Efficient Acceleration for Total Focusing Method Based on Advanced Parallel Computing in FPGA

Chong Wang

State Key Laboratory of Acoustics, Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, P. R. China.
University of Chinese Academy of Sciences, Beijing 100049, P. R. China.

Jie Mao, Tao Leng, Ze-Yu Zhuang and Xiao-Min Wang

State Key Laboratory of Acoustics, Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, P. R. China.

(Received 1 December 2016; accepted 12 June 2017)

Total focusing method (TFM) has improved resolution and accuracy over traditional ultrasonic phased array technology. In this paper, an advanced parallel architecture in field programmable gate arrays is suggested to significantly accelerate the imaging efficiency of TFM. Several techniques are investigated, including the real-time concurrent calculation for time of flight, parallel generation of multiple pixels, and the Hilbert transform to the pixels array. This architecture achieves the real-time computation of the flight times for each pixel and the concurrent generation of double pixels for TFM imaging. Compared to conventional methods, the efficiency of TFM imaging is greatly accelerated and the impact from the increase of element and pixel number is also effectively reduced. Simulation data was used to verify the architecture, and experiment results confirmed that the efficiency was only related to the pulse repeated frequency and element number, which reaches to the physical limitation of TFM inspection. This approach also shows that high efficiency is maintained when pixel number increases, and a strict real-time imaging can be achieved in this architecture. As a result, an effective way for the fast inspection with TFM is provided.

1. INTRODUCTION

Ultrasonic phased array testing for non-destructive evaluation has developed dramatically in recent years owing to the improved sensitivity and coverage over single element transducer.[1,2] Based on the full matrix capture, the total focusing method (TFM) is an advanced phased array imaging algorithm, which was proposed by Holmes in 2005.[3]

From a long time, the TFM can only be achieved offline due to the huge amount of data acquisition and computation requirements. Real-time inspection is a key aspect for ultrasonic testing in industrial fields, and many efforts have been devoted to accelerate the TFM inspection over the past years.

Parallel computing platforms, including CPU, GPU, and FPGA, are suggested to implement the TFM algorithm. Early efforts were suggested by using general purpose PC-based modules in 2012 and achieved an imaging speed at merely 20 Hz.[4] Today, GPU components have an improved ability in concurrency over CPU. However, limited by the transfer bottleneck of raw data and the nature of inherent serial execution, the real-time imaging in GPU could only be achieved at low configurations and the frame rate decreases seriously when the number of elements or pixels increases.[5,6]

The FPGA provides a suitable parallel characteristic and an implementation of TFM was also provided in 2013.[7] However, the maximum frame rate that it can achieve is 73 Hz and decreases to less than 20 Hz in a higher configuration of 128×128 pixels. The efficiency is not enough as the design is far from perfect for the parallel calculation of TFM algorithm. With the flight times usually computed by software, it takes several periods for FPGA to get them serially from external memory, which extends the time to generate an imaging pixel and limits the parallelization of the process. Moreover, as a result of the serial generation of pixels in the design, the imaging efficiency decreases dramatically when the number of pixels or elements increases. A fast beamformer with time-multiplexing was provided for several beam forming lines.[8] But the concurrency is sensitive to the sampling rate, which limits the fast calculation for TFM in industrial applications with a high sampling frequency normally.

Aiming to meet the need of real-time applications, this paper investigates an advanced parallel architecture in FPGA to accelerate the efficiency of TFM calculation, including the real-time concurrent time of flight calculation, parallel generation of multiple pixels, and the Hilbert transform to the pixels array. Compared to traditional designs, a higher degree of parallelization for the TFM calculation can be achieved. Experimental results on phased array system indicated that a strict real-time TFM imaging could be obtained and the high efficiency remained constant when the pixel number increased.

