A RISC CONTROLLED MOTION ESTIMATION PROCESSOR FOR MPEG-2 AND HDTV ENCODING

D. Charlot[1], J.-M. Bard[1], B. Canfield[3], C. Cuney[1], A. Graf[2], A. Pirson[1], D. Teichner[2], F. Yassa*[1]

[1]Thomson Consumer Electronics Components, Meylan, France
[2]Thomson Consumer Electronics, VS-Villingen, Germany
[3]Thomson Consumer Electronics, Indianapolis, U.S.A.

ABSTRACT

In this paper, we describe the architecture of a hierarchical motion estimation processor, with respect to the MPEG-2 encoding standard. This processor can also be used in HDTV applications. The motion estimation processing is in 2 steps: first full-pixel then half-pixel. Several modes are possible, depending on the image types (I, P or B - MPEG terminology, frame based or field based). A decision is taken in this processor to choose the best mode. The architecture is based on a RISC controller, external DRAMs to store anchor frames and specific hardware for processing the distortions. The architecture was chosen to achieve high performance, programmability and high memory bandwidth.

1. INTRODUCTION

In the new digital television world, an international standardization launched by the ISO was decided and MPEG-2 standard has recently appeared. It defines some algorithms to achieve high data compression and good picture quality. Among them, the motion compensation is a powerful tool: the encoder assumes that a 16*16 block of pixels ("macro-block" in MPEG terminology) has been translated by a motion vector, which is found using a "block matching" technique. The motion vector is then encoded and sent to the decoder.

A hierarchical approach is a good solution, using for example a first step to find a coarse motion vector in a big search window, and then to find a fine motion vector in a smaller search window. The goal of this article is to give an example of an algorithm to find the fine motion vector.

Due to the multiplicity of resolutions, bit rates or television formats (interlaced or progressive), it is important to obtain digital television representation techniques with maximum interoperability. A number of search modes are then defined and treated by the following algorithm.

2. HIERARCHICAL MOTION ESTIMATION DESCRIPTION

2.1. Global functionality

The algorithm needs 3 inputs for one given macro-block (16*16 pixels in luma, 8*8 chroma U pixels, 8*8 chroma V pixels) and for one given mode:

- the luma pixels of the given macro-block;
- the input vector for the mode, generated externally, for example from a decimated picture;
- the search window pointed to by the previous vector, whose size depends on the mode. The search window can be from a forward anchor frame or from a backward anchor frame.

The processor outputs will be then:

- the chosen mode (in luma)
- the luma vector(s) on half-pixel grid for this mode

The global processing algorithm is given on fig. 1.

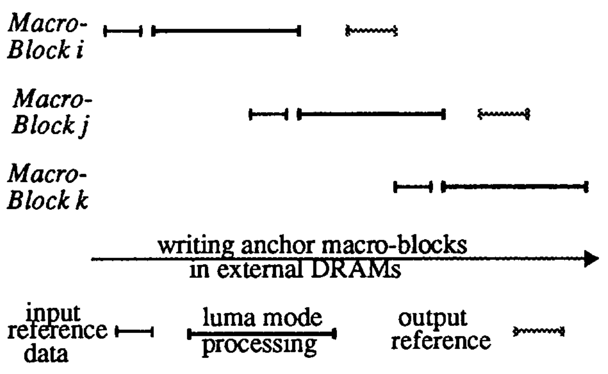


FIG. 1: **Global Processing Algorithm**

The macro-blocks are processed in pipe-line.

2.2. Hierarchical processing

* This work was completed while with TCEC (Thomson Consumer Electronics Components). Currently joined Motorola IC development Center.

0-7803-2431-5/95 $4.00 © 1995 IEEE

We describe here a processing for a given mode and for a reference macro-block (compute means here calculate the square error on each pixel and sum for the whole macro-block):

1. First get from the needed anchor (forward or backward) the search window, which may be reduced due to edge effects, knowing the input vector
2. Compute the distortions of the reference macro-block on the whole search window to find the best candidate (smallest distortion) in full-pixel.
3. Compute the distortions for the 8 half-pixel candidates which are around the best full-pixel candidate, getting rid of those which are outside the current search window.
4. Output the smallest distortion (which can be on half-pixel grid or full-pixel grid) and the corresponding vector on half-pixel grid, relative to the first pixel of the search window.

