International Journal of Engineering Research and Development
e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 12, Issue 7 (July 2016), PP.27-35

Novel Method of Realization of Scalable VLSI Adaptive Digital Beamforming Architecture for Phased Array Radar

D. Govind Rao[1], T. Kishore Kumar[2] N. S. Murthy[3] And A.Vengadarajan[4]
[1,4] Lrde, Drdo, Bangalore
[2,3]nit, Warangal

Abstract:- This This paper describes a novel method for the hardware Design and realization of adaptive filter for the application of Adaptive Digital beam former suitable for FPGAs. The planar phased array configuration considered in this case is sixteen element array. Design approach followed is modular design and each one of the modules is reused to make the sixteen element planar array configuration. One of the most efficient Adaptive algorithms from the hardware realization aspect is Inverse QRD-RLS algorithm. This algorithm is implemented in a systolic array structure where in the optimal weights are calculated in much less time and is numerically more stable compared to other traditional adaptive algorithms like LMS, RLS etc. Also Inverse QRD-RLS algorithm is suitable for parallel and pipelined implementation thus making it very useful in the applications where speed, accuracy and numerical stability are of utmost importance; such as Phased Array Radar. High performances FPGA Kintex--7 is employed to implement the VLSI architectures.

Keywords:- Phased Array, Beam Formation, Kintex-7, QRD-RLS, DDC, CIC filter, SOC, Complex Arithmetic.

I. INTRODUCTION

Beam former design is an essential part of radar[24] and information exchange architectures. There are numerous ways of beamforming[4] that are currently utilized. The optimization of the system used in signal reception decrease due to sensitivity in SNR caused by unwanted signal which enters the system neither by the main lobe nor by the side lobes of the patterned beam. The processing of the signal is done by the signal processors for the signals which are received and then the beam is formed in the directions as specified.

Presently efficient beamforming is employed in the filter algorithms which are adaptive and it can be utilized in the array system that comprises of the radar such that the intended signal is preserved in the occurrence of noise signal. Due to the raise in radar traffic the interference suppression becomes more essential in those systems where adaptive beamforming has its major usage .the signals and interference properties are distinguished in the active phased planar array through beamforming where it comprise a array of sensors that are independent so that it gives the samples of signal which are received in space.

Hereby outputs from the sensor are modeled by a filter that can be transverse to produce output in the form of the beam The basic aim of the modifying filter is to defend that the target signal when the noise signal is cancelled Moreover the modifying filter provides capacity to the system that consists of array of antenna such that it can automatically capture the occurrence of the intruding signals to precisely suppress the noise signals which simultaneously enhance the referenced signal the QRD RLS algorithms are opted to be a better approach for those applications where convergence speed is paramount.

II. ADAPTIVE FILTER ALGORITHMS

The important aspect of adaptive Beam forming is computation of adaptive weights on the hardware. In designing adaptive filters for radar applications, recursive least squares (RLS) and constrained recursive least squares (CRLS) algorithms were the promising methods compared to least mean squares (LMS) algorithm due to their fast convergence rate. CRLS and RLS algorithm uses direct inversion input data matrix. It has two major disadvantages. One is that this method has undesirable numerical characteristics when complex covariance matrix is ill conditioned. Another disadvantage is that the RLS and CRLS algorithms [2,12] cannot be implemented as parallel and pipelined array processors for real time signal processing applications.

III. FPGA IMPLEMENTATION OF SYSTOLIC ARRAY

The inverse QRD RLS Algorithm[6,8] compared to its variants hardware implementation is possible in pipelined and parallel architecture. This algorithm, also inherited numerical robustness of its family, without having to resort to the computationally tedious backward or forward substitution procedures it provide the coefficient vector at every iteration. Inverse QRD-RLS algorithm allows the calculation of the weight vector in completely pipelined manner. But Conventional QRD-RLS update scheme involves two computational steps which cannot be efficiently combined on a pipelined array.

Angle Processor: The Angle processor takes the input data from the input vector matrix and computes the rotation angles. It calculates the sine and cosine which is given to the rotation processor to perform the rotation and obtain the new set of values. Angle processor has its internal memory to store the rotated data which is to be used during the calculation.

Rotation Processor: It performs the Givens rotation on the input data by multiplying it with the sine and the cosine from the angle processor and generates the rotated data which is stored in its internal memory and the output is passed to the cell in the next row. The cells involved in the systolic array implementation of the Inverse QRD-RLS are same as in Conventional QRD-RLS except the cells in the last row. These cells are called as Weight processors as they give out the optimal weight vector.

