Proc. of the Second Intl. Conf. on Advances in Electronic Devices and Circuits -- EDC 2013

ISBN: 978-981-07-6261-2

An FPGA embedded ACCA architecture for high resolution target detection

Ridha Djemal

Electrical Engineering Department
College of Engineering, King Saud University
Box 800 CP 11421 KSA

***Abstract*—This paper presents an efficient FPGA-based architecture of CFAR target detector for radar system based on the automatic censored cell averaging (ACCA) detector based on ordered data variability (ODV). The ACCA–ODV detector estimates the unknown background level by dynamically selecting a suitable set of ranked cells and applying successive hypothesis tests. The proposed detector does not require any prior information about the non homogenous background environment. It uses the variability index statistic as a shape parameter to accept or reject the ordered cells under investigation. The detection process is achieved on the fly in real-time where the processing time must be lower than 0.5 µs for high resolution detection. The proposed architecture is based on the embedded software solution which consists on execution an the ANSI-C code of the detector over the Nios-II soft-core processor downloaded in the FPGA with the requires hardware components, such as on-chip memories, UART and JTAG interfaces and Avalon interfaces, to build the system on chip. Using the proposed approach for our embedded target detection system, the total delay is close to 0.38 µs for the ACCA-ODV algorithm, which satisfy the real-time constraints of 0.5 µs.**

Keywords— Constant False Alarm Rate (CFAR), Target Detection, Embedded System, Programmable device FPGA, Ordered Data Variability (ODV).

I. Introduction

The received signal in a radar system is computed to extract necessary information on the targets related to the object type (target or clutter) and the locations of the identified objects. If the echo is associated with a clear or empty background, it can be simply compared with a fixed threshold and the target is detected whenever the signal exceeds this threshold. However, in real cases, the echo is accompanied with clutter that varies in time and position, and therefore, in the extraction of the target, the threshold should be calculated dynamically from the local background noise/clutter power and not be a constant. In this respect, adaptive signal processing with a variable detection threshold is required to decide if there is a target present. The main idea is to define a window of cells around the cell under analysis and to determine the clutter information in that window to calculate dynamically the actual threshold [1].

Several constant false alarm rate (CFAR) techniques used for radar systems have been proposed in the literature, such as the application of cell averaging (CA) and ordered statistics (OS) [2,3]. For example, the OS-CFAR detector, for which an appropriate reference cell is used to estimate the background noise power level, has been proposed [4]. The OS-CFAR detector has a small additional detection loss over the CA-CFAR detector for homogeneous backgrounds but can resolve closely spaced interferences. However, it requires a longer processing time than the CA-CFAR detector, and in these terms, the CA-CFAR technique is the optimum CFAR approach for homogenous environments.

Other well-developed OS algorithms, such as the Greatest-of-CFAR (GO-CFAR) algorithm and the Smallest-of-CFAR (SO-CFAR) algorithm [5], the Censored Mean-Level Detector (CMLD) [6], and other OS algorithms [7,8], have been studied for different scenarios. However, the assumption of a homogenous environment is no longer valid when the number of targets changes abruptly. In such situations, the performance of the CA-CFAR processor is seriously degraded. Various classes of CFAR techniques have been proposed to enhance robustness against a non-homogeneous environment for different applications [9, 10] according to the background distribution but these implementations have been experimental in a software environment and not validated for a real-time system.

Although the theory of CFAR radar detection has been well established, the hardware implementation for a real-time environment is still beyond currently available high-computational signal processing operations. Owing to the real-time constraints of target detection by a high-resolution radar system, system-on-chip (SoC) architecture is an attractive solution for the real-time CFAR processor. In SoC architecture, all components of a computer, such as the processor, glue logic and memories, are integrated onto a single chip and operate in an organized manner.

In this paper, a Nios II processor FPGA-based platform is used to implement the Automatic Censored Cell Averaging Ordered Data Variability ACCA-ODV CFAR algorithm. This detector should be able to operate robustly to detect automatically target cell and determine the number of interferences close to the target. The SoC architecture of the CFAR detector is implemented on using Altera Stratix IV board with an embedded architecture organization based on the integration of the Nios-II soft-core processor in VHDL language. The Avalon switch fabric is also integrated within the same FPGA to interconnect the system-on chip components detailed in section IV. The proposed CFAR system is a typical embedded system example built in such way to achieve a processing delay of less than 500 ns, suitable for high-resolution radar applications in a desert environment [11]. The rest of this paper is organized as follows. In Section 2, the fundamentals of CFAR theory and related research on hardware realization for some types of CFAR algorithms are described. Section 3 presents the mathematical formulation with respect to the ACCA-ODV target detector. The embedded system FPGA-based design architecture for the proposed detector is explained in Section 4. Section 5 presents the simulation results and the realization of the target detection embedded system. In section 6, conclusions and future research plans are discussed.

