Journal of Engineering and Development, Vol. 11, No. 2, September (2007) **ISSN 1813-7822**

Design and Implementation of a TM-CFAR Processor using Field Programmable Gate Array (FPGA)

Asst. Prof. Dr. Waleed Khalid Al-Dulaimi
Electrical Power Eng. Technology Dept.
Al-Mamoon University College, Baghdad, Iraq

Eng. Ali Hadi Abdul Wahed
M.Sc. in Electrical Engineering
Technical College, Basrah, Iraq

Abstract

This paper proposes on a project whose aim is to implement architectures for Constant False Alarm Rate (CFAR) processor employed in radars. These proposed architectures are implemented by using Field Programmable Gate Array (FPGA) technique, with observance of feature desired as flexibility, and speed without performance loss. This design written in VHSIC Hardware Description Language (VHDL).

FPGA architecture of TM-CFAR processor is implemented to overcome the large processing time required in case of multi-target environment. The use of this technology is suitable because it offers high flexibility in modifying and even developing the required design with a reduction in the required number of hardware and cost.

In addition to the main features of the processing chain architectures such as degree of parallelism, small size, accuracy and high computational speed, there is a requirement for generality and adaptively i.e. the ability to handle many different models of operation in different environments.

The implementation of the design is achieved using Xilinx Virtex-II platform as a suitable selected programmable device. The development of the system software for this work has been done using Active HDL 3.5, and ISE Navigator 6.3i programs.

الخلاصـــــــة

ان هذا البحث هو مشروع هدفه تطبيّقَ هندسة معماريه لمعالجِ ***(CFAR)*** *مستخدم في الرادارات. هذه الهندسة المعماريه المُقتَرَحه مُطَبَّقَة بإستعمال تقنية مصفوفة البوابات المبرمجة* ***(FPGA)****، مَع مراعاة الميزّات المطلوبه كالمرونة، والسرعة بدون خسارةِ في الأداءِ. هذا التَصميم مكتوبِ بلغة* ***(VHDL)*** *والتي أستَعملتَ لبناء التطبيقَ.*

تم بناء معمارية تستخدم ***FPGA*** *للمعالج* ***(TM-CFAR)*** *لغرض معالجة مشكلة طول وقت المعالجة اللازم لهذه الطريقة عند التعامل مع ألاهداف المتعددة . أن استخدام تقنية برمجة الـ* ***(FPGA)*** *مهم للتصميم لانه يوفر مرونة في تغيير وتحديث مواصفات التصميم والتي تسهم في تقليل عدد المكونات المادية* ***(Hardware)*** *المستخدمة في التصميم وبالتالي تقليل الكلفة الكلية للمنظومة.*

أن اهم مايميز البناء المعماري لهذا المعالج اضافه الى الدرجة العالية من المعالجه وصغر الحجم ودقة وسرعة المعالجة العالية ، هو الشمولية وأمكانية التطوير او التحديث أي بمعنى امكانية المعالج من العمل بأطوار وأشكال مختلفة وبظروف عمل مختلفة بظهور تقنيات المعالجة الحديثة والمتمثلة بتقنية مصفوفة البوابات المبرمجة ***(FPGA)****.*

انجز تنفيذ التصميم في هذا البحث بأستخدام رقاقة المصفوفة البرمجية نوع ***(Virtex II)*** *التي تنتجها شركة* ***(Xilinx)*** *كاداة للكيان المادي. حيث تمت عملية التصميم بأستخدام برنامج* ***(Active-HDL 3.5)*** *في حين تمت عملية التنفيذ بأستخدام برنامج* ***ISE Navigator 6.3i*** *.*

1. Introduction

In the modern radar systems, equipped with automatic detection circuits, the use of CFAR technique is required to keep false alarms at a suitably low rate in a-priori unknown. This interference may have different dynamic ranges, probability distributions, which vary in both time and space, at the signal processing and detection level.

