C-slow retimed parallel histogram architectures for consumer imaging devices

Article

Accepted Version

Cadenas Medina, J. O., Sherratt, R. S. ORCID: https://orcid.org/0000-0001-7899-4445, Huerta, P., Kao, W.-C. and Megson, G. M. (2013) C-slow retimed parallel histogram architectures for consumer imaging devices. IEEE Transactions on Consumer Electronics,, 59 (2). pp. 291-295. doi: https://doi.org/10.1109/TCE.2013.6531108 Available at https://centaur.reading.ac.uk/33371/

Published in the IEEE Transactions on Consumer Electronics,
Volume 59(2), May 2013, pp. 291 - 295,
DOI: 10.1109/TCE.2013.6531108

Title: **C-slow Retimed Parallel Histogram Architectures for Consumer Imaging Devices**

José Cadenas, School of Systems Engineering, the University of Reading, RG6 6AY, UK (e-mail: o.cadenas@reading.ac.uk).

R. Simon Sherratt, School of Systems Engineering, the University of Reading, RG6 6AY, UK (e-mail: sherratt@ieee.org).

Pablo Huerta, Department of Computer Architecture and Technology, the Universidad Rey Juan Carlos, 28922, Spain (e-mail: pablo.huerta@urjc.es).

Wen-Chung Kao, Department of Applied Electronics Technology, National Taiwan Normal University, Taipei, 10610, Taiwan, (e-mail: jungkao@ntnu.edu.tw).

Graham M. Megson, School of Electronics and Computer Science, University of Westminster, London W1T 3UW, UK, (e-mail: g.megson@westminster.ac.uk).

Abstract

A parallel pipelined array of cells suitable for real-time computation of histograms is proposed. The cell architecture builds on previous work obtained via C-slow retiming techniques and can be clocked at 65 percent faster frequency than previous arrays. The new arrays can be exploited for higher throughput particularly when dual data rate sampling techniques are used to operate on single streams of data from image sensors. In this way, the new cell operates on a p-bit data bus which is more convenient for interfacing to camera sensors or to microprocessors in consumer digital cameras.

Index Terms

Parallel Histograms, Pipelined Array, FPGA, Digital Imaging, Image Processing.

I. Introduction

Image analysis based on histograms is abundant and well used in many consumer applications [1]. A histogram is a mapping from a set of data values to a set of non-negative integers, or bins. The computation of histograms may take considerable time if it is performed by an embedded microprocessor. Thus a dedicated hardware histogram module is desirable for practical consumer applications.

An array of cells to perform the computation of *m*-bin histograms that takes *k* pixels per clock cycle offers a gain in speedup factor of *k*. Such a design has been previously proposed [2], but required a sensor or processor supplying four pixels per clock cycle ($k = 4$) to get a speedup of four. Given a set of *p*-bit pixel values, then $k \times p$ interface wires are required for a speedup of *k*. However, embedded microprocessors consist of *w*-bit data buses ($w = 8$, 16 is common [3]) and consequently are limited to supply *k* pixel per clock cycle in cases where $w < kp$. In order to overcome the limitation for this case, a histogram solution using C-slow retiming to create two sub streams of computation derived from a dataset arriving at one pixel per clock cycle is proposed, that is, fix $k = 2$. The proposal presented here, extended from previous work [4], is very easy to be implemented. For the case where $w \geq 2p$, the proposed hardware design contains a histogram array accepting two streams of pixels per clock cycle using the new C-slow retimed cells presented here. For the case where $w < 2p$, a mechanism operating on a fast single stream of data sampled at a dual data rate (on both clock edges of the clock) is also presented to generate two streams of computation. The mechanism acts as a front-end to the same array solution to the former case.

This paper briefly explains the principle of C-slow retiming in Section II and applies C-slow to fully develop the proposed cells in section III with a brief analysis of the results in section IV before presenting final conclusions. The essential result is that the proposed design provides speed-up while also facilitates easier interfacing to camera sensors or microprocessors compared to other designs.

