Efficient Sample Delay Calculation for 2-D and 3-D Ultrasound Imaging

Aya Ibrahim, Pascal A. Hager, Andrea Bartolini, *Member, IEEE*, Federico Angiolini, Marcel Arditi, *Member, IEEE*, Jean-Philippe Thiran, Luca Benini, and Giovanni De Micheli, *Fellow, IEEE*

***Abstract*—Ultrasound imaging is a reference medical diagnostic technique, thanks to its blend of versatility, effectiveness, and moderate cost. The core computation of all ultrasound imaging methods is based on simple formulae, except for those required to calculate acoustic propagation delays with high precision and throughput. Unfortunately, advanced three-dimensional (3-D) systems require the calculation or storage of billions of such delay values per frame, which is a challenge. In 2-D systems, this requirement can be four orders of magnitude lower, but efficient computation is still crucial in view of low-power implementations that can be battery-operated, enabling usage in numerous additional scenarios. In this paper, we explore two smart designs of the delay generation function. To quantify their hardware cost, we implement them on FPGA and study their footprint and performance. We evaluate how these architectures scale to different ultrasound applications, from a low-power 2-D system to a next-generation 3-D machine. When using numerical approximations, we demonstrate the ability to generate delay values with sufficient throughput to support 10 000-channel 3-D imaging at up to 30 fps while using 63% of a Virtex 7 FPGA, requiring 24 MB of external memory accessed at about 32 GB/s bandwidth. Alternatively, with similar FPGA occupation, we show an exact calculation method that reaches 24 fps on 1225-channel 3-D imaging and does not require external memory at all. Both designs can be scaled to use a negligible amount of resources for 2-D imaging in low-power applications and for ultrafast 2-D imaging at hundreds of frames per second.**

***Index Terms*—Beamforming, delay calculation, efficient FPGA architecture, ultrasound imaging, ultrasonography, volumetric ultrasound – 3D ultrasound.**

I. Introduction

ULTRASOUND imaging is used as a medical diagnostic technique in a broad variety of fields, from obstetrics to cardiology. The main assets of ultrasound imaging are its avoidance of ionizing radiation, non-invasiveness, moderate cost, and versatility.

Manuscript received September 15, 2016; revised December 17, 2016; accepted January 21, 2017. Date of publication May 25, 2017; date of current version July 26, 2017. This paper was recommended by Associate Editor G. Yuan.

A. Ibrahim, F. Angiolini, M. Arditi, and J.-P. Thiran are with the Swiss Federal Institute of Technology in Lausanne, Lausanne 1015, Switzerland (e-mail: aya.ibrahim@epfl.ch; federico.angiolini@epfl.ch; marcel.arditi@epfl.ch; jean-philippe.thiran@epfl.ch).

P. Hager, A. Bartolini, and L. Benini are with the Swiss Federal Institute of Technology in Zurich, Zurich 8092, Switzerland (e-mail: pascal.hager@iis.ee.ethz.ch; andrea.bartolini@iis.ee.ethz.ch; luca.benini@iis.ee.ethz.ch).

G. De Micheli is with the Swiss Federal Institute of Technology in Lausanne, Lausanne 1015, Switzerland, and also with the Swiss Federal Institute of Technology in Zurich, Zurich 8092, Switzerland (e-mail: giovanni.demicheli@epfl.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBCAS.2017.2673547

Ultrasound imagers range from small, portable devices for 2D imaging only [1]–[6] to bulky hospital equipment for both 2D and 3D imaging [7], [8]. Obviously, there is a significant disparity in terms of image quality and other features among these different ends of the spectrum. For example, simpler machines produce 2D images, e.g. along arbitrary sections of the patient's body, while more sophisticated equipment is also capable of 3D scans, acquiring echoes from a volume at a time [9]. This latter technique is particularly beneficial for volumetric analyses, such as when evaluating cardiac functions (e.g. Philips iE33 [10]), or for the study of the movement of complex body structures, such as heart valves. It is also popular in obstetrics, thanks to its realistic and impressive rendering of fetal features (e.g. GE Voluson E10 [11]).