Proc. of the Second Intl. Conf. on Advances in Electronic Devices and Circuits -- EDC 2013

ISBN: 978-981-07-6261-2

II. Related Work

For a radar system, a detection method is needed to determine the power threshold above which any return can be considered as coming from a target. If the threshold is too low, then more targets are detected, but the number of false alarms is high. Conversely, if the threshold is too high, then fewer targets are detected but the number of false alarms is low. The adaptive threshold can be used, where the threshold level is raised and lowered to maintain a constant probability of a false alarm. This is called CFAR detection.

A typical CFAR processor is shown in Fig. 1. The input signals are set serially in a shift register. The content of the cells surrounding the cell under test (X_0) are processed using a CFAR processor to obtain the adaptive threshold T. The value of X_0 is then compared with the threshold to make the decision. The cell under test is declared a target if its value exceeds the threshold value.

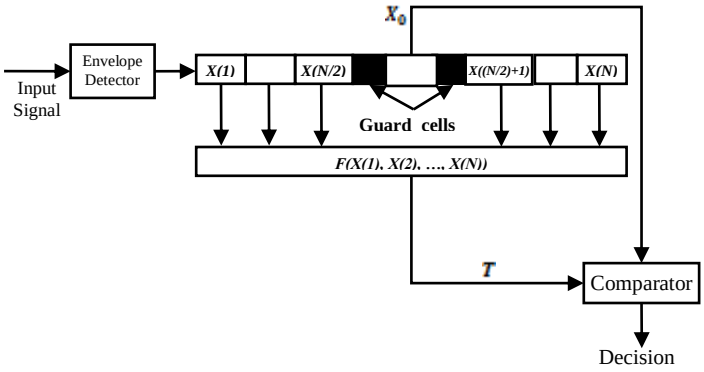

Fig. 1. Block diagram of a typical CFAR algorithm

The first and simplest CFAR detector is the CA-CFAR detector [3], for which the adaptive threshold is obtained from the arithmetic mean of the reference cells. Many CFAR algorithms have been recently developed. We can categorize a CFAR algorithm into one of three models according to the clutter power distribution and the interfering targets.

- When there is transition in the clutter power distribution, we can use, for example, greatest-of-selection logic for the CA-CFAR detector (GO-CFAR) [12] to control the increase in the probability of a false alarm. If one or more interfering targets are present, the GO-CFAR detector performs target detection poorly and it is suggested that an SO-CFAR algorithm employing smallest-of-selection logic is used instead for the CA-CFAR detector [13].
- When the clutter background is composed of homogeneous white Gaussian noise plus interfering targets, the CMLD can be used as a target detector. The CMLD censors target samples and estimates the noise level from the remaining noise sample. In addition, the trimmed mean-level CFAR (TM-CFAR) detector [4] implements trimmed averaging after ordering the samples in the window. When the number of interfering targets is not known a priori, the generalized CMLD (GCMLD), for which the number of interfering targets is determined and their corresponding samples are then sampled, can be used as well as the OS-CFAR detector, which chooses one ordered sample to represent the estimated noise level in the cell under test. If there is not only transition in the clutter power distribution but also interfering targets, a commonly used technique is the generalized two-level CMLD (GTL-CMLD) [14], which uses an automatic censoring algorithm of the unwanted samples when both interfering targets and extended clutter are present in the reference window of the cell under test.
- The last category deals with non-Gaussian clutter distribution. The lognormal distribution, Weibull distribution, gamma distribution, and K-distribution have been used to represent the envelope-detected non-Gaussian clutter distribution. Works on CFAR detection for Weibull clutter have been reported. [15-16]

However, the developments of the theoretical aspects of CFAR detection are not followed by hardware implementation. There are few attempts considering hardware implementations of CFAR processors have been reported. In particular, configurable hardware architecture for adaptive processing of noisy signals for target detection based on CFAR algorithms has been presented in [16-18]. The architecture has been designed to deal with parallel/pipeline processing and to be configured for Max, Min, and Cell-Average (CA) CFAR algorithms. OS-CFAR was implemented using parallel structure in [19]. In [20], CA-CFAR and OS-CFAR are combined and implemented in FPGA. In [21], TM-CFAR algorithm has been realized using FPGA. However, all these implementation were for simple CFAR algorithms and only suitable for Gaussian distribution type of clutter.

