Single-FPGA, Scalable, Low-Power, and High-Quality 3D Ultrasound Beamformer

W. Simon*, A. C. Yüzügüler*, A. Ibrahim*, F. Angiolini*, M. Arditi*, J.-P. Thiran*† and G. De Micheli*

* École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

†Department of Radiology, University Hospital Center (CHUV) and University of Lausanne (UNIL), Switzerland

***Abstract*—We present an efficient FPGA architecture suitable for a medical 3D ultrasound beamformer. We tackle the delay calculation bottleneck, which is the heart and the most critical part of the beamformer, by proposing a computationally efficient design that is able to perform volumetric real-time beamforming on a single-chip FPGA. The design has been demonstrated for a 32×32-channel receive probe, and we extrapolated the requirements of the architecture for 80×80 channels.**

I. Motivation

Medical ultrasound (US) imaging is well established, being used in a wide range of applications including detecting static structures, such as tumors, and studying dynamic phenomena like blood flow and valve functionality. US imaging is comprises three main processes: insonification, beamforming (BF), and visualization. Insonification is the process of emitting Radio Frequency (RF) acoustic waves from a piezoelectric transducer, called probe, through a body region. The waves are reflected from inhomogeneous tissues interfaces that act as scatterers due to acoustic impedance mismatches. The returned echoes are digitized and processed through an algorithm called *Beamforming (BF)*. Finally, a post-processing step should be performed, including mapping the beamformed signals into screen image pixels.

Recently, 3D US imaging has become available. A key advantages is that, since whole volumes are acquired at once, it is possible to remove the traditional dependence on having a trained sonographer operating the probe, in order to locate minute anatomical structures by fine adjustments of the position and orientation of the transducer. This enables telesonography, where even an unskilled operator can upload scans to a hospital where trained radiologists will issue a diagnosis. Unfortunately, present-day 3D imagers are bulky and expensive, suitable only for clinics and hospitals. A portable US platform with cheap, battery-operated electronics would be a breakthrough, enabling telesonography in rescue environments, in rural areas, and in developing countries, with major societal benefits. To this end, we undertake to implement 3D beamforming on a single FPGA.

II. Problem Definition and Previous Work

Beamforming is the core of any US imaging machine. It is the process of mapping the echoes to their origins by summing them along a certain delay profile, that represents the two-way time-of-flight of the acoustic wave from the origin to each scatterer, and back to the all the piezoelectric elements. BF also includes *apodization*, the weighting of the delayed echoes by a factor that compensates for antenna directivity effects.

In volumetric US imaging, a software-based implementation of the beamformer is not optimal if we target a battery-powered platform, whereas a hardware design offers major potential energy savings. One of the critical challenges of 3D US imaging is the number of receiving channels of high-end transducers, up to 100×100 elements, and the correspondingly massive computations required for image reconstruction. Different state-of-the-art systems, either commercial or academic, have only been able to deal with the 3D BF challenge by reducing the number of receive channels, hence simplifying the computation through the usage of far fewer elements, typically 256 or fewer. This can be achieved by using different methods like analog micro-BF [1] or multiplexing [2]. Nonetheless, these systems are still highly complex and expensive.

The critical kernel of beamforming is delay calculation. For example, in our configuration, to perform 3D imaging of a single 64×64×600 voxel volume using a 32×32-element probe, we need to calculate about 2.5 Gdelays. At a 50 volumes/s reconstruction rate, 125 Gdelays/s need to be computed. Each delay value is the outcome of a square root calculation. This is obviously a challenge, presently unmet, especially in the context of a portable, low-power, low cost imaging system. In this work we show a beamformer that leverages an efficient delay calculation method. Its architecture has been optimized for an FPGA platform. Our architecture is scalable and has the ability to perform the delay calculation of a fully sampled high-end transducer of thousands of elements in a single latest-generation FPGA. We demonstrate that our design can support as many as 32×32 receive channels within a single Kintex Ultrascale KU-040 FPGA, or more if using even more advanced chips.

III. Proposed Beamformer Architecture

As mentioned in Section II, delay calculation is the most computationally expensive part of ultrasound imaging. An approach has been proposed [3] that benefits from the first order Taylor approximation of the square root operation. This approach simplifies the delay calculation to:

$$t_p(O,S,D) \approx t_p(O,R,D) - \frac{x_D \sin\theta}{c} - \frac{y_D \sin\phi \cos\theta}{c} \quad (1)$$

where t_p is the propagation time for the sound wave to be emitted from the origin $O = (0,0,0)$ to the scatterer $S = (r,\theta,\phi)$ or reference point $R = (r,0,0)$, and back scattered to the probe element $D = (x_D, y_D, 0)$. c is the speed of the sound in the medium. θ and ϕ are the azimuth and elevation angles, respectively.