Weight Processor: It accept the data from the rotation processors in its preceding row and the sine and cosine of the rotation angle from the angle processor in its row. Part of the operation performed in the weight processor is similar to the rotation processor wherein it rotates the input data by multiplying it with the sine and cosine of the rotation angle.

$W(k) = -1(COS * MEM) + (SINE' * Xin)\,Y$ (1)

Along with this operation, weight processor multiplies the rotated input data value with the scaling factor (Y) and then negates the result to generate the final optimal weight vectors.

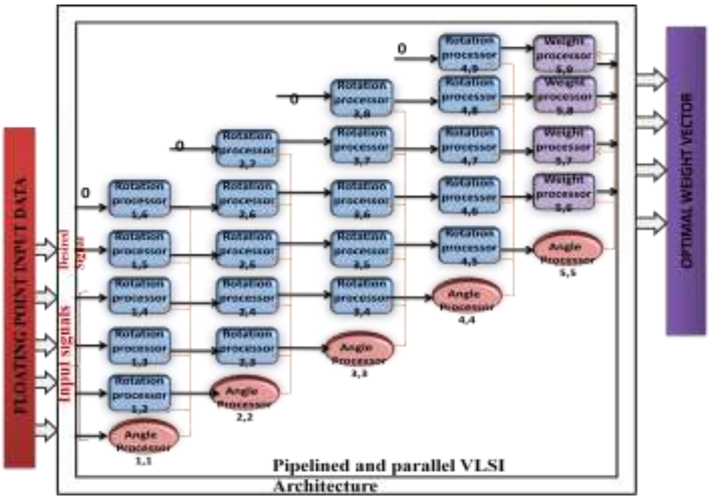


Figure 1. VLSI Architecture of Inverse QRD-RLS algorithm.

The optimal weights will be calculated with the latency of 4 input samples. Finally the weight processor as shown in Figure 1 gives the optimal weights for the adaptive beam formation application.

IV. ADAPTIVE BEAM FORMER ARCHITECTURE USING FIXED POINT OPERATIONS

The pipelined architecture of the IQRD-RLS algorithm is contained in FPGA1 and FPGA2 in the Array signal processor architecture. The Function of the algorithm is to adaptively calculate the weight according to the input signal. Then these adaptively calculated weights will be multiplied with the input in phase and quadrature signals with a complex multiplier. Then the complex multiplier outputs will be summed together to get the final beam.

The architecture of the Array Signal Processor is divided into three different FPGA's for the purpose of the modular design approach. First and second FPGAs are doing almost same work, taking input from the ADC's, frequency translation of signal using DDCs, weight calculating adaptively, followed by multiplying the input signal with the weights. These multiplier outputs are fed to the third FPGA which performs the summation of these signals from both FPGA1 and FPGA2, and produce the beam.

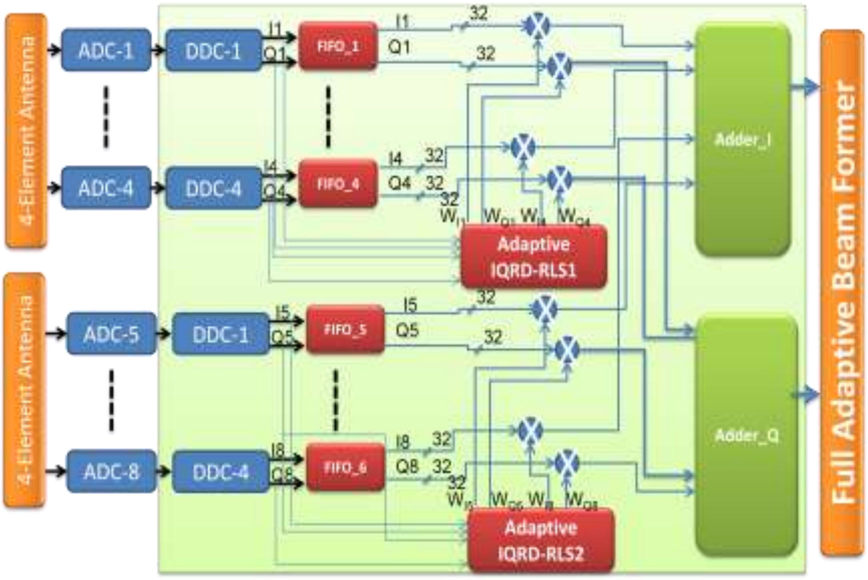


Figure. 2: VLSI Architecture of FPGA1 and FPGA2 with ADC and DDC

Eight inputs are connected to FPGA1[3] and another eight inputs connected to FPGA2. The net result is sixteen channel inputs; both are synchronously running under the control of FPGA3 The Fig. 2 is the architecture with ADC, DDC, IQRD-RLS weight calculator etc. It take input from the ADC, and then DDC will make two signals from the ADC input by multiplying the ADC output with the NCO(Numerically Controlled Oscillator) output which is sine and cosine waves, producing two frequency translated signals, one in phase with the input signal and the other is quadrature signal. The DDC is used as a frequency translator.

