Transactions Briefs

A Parallel Bit-Level Maximum/Minimum Selector for Digital and Video Signal Processing

Chen-Yi Lee, Shih-Chou Juan and Wen-Wei Yang

Abstract— **This paper presents a novel circuit for parallel bit-level maximum/minimum selection. The selection is based on a label-updating scheme which sequentially scans a set of data patterns from MSB to LSB and generates corresponding labels. The complete circuit realizing this scheme consists of a set of updating units and a global OR unit, where each updating unit is composed of only a few basic gates. Due to structure modularity, the developed circuit provides a very cost-effective hardware solution for comparing large volumes of data patterns as those required in digital and video signal processing.**

I. Introduction

In the domain of digital and video signal processing, exhaustive search within specified regions has to be done repeatedly in order to find the maximum or minimum for decision making. Very often the search is achieved by a set of comparators implemented at both word-level and bit-level. This results in too much hardware overhead when on-chip system integration is considered since the search procedure is only part of the algorithm used. Moreover, comparison for large volumes of data patterns at word-level cannot reach high-speed requirements due to carry propagation effect. Although many circuit techniques for handling carry have been proposed in the literature [1], extra hardware is needed and hence hardware overhead is increased too. Therefore a cost-effective hardware solution which takes into account area and timing for parallel comparison of large volumes of data is demanded. In this paper we first present a scheme which can search the required maximum or minimum for a group of data patterns at bit level. Then a very cost-effective architecture and circuit is presented to see how to reach the goal. This circuit has been used in one of our ASIC designs for a VQ-based coding system [2].

II. The Label-Updating Scheme

To explain how this algorithm works, we provide one example as shown in Fig. 1. This example gives us the feeling about how a maximum or minimum can be found from an input data set. For example, the minimum can be obtained by comparing these bit-patterns from MSB to LSB. Initially all labels ($L_n, n = 0 \cdots N-1$, where N is the number of data patterns) are assigned to "0." We start from the MSB and check all bit patterns. If all bit patterns are the same, *i.e.* either all "1" or "0," all labels L_n's remain unchanged. On the other hand, if some of them are different, then those labels whose bit patterns are "1" are assigned to "1" indicating that they are no longer minimum candidates. For the rest, they remain unchanged. This implies that those labels with "0" are still the minimum candidates. This detection procedure is performed until the LSB is detected. Then the minimum can be found at those data patterns whose labels are "0."

To find the maximum, the same procedure can be applied. However we now focus on detecting "1" instead of "0." Thus the required timing to find maximum or minimum is dependent on the word-length of input data patterns and independent of input number (N). In other words, only the word-length influences the cycle-count for maximum/minimum search.

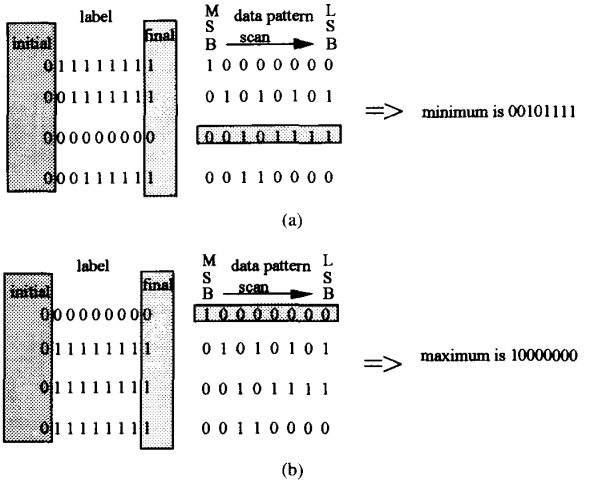

Fig. 1. An example demonstrates the label updating scheme for (a) minimum and (b) maximum detection.

