Efficient Implementation of Video Postfiltering on the TI DSC25 Platform

Murali Babu, Amitava Deb,
Sankaranarayanan Parameswaran,
Dileep Tamia & Sriram Sethuraman

Ittiam Systems Pvt. Ltd.

Presented by: Sriram Sethuraman

TI DSP Conference, 5 December 2002, Bangalore

Outline of Presentation

- Video Postfiltering for artifact reduction
  - Coding artifacts (blocking/ringing)
  - State of the art algorithms
    - Deblocking
    - Deringing
  - Complexity analysis
- DSC25 considerations
  - Ittiam's MuVPlayer™ on DSC25
- Proposed Postfiltering algorithm
  - Improving data access patterns
  - Reducing conditional execution
  - Multithreading on DSP, iMX, and DMA
- Results / Demo

Video Coding Artifacts

- Block-based coding (8x8 blocks in MPEG-4)
  - Continuity not maintained at block boundaries
- Quantization
  - At low bit-rates, high frequency coefficients get coarsely quantized
  - Spatially adaptive quantization (adjacent blocks have different QPs)
- Motion compensation
  - Compensates from a reference with coding artifacts
- Resulting in…

Blocking artifacts

Goto deblocked

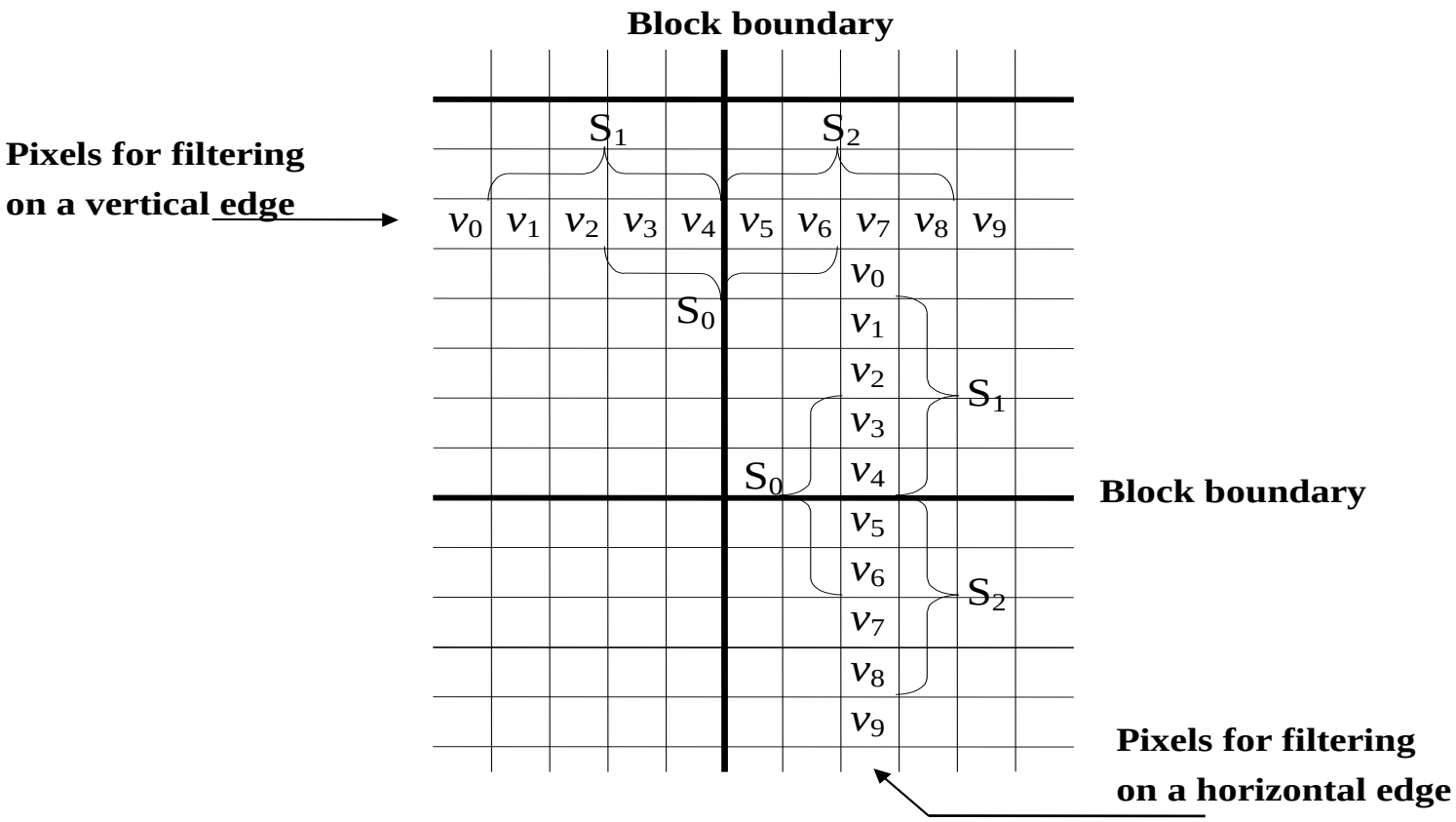
Ringing Artifacts

Goto deringed

State of the Art Postfiltering