We can also make another structure of the FPGA1 and FPGA2[3] such that a DDS is designed inside the FPGAs so that the DDC module can be avoided, in turn, resulting the reception of signals from the ADC modules are not needed, which is best method for the testing environment. We can change the Frequency, Phase, and Amplitude of the DDS output signals. The DDS module will generate sine and cosine signals in IF frequency or baseband region, upon our commands. For that, in FPGA3, we have built a RS232 interface such that we can give commands through PC and then FPGA3 will give these signals to both FPGA1 and FPGA2, which is very useful for the testing of the Array Signal processor. Such architecture of FPGA1 and FPGA2 is shown in Figure. In this case, architecture of the FPGA3 will be giving the control signals to the FPGA1 and FPGA2 about the Frequency, Phase and Amplitude of the signal to be made by the DDS. FPGA3 will do the final summing of the complex multiplier outputs from FPGA1 and FPGA2. When the FPGA1 and FPGA2 outputs are available, it will give output available signal to the FPGA3[15]. On receiving the output available signal from FPGA1 and 2, FPGA3 will initiate dataflow through the data bus synchronously. This implemented architecture is for eight element array, but can be extended to any number of antenna arrays.

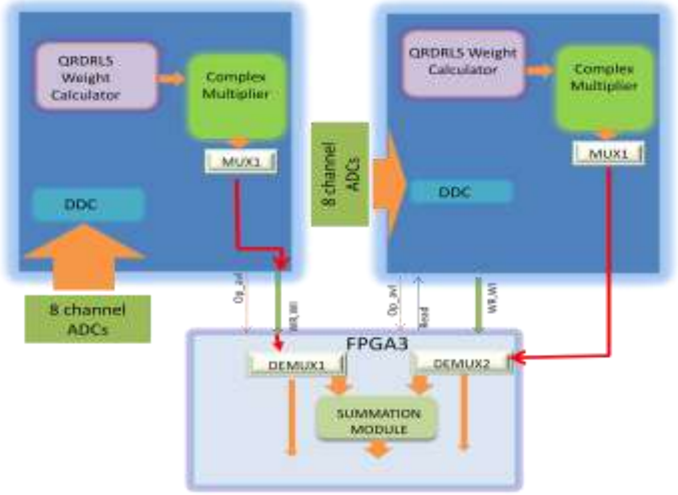


Figure. 3: Complete test setup including Hardware Emulator

In 50 MHz sampling clock The digital data is received and then passed via a digital Mixer which consisting of a multiplier (14x14 bit), a 50 MHz Numerically Controlled Oscillator (NCO), suitable low pass decimation and compensating filters (CIC and CFIR filters) of bandwidth 5 MHz to filter the entire unwanted signal outside the band and a 10 rate decimator to bring down the sampling rate to 5 MS/s for further processing Finally the DDC output will be In phase (I) and Quadrature (Q) signals. By multiplying the input data, we can achieve a frequency translation to the base band, by the quadrature sine and cosine waveforms. For sixteen elements to form one beam we need to have sixteen weights and for N number of beams, N different sets of sixteen weights are required.