At the end of this process, we know what the distortion is for the given mode, and what the relative half-pixel vector is. Knowing the input vector, the size of the search window and the relative vector, we know the absolute half-pixel grid vector for this mode.

2.3. Motion estimation modes

Several modes to estimate the best candidate for a macro-block are possible in the processor, depending on the picture type, on the user who can program the number of modes to limit the process timing, and also on the edge effects which can forbid some modes.
The distortions can either be calculated on 16*16 pixels (frame luma macro-block, see Fig. 2) or on 16*8 pixels (field luma macro-block, see Fig. 3).

*2.3.1. 16*16 pixel modes*

MV0 mode: only one distortion computed, with an input vector equal to zero, on forward anchor frame and also on backward anchor frame (for bidirectional pictures).

Frame mode: a search window is defined, pointed to by an input vector, in which the best candidate 16*16 is searched, in a forward or backward frame.

Interpolated frame mode: in bidirectional picture, the best forward frame candidate and the best backward frame candidate are averaged to give one candidate 16*16. In that case, the result will be one distortion but two half-pixel vectors (one forward and one backward).

*2.3.2. 16*8 pixel modes*

The macro-block is divided in two sub-macro blocks - a and b, taking respectively even lines or odd lines, especially used

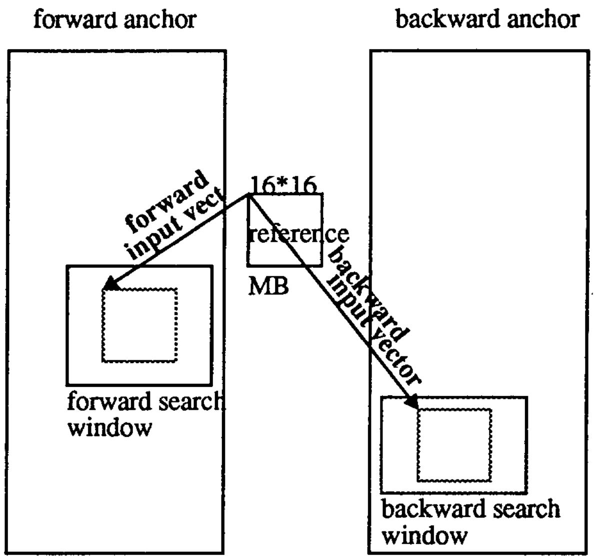
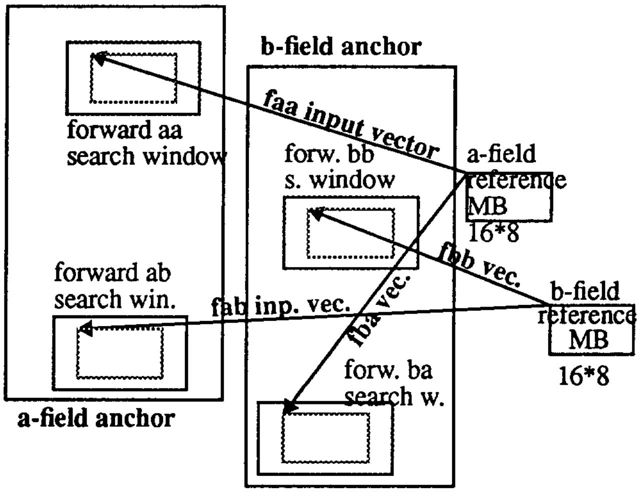


FIG. 2: **Frame mode**

in interlaced video. The following modes are processed on the sub-macro-blocks:

Field modes: a search window is defined, pointed to by an input vector, in which the best candidate 16*8 is searched, in a forward or backward a-field (even lines) or b-field frame (odd lines). One best candidate is kept from the a sub-macro-block, and one from the b sub-macro-block. The two corresponding distortions are added to be compared with frame modes. Two half-pixel vectors will be output if selected..



Same features for backward searches

FIG. 3: **Field mode**

Interpolated field mode: in bidirectional picture, the best forward a-field candidate and the best backward a-field candidate are averaged to give one candidate 16*8. One distortion is calculated with the a-sub-macro-block.

The same process on the b parts, a second distortion is then calculated.

Dual-prime mode [1]: In interlaced video, field candidate vectors are averaged with small search windows of the opposite field. Resulting distortions are composed to each other to find the best match.