In both 2D and 3D systems, acoustic impedance discontinuities in tissue structures (e.g. due to density or stiffness changes) cause ultrasound pulse waves to be scattered, with a fraction of that energy being detected back by the probe elements as pressure signals. The echo amplitude and phase shift depend on the amplitude and position of such reflectors. An image can be reconstructed by summing together the echo signals according to appropriate *delay profiles* through a process called *beamforming*. Beamforming requires high-throughput operations, and represents the key processing stage in digital ultrasound imaging.

3D imaging requires probes composed of matrices - rather than one-dimensional arrays - of vibrating elements, for example a grid of 9212 in [12]. This is the square of the typical amount in a linear probe for 2D imaging, and entails orders of magnitude more beamforming calculations. A way to bypass the computation bottleneck is to resort to analog pre-beamforming [13]–[15], i.e. summing the readout of multiple probe elements according to fixed delay profiles to generate a single output signal. This effectively turns a matrix probe into another with far fewer elements, simplifying computation, but sacrificing image quality and resolution in the process. Other approaches to work around the high number of probe elements in matrices and their related challenges are to use either multiplexing [16] or sparse 2D arrays [17], [18], where the matrix probe is undersampled by activating only some of the probe elements at a time, in patterns. However, it has been concluded that there is a direct relationship between the density of the probe elements and the quality of the reconstructed images, determined by the beam profile with its mainlobe width and sidelobe levels; therefore, a high channel count is still desirable. All academic and commercial 3D systems are forced to choose a trade-off between supporting a higher channel count for better image quality and managing the implementation cost. Even so, powerful computation resources, incompatible with portable operation, are needed, and the highest achievable refresh rates are just a few frames per second [19].

In several systems, beamforming is achieved by software, on CPUs [20], GPUs [20], [21] or DSPs [22]–[24]. However, software implementations are not optimal in the case of battery-powered operation, where dedicated hardware designs can offer major potential energy savings. More crucially, software beamforming faces critical limitations in 3D imagers. In this work, we choose to focus on a hardware, rather than software, implementation of the beamformer logic. This is with two objectives in sight: on one hand, optimized energy consumption for portable 2D systems; at the opposite extreme of the spectrum, the unprecedented capability to reconstruct 3D images while exploiting the full-resolution probe readout. The challenge is to do so in a manageable hardware footprint; for example, the academic SARUS [25] system, which is very advanced but only supports up to 1024-element matrix probes, runs on 320 FPGAs. The more recent second-generation ULA-OP platform [26] uses 8 high-end FPGAs and 16 DSPs to support only 256-channel beamforming.

In this paper, we focus on one specific part of the beamformer design, i.e. the *delay profile computation*, which is its innermost, and thus most critical, kernel. As will be shown in Section II, in the 3D case, trillions of delay values are needed per second. This makes pre-calculation impractical, due to either on-chip storage or off-chip bandwidth constraints. Therefore, specialized optimization techniques are essential. The paper contributions are as follows:

1) We investigate circuit architectures that compute delay values with suitable parallelism to tackle even next-generation 3D beamformers, with the crucial ability to take into account each single transducer element individually for maximum image quality. This paper builds upon and extends our previous publications [27]–[30]. To the best of our knowledge, this has not been attempted by any other research group and is beyond the capabilities of current commercial products.
2) We assess the performance/cost of such architectures. It is our goal to devise implementations that are suitable for single-chip implementations even for next-generation 3D systems. The architectures we propose can be realized on either FPGA or ASIC; in this paper we will show an FPGA mapping.
3) We demonstrate the scalability of these architectures from low-power to very-high-performance systems. We will identify three design points representative of a spectrum of ultrasound imagers, from a battery-operated 2D system focused on minimum power consumption to a next-generation 3D device, and assess the scalability of our proposed techniques.