V. ADAPTIVE BEAM FORMER ARCHITECTURE USING FLOATING POINT OPERATIONS.

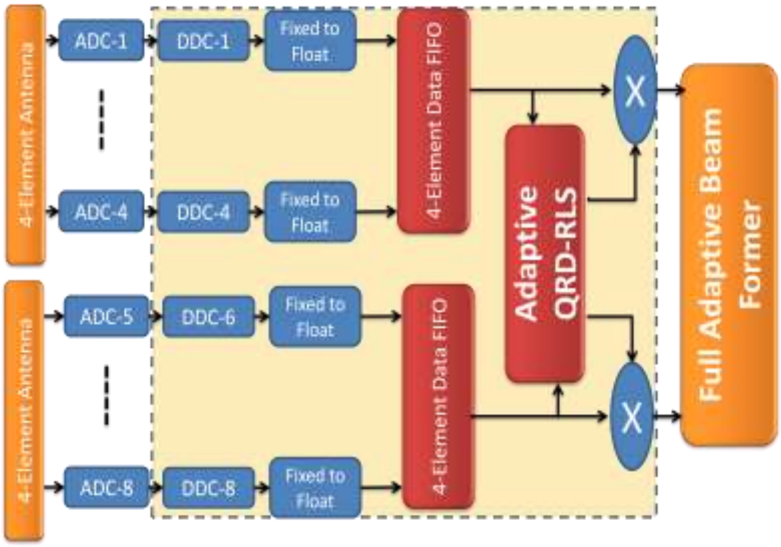


Figure 4: Adaptive Beamformer VLSI Architecture With Floating Point Operation

Approximations of the real numbers can be categorized as finite -precision and the floating-point[10] numeration entities. The floating-point arithmetic permits us to symbolize broad range of numbers with constant precision. Finite-point arithmetic symbolizes a decreased range of numbers with appropriate precision absolute. Employing floating precision arithmetic is highly expensive when considered in terms of hardware and results in inefficient architecture, particularly. when implemented using FPGA. Alternatively the fixed precision arithmetic representation leads to hardware design that is efficient and also introduces a small amount of error. Here the design employs two’s compliment method and the representation of finite precision arithmetic comprise of the sign bit, integer part along with the fractional part. Quantities that find occurrence in the algorithm are characterized with mn bits designed for the integer along with m bits which are employed for the representation of the fractional bits. Finally, representing a finite precision illustration would entail mn+m+1 bit, were one bit is utilized as a sign bit representation.

QR decomposition[14] needs the usage of arithmetic operations, the calculations required finds an expansion when there is growth in the dimension of the matrix. Large number of calculations is observed during the matrix decomposition process where the upper triangular matrix and the orthogonal matrix are obtained. Calculations employed for the decomposition process are the basic arithmetic computation which is a direct approach; complicated computations significantly alter the precision of the entity likely producing the inefficient values during the implementation in the FPGA. Finite-precision arithmetic reduces the obtained precision and thereby introducing two different types of errors which are coined as the round-off and the truncate error. Round-off error makes its presence when the additional bits are required in the outcome than the bits that are reserved usually after the arithmetic computations. Due to the restricted range of bits that are required to produce numbers, truncation error occurs. These problems must be handed rigorously in order to stop the overflow which results in the erroreneous outcome.

5.1 Resource comparison

The FPGA based Adaptive Beam Former Architecture [24] is designed using VHDL. The VLSI architecture modeling has been carried out and simulated. The functional verification has been done to validate the correctness of the results. Initially the 8 element linear array and 16 element linear array configuration is simulated. Then a planar array of 16 element configuration has been implemented. The implementation of

ADBF for 1 beam is done on Kintex-7 series FPGA and later 2 beams also. Due to the requirement of more resources on hardware, for multiple beams, some reuses of the modules are necessary and hence time required to calculate the multiple weights will be large. The resource utilization for this implementation is given in Table 1 and 2.

Table 1: Resource utilization of FPGA-I and FPGA II for implementation of Inverse QRD RLS for Adaptive beam former of 8 elements with Floating point operations

Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	119840	508400	23%
Number of Slice LUTs	144126	254200	56%
Number of fully used LUT-FF pairs	82594	181372	45%
Number of bonded IOBs	73	500	14%
Number of Block RAM/FIFO	8	795	1%
Number of BUFG/BUFGCTRLs	5	32	15%
Number of DSP48E1s	643	1540	41%

Table 2: Resource utilization of FPGA-III for implementation of Adaptive beam former for sixteen elements. (Floating point)

Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	16316	508400	3%
Number of Slice LUTs	13396	254200	5%
Number of fully used LUT-FF pairs	10776	18936	56%
Number of bonded IOBs	66	500	13%
Number of Block RAM/FIFO	1	795	0%
Number of BUFG/BUFGCTRLs	1	32	3%

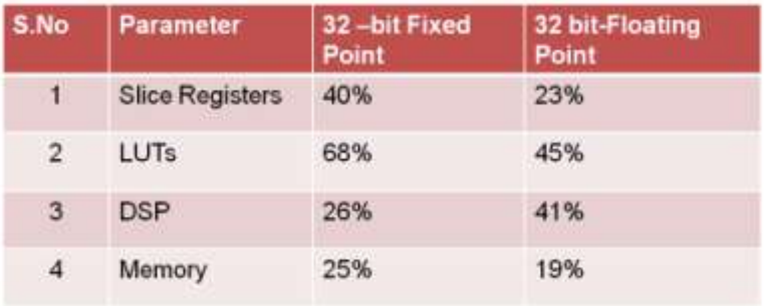
Table 3: Resource utilization of FPGA-I and FPGA II for implementation of Inverse QRD RLS for Adaptive beam former of 8 elements with fixed point operations.