2. PARALLEL COMPUTING ARCHITECTURE FOR TFM

2.1. TFM Algorithm

Figure 1 shows the concept view of the TFM algorithm. The algorithm meshes the region of interest in a grid of pixels. A pixel is generated by summation of data from all the transmission-reception pairs.[3]

The intensity I of a pixel $P(x,z)$ is expressed by the Eq. (1).

$$I\left[P(x,z)\right]=\left|\sum_{i=1}^{N}\sum_{j=1}^{N}h_{ij}(T_{ip}+T_{pj})\right|; \qquad (1)$$

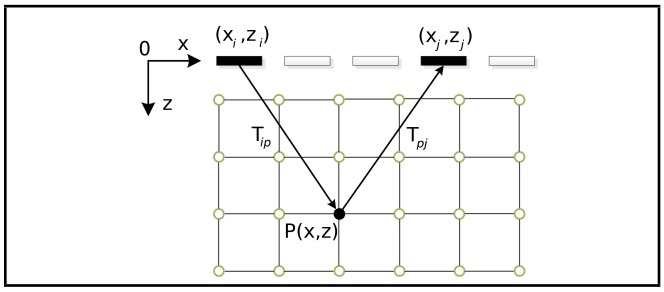

Figure 1. Total Focusing Method.

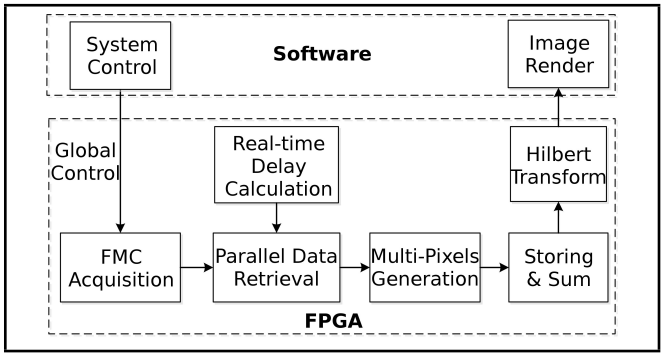

Figure 2. TFM calculation architecture.

where, h_{ij} is the analytical version of the echo received by element j when element i transmits. $T_{ip} + T_{pj} = \frac{\sqrt{(x_i-x)^2+z^2}+\sqrt{(x_j-x)^2+z^2}}{c}$, is the time of flight between P and element pairs (i, j). N is the elements number. x_i and x_j are the coordinates of element i and j, respectively. c is the longitudinal velocity of sound.

2.2. Analysis of Parallel Calculation for TFM

Pulse repeated frequency (PRF) represents the time interval between two adjacent FMC sequences. In fact, a sequential TFM image could be generated after each FMC sequence.7 When the calculation could always be completed in this time interval, the frame speed of TFM would reach to PRF/N, which is the physical limitation for TFM testing.

The TFM is a process of superimposing to generate imaging pixels, using the FMC data retrieved by time of flight. The efficiency to generate a single pixel depends on the degree of concurrence to obtain the flight times and the bandwidth to extract the corresponding data. Maximum efficiency would be achieved if the N flight times for a pixel are concurrently computed, as well as the data can be retrieved in parallel. When multiple pixels could be executed concurrently in their maximum efficiency, the imaging speed of TFM would be significantly improved, and the impact of the pixels number could be reduced.

2.3. Parallel Architecture for TFM Calculation

The parallel architecture proposed in the paper is illustrated in Fig. 2. Here, FMC acquisition and TFM calculation are both implemented in FPGA while the global system control and image rendering are designed in software.

The following advantages and characteristics are achieved in this parallel architecture:

(a) Times of flights are concurrently calculated in real-time in FPGA, rather than the serial reading from memories.

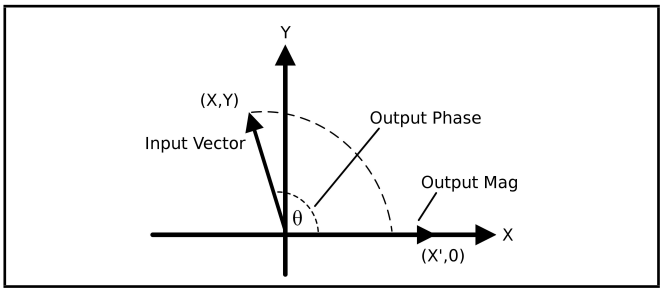

Figure 3. Vector Translation (Polar to Rectangular).

(b) Multiple pixels could be generated in parallel to improve the imaging speed.