In many radar systems, decision making has to be done in a greater speed than the capability of the operator. Thus, replacing the human operator by adaptive signal processing and data processing units is the general trend of today's radar technology. The philosophy of such devices is based upon the statistical decision theory, which gives an explanation and understanding of the two problems of signal processing and detection. In addition, it satisfies the needs to solve many hard problems like multiple target discrimination. However, with automatic detection and decision making devices come the problem of false alarm. Since the early 1960's, many techniques can yield CFAR, under specific restrictions [1].

In the late 1960s, when digital signal processing was introduced in radars, a general-purpose processor was provided to control the radar's operation. But processing speeds were so low and memory so limited that a hard-wired processor had to be provided for signal processing. To run predictably in real time and fit within the available memory, software for the data processor had to be written in assembly language. The size of the program was limited by what could be executed in real time at the processor's limited speed and would fit in its memory [2].

For the past decade, reconfigurable computing became widely used when Xilinx Corporation introduced the first commercial (FPGAs) [1].

The United States Department of Defense (DoD) sponsored a workshop on hardware description language; this workshop was to study various hardware description methods, the need for standard language, and the features that might be required by such standard. This approach is economical, flexible and easy in implementation. It is suitable for real time application for speed less than 100 MSPS, which is an advantage over the TMS.

2. CFAR Algorithm

The Cell-Averaged CFAR (CA-CFAR) is the most common CFAR detector used for target detection. The CA-CFAR detector is used to regulate the false alarm probability to a desired level in varying background environments through averaging. **Figure (1)** shows a block diagram of the CA-CFAR algorithm structure. In CA-CFAR detectors, a reference window of N samples which surround the cell or data under test is taken to compute the average value [3].

A block diagram of conventional CFAR processor is shown in **Fig.(1)**. The square-law detected range samples are sent serially into a tapped delay line of length. The output of the test cell, which is the one in the middle of the tapped delay line, is denoted xo. The statistic Z,

which is proportional to the estimate of total noise power, is computed by processing the contents of N reference cells surrounding the Cell Under Test (CUT(x_0)).

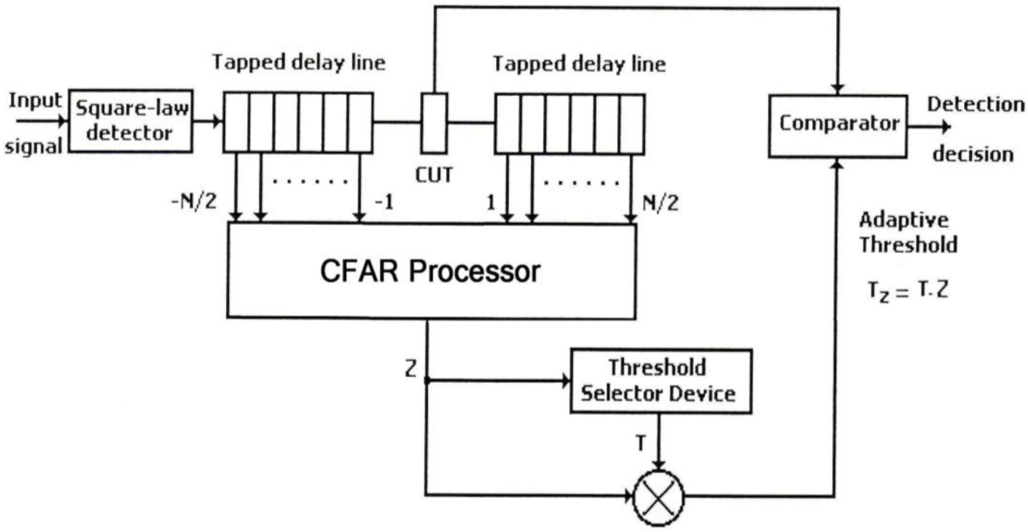


Figure (1) Block diagram of a CFAR processor

The estimate Z is then multiplied by a constant scale factor T, depending, first, on the estimation method applied and, secondly, on the false alarm rate required. A target is declared to be present if x_0 exceeds the adaptive threshold T_Z .The scaling constant T is selected so that the design probability of false alarm is achieved. The procedure or estimating Z varies with different CFAR schemes [4].

There are many types of CFAR processing algorithms based on the method of regulating the false alarm rate, where all these types mainly use adaptive threshold setting [5].

