Microprocessors and Microsystems 35 (2011) 297–307

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Parallel processing speed increase of the one-bit auto-correlation function in hardware

Nicolas Huber [a,*], M.S. Hromalik-Pouchet [b], T.D. Carozzi [c], M.P. Gough [c], A.M. Buckley [c]

[a] Biomedical Engineering Group, University of Sussex, Brighton BN1 9QT, United Kingdom
[b] Lab. of Atomic & Solid State Physics, Cornell University, Ithaca, NY, USA
[c] Space Science Centre, University of Sussex, Brighton BN1 9QT, United Kingdom

ARTICLE INFO

Article history:
Available online 19 January 2011

Keywords:
Parallelization
Counterbased
Auto-correlation
FPGA

ABSTRACT

Recently, a serial implementation of the one-bit auto- and cross-correlation functions (ACF and CCF respectively) in a field programmable gate array (FPGA) has been developed, based on asynchronous delay elements and counters, known as the counterbased correlation. This paper proposes a method of parallelizing this otherwise serial process, offering significant improvements in the applicability of this approach to more types of ACF. Furthermore, the possibility of obtaining lag results from a parallel data sequence without first shifting the entire sequence has been realized, hence decreasing the number of clock cycles necessary for the calculation of the ACF. A synchronous design was preferred here for reasons of stability and portability, the technology of choice again being an FPGA. The advantages offered by the counterbased implementation in terms of device area usage and speed still apply. A practical implementation in the instrumentation of an upcoming space mission is also discussed.

1. Introduction

A wide sense, random, stationary signal is fully characterized by its auto-correlation function (ACF). Hence the ACF in various forms enjoys extensive use in a wide range of applications. It can be found in such diverse fields as economics [1], image processing [2], multiple areas of astronomy [3] and Brownian motion studies [4], to name but a few.

An ACF calculates the degree of association between the elements of a single series separated by some lag. Its purpose is to detect non-randomness in data, and to identify an appropriate time series model if the data is found to be non-random [5]. For a periodic sequence $(X_i)_{i=0}^{N-1}$, it generally takes the form of

$$R_X[i] = \sum_{k=0}^{N-1} X[k]X[k+i] \tag{1}$$

Significant effort has been made to implement correlation functions in hardware, due to the speed-up offered by the possibility of parallel processing. These functions are currently being deployed in FPGAs, while traditionally DSP devices have been the technology of choice [6], as well as custom-made ASIC devices [7]. In [8], a CCF was implemented in a FPGA, where the function was defined using low-level schematic capture, probably due to the availability of optimized schematic components. The significant work by Von Herzen [9] shows the possibility of very fast (up to 250 MHz) CCF calculation in reconfigurable computing engines. Further to this, there has also been some investigation into ACFs in particular, and their implementation in a FPGA. In the work by Bezerra et al. [10] an ACF developed using VHDL targeting an FPGA was directly compared with the exact same algorithm as carried out by a pair of microcontrollers of the 8051 family, as found on-board the NASA 36.152 CUSP sounding rocket. The significant performance gains from the deployment of correlation functions in reconfigurable devices over purely software implementations were proven, leading in part to the work presented here. In this paper, we investigate the parallelization of a special type of ACF, the 1-bit version, in hardware.

2. 1-Bit correlation

A two-level (Bernoullian) quantization can be performed on a series before it is correlated without significant loss of correlation information, assuming the series investigated is wide sense stationary and bandlimited. Correlation coefficients before and after two level quantization are related by what is known as the Van Vleck correction [11], which states that the result of a measured correlation ρ_2 is proportional to the two-level quantized correlation ρ, as given by:

$$\rho_2 = \frac{2}{\pi}\sin^{-1}\rho \tag{2}$$

* Corresponding author. Tel.: +44 1273872821.
E-mail address: n.huber@sussex.ac.uk (N. Huber).

0141-9331/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.micpro.2011.01.001

Assuming this correction factor is applied, it follows that instead of carrying out a multi-bit correlation, a 1-bit (or binary) equivalent may be preferred without significant loss of accuracy. Hence, a 1-bit ACF or CCF can be a much more computationally efficient and realizable statistical tool in hardware systems where there are heavy restrictions, such as limited power, processing capabilities or transmission bandwidths. The most prominent example area is radio astronomy [12], where 1-bit cross-correlation functions have been extensively used due to bandwidth restrictions. Brisken [13] made a strong case for using dedicated hardware to perform correlations in this field, instead of software, in that the currently under development Expanded Very Large Array (EVLA) radio telescope would require a total of 200,000 3 GHz Pentium processors to carry out the excessively parallel computations needed. This approach gains more weight in [14], postulating that the best approach to the processing necessary for the correlator of the Square Kilometric Array Radio Telescope will probably be dedicated hardware. For this project, a series of dedicated correlator chips were always going to be the most important processing elements [15], focusing on FPGAs [16], whatever the number of quantization levels used.