Alsuwailem et al. [22] implemented an automatic censoring CFAR detector called Automatic Censored Cell Averaging (ACCA) ODV CFAR. However, the implementation does not consider the real time aspects where an offline validation is done without allowing interactive interaction with the architecture. Furthermore not standard interface is given in order to facilitate the communication with the Radar System environment. Winkler et al. [23] used SoC with reconfigurable processor inside for an automotive radar sensor. The processor is responsible for controlling the custom logic and IO tasks.

In this respect, we propose to implement the ACCA-ODV CFAR detector using the embedded system organization integrating both hardware and software in the same FPGA and satisfying the real-time constraint related to the high resolution of non-homogenous environment.

III. The ACCA-ODV Detection Algorithm

In ACCA-ODV CFAR algorithms, the detection consists of two steps: removing the interfering reference cells (censoring step) and the actual detection (detection step). Both steps are performed dynamically by using a suitable set of ranked cells to estimate the unknown background level and set the adaptive thresholds accordingly. In a CFAR processor, the radar outputs { are stored in a tapped delay line (Fig 2). The cell with the subscript is the cell under test, where it contains the signal which should be detected as a target or not. The last surrounding cells are the auxiliary cells used to construct the CFAR procedure. In the ODV-CFAR, the surrounding cells are ranked in ascending order according to their magnitudes to yield

$$X(1) \le X(2) \le \cdots \le X(p) : \qquad (1)$$

The test cell X_0 is to be compared with the threshold T_k, to decide whether a target is present or not. Selecting

$$T_k = t_k \sum_{i=1}^{j} X(i) \qquad (2)$$

leads to a CFAR processor in Rayleigh clutter. The threshold T_k is parameterised by the variable t_k. The subscript j is taken to represent the largest rank possible, since CFAR loss would increase with the decrease in the value of j. In particular, the numerical results obtained in [5] show that the appropriate value of j, when detection is performed in homogeneous environments, is j = N.

Proc. of the Second Intl. Conf. on Advances in Electronic Devices and Circuits -- EDC 2013

ISBN: 978-981-07-6261-2

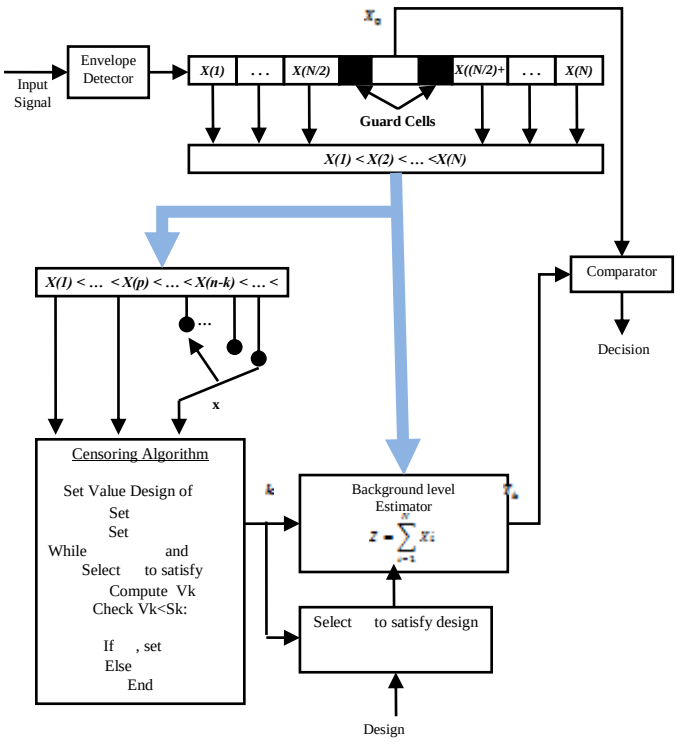

Fig. 2. Block diagram of the ACCA-ODV CFAR algorithm

However, in the presence of k interfering targets in the reference window, the value of j is best selected such that j = N- k. Therefore the main objective of the ACCA–ODV censoring algorithms is to have the task of determining the best value of k. Once the number of interfering targets is determined automatically, the output of the test cell X_0 is then compared with the adaptive threshold T_k according to