In TM-CFAR, the outputs of the reference cells {Xi}, i=1,2,….,N are fed into a ranking device which, in turn, outputs the samples in ascending order according to their magnitude $X_{(1)} \le X_{(2)} \le \ldots \le X_{(k)} \le \ldots \le X_{(N)}$ to yield the N ordered samples, and then trims T_1 cells from the lower end and T_2 cells from the upper end before summing the rest. The Ordered Statistic CFAR (OS-CFAR) and CA-CFAR schemes are special case of the TM-CFAR scheme, as shown in **Fig.(2)**.

$$Z = \sum_{k=T1+1}^{N-T2} X_k \quad \ldots\ldots\ldots\ldots (1)$$

where:

Z: is the sum of the reference cells ranked according to their output level.

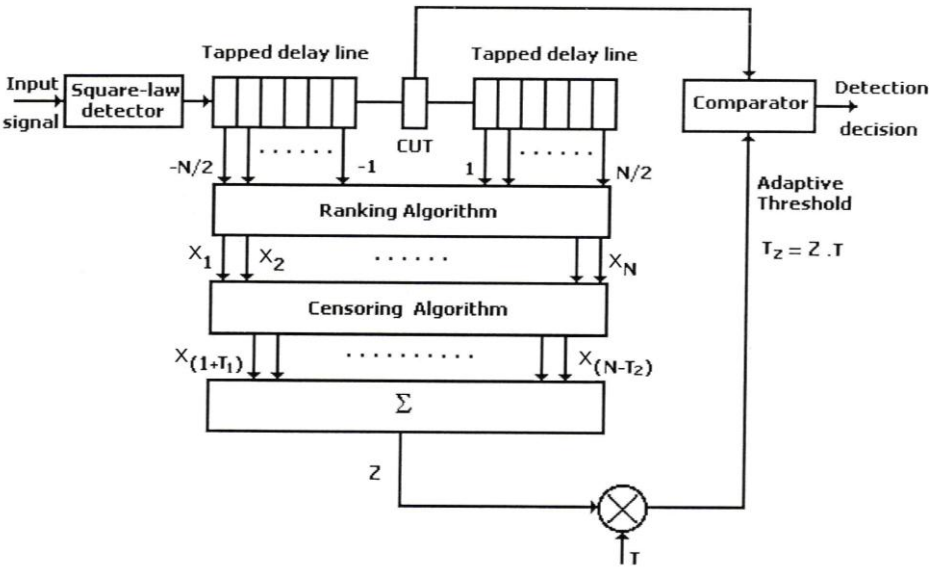


Figure (2) Block diagram of a TM-CFAR processor

$$T_Z = Z \cdot \frac{T}{N - T_1 - T_2} \quad \text{..... (2)}$$

The adaptive threshold level T_Z is obtained by multiplying the output of the representative rank cell by a scaling factor T [5].

The values of the Cell Under Test CUT and the adaptive threshold TZ, are fed into a comparator circuit that compares both amplitudes. If the value of (CUT) exceeds (T_Z) the target is declared to be present, and otherwise absent. Therefore the Threshold is found by [4].

$$P_{fa} = \prod_{k=1}^{N-T_1-T_2} M_k(T) = \prod_{k=1}^{N-T_1-T_2} \frac{N!}{T_1!\,(N-T_1-1)!\,(N-T_1-T_2)} \times \sum_{j=0}^{T_1} \frac{\binom{T_1}{j}(-1)^{T_1-j}}{\frac{N-j}{(N-T_1-T_2)}+T} \quad \text{..... (3)}$$

where:

$M_j(T)$: represents the moment generating function (MGF) of ordered sample X.