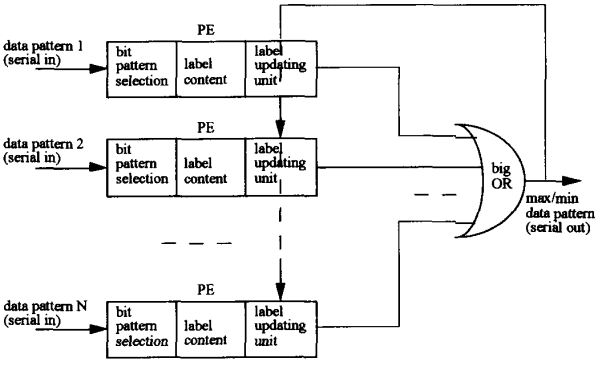

Fig. 2. PE-based architecture for the parallel bit-level maximum/minimum selector.

III. Circuit Design Techniques

The label-updating scheme can be realized on a PE-based architecture as shown in Fig. 2. Each PE handles single data pattern respectively and then updates the label content. The big OR gate is inserted to detect if there is any "0" (for minimum) or "1" (maximum) pattern from the possible candidates. This big OR gate has N inputs,

Manuscript received September 28, 1992; revised May 7, 1994. This paper was supported by the National Science Council (NSC) of Taiwan, ROC under Grant NSC-82-0404-E009-113 and partly supported by the NSC under grant NSC81-0404-E009-114. This paper was recommended by Associate Editor G. DeMicheli.

The authors are with the Department of Electronics Engineering and Institute of Electronics, National Chiao Tung University, 1001, University Rd., Hsin-Chu 300, Taiwan, ROC.

IEEE Log Number 9404007.

1057–7130/94$04.00 © 1994 IEEE

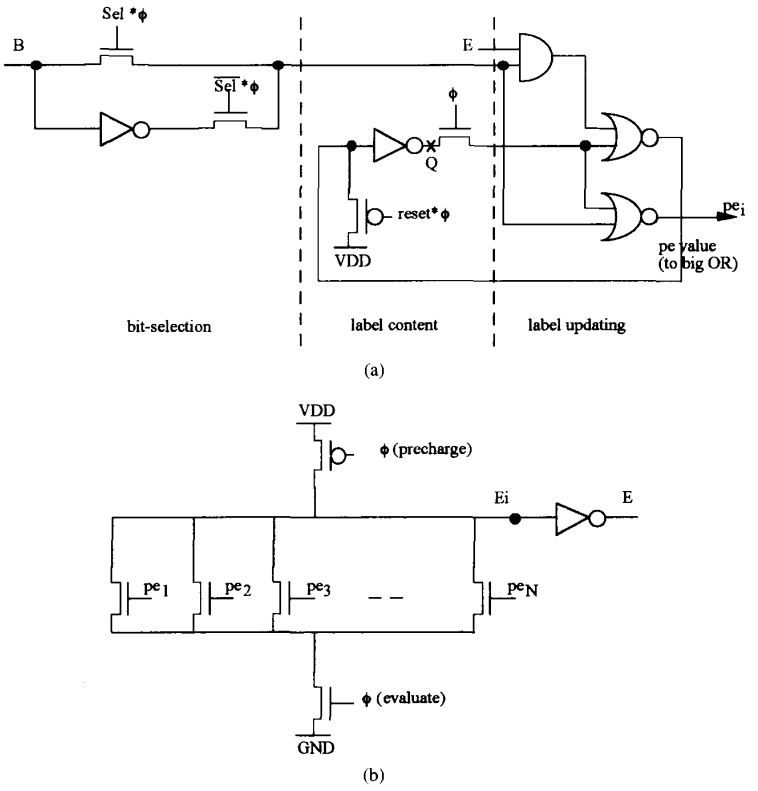

Fig. 3. (a) Circuit diagram of each PE for the parallel bit-level maximum/minimum selector; (b) Pre-charge circuit diagram for the big-OR unit.

where each input is integrated with the PE in layout design. In addition, driving circuit is added every 8-PE to gain speed. In the following, we describe how the complete circuit for such a PE is designed.