II. C-slow Retiming

C-slow retiming is a method used to reduce the critical path delay in digital circuits especially when feedback loops exist [5]. Every register in the data path is replaced by C registers and then all registers are moved along the critical data paths using a retiming algorithm. C-slow retiming separates the calculation performed in the original data path into C instances. Fig. 1 shows an excerpt of the data path of a histogram cell as previously presented [4] that includes a feedback path (left), its C-slow version by a C factor of two (center) and after retiming to get a C-slow retimed version (right). A simple example using Fig. 1 illustrates the principle of retiming. For input sequence u = 3, 5, 4, 1 the left diagram in Fig. 1 produces r = 0, 3, 8, 12, 13; the leading zero reflects the register delay with output r being the running accumulation on input u. The diagram on the right of Fig. 1 gives r = 0, 0, 3, 5, 7, 6 for the same input u. The output corresponds to the accumulation as if there were two separate input streams: u_0 = 3, 4 and u_1 = 5, 1 and as such the output has been separated into r_0 = 3, 7 and r_1 = 5, 6; and the two interleaved into output r. In general C-slow retiming creates C interleaved streams of computation and as such also requires C input data streams. For practical reasons related to the design, only the factor C = 2 is considered in the rest of the discussion.

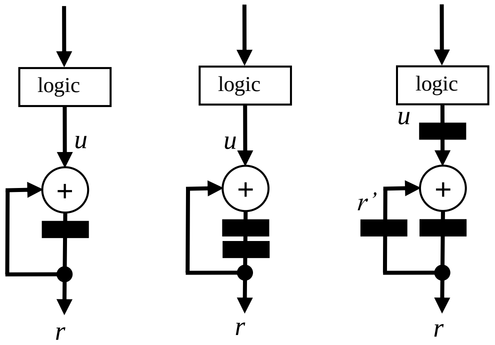

Fig. 1. Pipelined datapath with feedback (left), C-slow with C = 2 (center) and C-slow retimed (right) [4].

III. Implementation

A. C-slow Retimed Histogram Cell

A C-slow retimed (C=2) processing cell for the computation of histograms in a pipelined fashion is presented in Fig. 2. Note this cell is not parallel as it is processing a single data item (of p-bit) per clock cycle; it is included here to make it easier to understand the later discussion on the parallel case. The cell structure follows straightforwardly from the discussion in Section II and the pipelined histogram cell structure [2]. It follows that, the separation of the computation into two streams does require the use of the extra adders as seen at the bottom of Fig. 2. Registers in solid black are the original ones in the pipelined cell. The new registers introduced by C-slow retiming are shown in gray. The registers filled in with white are registers that should be there from the C-slow transformation, but that can be conveniently removed. This is justified from the observation that the bin update mechanism (which is the one having the feedback loop) proceeds down the cell for a data item value *xin* equal to *sin*, otherwise *xin* simply moves on to the next cell on the right of the histogram array. Thus, there is no need to slow down the data moving left to right on Fig. 2. This is convenient as array latency is not increased from left to right (histogram calculation); all cells simply have added an extra delay for the bin count update.

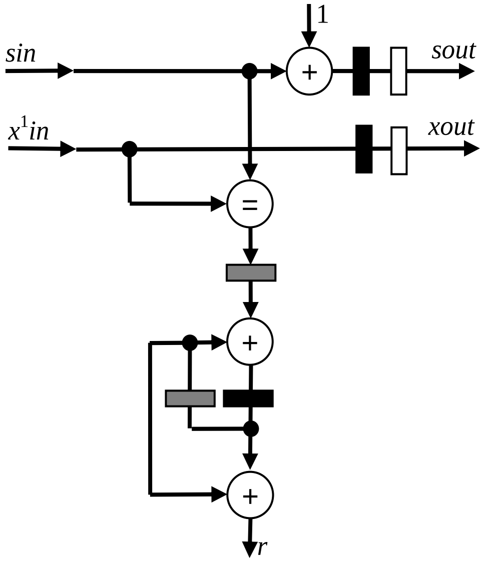

Fig. 2. C-slow retimed (*C*=2) scalar pipelined histogram cell requiring an extra adder per cell to merge the two created C-slow streams.