This approach of calculating the delays is computationally efficient because it simplifies the complex square root to just a calculation of a small reference set of delays plus two additions. Since the possible values of θ, ϕ, x_D, and y_D are discrete and few, the correction terms can be fully pre-calculated and stored in small tables.

In this work we designed an FPGA architecture that leverages this method in the context of a full beamformer. Fig. 1 shows the whole FPGA system including our beamformer custom block. The latter communicates via an AXI interface, and houses 1024 BRAM memories for storing slices of the input data, a block to compute 1024 delays per cycle according to the algorithm shown above, a 1024:1 adder tree comprising ten 2:1 levels, an output FIFO storage, and control circuitry. The overall system includes a MicroBlaze processor subsystem and an Ethernet interface that is presently used for all I/O. The beamformer FPGA architecture has been designed based on nappe-by-nappe processing [4], i.e. by beamforming "onion layers" of voxels at constant radius and then moving to the next radius. This order of computation is alternative to the traditional scanline-based one, and is more efficient since it optimizes the consumption of the data coming from the probe elements.

The delay calculation logic must first of all compute the reference delay value $t_p(O,R,D)$. This delay is itself the output of a square root computation, but is much more slowly-varying than the desired $t_p(O,S,D)$ as it holds constant over a nappe of 64×64 voxels. Therefore, we can afford to implement this square root on a Xilinx CORDIC core, minimizing design effort. Note that this calculation on-the-fly is an improvement over [3], where the reference delay was fetched from an off-chip table, resulting in memory bandwidth issues. We chose to use the "optimum" pipelining configuration of the CORDIC core; this saves area and latency in return for a lower

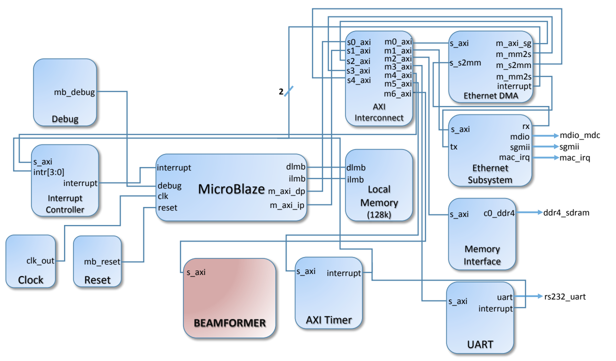


Fig. 1. The block diagram of the FPGA including the *Beamformer* custom block and its interconnection with other blocks.

operating frequency, that we still found to be in excess of 125 MHz. The delay calculation architecture shown in Fig. 2(a) is composed mainly of 32 sub-blocks; each contains a BRAM in which we store precalculated elevation correction coefficients ($c_2 = \frac{y_D \sin\phi \cos\theta}{c}$), and LUTs for storing precalculated azimuth correction coefficients ($c_1 = \frac{x_D \sin\theta}{c}$). Since the size of the c_2 table is much larger than the c_1 table, we used BRAMs instead of LUTs to save logic. Using adders, for each elevation step, we add the stored c_2 to the reference delays and then for each azimuth step we add the stored c_1.

In the design of Fig. 2, we store a time slice of the 32×32 input channels, using one BRAM per element. The delay calculation logic produces 32×32 delay values per cycle, used to address all the BRAMs simultaneously, thus selecting one input sample for each probe element. The resulting 1024 samples are fed into a 1024:1 adder tree, resulting in an output of one reconstructed voxel per cycle (Fig. 2(b)). For our reference volume of 2.5M voxels (Table I), the architecture reconstructs ideally one volume every 2.5M clock cycles, i.e. 50 volumes/s at 125 MHz.