$$X0 \underset{H_0}{\overset{H_1}{\gtrless}} T_k \qquad (3)$$

where the adaptive threshold T_k (or equivalently the parameter t_k) is selected so that the design Pfa is achieved. Hypothesis H1 denotes the presence of a target in the test cell, whereas H0 is the null hypothesis (i.e. no target is present). To determine the number of interfering targets k, the ODV statistic V0 is first compared with the ODV threshold S0, which is selected so that a low probability of false censoring Pfc is maintained. The statistic V0 is defined as follows:

$$V_0 = \frac{\mu_p + X(N)^2}{\left(\sigma_p + X(N)\right)^2} \qquad (4)$$

Where $\sigma_p = \sum_{i=1}^{p} X(i)$

And $\mu_p = \sum_{i=1}^{p} X^2(i)$

The parameter p has to be carefully selected to yield a robust performance in both homogeneous and non-homogeneous environments. Values of p > N/2 have been found to yield a reasonable performance [4]. If $V_0 < S_0$, the algorithm decides that X(N) corresponds to a clutter sample without interference, and it terminates. If, on the other hand, $V_0 > S_0$, the algorithm decides that the sample X(N) is a return echo from an interfering target. In this case, X(N) is censored and the algorithm proceeds to compare the statistic V_1 with the threshold S_1 to determine whether X(N - 1) corresponds to an interfering target or a clutter sample without interference. In this case, we have: $V_1 = \frac{\mu_p + X(N-1)^2}{\left(\sigma_p + X(N-1)\right)^2}$ (7)

At the $(k + 1)^{th}$ step, the ODV statistic V_k is compared with the threshold S_k and a decision is made according to the test .

$$V_k \underset{H_0}{\overset{H_1}{\gtrless}} S_k \qquad (8)$$

Where $V_k = \frac{\mu_p + X(N-k)^2}{\left(\sigma_p + X(N-k)\right)^2}$ (9)

Hypothesis H_1 represents the case where X(N- k) and thus the subsequent samples X(N -2 k + 1), X(N - k + 2), . . . , X(N) correspond to clutter samples with interference, whereas H0 denotes the case where X(N - k) is a clutter sample without interference.

The successive tests are repeated as long as the hypothesis H1 is declared true. The algorithm stops when the cell under investigation is declared homogeneous (i.e. clutter sample only) or, in the extreme case, when all the N- p highest cells are tested (i.e. k = N- p). It is quite clear from Fig. 2 that the threshold selection is a key element in the implementation of the ACCA–ODV algorithm. The threshold parameter t_k is determined for a design P_{fa} by [4, 5]

$$p_{fa}(k) = \binom{N}{N-k} \prod_{j=1}^{N-k} \left(T_k + \frac{N-j+1}{N-k-j+1}\right)^{-1} \qquad (10)$$

As of S_k, these thresholds are selected such that a low probability of hypothesis test error is achieved in a homogeneous environment. For the ACCA–ODV algorithm, this probability is defined, at each value of k, as:

$e_k = Prob(V_k > S_k \mid homogeneous\ environment)$ (11)

The ODV thresholds S_k are selected such that a low P_{fc} is maintained at each step [4]. Hence, the values of S_k are determined by setting

$e_0 = e_1 = ... = e_{N-p-1} = design\ P_{fc}$ (12)

Threshold Values

The threshold selection is a key element in the proposed algorithm. These thresholds should be selected in order to reach low probability of hypothesis test error in a homogeneous environment. Monte Carlo simulation employed to obtain the threshold values with exponential probability density function . Table I gives the threshold parameters obtained using ACCA-ODV with different values of .

TABLE I: Threshold parameters for ACCA-ODV

(N,p)	P_{fc}	S_k					
		S_0	S_1	S_2	S_3	S_4	S_5
(16,12)	10^{-2}	0.356	0.246	0.199	0.173	-	-
	$5x10^{-3}$	0.389	0.267	0.213	0.183	-	-
	10^{-3}	0.456	0.320	0.246	0.206	-	
(24,16)	10^{-2}	0.332	0.235	0.189	0.162	0.143	0.131
	$5x10^{-3}$	0.362	0.255	0.204	0.173	0.152	0.138
	10^{-3}	0.422	0.305	0.240	0.200	0.174	0.155

