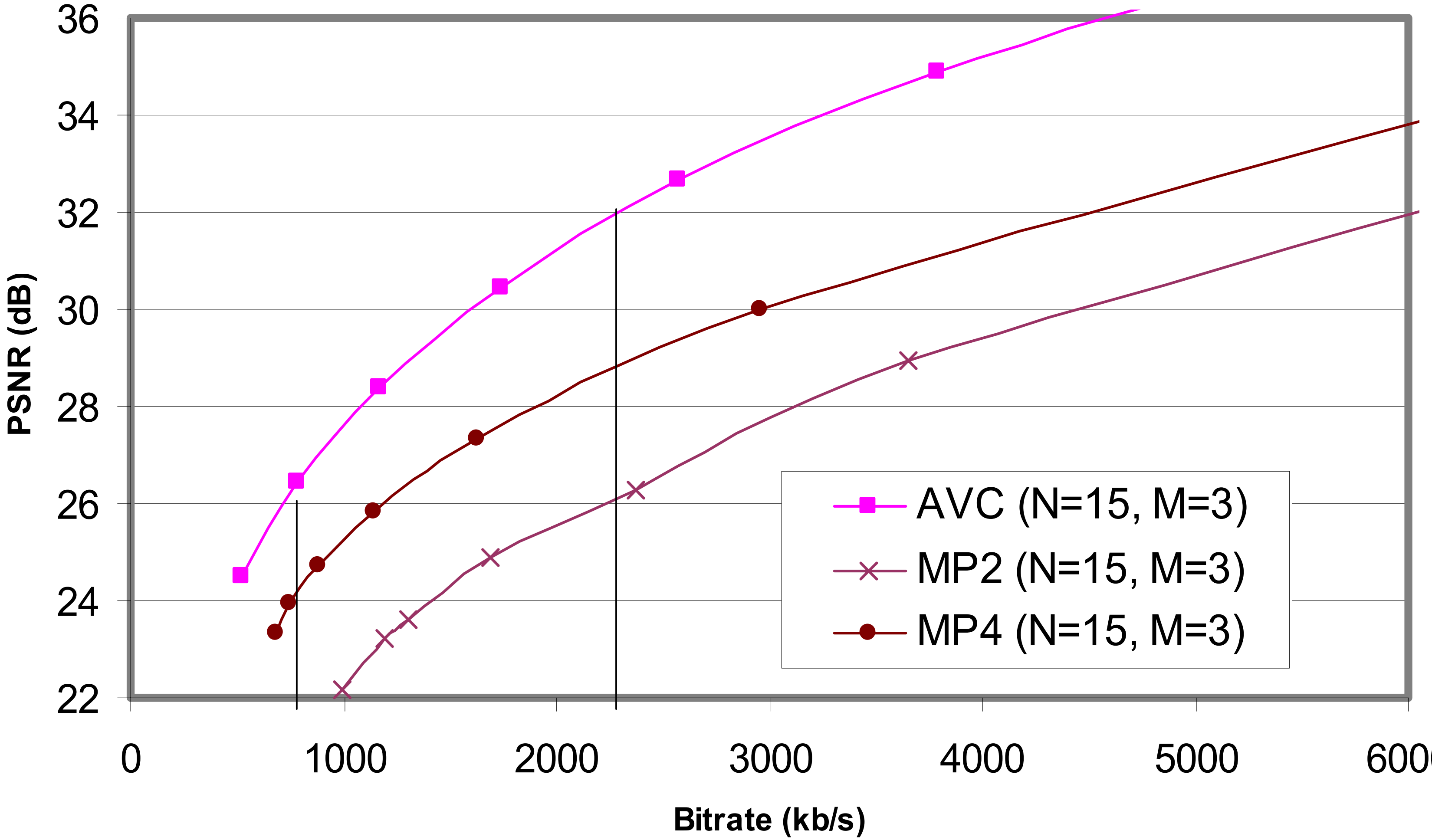
Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another

MPEG-2, MPEG-4 ASP, MPEG-4 AVC (WD2)
BUS (CIF)