A similar situation is presented for the logic required to read a histogram from an array. This logic is shown in Fig. 3, given for two adjacent cells in an array. The *r* output from cell *i*+1 is selected at multiplexer from cell *i*; this occurs when reading bin values after a histogram computation is ready. It has been fully explained previously how the reading proceeds from left to right in an array in order to hide some latency when operating in streaming mode [2]. The main point here is to note that the outputs *r* (bin counts) from the cell in Fig. 2 are registered, and the histogram reading outputs *hout* in Fig. 3 are also registered. By the same reasoning, there is no need to duplicate the pipeline registers in this reading mechanism by C-slowing with a factor of two. The mechanism to read bins out from the cell in a pipelined fashion is omitted for simplicity in the rest of this paper.

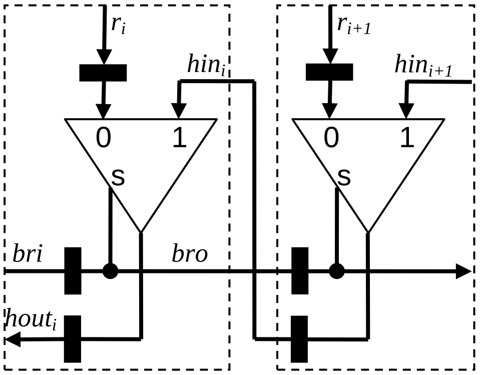

Fig. 3. Logic to read histograms bins r_i from an array through a single output port *hout*; any two adjacent cells are shown.

B. C-slow Retimed Parallel Histogram Cell

A C-slow retimed (*C*=2) processing cell for the computation of two bins per cell while accepting two input data items in parallel is presented in Fig. 4. The cell's structure follows the above discussion and the histogram cell presented [2]. The registers introduced by C-slow retiming are still shown in gray. Notice there is a pipeline register at each *r* output because this has been moved onto Fig. 4 from the register on input '0' of the multiplexer in Fig. 3, to clearly illustrate that the critical path of the cell is most likely to be just the adder block. This is because, since comparisons are shallower; comparisons work on pixels values while the adders are working on bins values (the block of Logic is a simple three bit logic function). For an image of 640×480, bin values are of 19 bits, thus adders are wider than comparisons since typical pixels values are expressed in 8, 10, 12 or 16-bit. Without C-slow retiming the critical path is due to the compare-logic-accumulation chain.

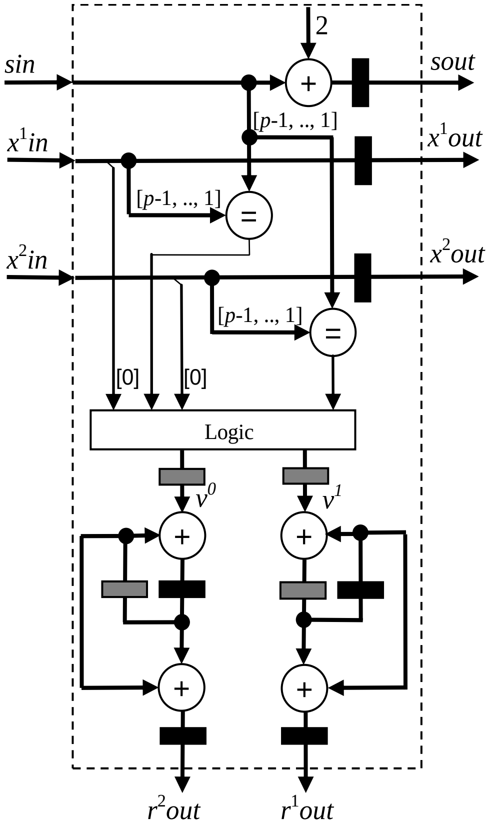

Fig. 4. C-slow retimed internal cell structure processing two data items per clock cycle and computing two histogram bins within the cell.

IV. RESULTS

Table I shows the ratios in area and frequency of operation for the new cells and arrays in Fig. 2 here, in relation to the cell and arrays as previously presented [2]. These ratios are given for 0.35 µm ASIC (Application Specific Integrated Circuits) and modern FPGA (Field Programmable Gate Array) technology. The table indicates that the designer can pay a penalty of 70% more area to have the advantage of a 20% increase in frequency of operation. Note this is not a parallel case, but a simple pipelined case. Later, it will be shown the results improve for the parallel case.

A parallel design based on the cell in Fig. 4 was fully implemented and validated using ASIC technology of 0.35 µm giving the results in Table II. Although the C-slow cell is only around 25% faster than a comparable cell [2], the real advantage can be seen when arranging C-slow cells as an array. There are two obvious ways of exploiting the gain in the frequency of operation of an array of C-slow parallel cells; one merging the two parallel inputs into a single one for scalar operation and the other preserving the parallelism. These are detailed below.