The detection probability P_d is obtained by replacing T with T/(1+SNR) in (3). The loss of detection in terms of ADT is computed by differentiating (3) with respect to SNR which yields [4].

$$ADT = \frac{TN\ !}{(N-T_1-1)!} \sum_{j=0}^{T_1} \frac{(-1)^{T_1-j}}{(N-j)\ j\ !\ (T_1-j)!} \times \left[\frac{N-T_1-T_2}{N-j} + \sum_{i=2}^{N-T_1-T_2} \frac{1}{\Delta i} \right] \quad \text{..... (4)}$$

where:

$$\Delta i = (N - T_1 - i + 1)/(N - T_1 - T_2 - i + 1) \quad \text{..... (5)}$$

It is clear that with symmetric trimming, Z is given by the sum of the middle $N-T_1-T_2$ reference cells of the ordered window. Consider the situation where the leading half of the reference window contains cells from clutter-plus-noise background and the lagging half from clear background. The noise power estimate y will include samples from both the clear and clutter-plus-noise background. The corresponding threshold will not be high enough to regulate the false alarm rate if the cell contains a return from the clutter-plus-noise background. The TM-CFAR processor performance for asymmetric trimming is more interesting [5].

The flowchart shown in **Fig.(3)** illustrates the steps of the TM-CFAR program.

3. CFAR Hardware Implementation

Let X be the raw data samples of the signal to be processed and N the number of reference cells in the TM-CFAR detector. For the sake of simplicity but without lost of generality, let consider a sequence of reference data samples around a cell under test as one-dimension windows. Each rectangle includes data from the reference cells. The windows share data and each time that the window slides leftwards one position on data, all its data samples, except the data located on the edges, belong to the domain of the next window [4,8].

A block diagram of the proposed architecture is shown in **Fig.(4)**. The main components of the architecture are:

1. Delay
2. Ranking
3. Trimming &Average
4. Adaptive Threshold
5. Comparator.

Figure (4) show the block diagram of the main core of the CFAR architecture.