(c) Analytical version of the pixels array can be achieved by Hilbert transform in FPGA without any operations of software.

This architecture makes a better utilization of the parallel characteristic in FPGA, and an improved degree of parallelization for TFM calculation is obtained.

3. REAL-TIME CALCULATION OF TIME OF FLIGHT BASED ON CORDIC

3.1. CORDIC Algorithm

The coordinate rotational digital computer (CORDIC) algorithm can be used as vector translation from polar to rectangular by rotating the input vector (X, Y) around the circle until the Y component equals zero,[10] as Fig. 3 shows.

The magnitude of (X, Y) can be obtained from the outputs (X', Y') after the rotation, given by Eq. (2).

$$X' = \sqrt{(X^2 + Y^2)}. \tag{2}$$

The CORDIC translation provides convenience for the calculation of propagation distances, which is the first step to obtain the flight times. The input vector (X, Y) can be presented as the difference of coordinates between the elements and pixel. The output X' after the translation, is the one-way distance from element i to pixel P. This can be given by Eq. (3).

$$X' = \text{OutputMag} = \sqrt{(x_i - x)^2 + z^2}. \tag{3}$$

4. CONCURRENT CALCULATION OF FLIGHT TIMES IN REAL-TIME

Bidirectional propagation path is the sum of the two one-way distances from the element pair to the pixel. For example, when the first element transmits, the concurrent calculation of the N flight times for one pixel is illustrated in Fig. 4.

For each element to the pixel, N one-way distances are calculated from CORDIC_1 to CORDIC_N, and N bidirectional paths are the sum between the output of CORDIC_1 and the outputs of the N modules respectively. The flight times are obtained by dividing the propagation distances by the longitudinal velocity of sound.

All the flight times are calculated and used for corresponding pixels in real-time without any storage. This effort achieves a higher degree of concurrence as we can get N flight times simultaneously in just a clock period of FPGA. When pixel resolution increases, more CORDIC modules could be instantiated with the benefits of rich DSP resources in current FPGAs.

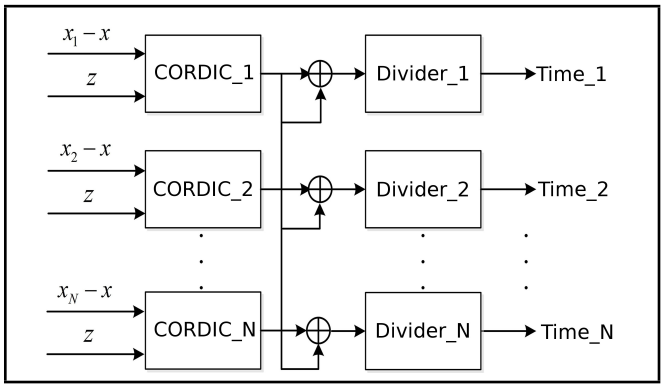

Figure 4. Parallel calculation for N flight times.

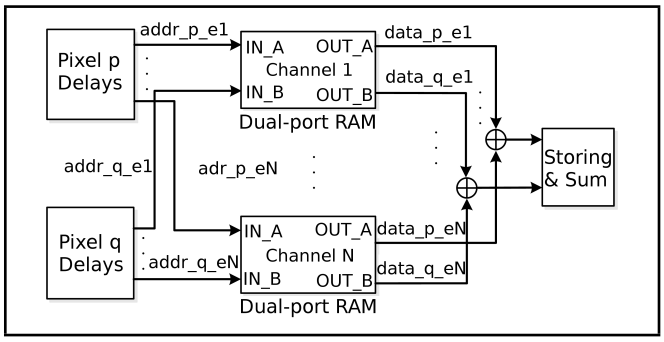

Figure 5. Parallel pixels generation.

4.1. Output Quantization Error

In vector translation, the input quantization noise results in the output quantization error of a half least significant bit (LSB) on the magnitude output. For example, an 8 bits magnitude number of the input will cause a quantization noise of $1/2^8 = 0.4\%$. In this paper, all the coordinates are represented in nanometre, using 30 bits to guarantee the precision of computation.