TABLE I, Area and Frequency Ratios for Cells and Arrays

	ASIC		**FPGA**	
	Area	Frequency	Area	Frequency
Cell Fig. 2/Cell [2]	2.19	1.09	1.71	1.03
Array Fig. 2/Array [2]	2.20	1.08	1.72	1.20

TABLE II, Histogram Array Frequency and Area

	MHz	**No. gates**
Cell [2]	226	562
C-slow retimed Fig. 4	282	1366
Histogram array [2]	144	86336
Histogram array of C-slow cells of Fig. 4	238	194840

A. Processing one data item per clock cycle

The C-slow cell in Fig. 4 requires and processes two input data items per clock cycle. Assume the cell of Fig. 4 is fed with every other data item (from an input dataset of n items) every clock cycle: half the items go into the array stream piped through x^1*in* input and the other half into through x^2*in* input. This is easily achieved by a de-multiplexer and very similar to the operation explained in Section II. As a result, an array processes a single data item per clock cycle (in this case a trivial modification is required when reading the histogram that has been omitted.) Thus, the histogram is computed in $n + m$ clock cycles. A parallel pipelined array accepting two data items per clock cycle computes the histogram in $n/2 + m/2$ clock cycles with each cell processing two bins; $m/2$ is the latency. As $n >> m$ for typical image sizes, latency can be ignored for a quick analysis. Arrays of C-slow retimed cells can be clocked 65% faster than the histogram arrays previously proposed [2]. In fact, from Table II, ratio $T_{\text{pipe}}/T_{\text{C-slow}} = 1.65$ between the pipelined array and the C-slow array, then the time to compute the histogram for any dataset of size n with the C-slow array (one data item per clock cycle) reaches over 80% of the throughput delivered by a parallel (of two data items per clock cycle) pipelined array.

B. Processing two data items per clock cycle

Fig. 4 processes two data items per clock cycle. For a data item of p-bit, the cell requires $2p$ wires on the datapath. It would be most convenient to maintain the throughput due to the inherent parallelism and higher clock rate gain without the need of making the data path wider. Assuming a single stream of data s arrives at double speed rate, two data items per clock cycle can be separated from s using a Double Data Rate (DDR) arrangement [7]. The principle of operation of the DDR is shown in Fig. 5. The single stream s is distributed into two sub streams s^1 and s^2 by a dual-edge triggered flip-flops arrangement, so streams s^1 and s^2 are each generated at a frequency f_{clk}. Then s^1, s^2 are fed into x^1*in* and x^2*in* of Fig. 4 respectively. This implies that the registers associated with r^1*out* in Fig. 4 operate on the rising edge of the clock, while the registers associated with r^2*out* in Fig. 4 must operate on the falling edge of the clock. This requirement is unnecessary; adding an extra register to the stream s^2 (before is taken by the histogram array) makes all registers in the cell of Fig. 4 to operate on the same edge of the clock at the expense on an extra clock cycle delay. Overall, latency is $m/2 + 3$ clock cycles, and the histogram is computed in $n/2$ clock cycles. As seen from Table II, this arrangement is 1.65 times faster than the equivalent design previously presented [2].

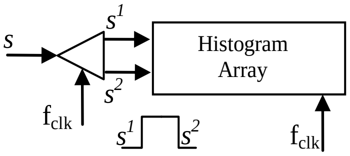

Fig. 5. Principle of operation of a double data rate to create two input streams out of a single input stream

The operation of a histogram array is shown in Fig. 6 where the *input* (stream s) arrives at double the speed compared to the clock *clk*. For simplicity, the input data values are restricted to values in the range 0-7, and 8 bins are being computed in the histogram. It is seen the histogram for the 18 data values is computed in 18/2 + 3 clock cycles at the frequency of *clk* signal. A DDR mechanism based on dual-edge flip-flops suitable for ASIC technology and current synthesis tools was evaluated for a data item of $p = 8$ bits. This resulted in an extra cost of 186 gates and can operate at 345 MHz; thus such a mechanism can easily be used as a front-end to feed data to an array of C-slow cells with a negligible impact on area. The DDR mechanism is supported in all inputs and outputs pins of modern FPGAs devices and also the rich configuration capabilities of their clock manager blocks makes the generation of operation as shown in Fig. 6 particularly simple [8].