Proc. of the Second Intl. Conf. on Advances in Electronic Devices and Circuits -- EDC 2013

ISBN: 978-981-07-6261-2

IV. Embedded ACCA-ODV based CFAR system architecture

A. *Generic hardware architecture for Radar system*

The overall SoC design consists of five main modules as shown in Fig.3: the Nios II processor dedicated to the execution of the ACCA-ODV CFAR algorithm, on-chip ROM input/ROM interface, output/RAM interface and JTAG UART interface. All blocks are connected by an Avalon interface. This interface allows our system to interact easily with external devices such as external memories or any other component capable of integration with the Avalon interface. The processor masters all communications between the hardware and executes the CFAR program using the MicroC/OS II operating system.

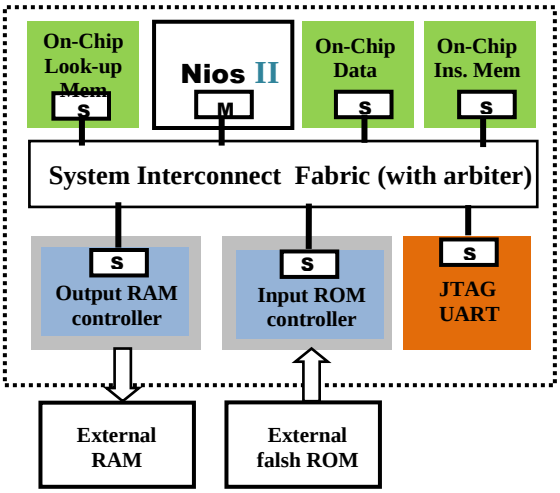

Fig.3. ACCA-ODV Nios II-based embedded System:

B. *Embedded Design flow*

To design the ACCA-ODV System on Chip, we propose to follow a typical embedded system design flow as shown in Fig. 4 integrating three main steps as:

- *Hardware design steps*: In this step, the embedded system based hardware architecture is defined. It incorporates a fast version of Nios-II core processor with on-chip memories and JTAG-UART interface interconnected using the Avalon fabric. A MicroC/OS real-time operating system is also selected and integrated within the Nios-II core processor to execute the ANSI-C software code related to the proposed CFAR application.
- *Software design steps*: This step consists on the design of a pure software architecture using a high-level language (HLL). The target detection code is developed with ANSI-C language and is running at first on the instruction set simulator (ISS) of the Nios-II core processor in the NiosII-IDE environment of Aletra. Once the code is simulated and checked this code is integrated on the FPGA code runs on the Nios-II processor within the FPGA using a micro-C operating system and a real-time validation is processed.
- *System design step*: It consists on the integration of both FPGA-based hardware architecture and software code within the same platform. An adequacy between the system architecture and the target techniques is explored by operating many optimizations on the algorithm (sorting, look up, threshold computation) as well as the system architecture (cache optimization, memory organization) in order to meet the high resolution and real-time requirements.