5. PARALLEL GENERATION OF MULTIPLE TFM PIXELS

5.1. Parallel Generation of Pixels

This paper proposes a way to accelerate the retrieving bandwidth doubly by the true dual-port RAM in FPGA. When write and read operations are executed simultaneously, retrieving can be conducted while FMC data is captured into the dual-port RAMs. When only read operation is needed, two ports of the RAMs can be used for retrieving independently. The bandwidth then will be doubled after the FMC acquisition.

Some pixels could be generated during the FMC acquisition. The efficiency for the remaining pixels can be doubled as the data is retrieved concurrently for two pixels. As Fig. 5 shows, modules Channel 1 to Channel N cache the FMC data for each channel. Next, addr_p_e1∼addr_p_eN are the N retrieving addresses for pixel p while addr_q_e1∼addr_q_eN are the ones for q. They are determined from the flight times divided by the sampling period. Then the outputs are summed respectively to generate the pixels, which are stored temporarily into a FIFO.

The parallelization is improved with the parallel generation of two pixels in one FPGA clock period. With abundant memory resources in current FPGA chips, more dual-port RAM could be used when the array becomes larger. This benefit makes it reliable to reduce the impact from the increase of element number.

Table 1. Frame rates of TFM.

Elements	Pixels	CORDIC modules	Memory utilized	Calculating time μs	Frame rate Hz
16	60×60	32	8%	40	312.5
	100×100	32	9%	40	
	200×200	32	15%	100	
32	60×60	64	10%	40	156.25
	100×100	64	11%	40	
	200×200	64	17%	100	

5.2. Efficiency of the Parallel Architecture

The execution time of FPGA can be concluded due to its strict timing sequence. When the PRF = 5 kHz and FMC acquisition works at 50 MHz, data captured for each channel is 2000 and the TFM calculation works at 250 MHz, the imaging frame rates in a Xilinx XC7K325T FPGA in different configurations are shown in Table 1.

All the configurations in Table 1 can be completed within 200 μs, which is the time interval of FMC sequences when PRF is 5 kHz. A high frame rate of 312.5 Hz in 16 elements can be achieved and is equal to PRF/N. The number of CORDIC modules and the memory utilized in FPGA are used to represent the computational complexity in different configurations. The number of CORDIC modules is twice of the elements as the parallel generation of two pixels. More memories will be consumed when either the element or pixel number increases. When the pixel number increases to 200×200, the frame rate remains unchanged. When the element number increases, the frame rate will be inversely proportional to N. So the frame rate of 32 elements is 156.25 Hz in the same pixels configuration, which is half of the one of 16 elements. It can be seen from Table 1 that the TFM efficiency is dramatically improved with higher parallelization, and the impact from the increase of the complexity in different configurations is reduced.

6. HILBERT TRANSFORM TO PIXELS ARRAY

For subsequent processing it is necessary to obtain the analytical version of the FMC signals.[9] The benefits to TFM image are observed when the Hilbert transform is used to convert the real-time domain signal into complex form.[6] In this paper, a Hilbert transformer was designed in FPGA to convert the real signals into complex form. It should be pointed out that an ideal Hilbert transformer cannot be realized since the impulse response is non-causal. Nevertheless, Hilbert transformers can be approximated as finite impulse response (FIR) digital filters.[11] As shown in Fig. 6, a Hilbert FIR filter with 128 orders was designed as an approximate all-pass filter with strict linear phase response. A high order number of 128 helps to get steeper transitional band and small ripples in the passband of 0 ∼ Fs/2.

In the post-processing of GPU or CPU, the Hilbert transform is usually applied to the FMC signals before the calculation of pixels. 3,6 It should be observed that performance of this process is directly proportional to the size of the full matrix of data and the operations for each A scan cause an extended computation time. Moreover, additional memory will be needed to store the complex version of the full matrix data. Hilbert transform for the pixels rather than the original data will make the process simplified and low cost. As it is applied to the pixels,