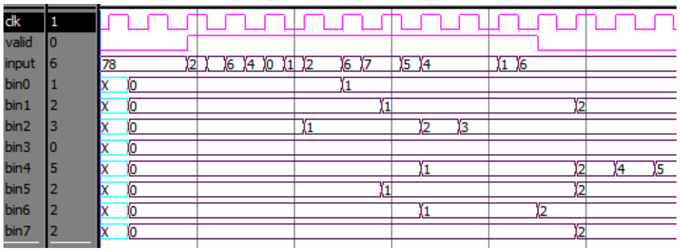

Fig. 6. A 8 bins histogram is computed on a single *input* stream of 18 data values in 18/2 + latency clock cycles; *input* is fed on each *clk* edge.

V. Discussion

Many image pipelines used in digital cameras are designed with 32-bit data paths to enable image transformations, e.g. color correction and Automatic While Balance (AWB) which is typically computed using floating point operations. Also, some operations (e.g. tone reproduction and gamma correction) are non-linear in nature which does require high precision calculations. However, histogram analysis is mostly used to provide statistics of the distribution of the intensity data and in practice sixteen bits are considered to be enough (p = 16) for consumer applications. Histogram analysis might be applied on several stages inside the image pipeline where different precisions might be required in different stages. For auto exposure, 10-bit resolution is considered a good compromise whereas tone reproduction or contrast enhancement higher resolution is often required. This work allows providing histogram analysis on p-bits and a designer may judge the corresponding tradeoff associated with the area/throughput on an available data path engine of w bits based on the relationship between w and $2p$. For example, this paper provides a simple interface for processing two pixels of 8-bit on 16-bit data paths, or for sustaining an equivalent throughput on a 16-bit data path for pixels of 12-bit.

VI. Conclusion

This paper has presented a new array of cells to compute m-bins histograms on streams of one pixel per clock cycle at over 80% of the performance of a fully pipelined array, working on streams of two pixels per clock cycle. The developments are due to arrays of C-slow cells achieving 65% faster clocks than previous pipelined arrays. Yet another way of exploiting the faster clock rate of the new C-slow cells is using a dual data rate mechanism that results in 65% higher throughput. A dual rate mechanism is easily implemented in ASIC technology and already available in FPGA devices. In either way, a sensor or microprocessor only needs to interface as many wires as required by a pixel value to the histogram array; a 1.65 higher throughput is reported here using an interface of p-bit wires compared to previous designs of $2p$-bit wires.

References

[1] H.-C. Huang, F.-C. Chang and W.-C. Fang, “Reversible data hiding with histogram-based difference expansion for QR code applications,” *IEEE Trans. Consumer Electron.*, vol. 57, no. 2, pp. 779-787, May 2011.

[2] J. O. Cadenas, R. S. Sherratt, P. Huerta and W. C. Kao, “Parallel pipelined arrays for real-time histogram computation in consumer devices*,” IEEE Trans. Consumer Electron*., vol. 57, no. 4, pp. 1460-1464, Nov. 2011.

[3] K. Yoon, C. Kim, B. Lee and D. Lee, “Single-chip CMOS image sensor for mobile applications,” *IEEE J. on Solid State Circuits*, vol. 37, no. 12, pp. 1839-1845, Dec. 2002.

[4] J. O. Cadenas, R. S. Sherratt, P. Huerta, and W. C. Kao, “Parallel pipelined histogram architecture via C-slow retiming,” IEEE Int. Conf. on Consumer Electronics, 2013, accepted.

[5] C. Leiserson, F. Rose and J. Saxe, “Optimizing synchronous circuits by retiming,” 3rd Caltech Conf. on VLSI, 1993.

[6] N. Weaver and J. Wawarzynek, “The effects of datapath placement and C-slow retiming on three computational benchmarks,” 10th Annual IEEE Symp. on Field-Programmable Custom Computing Machines, 2002, pp. 303-313.

[7] R. S. Sherratt and Oswaldo Cadenas, “A double data rate architecture for OFDM based wireless consumer devices,” *IEEE Trans. Consumer Electron.*, vol. 56, no. 1, pp. 23-26, Feb. 2010.