From the scheme description in the previous section, it can be found that for maximum and minimum search, they are very similar except that different input phases (inverted and non-inverted) are needed to be detected. For minimum search, "0" should be detected while "1" is detected for maximum search. Thus an input selector should be provided at the input node. This is shown in Fig. 3, where a control signal *Sel* is used to indicate the search mode and in the meantime, to select the correct bit-pattern. To provide a systematic way, we use several symbols as explained below: B stands for *input bit pattern*, E stands for *enable*, pe stands for each PE status, and Q stands for *label's content*. For minimum search, we would like to find if there any bit pattern "0" exists among those candidates. Thus we need to check each input bit pattern by the following equation:

$$\overline{Q}\,\overline{B} = \overline{\overline{Q}\,\overline{B}} = \overline{Q + B} \tag{1}$$

This implies that for minimum search, only those possible candidates whose bit patterns are "0" are needed to detect. For maximum search, we need to check each input bit pattern by the following equation:

$$\overline{Q}\,B = \overline{\overline{Q}\,\overline{B}} = \overline{Q + \overline{B}} \tag{2}$$

The equation shows that for maximum search, only those possible candidates whose bit patterns are "1" are needed to detect. These two equations are derived from label content Q and input bit pattern B. They also imply that for minimum search *Sel* should be assigned to "1" while "0" for maximum.

Next we consider how to update label content. Since initially all labels are assigned to "0" and only those labels remaining unchanged becomes possible candidates, the updating procedure can be described as follows:

- for possible candidates, the content to be assigned is based on the input bit pattern and the *enable* signal obtained from the big OR gate. For example, during minimum search, only those bit patterns with "0" can survive. This implies that bit pattern B can be directly loaded to Q. However for maximum search, the inverse bit pattern $\overline{B}$ has to be loaded to Q to indicate that only those bit pattern with "1" can survive. Fortunately, these signals are the same as those needed in (1) and (2) mentioned earlier. Thus the signals can be connected directly as those shown in Fig. 3(a).
- for non-candidates, the content remains unchanged, *i.e.* "1."

The other part of the circuit is the big-OR which is realized by a clock-mode circuit as shown in Fig. 3(b). During $\overline{\phi}$ input node at Ei is pre-charged to high and hence the output node E remains at low. During ϕ, outputs from all PEs are evaluated to check if any desired patterns exist, *i.e.* $\overline{Q + B}$ or $\overline{Q + \overline{B}}$. Through this technique, a very high-speed circuit can be obtained.

The complete circuit works as follows. During $\overline{\phi}$, E is assigned to "0" while during ϕ bit patterns are selected and passed to the label updating units to generate *evaluate* (pe) signals which are connected to the bit-OR unit so that the *enable* signal is produced to orchestrate the label content. Therefore only one-phase clock is required. Note that these pe signals do not change during evaluating the OR gate because (1) if $Q = 1$, then pe always remains 0 and (2) if $Q = 0$, pe will probably change only after the evaluation of big-OR

is done. One very important feature about this circuit is the required maximum/minimum bit patterns can be obtained respectively right after each parallel bit detection. This is very useful for extension when large volumes of data patterns are to be searched.

To evaluate the efficiency of this circuit, we have used an example of a VQ-based video encoder [2], which requires parallel comparison of 32 8-bit inputs to determine an index with minimum distortion. We first implemented the parallel comparison by using carry propagation and pass logic functions [1]. The parallel comparison is organized in a tree-structure and consists of 31 comparators which can be speed up by using carry-look-ahead or carry-save techniques. We then implemented the parallel comparison by the circuit proposed in this paper. Results show that our proposed method achieves 8 times faster and with only one-tenth of the area of the other methods [3], indicating that our proposed method does outperform the other comparator-based approaches in terms of area and timing.