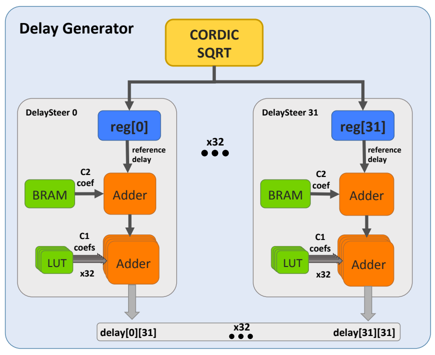
Results: The first row of Table II shows that we can fit the delay generation logic for 32×32 receiving channels at a theoretical reconstruction rate of 50 volumes/s on a single Kintex UltraScale KU040 FPGA [5]. Presently, this rate cannot be fully exploited due to the bottleneck caused by the fact that in our testing setup the input data is fed via an Ethernet interface; in a real system, the inputs would be fed via a higher-bandwidth interface from the ultrasound probe. The estimated power consumption is 4 W, compatible with a portable implementation. The architecture can scale to a larger channel count, but is limited on the KU040 by the BRAM resources. The channel count could be increased up to 80×80 on a larger FPGA like the Virtex UltraScale XCVU190 [5], with a further BRAM optimization - we are currently using a full BRAM for each channel, but two channels could share one, since Ultrascale macros feature dual-ported read access.

IV. Conclusion and Future Work

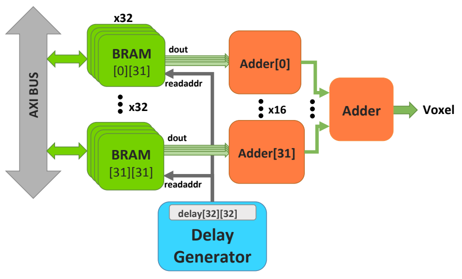
In this work we have developed a design that is able to perform single-chip beamforming for realtime 3D US. It supports fully-digital matrix probe BF with 1024 elements by minimizing the required storage space and optimizing the processing cost. We are currently working on implementing and integrating the apodization block to the

TABLE I
System Specifications

Parameter	Value
Speed of sound in tissue	1540 m/s
Transducer center frequency	4 MHz
Transducer bandwidth	4 MHz
Transducer matrix size	32 × 32
Wavelength	0.385 mm
Sampling frequency	32 MHz
Focal points	64 × 64 × 600 = 2.5 M voxels



(a)



(b)

Fig. 2. Proposed architecture of the delay computation blocks. The receive delay is computed by applying steering coefficients to the calculated reference delay (a), then the calculated 32 × 32 delays index the input sample BRAMs. Finally the samples are summed to reconstruct a voxel (b).

beamformer, as well as ancillary logic for pre- and post-processing. We plan on further improving the reconstruction rate by raising the clock frequency, and by using more efficient I/O interfaces, like PCI Express, instead of the current Ethernet. Our final objective is a complete and efficient 3D US back-end system on a single FPGA board.

Acknowledgment

The authors acknowledge Swiss Confederation funding through the UltrasoundToGo project of the Nano-Tera.ch initiative.

References

[1] Philips Electronics N.V., “iE33 xMATRIX echocardiography system,” www.healthcare.philips.com.
[2] R. Sampson, M. Yang, S. Wei, C. Chakrabarti, and T. Wenisch, “Sonic millip3de: A massively parallel 3d-stacked accelerator for 3d ultrasound,” in *High Performance Computer Architecture (HPCA2013), 2013 IEEE 19th International Symposium on*, Feb 2013, pp. 318–329.
[3] A. Ibrahim, P. A. Hager, A. Bartolini, F. Angiolini, M. Arditi, L. Benini, and G. De Micheli, “Tackling the bottleneck of delay tables in 3D ultrasound imaging,” in *Proceedings of the 2015 Design Automation and Test in Europe (DATE 2015) Conference*, March 2015, pp. 1683 – 1688.
[4] P. Vogel, A. Bartolini, and L. Benini, “Efficient parallel beamforming for 3D ultrasound imaging,” in *Proceedings of the 24th Edition of the Great Lakes Symposium on VLSI*, ser. GLSVLSI ’14. New York, NY, USA: ACM, 2014, pp. 175–180. [Online]. Available: http://doi.acm.org/10.1145/2591513.2591599
[5] Xilinx Inc., “Ultrascale FPGA: Product tables and product selection guide,” 2016, http://www.xilinx.com/support/documentation/selection-guides/ultrascale-fpga-product-selection-guide.pdf#KU.

TABLE II
Beamformer Architecture Results.
Kintex UltraScale KU040 implementation results.
Virtex UltraScale XCVU190 extrapolated results.

Supported Channels	Logic LUTs	Regs	BRAM	DSP	Clock	Volume Rate
32×32*	78%	25%	100%	0.3%	125 MHz	50 vps
80×80**	86%	19%	43%	0.3%	125 MHz	50 vps