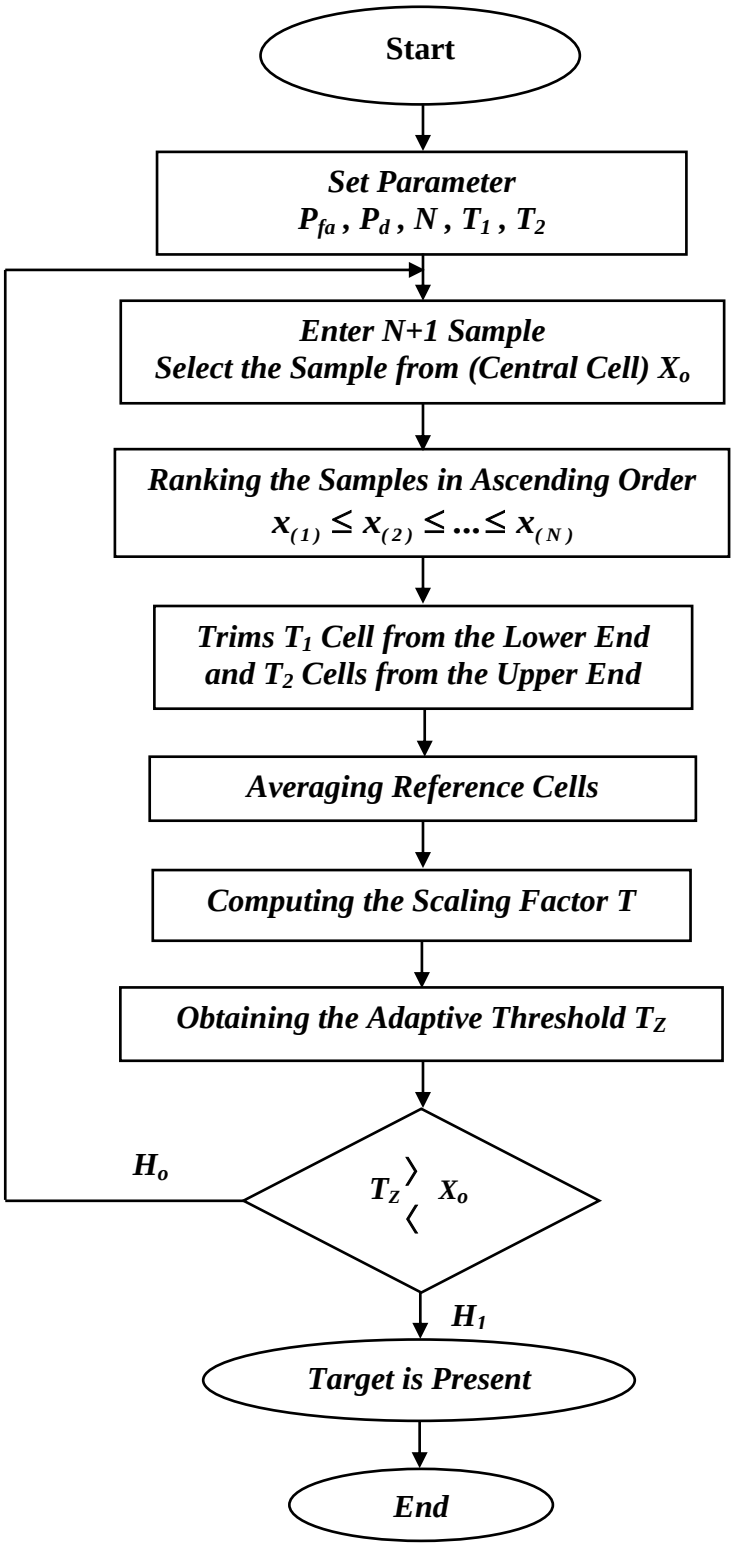
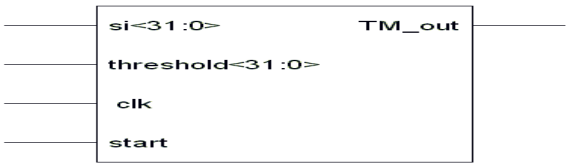
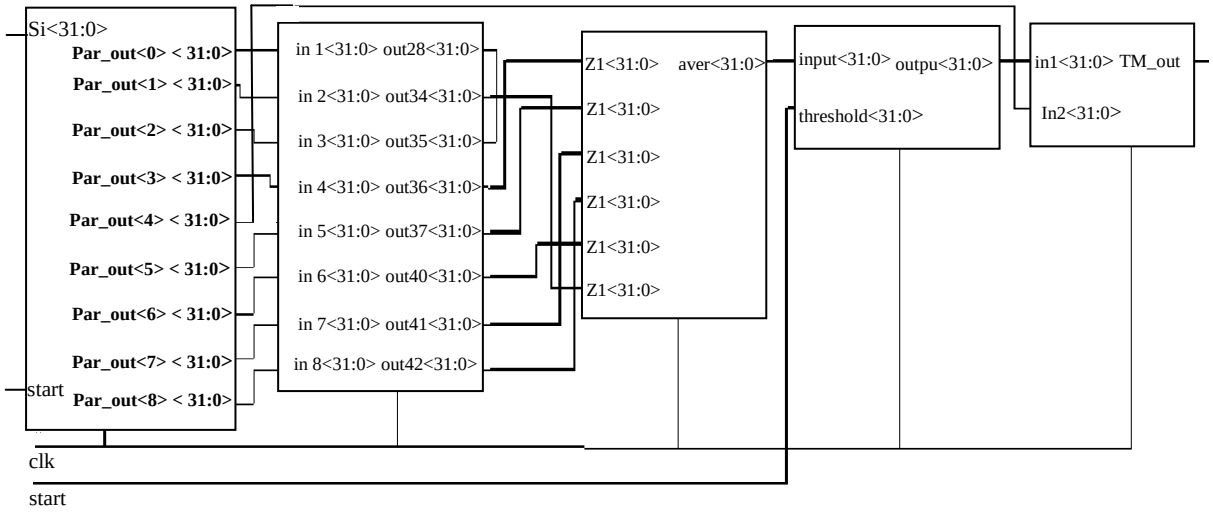


Figure (3) Flowchart illustrates the steps of the TM-CFAR program



(a)



(b)

Figure (3) a. Hierarchical block diagram of TM-CFAR component
b. Inner content of top-level block

Figure (4) Block diagram of the internal structure of the processing elements, b) main components of the TM-CFAR

Figure (5) illustrates hierarchy block diagram of Delay component. It has input ports named si<31:0>, and control signals: start, clk. It has par_out<0> <31:0> to par_out<8><31:0>, as output ports.