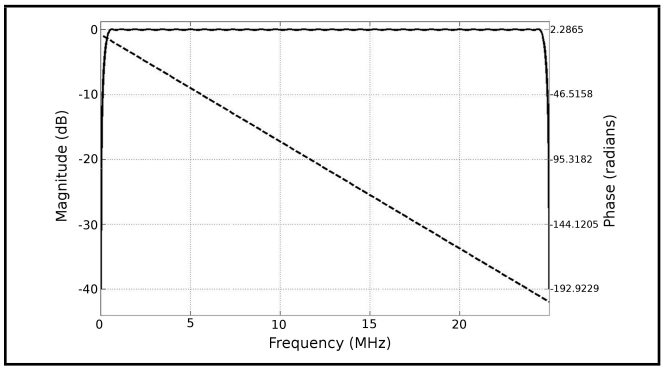

Figure 6. Magnitude (solid line) and phase responses (dashed line) of the Hilbert FIR filter.

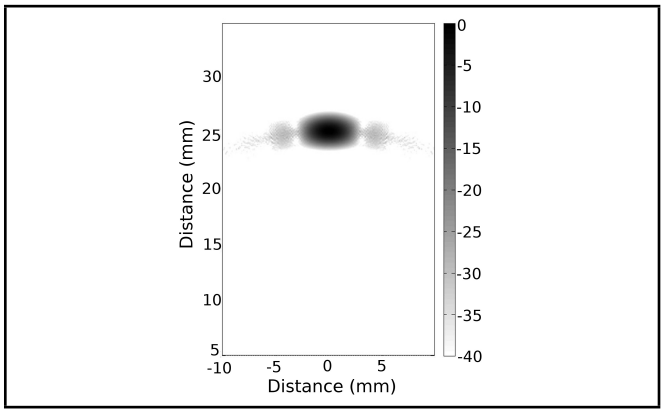

Figure 7. Result when Hilbert transform is applied to original FMC signals.

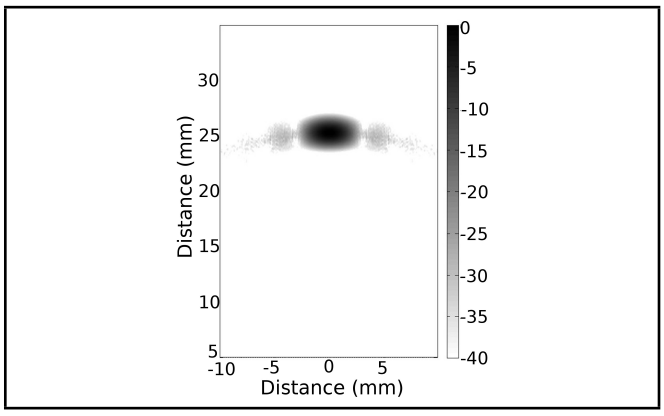

Figure 8. Result when Hilbert transform is applied to final pixels array.

no additional memory is needed and only one transformation is used for the final pixels array, while N transformations will be necessary when it is applied to the original A scans.

A simulation by MATLAB was performed to compare the two methods. The array parameters are in the Table 2 and the velocity of sound is 6300 m/s. The output of each element was a five cycle, Gaussian windowed tone burst with a 5 MHz centre frequency and a -6 dB bandwidth of 50%. Figure 7 shows the result scaled in dB to display a point-like reflector $(0, 20\lambda)$ when the Hilbert transform is applied to the original FMC signals before the generation of pixels. Figure 8 shows the result when the same simulation data was calculated in FPGA with the Hilbert transform applied to the final pixels array. It can be seen that the results are equivalent with each other. However, the latter is more suitable for the pipeline design of FPGA with less resources utilization. It should be pointed out that

Table 2. The array parameters.

Element number	16
Element width	0.53 mm
Element pitch	0.63 mm
Central frequency	5 MHz
Bandwidth (-6 dB)	50%

Table 3. Frame rates at various configurations.