Logic Utilization	**Used**	**Available**	**Utilization**
Slice Registers Number	119840	508400	0%
Slice LUTs Number	144126	254200	0%
Fully used LUT-FF pairs	82594	181372	72%
Bonded IOBs Number	73	500	0%
Block RAM/FIFO Number	8	795	1%
BUFG/BUFGCTRLs Number	5	32	12%
DSP48E1s Number	643	1540	1%

Table 4: Resource utilization of FPGA-III for implementation of Adaptive beam former for sixteen elements with fixed point operations.

Logic Utilization	**Used**	**Available**	**Utilization**
Slice Registers Number	16316	508400	0%
Slice LUTs Number	13396	254200	0%
Fully used LUT-FF pairs Number	10776	18936	48%
Bonded IOBs Number	66	500	13%
Block RAM/FIFO Number	1	795	0%
BUFG/BUFGCTRLs Number	1	32	12%

Table 5: Resource utilization Comparison for 32 bit Fixed and Floating point operations.



S.No	Parameter	32 –bit Fixed Point	32 bit-Floating Point
1	Slice Registers	40%	23%
2	LUTs	68%	45%
3	DSP	26%	41%
4	Memory	25%	19%

VI. SIMULATION AND IMPLEMENTATION RESULTS OF ADAPTIVE BEAM FORMER

The Architecture developed is simulated using the Xilinx FPGA Models and the results are as shown in Figure 5.

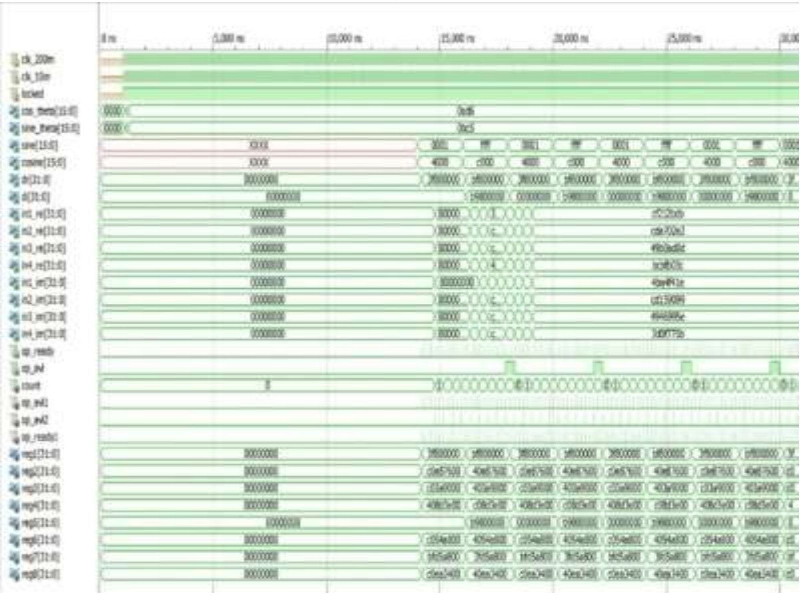
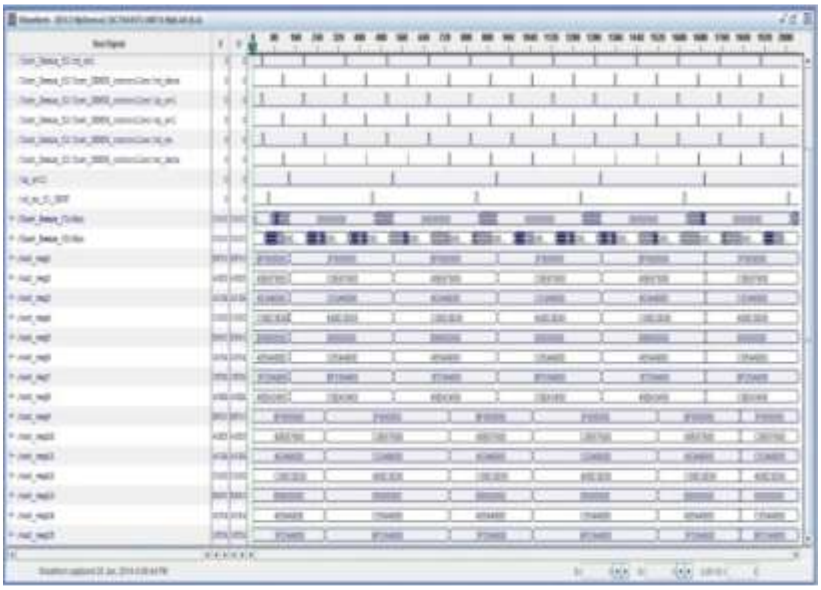


Figure 5. Adaptive Weight calculation for 16 Element Array at Simulation level.