- Postfiltering
  - Outside the scope of a standards based codec
  - Essential at low bit-rates
- Spatial domain (more common)
- DCT domain
- Filtering is adapted based on coding parameters (e.g. quantizer)
- Extent of filtering is adapted to spatial details
  - To preserve edges/details
- Independent filtering in luma and chroma
- Good Baseline from the quality perspective
  - MPEG-4 VM algorithms

MPEG-4 VM Deblocking



- Spatially adaptive modes
  - DC offset mode
    - Smooth except for a discontinuity due to blockiness
  - Non DC offset mode
    - Has local details or edges

MPEG-4 VM Deblocking (Cont.)

- Steps
  - Detection of the mode
    - DC offset: 6 out of 9 differences are below T_{pd}
    - Else, non DC offset mode
  - Correction for the mode
    - DC offset: If $|MAX(v_i) - MIN(v_i)| < 2*QP$,
      - Apply [1 1 2 2 4 2 2 1 1]/16 filter with boundary padding
    - Non DC offset:
      - Apply [2 -5 5 –2]/8 kernel to S_0, S_1, S_2 to get a_{30}, a_{31}, a_{32}
      - Let $a_{3,0}' = SIGN(a_{3,0}) \cdot MIN(|a_{3,0}|, |a_{3,1}|, |a_{3,2}|)$;
      - If $| a_{30} | < QP$,
        - » d = (v4 – v5)/2 [clipped by $5 \cdot (a_{3,0}' - a_{3,0})//8$]
        - » v4 = v4 – d and v5 = v5 + d