[8] Xilinx, Inc., “Virtex-6 Family Overview,” DS150 (v2.4) January 19, 2012.

Biographies

José Cadenas (M’10) received the M.Eng. in Electronics from Unexpo, Venezuela. After a brief period in industry he joined ULA, Venezuela as lecturer in digital logic design and completed the M.Sc. degree in Control Systems in 1994. He later received the Ph.D. degree in computer science from the University of Reading in 2002 where he currently is a lecturer in the School of Systems Engineering. Dr Cadenas works mainly on hardware parallel algorithms and its modeling and simulation as digital designs in RTL. Some of his designs, acting as private consultant, are used worldwide by commercial customers.

R. Simon Sherratt (M’97-SM’02-F’12) received the B.Eng. degree in Electronic Systems and Control Engineering from Sheffield City Polytechnic, UK in 1992, M.Sc. in Data Telecommunications in 1994 and Ph.D. in video signal processing in 1996 both from the University of Salford. Since 1996, he has been a Lecturer in Electronic Engineering at the University of Reading, currently a Senior Lecturer in Consumer Electronics and a Director for Teaching and Learning. His research topic is signal processing in consumer electronic devices concentrating on equalization and DSP architectures.

Eur Ing Dr. Sherratt has served the IEEE Consumer Electronics Society as a Vice President (Conferences) (2008/9), AdCom member (2003-2008, 2010-) and Awards chair (2006/7). He is a member of the IEEE Transactions on Consumer Electronics Editorial Board (2004-) and the current Editor-in-Chief, the IEEE International Conference on Consumer Electronics (ICCE) general chair in 2009 and the IEEE International Symposium on Consumer Electronics (ISCE) general chair in 2004. He received the IEEE Chester Sall 1st place best Transactions on Consumer Electronics paper award in 2004 and the best paper in the IEEE International Symposium on Consumer Electronics in 2006.

Pablo Huerta graduated in Communication Engineering from the Universidad de Cantabria, Spain in 2003. He received the Ph.D. degree in Computer Science from Universidad Rey Juan Carlos, Spain in 2009, where he was a lecturer in the Department of Computer Architecture. He has now joined a company in the private sector implementing digital designs for FPGA technology.

Wen-Chung Kao (M’05-SM’08) received the M.S. and Ph.D. degrees in Electrical Engineering from the National Taiwan University, Taiwan, in 1992 and 1996, respectively. From 1996 to 2000, he was a Department Manager at SoC Technology Center, ERSO, ITRI, Taiwan. From 2000 to 2004, he was an Assistant Vice President at NuCam Corporation in the Foxlink Group, Taiwan, where he was responsible for leading embedded software team to develop digital still/video cameras. In 2002, he was also invited to form SiPix Technology Inc., Taipei, Taiwan, where he was in charge of setting up the research team of the company and studying the flexible electrophoretic display. Since 2004, he has been with National Taiwan Normal University, Taipei, Taiwan, where he is currently a Full Professor and the Chairman of Department of Applied Electronics Technology. His current research interests include system-on-a-chip (SoC) as well as embedded software design, flexible electrophoretic display, digital camera system, robotic vision system and color imaging science.

In 2006, he received the Best Paper Awards at the 2006 IEEE International Symposium on Consumer Electronics and the IEEE 2006 Asia-Pacific Conference on Circuits and Systems. In 2011, he received the Best Paper Awards again at the 2011 IEEE International Symposium on Consumer Electronics. He currently serves on the Editorial Board of the IEEE Transactions on Consumer Electronics. He is a member of the Technical Program Committee of the IEEE International Conference on Consumer Electronics and IEEE International Symposium on Consumer Electronics general co-chair in 2013.

Graham Megson received a BSc in Computational Science from Leeds University (1984), and Ph.D. in systolic algorithm design from Loughborough University (1987). He was a research fellow at Oxford University before joining Newcastle University (1989) and appointed Professor at Reading University (1995). He is currently with Electronics and Computer Science at University of

Westminster. He was editor-in-chief of the journal of Parallel Algorithms and Application (1997-2005) and continues to act on the editorial boards of a number of journals. He has published over 200 research papers and eight books on the areas of algorithmically specialized and parallel processing.