The architecture developed was simulated with 32 bit fixed point arithmetic as shown in Figure 6. The weights were calculated for various look angles for 16 element phased Array.

- Fixed point 32 bit input values the total time for the first out is 720 ns
- All the output values are rounded off to 32 bit instead of 64 bit[24].
- This is carried out in order to prevent the bit growth.
- This has resulted into great reduction of the Resource utilisation
- The optimal weights are available after 720 ns instead of 800 ns
-



- **Figure 6.** Adaptive Weight calculation for 16 Element Array at Chip level.
-

VII. RESULTS OF ADAPTIVE BEAMS FORMED WITH FIXED POINT OPERATIONS

1) Following is the result of the implementation of adaptive beam forming using inverse QRD-RLS algorithm for Eight element planar array with, Input azimuth arrival angle for desired signal= -30 deg. Input elevation arrival angle for desired signal =10 deg.

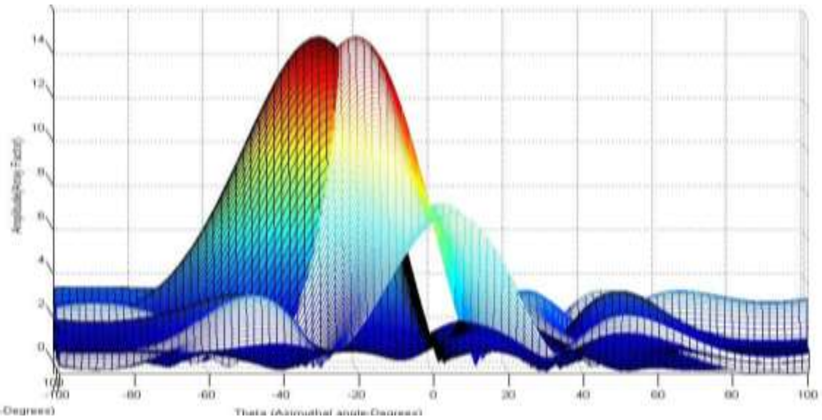


Figure.7. Planar array of eight elements Two adaptive beams formed from the hardware.

2) Following is the result of the implementation of adaptive beam forming using Inverse QRD-RLS algorithm for 16 element planar array. Input azimuth arrival angle for desired signal = 0 deg, Input elevation angle of arrival for desired signal =0 deg.

3)

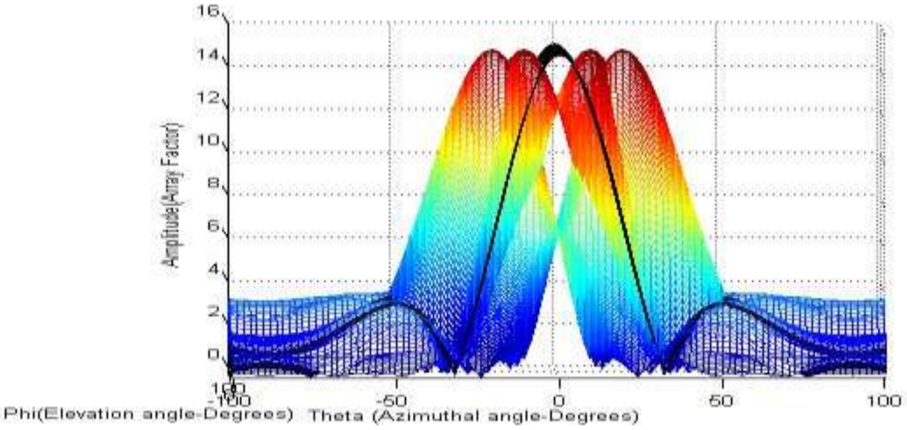


Figure.8. Planar array of sixteen elements Five adaptive beams.

VIII. RESULTS OF ADAPTIVE BEAMS FORMED WITH FLOATING POINT OPERATIONS

IX. RLS algorithm for 16 element planar array is shown in Fig Input arrival angle for desired signal = 0 deg, Interference angle =0 deg, 2 beams are produced.