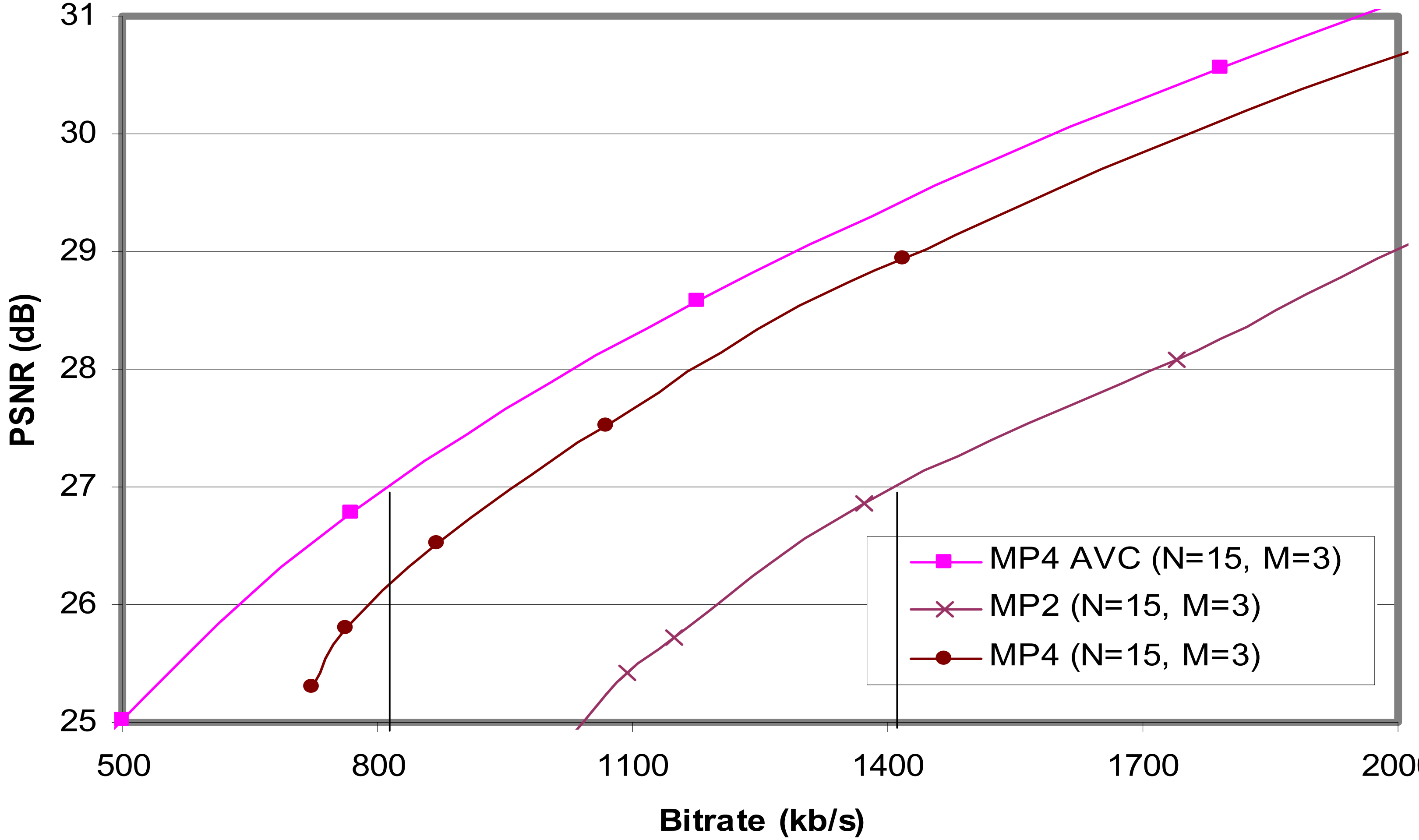
Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another

MPEG-2, MPEG-4 ASP, MPEG-4 AVC (WD2)
Mobile & Calendar (HHR)