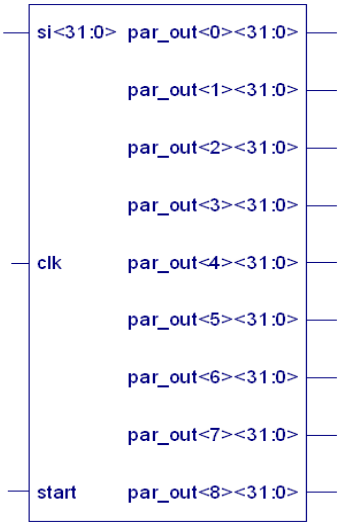


Figure (5) Top-level FPGA hierarchical structure of delay component

As shown in **Fig.(4-b)** the second component is achieve Ranking algorithm, the voltage of a reference cells (leading and lagging parts about the cell under test) are rank ascending according to their magnitude. **Figure (6)** illustrates hierarchy block diagram of ranking component. It has 8 input ports named in1<31:0> to in8<31:0>.and 8 output ports, which are: out28<31:0>, out34<31:0>, out35<31:0>, out36<31:0>, out37<31:0>, out40<31:0>, out41<31:0>, and out42 <31:0>.

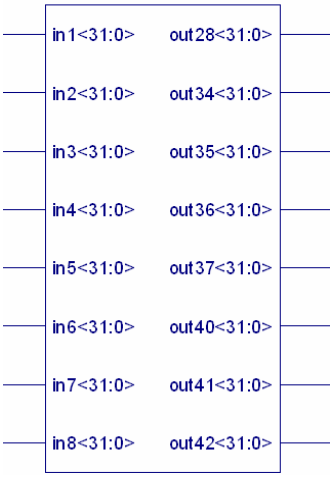


Figure (6) Top-level FPGA hierarchical structure of ranking component

The Ranking component consists of Twenty-One subcomponents. **Figure (7)** illustrates hierarchy block diagram for these subcomponents. Each of them has input named in1<31:0>, in2 <31:0>, It has out1<31:0>, out2 <31:0>, as an output.

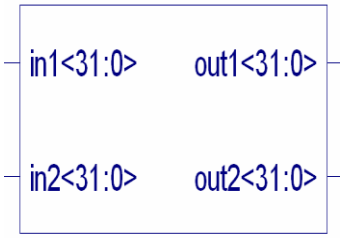


Figure (7) Top-level FPGA hierarchical structure of (subcomponents)

The third component as shown in **Fig.(8)** is Trimming and average, this component is responsible for remove a predetermined number reference cells that contain the highest outputs than the rest. The remaining reference cells are combined together, the background estimate is obtained from the average of the N-T_1-T_2 reference cells of ordered window, equation (1). **Figure (8)** shows the hierarchy block diagram of trimming component. It has six input ports named z1<31:0> to z6<31:0>, with output port named aver<31:0>.

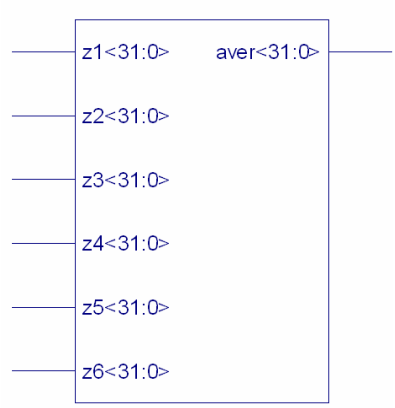


Figure (8) Top-level FPGA hierarchical structure of trimming component

The adaptive threshold component, which is the fourth component in the TM-CFAR structure is shown in **Fig.(9)**, this component is used for calculate the adaptive threshold. This is obtained by multiplying the output of previous component by a scale factor T to yield the T_Z. **Figure (9)** illustrates hierarchy block diagram of adaptive threshold component. It has input ports named input<31:0> and threshold <31:0>, and output port is output<31:0>.