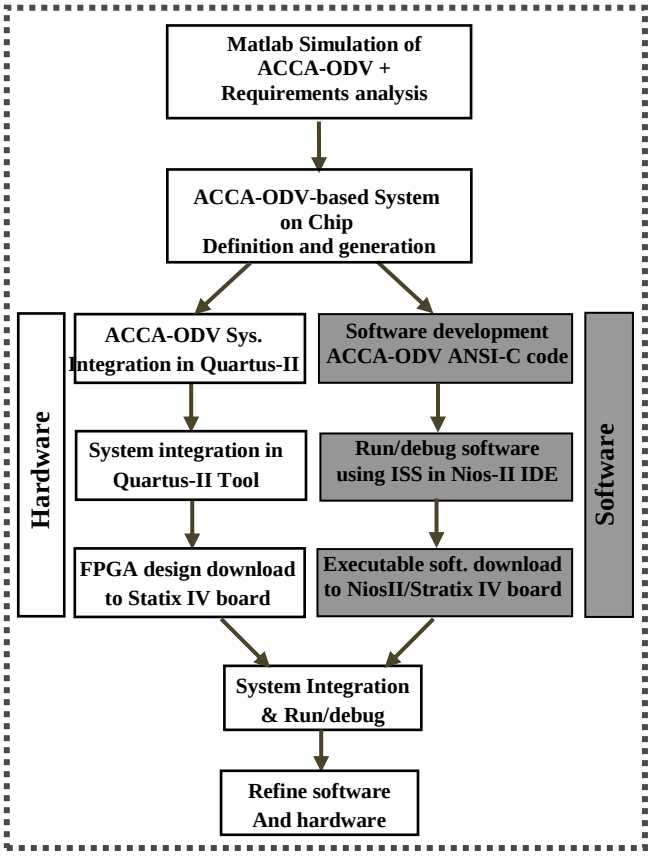

Fig.4. Typical HW/SW Design Flow

V. Experimental Implementation and Validation

The ACCA-ODV CFAR architecture has been built around an FPGA single chip, with bloc diagram architecture similar than the one presented in Fig.3 where the architecture is mapped onto an embedded system configuration. In this respect, the embedded software developed in ANSI-C code is executed over the Nios II core processor. The following table gives the execution time of each component of the ACCA-ODV architecture as a pure solution embedded into the Stratix IV board.

TABLE II: Embedded Software Execution Time of ODV module

ACCA-ODV Modules	Delays in µs
Sorting module	12
Censoring module	10
Detection module	4
Total delay	26

ISBN: 978-981-07-6261-2

According to the computation results, we have found that the critical point is located in computing σp and μp of the censoring module to compute the adaptive threshold and a part of the detection one. The sorting module represents also a critical task. We decide to export two custom instructions to integrate the sorting module and the censoring module with a part of the detection. The rest of the algorithm is dedicated for the Nios II core processor. After integrating the overall architecture including the hardware and the software modules, we have simulated the design and evaluated their complexity before and after adding the custom instruction. In Table III, we presented the complexity of only the Nios II core processor downloaded within the FPAG to execute the ACCA-ODV CFAR detector. The processing time is decreasing from 26μs to 0.46 μs which is a big delay saving.

TABLE III: Nios II processor resources (pure software solution)

	Without Custom Inst.
Comb.LUTs	3368 (2%)
Logic Reg.	2468(1%)
On-chip Memory	4547584 (31%)

The FPGA implementation result for SoC with $N = 16$ and $p = 12$ shows that the NIOS processor can achieve a maximum operating frequency of 250 MHz. After many optimizations the processing time to perform a single run is 0.46 μs.

VI. Conclusion

In this paper, a hardware software implementation of ACCA-ODV CFAR target detector algorithm is reported. This proposed system on chip system has the advantages of being simple, fast, and flexible with low development cost. The performance of the prototype hardware setup proved the concept of the co-design within a reasonable time of design. We have considered the custom instruction approach to export and design the hardware components having critical delays.

The proposed FPGA implementation integrates Nios II c, custom logics, on-chip memories, Avalon switch fabric and additional interfaces. The proposed architecture allows detection of each cell under test within a delay of 0.46 μs, below the real-time requirement of 0.5 μs . The architecure has been synthesized and validated using the Stratix IV development Kit (EP4SGX230KF4C2 device). As a future work, we have to extent the study for multi-cell target detection to yield an interfering target within homogenous and non homogenous environment.

Acknowledgement: This work is supported by the NPST project number ELE1730 of the King Saud University.

References

[1] M. Barkat, Signal Detection and Estimation. Norwood, MA: Artech House, 2005.

[2] R.S. Johnson H.M. Finn, "Adaptive detection mode with threshold control as a function of sampled clutter-level estimates," RCA Review, vol. 29, pp. 414-463, Sept. 1968.

[3] H. Rohling, "Radar CFAR thresholding in clutter and multiple target situations," IEEE Trans. Aerospace and Electronics Systems, vol. 19, no. 4, pp. 608-621, Jul. 1983.

[4] S.A. Kassam P.P. Gandhi, "Analysis of CFAR processors in nonhomogenous background," IEEE Trans. Aerospace and Electronics Systems, vol. 24, no. 4, pp. 427-455, July 1988.

[5] H.A. Meziani and F. Soltani, "Performance analysis of some CFAR detectors in homogenous and non-homogenous Pearson-distributed clutter," Signal Processing, vol. 86, pp. 2115-2122, April 2006.

[6] G.M. Dillard J.T. Rickard, "Adaptive detection algorithms for multiple target situations," IEEE Trans. Aerospace and Electronics Systems, vol. 13, no. 4, pp. 383-343, Jul. 1977.

[7] A. Mezache and F. Soltani, "A novel threshold optimization of ML-CFAR detector in Weibull clutter using Fuzzy-neural networks," Signal Processing , vol. 87, pp. 2100-2110, Feb. 2007.