IV. Conclusion

We have presented a novel scheme and circuit for parallel bit-level maximum/minimum selection. The circuit does need less area and also has features of high-speed and extension capability. We are currently investigating the practicality of applying this circuit to several video coding algorithms which require extensive block search operations.

Acknowledgment

The authors would like to thank their colleagues within the VLSI/CAD group of NCTU for many fruitful discussions and suggestions.

References

[1] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Reading, MA: Addison Wesley, 1985.

[2] C. Y. Lee and S. C. Juan, "VLSI implementation of a modified-VQ encoder suitable for image/video coding," *Proc. of EUSIPCO '92*, Brussels, Belgium, Aug. 1992.

[3] S. C. Juan, "Fast algorithm and architecture for VQ-based image/video coding," NCTU/DEE Master Thesis, Hsinchu, China, July 1992.

Unitary FIR Filter Banks and Symmetry

R. A. Gopinath and C. S. Burrus

***Abstract*— Recently, unitary FIR filter banks with linear-phase have been completely parameterized [8] by exploiting the eigenstructure of the exchange matrix. This correspondence gives new characterizations of polyphase matrices of filter banks with various types of symmetry on the filters. Using matrix extensions of the well-known hyperbolic and orthogonal lattices, we give a alternative proof for the parameterization of linear-phase unitary filter banks. A complete parameterization of FIR unitary filter banks with each of the different types of symmetries considered (not just linear-phase) is also given. These results can also be used to generate non-unitary filter banks with symmetries, though no completeness results can be obtained. In some cases implicit, and in others explicit parameterization of wavelet tight frames associated with these filter banks are also given. This paper only considers filter banks with an even number of channels. A similar theory can be developed if the number of channels is an odd integer.**

I. Introduction

Consider an M-channel filter bank (see Fig. 1) with filters whose passbands approximate ideal filters (see Fig. 2). Clearly several transformations relate the M ideal filter responses. For example, the ideal filters could be related by modulation as seen in the theory of modulated filter banks [5], [2].

From both design and implementation points of view, unitary modulated filter banks are the most efficient. However, a unitary modulated filter bank (with desired response given by Fig. 2) cannot have linear-phase filters [1]. On the other hand, recently, a complete parameterization of unitary filter banks with linear-phase has been obtained [8].

This paper considers the following types of symmetries in filter banks: In Fig. 2, the response of the $M-1-i^{th}$ filter can be obtained by shifting the response of the i^{th} filter by π. Therefore, they could be related as

$$\begin{aligned} h_{M-1-i}(n) &= (-1)^n h_i(n) \\ H_{M-1-i}(z) &= H_i(-z) \\ H_{M-1-i}(\omega) &= H_i(\omega+\pi) \end{aligned} \tag{1}$$

or

$$\begin{aligned} h_{M-1-i}(n) &= (-1)^n h_i(N-1-n) \\ H_{M-1-i}(z) &= H_i^R(-z) \\ H_{M-1-i}(\omega) &= H_i^*(\omega+\pi) \end{aligned} \tag{2}$$

where N is the filter length and for causal $H(z)$, $H^R(z)$ denotes its reflection (*i.e.* causal time-reversed sequence). The former will be called *pairwise-shift* (or PS) symmetry (also known as pairwise-mirror image symmetry [6]), while the latter will be called *pairwise-conjugated-shift* (or PCS) symmetry (also known as pairwise-symmetry [6]). Both these symmetries relate pairs of

Manuscript received September 29, 1992. This work was supported by AFOSR under grant 90-0334 funded by DARPA. This paper was recommended by Associate Editor M. A. Soderstrand.

R. A. Gopinath is with IBM Watson Research, 30 Saw Mill River Road, Hawthorne, NY 10532 USA.

C. S. Burrus is with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77251 USA.

IEEE Log Number 9404008.

1057-7130/94$04.00 © 1994 IEEE