Imaging area	Pixels number and resolution	Frame rates
12×12 mm^2	60×60@0.2 mm	312.5 Hz
20×20 mm^2	100×100@0.2 mm	312.5 Hz
20×20 mm^2	200×200@0.1 mm	312.5 Hz

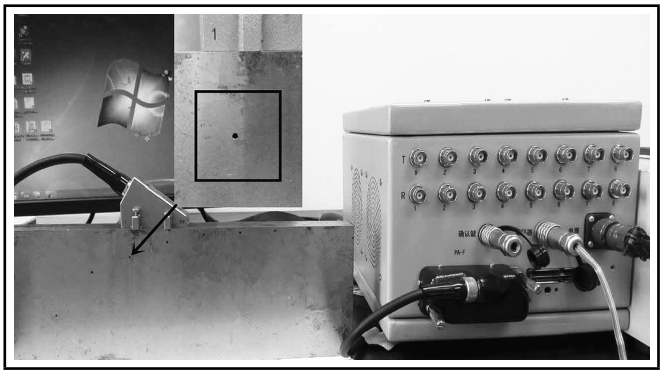

Figure 9. Phased array platform.

the calculation in FPGA was completed instantly, while it took several minutes in MATLAB as its poor calculating efficiency. The output of Hilbert transformer is the imaginary part of the real pixels. The envelope of the complex version will be obtained in pipelining mode in FPGA and then sent to software for TFM image rendering.

7. EXPERIMENT RESULTS

An experimental test was devised on phased array system to benchmark performance of the design. FMC acquisition and TFM calculation are both designed in Xilinx XC7K325T FPGA. An Olympus 5L64-A2 linear array transducer, consisting of 64 elements with 5 MHz central frequency was used. The first quarter of the array was used as the system could support 16 channels. The test sample was a steel block with a 1 mm side drilled hole at a depth of 20 mm. The hole was placed under the center of the first quarter of the array. Figure 9 shows the experimental platform.

The region of interest is an area of 20×20 mm^2 with 100×100 pixels at a resolution of 0.2 mm. The TFM imaging is illustrated in Fig. 10. It can be seen from the image that some trailing signal lies below the hole, probably caused by the creeping waves along the circumference of the hole. The frame rate of TFM imaging is 312.5 Hz when PRF is 5 kHz, consistent with the analysis in Section 4.2. It remains constant when the resolution increases to 200×200@0.1 mm. The frame rates at various configurations are tested and shown in Table 3.

The results verify the significant improvement to the efficiency of TFM imaging in this parallel architecture. The high efficiency will not reduce as the pixel number increases. When the element number is constant, the frame rate reaches to PRF/N. Resource utilization rate of this FPGA in the experiment with 16 elements and 200×200 pixels is shown in Table 4.

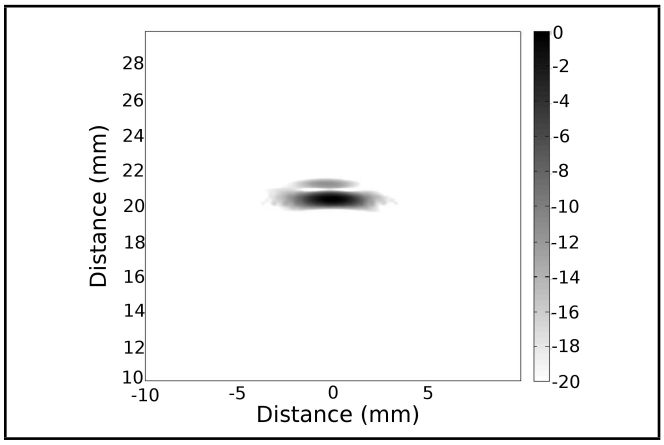

Figure 10. TFM with 16 elements.

Table 4. Resource utilization of FPGA.

Resource	FFs	LUTs	RAMs	DSPs
Utilization (%)	24	34	15	18

Experimental results have shown the effect of the FPGA architecture based on the phased array system with sixteen channels. The developments to get a larger inspection area and better images using more elements in this system are the main works in future.

8. CONCLUSIONS

This paper presents an advanced parallel computing architecture in FPGA to significantly improve the TFM imaging efficiency and effectively reduce the impact from the increase of elements or pixels number. Two efforts based on CORDIC translation and the True Dual-Port RAM in FPGA are designed to achieve a higher degree of parallelization for the calculation of TFM algorithm compared with conventional designs. Hilbert transform is applied to the pixels array to get effective images with small resources consumption.