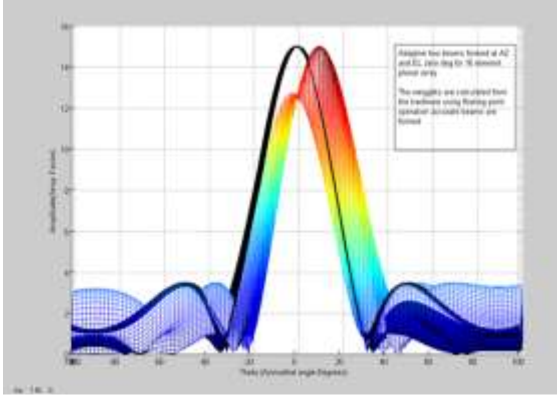


Figure 9: Planar array of sixteen elements two adaptive beams.

X. Result of implementation of adaptive beam forming using Inverse QRD-RLS algorithm for 16 element planar array is shown in Fig. Input arrival angle for desired signal = 0 deg, Interference angle =0 deg, 5 beams are produced.

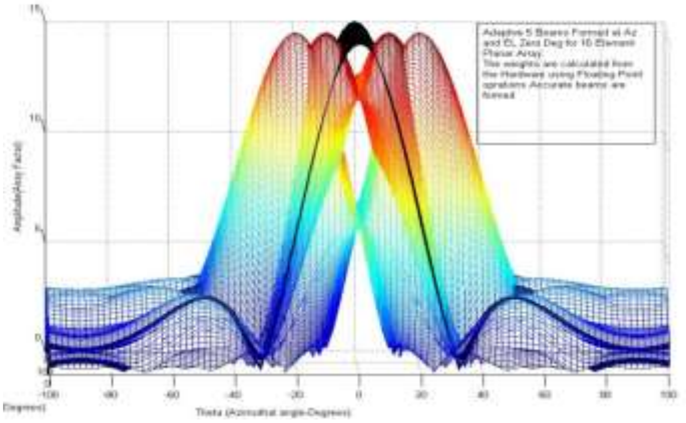


Figure 10: Planar array of sixteen elements five adaptive beams.

I. CONCLUSION AND FUTURE SCOPE

We have developed a 16-element Planar and Linear array Adaptive Digital Beam Former system. The weights are calculated using the highly efficient Inverse QRD-RLS algorithm. The Kintex-VII, FPGA is used to form adaptive beams, and it has enabled a remarkable reduction in the area utilization compared to the discrete and analog versions. This pipelined architecture generates multiple beams simultaneously from a given array matrix of 16 elements. FPGA based implementation finds huge applications in modern radars as this implementation makes the system immune to the limitations that the analog methods face. At the same time, the proposed Adaptive beam-forming system enjoys advantages of a reconfigurable design and low cost. The developed ADBF system is doing the computation of weights on the fly inside the FPGA, so as to enable radar to track the changes in continuously varying environment. Thus making the ADBF system robust and efficient. This system enjoys the benefit of Anti-Jamming, so that a Null can be formed in the direction of the Jammer automatically.

High throughput rate of 1 MHz enables the radar to track the target moving with the high speed. Depending on the availability of the resources and the input sample rate; conventional or the inverse QRD-RLS algorithm can be selected. The same work can be extended for a planar array of larger array dimensions.

ACKNOWLEDGEMENT

We are thankful to Director LRDE for giving the opportunity of providing the hardware for testing and performance evaluation. We are thankful to Mr.Bhairan.G.C Project Assistant KIT Tiptur and Ms.Harshitha.P. Project Assistant from SIT, Mangalore for helping us in making the report and compilation of the results and drafting the complete report.

REFERENCES

[1]. Owais Talaat Waheed, Ayman Shabra, Ibrahim (Abe) M. Elfadel FPGA Methodology for Power Analysis of Embedded Adaptive Beamforming IEEE,2015

[2]. Bing Han, Zengli Yang, and Yahong Rosa Zheng “FPGA Implementation of QR Decomposition for MIMO-OFDM Using Four CORDIC Cores” - Signal Processing for Communications Symposium IEEE ICC 2013.

[3]. Gayathri R Prabhu,Bibin Johnson,Sheeba Rani J “FPGA based Scalable Fixed Point QRD core using Dynamic Partial Reconfiguration” 28th International Conference on VLSI Design and 14th International Conference on Embedded Systems,2015.

[4]. Anjitha D, Shanmugha Sundaram G “Fpga Implementation Of Beamforming Algorithm For Terrestrial Radar Application”, International Conference on Communication and Signal Processing, April 3-5, 2014, India.

[5]. S.Niu, S. Aslan, and J. Saniie, “FPGA based architectures for high performance adaptive FIR filter systems,” in Proceedings of the IEEE International Instrumentation and Measurement Technology Conference ,I2MTC ’13, pp. 1662–1665, 2013.