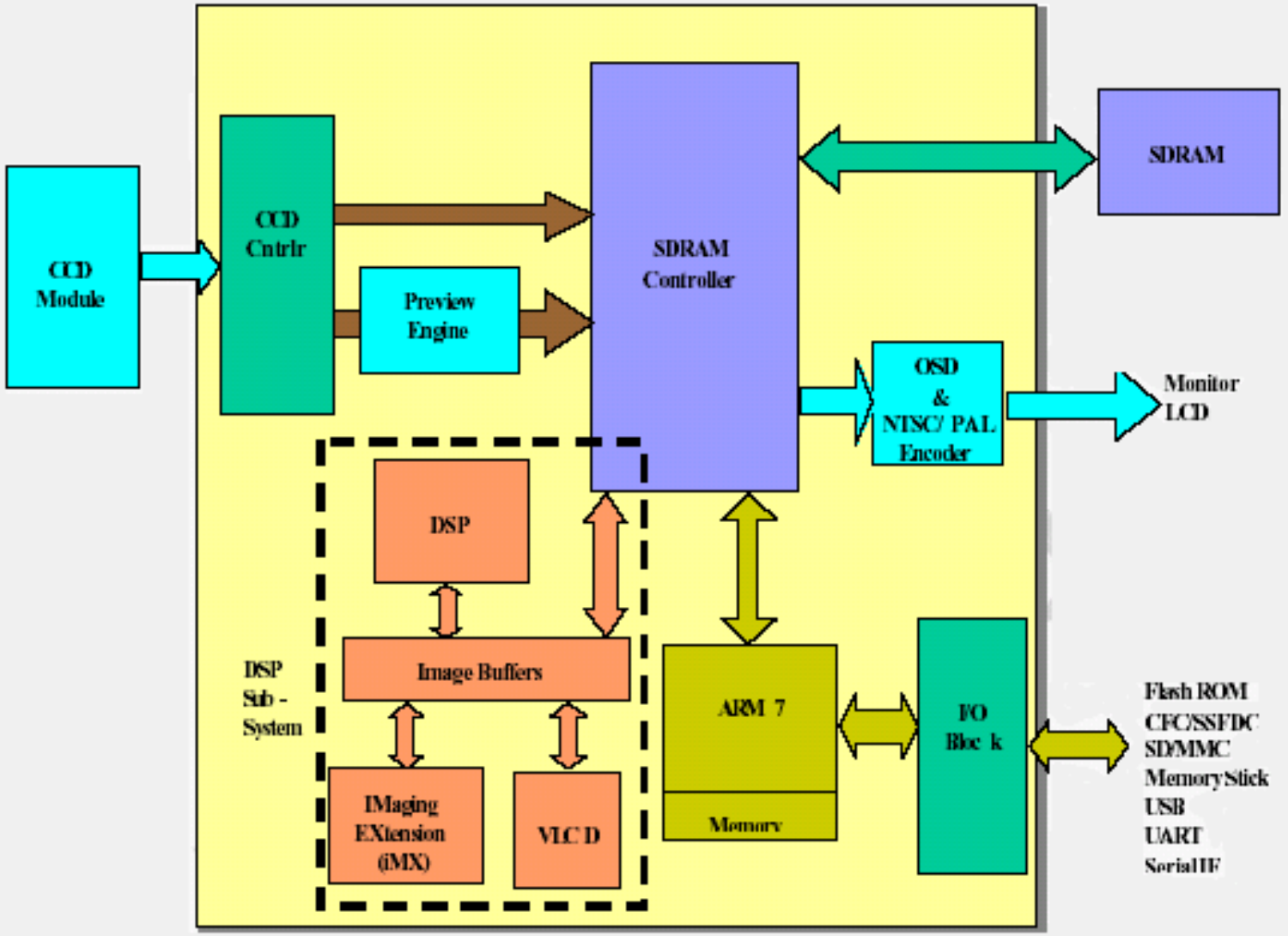
»Back Back2

MPEG-4 VM Deringing

	0	0	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	0	1	1	
	1	1	0	0	0	0	0	1	1	1	
	1	1	1	0	0	0	1	1	1	1	
	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1	1	1	0	
	1	1	1	1	1	1	1	1	0	0	

- Spatially Adaptive Index Acquisition:
  - Thr = (MAX + MIN + 1)/2
  - Index = 1 if P_{ij} >= Thr; Else 0
- Correction
  - Apply a 3x3 LPF if all indices in that 3x3 window are all ones or all zeros; clip the difference by QP/2