The architecture in FPGA was verified firstly by simulated data, and then experiment was devised to benchmark the performance of imaging and frame rates. Experimental results showed that a high frame rate of 312.5 Hz was achieved in a normal configuration and the reduction of efficiency could be effectively avoided when pixels increase. The work makes great benefit for TFM to be a strict real-time imaging technology, reaching to the physical limitation of PRF/N. It makes it possible for the fast automatic TFM scanning applications in industrial fields. As the frame rate is dependent on the PRF and element number, a higher frequency of pulse repetition is needed to further improve the frame rate when an array is given. However, a sparse array method based on the theory of effective aperture could be considered in the future to reduce the number of active elements during the full matrix capture, which will further improve the TFM efficiency and the image quality.

ACKNOWLEDGEMENTS

This work was supported by the Scientific Research and Equipment Development Project of CAS under Grant No. Y329011331, and the National Natural Science Foundation of China under Grant No. 11427809. Supports from Zhao-Bin Peng and Guo-Xuan Lian are gratefully acknowledged.

REFERENCES

[1] McNab, A. and Campbell, M. J. Ultrasonic phased arrays for nondestructive testing, *NDT International*, **20** (6), 333–337, (1987). https://dx.doi.org/10.1016/0308-9126(87)90290-2

[2] Jensen, J. A., Nikolov, S. I., Gammelmark, K. L., and Pedersen, M. H. Synthetic aperture ultrasound imaging, *Ultrasonics*, **44**, e5–e15, (2006). https://dx.doi.org/10.1016/j.ultras.2006.07.017

[3] Holmes, C., Drinkwater, B. W., and Wilcox, P. D. Post-processing of the full matrix of ultrasonic transmit-receive array data for non-destructive evaluation, *NDT & E Int.*, **38** (8), 701–711, (2005). https://dx.doi.org/10.1016/j.ndteint.2005.04.002

[4] Lines, D. I. A., Wharrie, J., and Hottenroth, J. Real-time full matrix capture + total focusing and other novel imaging options using general purpose PC-based array instrumentation, *Insight - Non-Destructive Testing and Condition Monitoring*, **54** (2), 86–90, (2012). https://dx.doi.org/10.1784/insi.2012.54.2.86

[5] Lambert, J., Pedron, A., Gens, G., Bimbard, F., Lacassagne, L., and Iakovleva, E. Performance evaluation of total focusing method on GPP and GPU, *Design and Architectures for Signal and Image Processing (DASIP)*, Karlsruhe, Germany, (2012). http://ieeexplore.ieee.org/document/6385392/#full-text-section

[6] Sutcliffe, M., Weston, M., Dutton, B., Charlton, P., and Donne, K. Real-time full matrix capture for ultrasonic non-destructive testing with acceleration of post-processing through graphic hardware, *NDT & E Int.*, **51**, 16–23, (2012). https://dx.doi.org/10.1016/j.ndteint.2012.06.005

[7] Njiki, M., Bouaziz, S., Elouardi, A., Casula, O., and Roy, O. A Multi-FPGA Implementation of Real-Time Reconstruction Using Total Focusing Method, *2013 IEEE 3 rd Annual International Conference on Cyber Technology in Automation, Control and Intelligent Systems (CYBER)*, Nanjing, China, (2013). https://dx.doi.org/10.1109/cyber.2013.6705491

[8] Cruza, J. F., Perez, M., Moreno, J. M., and Fritsch, C. Real time fast ultrasound imaging technology and possible applications, *Physics Procedia*, **63**, 79–84, (2015). https://dx.doi.org/10.1016/j.phpro.2015.03.013

[9] Kino, G. S. *Acoustic waves: devices, imaging and analog signal processing*, Prentice-Hall, New Jersey, (1987).

[10] Xilinx Inc., LogiCORE IP Product Guide CORDIC v6.0, Retrieved from http://www.xilinx.com/support/documentation/ /ip_documentation/cordic/v6_0/pg105-cordic.pdf, (Accessed July 10, 2016).

[11] Romero, D. E. T and Dolecek, G. J. *Digital FIR Hilbert transformers: fundamentals and efficient design methods*. INTECH Open Access Publisher, (2012). https://dx.doi.org/10.5772/46451