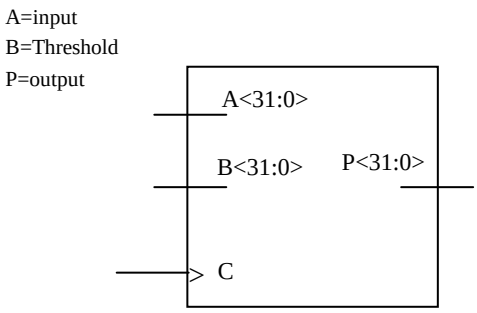


Figure (9) Lower-level FPGA structure of threshold component

Figure (10) illustrates hierarchy block diagram of comparator component. This component is responsible for detection decision. The sample from the cell under test is compared with the adaptive threshold. The target signal detection is declared if the output of CUT (in1) exceeds the adaptive threshold (in2). It has input port named in1<31:0> and in2<31:0>, the output is named Q.

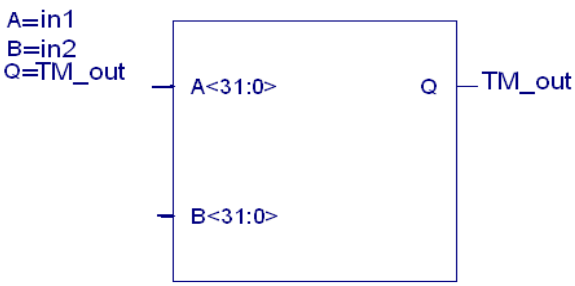


Figure (10) Lower-level FPGA structure of comparator component

4. FPGA Implementation and Results

The proposed architecture is modeled using the VHDL Hardware Description Language [6,7]. After simulating the VHDL model of TM-CFAR processor, it is implemented using ISE Navigator 6.3i software. The target device for implementation is XC2v300e-6fg456 FPGA chip. This FPGA chip is chosen according to its architecture, which is optimal for high density and high performance logic design, according to the repots obtained from the implementation process.

Table (1) Synthesis summary and timing for the FPGA implementation of the TM-CFAR processor

Synthesis summary for the TM-CFAR processor targeted for a XC2v300e-6fg456 Vertex-II device	
Number of Slices	1727
Number of 4-Input LUTs	3,145
Number of Flip Flops	265
FPGA occupation Percentage	56%
Maximum clock Frequency	101.163 MHz

5. Conclusions

Using FPGA technology for such an environment instead of other technology would give many benefits, including lower cost, high processing speed, and high degree of flexibility, the ability to create entirely new types of systems and applications can dynamically change the hardware in system. FPGA allows the implementation of CFAR processor to real time, high speed. The maximum delay is 7.911 ns. All these specifications are achieved by using a single chip and only up to 56% of single chip resources are utilized.

6. References

1. Cumplido, R., Torres, C., and López, S., ***"Implementation of an Efficient FPGA Based CFAR Processor for Target Detection"***, IEEE International Conference on Electrical and Electronic Engineering, Mexico, Septemper (8-10), 2004, pp. 214-218.

2. Levanon, N., ***"Radar Principles"***, John Wiley and Sons, USA, 1988.

3. Farina, A., and Studer, F. A., ***"A Review of CFAR Detection Techniques in Radar Systems"***, Microwave Journal, Vol. 29, No. 9, September 1986, pp. 115-128.

4. Gandhi, P. P., and Kassam, S. A., ***"Analysis of CFAR Processors in Nonhomogeneous Background"***, IEEE Transaction on Aerospace and Electronic System, Vol. AES-24, No. 4, July 1988, pp. 427-444.

5. El-Mashade, M. B., ***"Detection Performance of the Trimmed-Mean CFAR Processor with Noncoherent Integration"***, Proceedings of IEEE Radar, Sonar Navig., Vol. 142, No.1, February 1995, pp.18-24.

6. ***"VHDL Design Methodology Syntax and Commands"***, VHDL for XILINX Folie 1 Von 478, 2003.

7. Yalamanchili, S., ***"Introductory VHDL from Simulation to Synthesis"***, Prentice Hall Xilinx Design Series, Prentice Hall, 2001.

8. Huitzil, C. T., Parra, R. C., and Estrada, S. L., ***"Design and Implementation of a CFAR Processor for Target Detection"***, Computer Science Department, INAOE, Apdo. Postal 51 & 216 Tonantzintla, Puebla, Mexico, 2004.