[8] M. Barkat T.Larouissi, "Performance Analysis of order-statistic CFAR detectors in time diversity systems for partially correlated chi-square targets and multiple target situations," Signal Processing, vol. 86, no. 7, pp. 1617-1631, July 2006.

[9] M.A. Khalighi and M. H. Bastani, "Adaptive CFAR processor for nonhomogenous environemnt," IEEE Trans. Aerospace and Electronics Systems, vol. 36, no. 3, pp. 889-897, Jul. 2000.

[10] P. Henttu, and M. Juntti H. Saarnisaari, "Iterative multidimensional impulse detectors for communications based on the classical diagnostic methods," IEEE Trans. Communication, vol. 53, no. 3, pp. 395-398, Mar. 2005.

[11] S. Alshebeili, S.M. Alhumaidi, and A. M. Obied Y.M. Seddiq, "FPGA-Based Implementation of a CFAR Processor using Batcher's sort and LUT arithmetic," in 4th International Design and Test Workshop (IDT), Riyadh-KSA, 2009, pp. 1-6.

[12] J.H. Sawyers V. G. Hansen, "Detectability loss due to greatest of selection in a cell-averaging CFAR," IEEE Trans. Aerospace and Electronics Systems, vol. 16, pp. 115-118, Jan. 1980.

[13] M. Weiss, "Analysis od some modified cell-averaging CFAR processors in multiple target situations," IEEE Trans. Aerospace and Electronics Systems, vol. 15, no. 1, pp. 102-114, Jan. 1982.

[14] S. D. Himonas, and P. K.Varshney M. Barkat, "CFAR detection for multiple target situations," IEE Proceeding, Part F: Radar and Signal Processin, vol. 136, no. 5, pp. 193-210 M. Barkat, S. D. Himonas, and P. K.Varshney, "CFAR detection for multiple target situations," IEE Proceeding, Part F: Radar and Signal Processing, Oct. vol. 136, no. 5, pp. 193–210, Oct. 1989.

[15] R. Ravid and N. Levanon, "Maximum-likelihood CFAR for Weibull background," [16] R. Ravid and N. Levanon, "Maximum-likelihood CFAR for Weibull background," IEE Proceeding, Part F: Radar and Signal Processing , vol. 139, no. 3, pp. 256-264, Jun. 1992.

[16] V. Anastassopoulos and G. Lampropoulos, "Optimal CFAR detection in Weibull clutter," [17] V. Anastassopoulos and G. Lampropoulos, "Optima IEEE Trans. Aerospace and Electronic System, vol. 31, no. 1, pp. 52-64, Jan. 1995.

[17] C. Torres, and S. Lopez R. Cumplido, "A configurable FPGA-based Hardware Architecture for Adaptive Processing of Noisy Signals for Target Detection Based on Constant False Alarm Rate (CFAR) Algorithms," in Global Signal Processing Conference, Santa Clara CA, 2004, pp. 214-218.

[18] M.L.Bencheikh B. Magaz, "An Efficient FPGA Implementation of the OS-CFAR Processor," in International Radar Symposium, Wroclaw, 2008, pp. 1-4.

[19] R. Cumplido, C. Uribe and F. Del Campo R. Perez, "A versatile hardware architecture for a constant false alarm rat processor based on a linear insertion sorter," Digital Signal Processing, vol. 20, pp. 1733-1747, 2010.

[20] J. K. Ali, and Z. T. Yassen T. R. Saed, "An FPGA-based implementation of CA-CFAR processor," Asian Journal of Information Technology, vol. 6, no. 4, pp. 511-514, 2007.

[21] A. M. Alsuwailem, S. A. Alshebeili, and M. Alamar, "Design and implementation of a configurable real-time FPGA-based TM-CFAR processor for radar target detection," Journal of Active and Passive Electronic Devices, vol. 3, no. 3-4, pp. 241-256, 2008.

[22] A. M. Alsuwailem, M.H. Alhowaish, S. A. Alshebeili, and S.M Qasim, "Field programmable gate array-based design and realization of automatic censored cell averaging constant false alarm rate detector based on ordered data variability," IET Circuits, Devices & Systems, vol. 3, no. 1, pp. 12-21, Feb. 2009

[23] J. Detlefsen, U. Siart, J. Buchlert, and M. Wagner V. Winkler, "FPGA-based signal processing of an automotive radar sensor," in First European Radar Conference, Amsterdam, 2004, pp.