TI TMS320DSC25



Reproduced from TI's Reference Manual, © Texas Instruments

TI TMS320DSC25

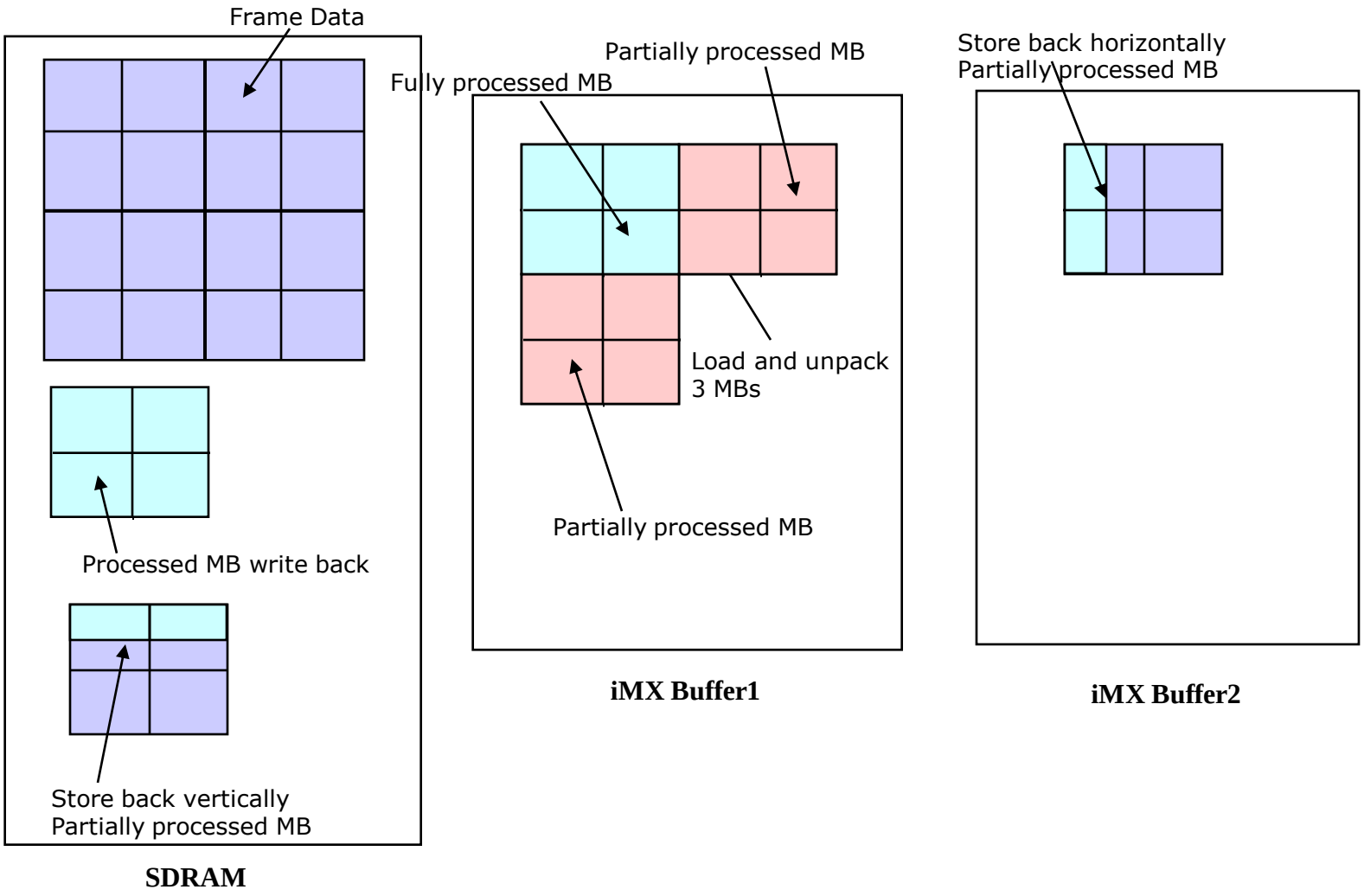
- ARM7
  - 40 MHz, 32-bit RISC, 32 kB RAM
- C54x
  - 94 MHz, 16-bit DSP, 32 kW memory
  - Access to SDRAM only thro' image buffers
  - 2 cycle access to image buffers
- Image buffers (16kW)
  - BUF A and BUF B (shared between iMX and DMA)
- Imaging Extension (iMX) coprocessor
  - 4 MAC units (parallel reads/writes from image buffer)
  - TI provided APIs for array op.s, transforms, etc.
  - Support for saturation
  - Cannot do conditional execution
- VLD and Quant/DeQuant coprocessors

Ittiam's MuVPlayer™ on DSC25

- Implements MPEG-4 Video Decoding
  - Simple Profile compliant
  - Multithreaded to maximally utilize DSP, iMX, VLD/Dequant units, and DMA
  - To use the motion compensation iMX APIs, the reconstructed frame is kept in 4:2:2 (YCbYCr) interleaved format
- Optionally implements MP3 audio decoding
  - Optimized to use DSP and iMX effectively
- ARM7 is used for RTOS, networking, and A/V sync
- Goal is to add video postfiltering to this player
  - MOPS required for CIF@30fps (ignoring branching)
    - DC offset: ~200 MOPS
    - Non DC offset: ~120 MOPS
  - Processing on DSP alone is not possible; need to use iMX