2.3.3. Mode selection

To select the best mode, the mode with the best distortion is preferred.

3. ARCHITECTURE DESCRIPTION

On the processing part, we can distinguish the anchor data block, the reference block and the luma mode processing block (see Fig. 4). On the controlling part, we have the MMU (Memory Management Unit) and the RISC on-chip controller.

3.1. Main blocks

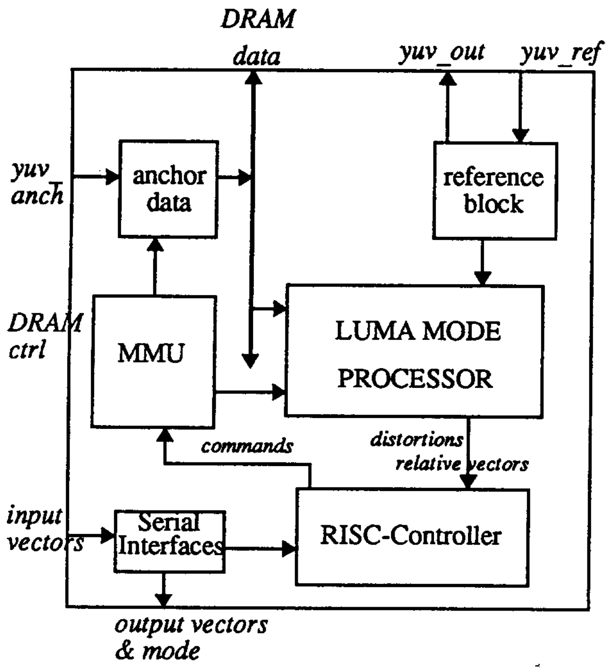


FIG. 4: **Architecture of the motion estimation processor**

3.1.1. Anchor data

This block is used to store the input data which are to be written in the external dram, that will be used as anchor frame for future macro-blocks. The pixel inputs (*yuv_anch*) are received serially and output 8 pixels at a time. Luma and chroma are treated here.

3.1.2. Reference block

This block is used to store the input macro-block which will be used as reference macro-block to be processed. The pixel inputs are received serially (*yuv_ref*). The luma output is 8 pixel wide to the block matching arrays (These 8 pixels are vertical, chosen among the 16 pixels depending on the mode, frame, a-field or b-field). The luma pixels and the chroma pixels are also output serially (*yuv_out*), at the same time as the half-pixel vectorsand the chosen mode.
ond BMA computes the search window delayed of one pixel. So after 17 cycles in field mode, 34 in frame mode, 2 distortions are available. Then the mincell selects the best one, and keeps the minimum for the full pixel search in the whole search window.

3.1.3. Luma Mode Processor

This block will calculate the distortions, depending on a search mode. The search windows, whose size is a parameter, are read from the external DRAM and the reference macro-block is read from the reference block. The process is done in full-pixel first then on half-pixel grid.
The outputs are then a relative half-pixel grid vector in the search window and the calculated distortions.

3.1.4. Serial interfaces

The input vectors are written bit-serially and read in parallel. The output half-pixel vectors and the chosen mode code are written in parallel and output bit-serially.

3.2. RISC controller

The functionality of this processor depends on many parameters (picture size, picture type, HDTV or not, search window size, mode choosing algorithm,...). A solution for the architecture is to have a programmable controller. A RISC is chosen for its simplicity. A RAM is also used for a part of the program, which is down-loaded at the initialization.
The RISC on-chip-controller has to manage the global sequencing of the process and to ensure a correct processing whatever the parameters are.

3.3. MMU synchronizer

The RISC does not drive directly the other blocks. The commands are given to the MMU (Memory Management Unit) at the same time as the external DRAM address and burst length. The MMU will give these commands to the chosen blocks only when the DRAM exchange starts. This enables an automatic synchronisation between commands and data.

4. CONCLUSION

The chosen architecture matches the main requirements:
- high memory bandwidth

- high flexibility.

ACKNOWLEDGMENTS

We want to acknowledge all those who facilitated the achievement of this work.

REFERENCES

[1] Preliminary Working Draft of MPEG-2/System, ISO/IEC JTC1/SC29/WG11 MPEG92, 6 Nov. 1992.

[2] Special issue on VLSI implementation for digital image and video processing applications, IEEE Transactions on circuits and Systems, Oct. 1989.