Traditionally, 1-bit correlations in digital logic are carried out by duplicating the original set of bits. The duplicate is then shifted one bit at a time against the original, until all but the last bit have "rolled-off", i.e. until the highest lag has been computed. The rolling-off is achieved by correlating the shifted-out part of the bitstream padded with '0'. Depending on whether coincidences strictly of 1's or of both 1's and 0's in the bitstreams are desired, logic AND or NXOR gates respectively carry out the multiplication. The product is a bitstream of length equal to that of the original for each lag/shift (Fig. 1). The result of the ACF for that lag is then extracted by finding the number of 1's in this new bitstream. This process is carried out by a 1's counter, an asynchronous design of which forces the sequential propagation of the bitstream through many levels of logic. Each lag result is available at the end of each shift cycle through this process. The above described procedure can formally be viewed as a classical ACF of a Bernoullian sequence.

An example of the above is the 'Bit Correlator' by Xilinx, which is a freely distributed, optimized, drop-in module for Xilinx devices [17]. It performs a bit-by-bit comparison between input data and a user defined bit match pattern, producing an unsigned output value of the number of bits that match. Limited to just single bit correlations, this function will not allow the setting of the match pattern during runtime, thus preventing the core from becoming as suitable to our needs as an ACF. In a broader implementation, a 1-bit version is also discussed in [18], following the principle described above, carried out in an ASIC as part of an XF-type spectro-correlator. A 32-lag, 1-bit correlation is performed in the traditional manner, returning a lag result at the end of clock cycle of 32 MHz.

3. Counterbased ACF

Recently in the work by Pouchet et al. [19] a novel method of implementing the 1-bit ACF, dubbed "Counterbased ACF" (CBACF), was proposed and explored in some detail. A chain of matched delay elements through which the original bitstream is serially propagated is used to produce the shifted coincidences, as a function of the position of each element in the chain. The chosen delay elements were latches found in the fabric of a Xilinx FPGA. The multiply-add-shift operation is replaced by a set of counters, which asynchronously count the number of coincidences. A running sum is kept for each degree of shift in the counters, and with each new bit input all sums are updated in parallel. Fig. 2 displays this method, along with the use of two delay lines to capture coincidences of '1's and coincidences of '0's independently, at will. Similarly, a counterbased implementation of a CCF involves only the shifting of two separate sequences through the delay elements. The

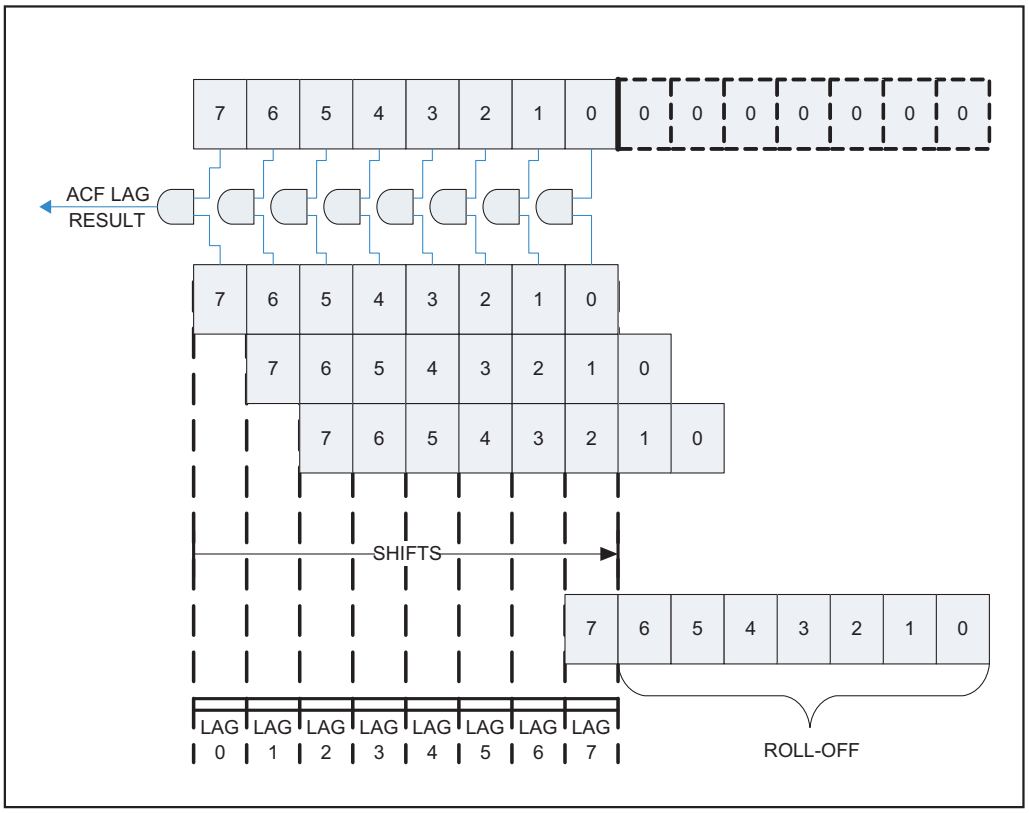

Fig. 1. Traditional implementation of a 1-bit ACF, showing "roll-off".