[6]. U. Vishnoi and T. G. Noll, “A family of modular area- and energy-efficient QRD-accelerator architectures,” in Proceedings of the International Symposium on System on Chip, SoC '13, pp.1–8, Tampere, Finland, October 2013.

[7]. B. Han, Z. Yang, and Y. R. Zheng, “FPGA implementation of QR decomposition for MIMO-OFDM using four CORDIC cores,” in Proceedings of the IEEE International Conference on Communications ,ICC ’13, pp. 4556–4560, June 2013.

[8]. J. Rust, F. Ludwig, and S. Paul, “Low complexity QR-decomposition architecture using the logarithmic number system,” in Proceedings of the Conference on Design, Automation and Test in Europe, pp. 97–102, EDA Consortium, 2013.

[9]. S. Aslan, S. Niu, and J. Saniie, “FPGA implementation of fast QR decomposition based on givens rotation,” in Proceedings of the IEEE 55th International Midwest Symposium on Circuits and Systems ,MWSCAS ’12, pp. 470–473, IEEE, August 2012.

[10]. D. Chen and M. Sima, “Fixed-point CORDIC-based QR decomposition by givens rotations on FPGA,” in Proceedings of the International Conference on Reconfigurable Computing and FPGAs ,ReConFig ’11, pp. 327–332, IEEE, Canc´un,Mexico, December 2011.

[11]. F. Sobhanmanesh and S. Nooshabadi, “Parametric minimum hardware QR-factoriser architecture for V-BLAST detection,” IEEE Proceedings: Circuits, Devices and Systems, vol. 153, no. 5, pp. 433–441, 2006.

[12]. I. H. Kurniawan, J.-H. Yoon, and J. Park, “Multidimensional Householder based high-speed QR decomposition architecture for MIMO receivers,” in Proceedings of the IEEE International Symposium on Circuits and Systems ,ISCAS ’13, pp. 2159–2162, Beijing, China, May 2013.

[13]. M. Shabany,D. Patel, and P. G.Gulak, “Alow-latency low-power QR-decomposition ASIC implementation in 0.13 □m CMOS,” IEEE Transactions on Circuits and Systems I: Regular Papers, vol.60, no. 2, pp. 327–340, 2013.

[14]. D. Patel, M. Shabany, and P. G. Gulak, “A low-complexity high speed QR decomposition implementation for MIMO receivers,”in Proceedings of the IEEE International Symposium on Circuits and Systems ,ISCAS ’09, pp. 33–36, Taipei, Taiwan, May 2009.

[15]. R. Gayathri and J. Sheeba Rani, “Fixed point pipelined architecture for QR decomposition,” in Proceedings of the IEEE International Conference on Advanced Communication Control and Computing Technologies ,ICACCCT ’14, pp. 468–472,IEEE, 2014.

[16]. J. A. Apolinário Jr., QRD-RLS Adaptive Filtering. New York: Springer, 2009.

[17]. M. Shoaib, S. Werner, and J. A. Apolinário Jr., “Reduced complexity solution for weight extraction in QRD-LSL algorithm,” IEEE Signal Process. Lett., vol. 15, pp. 277–280, 2008.

[18]. F. Riera-Palou, “Reconfigurable structures for direct equalization in mobile receivers [Ph.D. thesis],” University of Bradford, Bradford, UK, 2014.

[19]. P. Kabilan and K. Meena, “Performance comparison of a modified LMS algorithm in digital beam forming for high speed networks,” IEEE International Conference on Computational Intelligence and Multimedia Applications, Vol. 4, pp. 428-433, 2007

[20]. P. S. R. Diniz, “Adaptive Filtering: Algorithms and Practical Implementation”. 3rd edition, Springer, New York, NY, USA 2008,

[21]. Sumit Verma, Arvind Pathak, “Digital beam forming using RLS – QRD algorithm”, International Journal of Engineering Research & Technology ,Vol. 1 Issue 5, ISSN: 2278-0181 , 2012.

[22]. X. Wang, M. Leeser, “A truly two-dimensional systolic array FPGA implementation of QR Decomposition”. ACM Transactions on Embedded Computing Systems, Vol. 9 No. 1, Article 3, pp 1-10. October 2009.

[23]. S.Haykin, “Adaptive Filter Theory,” 4th Edition, Pearson, ISBN 978-81-317-0869-9, pp 4-22, 2011.

[24]. D.Govind Rao, N.S.Murhty, Vengadarajan, ”Adaptive VLSI Architecture of Beam Former for Active Phased Array Radar”, International Journal of New Computer Architectures and their Applications (IJNCAA) 3(2): pp.19-29, 2013.