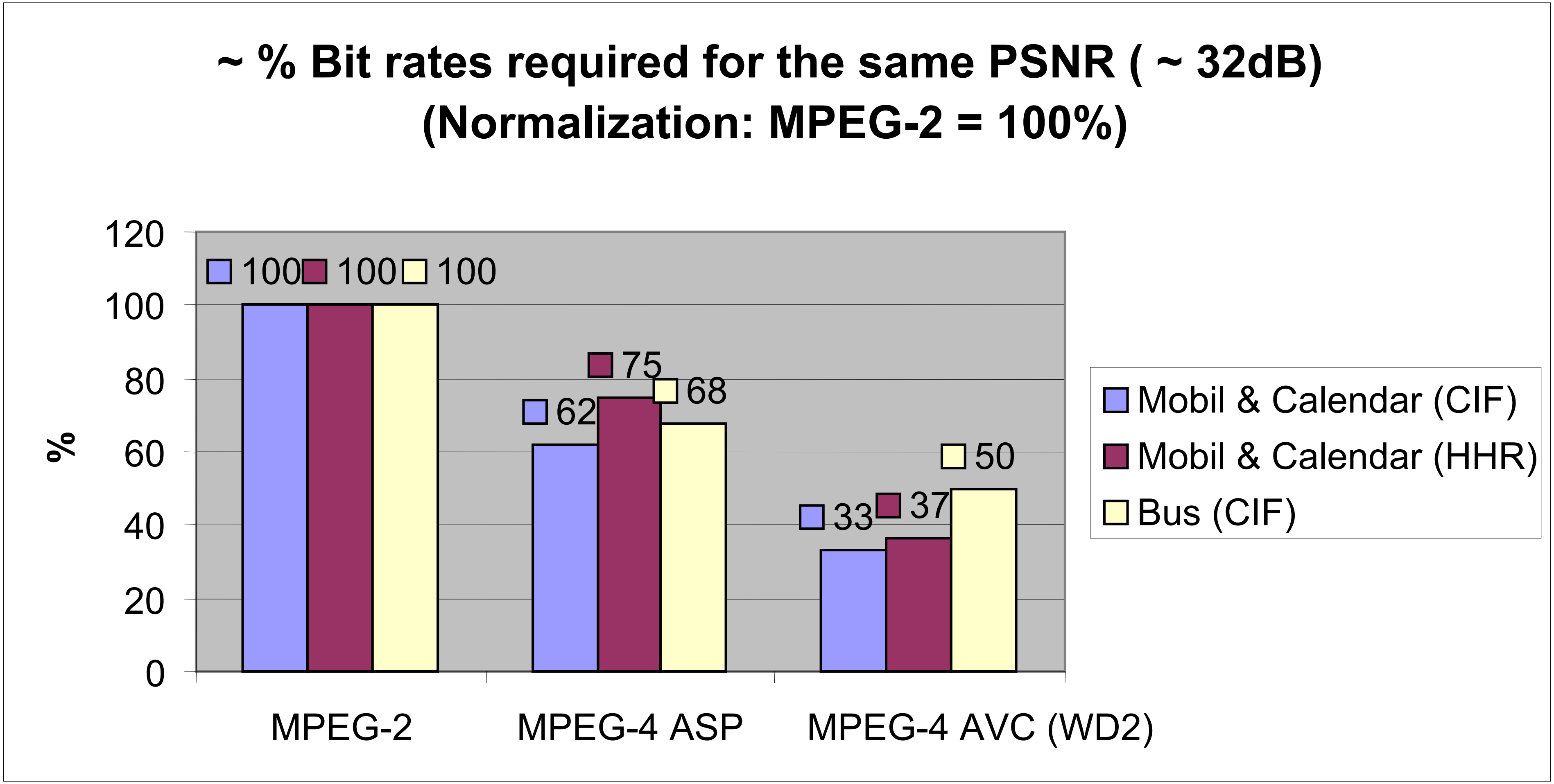


Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another

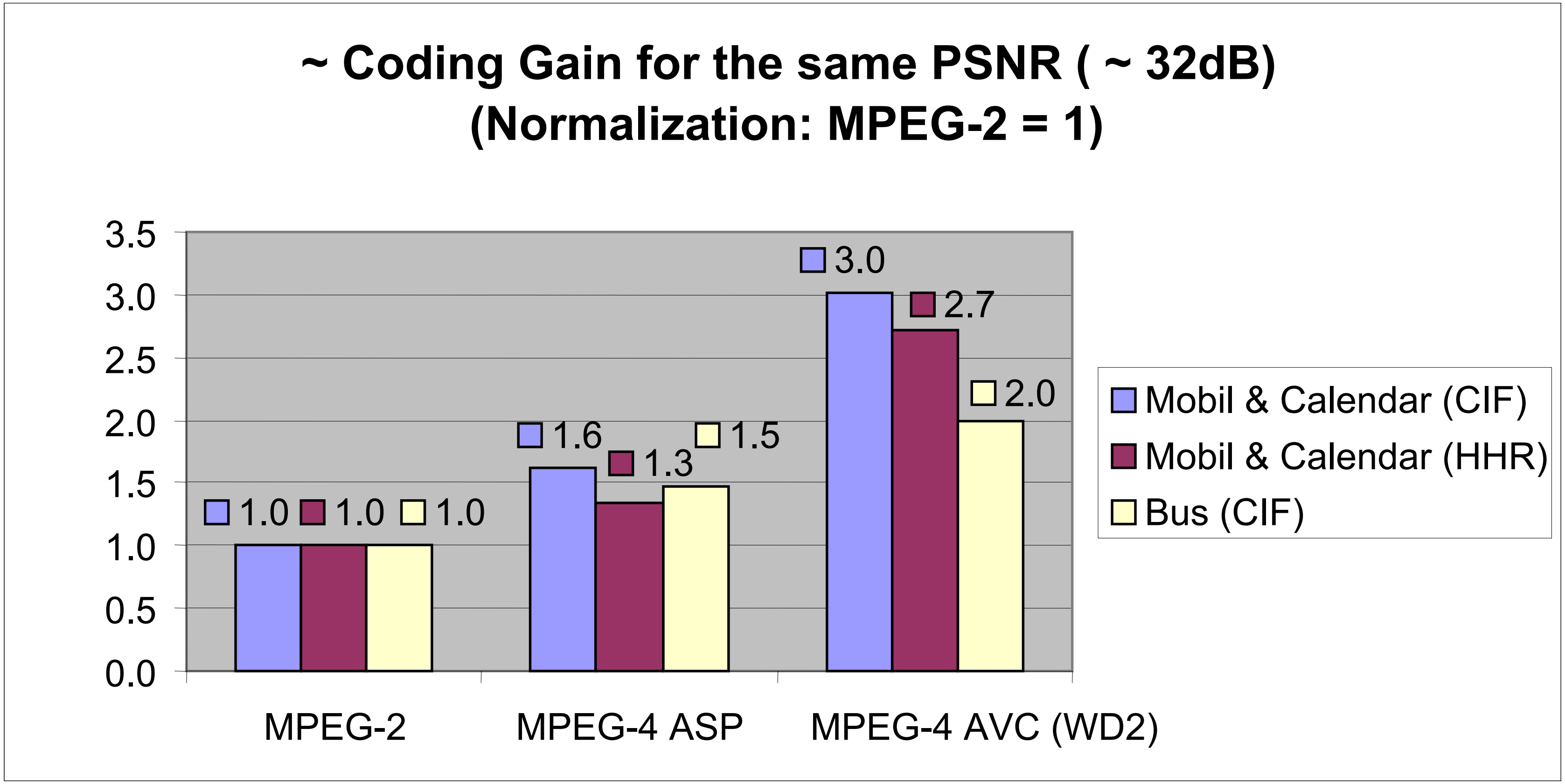
MPEG-2, MPEG-4 ASP, MPEG-4 AVC (WD2)
BUS (HHR)



Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another



Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another



Note: PSNR at a given bit rate is encoder dependent and will vary from one encoder to another

Demonstration

- MPEG-4 AVC vs MPEG-2
- MPEG-4 AVC vs MPEG-4 ASP