VM Deblocking on DSC25

- Data Access Issues



VM Deblocking on DSC25

- Adaptive Processing Issues

- DC/NonDC decision and corresponding corrections are independently performed for every row/column of a block VMDeblocking
  - iMX has to compute one decision for 8 pixels
  - DSP has to schedule corrections for every 8 pixels
    - Setup overheads outweigh any iMX gains
- DC mode
  - MAX and MIN cannot be done using iMX efficiently
  - iMX cannot pad with v0/v1 and v8/v9 for 9-tap filtering
- Non DC mode
  - 4-point inner products are needed; inefficient on iMX
  - Touches only 2 pixels (not good enough smoothing)
- 2-3 conditions per row/column
    - Branching penalty on DSP is 5 cycles
    - This alone would be around 12-18 MCycles/s
- Frequent interaction between iMX and DSP makes multithreaded scheduling hard

Proposed Approach
- Solution to Data Access Issues

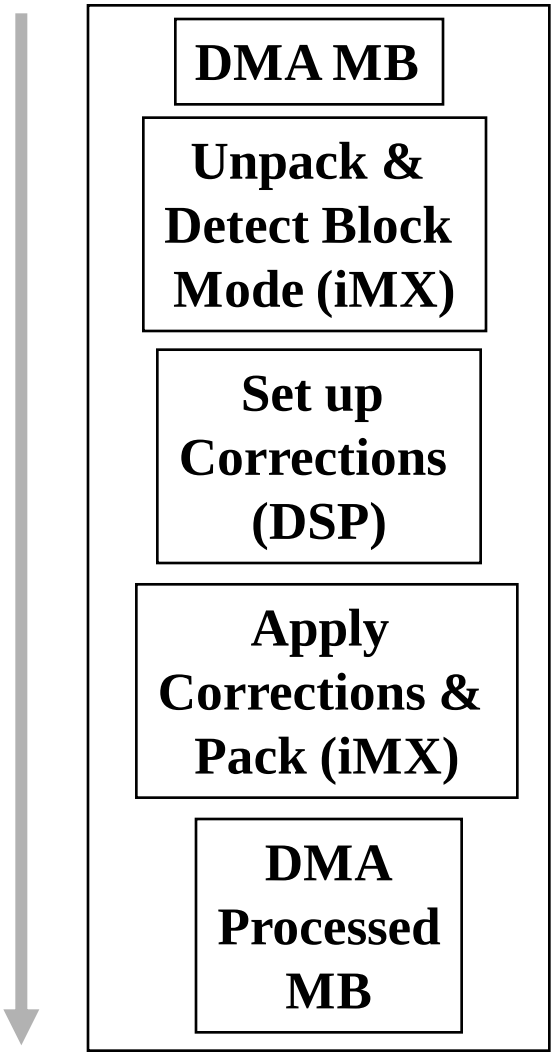
Issue	Solution
Horizontal and vertical deblocking happen on different set of pixels VMDeblocking	Perform both on the same set of pixels; exclude v0 and v9. For each MB in frame, leads to: • 1 MB load vs. 2MB loads • 1 MB store vs. 2 MB stores and a copy within image buffer

Proposed Approach

- Solutions to Adaptive Processing Issues

Issue	Solution
Adaptive decision per row/column	Combined decision for each block edge
Adaptive correction per row/column	Combined correction for each block edge
DC offset mode issues	Simplified DC offset smoothing
Non-DC offset mode issues	Adaptively correct up to 6 pels Extent of filter computed at the block level
Conditional scheduling on iMX for each row/column	Need to schedule only one per block edge
Multithreading is hard	Reduced DSP/iMX interaction

Scheduling & Multithreading



- Need to strike a balance between DSP and iMX processing times
- iMX operations are sensitive to the buffers in which the source and destination data are kept
- Using BUF-A and BUF-B, efficiently pipeline and hide the DSP and DMA operations by performing them in parallel with iMX

Deblocking - Results

Goto VMDeblocked
Goto Blocked

Deringing

- Since deringing need not be performed in flat regions, we can skip deringing for DC offset mode blocks
- Data from DC mode detection process is used to perform edge detection and index acquisition
  - instead of computing 2D MAX and MIN and then thresholding as in VM
- iMX can be used to efficiently perform 3x3 filtering
- iMX can be used to efficiently sum indices on 3x3 area

Deringing - Results

Goto Ringing

Conclusions

- Proposed approach
  - results in visual quality comparable to VM's
    - PSNRs were also comparable
  - Considerably simplifies porting to DSC25
  - effectively utilizes DSP, iMX, and DMA resources
  - enables postfiltering to be added to MuVPlayer™

Ittiam

END

VM Deblocking - Results

GoBack