We will explore two alternative methods to compute delay values. First, we will revisit an architecture originally presented in [28], where all delays are computed on-the-fly with an optimized circuit; we will call this approach TABLEFREE. Next, we will show an alternative technique, TABLESTEER, which relies on a much smaller precalculated delay table [29], fit for in-FPGA storage, compounded by a small circuit that completes the computation. We will then map both on a state-of-the-art FPGA and analyze their trade-offs. Both methods will be shown to meet the three key challenges of delay computation: accuracy, compactness, and throughput. However, the two methods have slightly different objectives. TABLEFREE targets precise computation and highest-quality imaging, and due to its inherent complexity, is best suited to an ASIC realization, where area and power can be most optimized. TABLESTEER adopts approximations to simplify the hardware design, and therefore is more suitable for an FPGA implementation, which allows for a much quicker and simpler path to a hardware prototype. The proposed techniques could be exploited standalone or as a complement to sparse matrix approaches, further improving the reconstruction quality by supporting more probe elements at the same hardware cost.

The remainder of the paper is organized as follows. In Section II, we will summarize some background information on the importance of delay calculation in ultrasound systems, while Section III will put our contribution in the context of the most closely related works. Sections IV and V will present the key ideas behind our proposed delay calculation methods, and Section VI will show how they can be implemented in hardware, with particular emphasis on the FPGA back-end that is chosen for comparison in this paper. Section VII will explain the configurations of 2D and 3D imaging under which we chose to study the performance of our proposals, which will be done in Section VIII. Finally, Section IX will draw conclusions.

II. Delay Calculation in 3D Ultrasound Systems

In this section, we briefly review some basics of ultrasound imaging to put our work in context.

A. Transmit Focusing

Ultrasound imaging systems can apply focusing at transmit time and at receive time. *Transmit focusing* consists of exciting transducer elements with such a timing that the sound field in front of the transducer has specific intensity maxima, corresponding to constructive interference, while the rest of the field is insonified less intensely. Image resolution depends on the acoustic pressure, and can thus be increased locally. “Unfocused” waves can also be emitted, with a plane or diverging wavefront, to insonify and image the whole volume in a single pass. In this paper, we generally assume imaging strategies based on diverging beams, but as will be shown in the following, this is without loss of generality.

B. Receive Focusing and Beamforming

On receive, the echoes sampled at each transducer element must be summed according to a delay profile that models the round-trip propagation time necessary to travel to a body scatterer and back towards each element of the probe (*beamforming*). During reception, all modern systems improve resolution

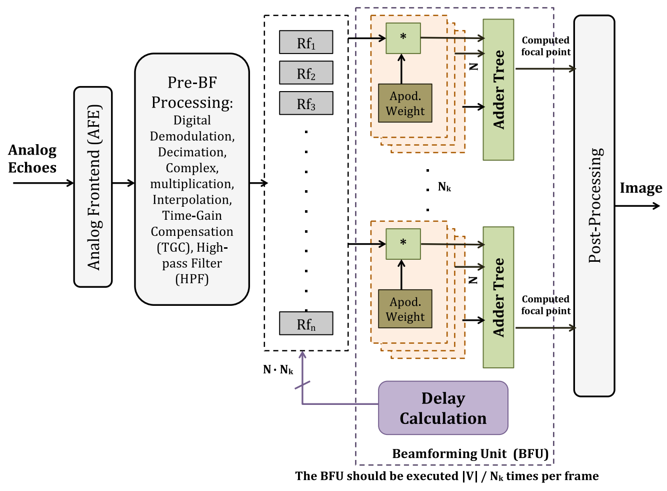

Analog Echoes
Analog Frontend (AFE)
Pre-BF Processing: Digital Demodulation, Decimation, Complex, multiplication, Interpolation, Time-Gain Compensation (TGC), High-pass Filter (HPF)
Rf1
Rf2
Rf3
Rfn
N · Nk
Apod. Weight
*
N
Adder Tree
Computed focal point
Nk
Delay Calculation
Beamforming Unit (BFU)
The BFU should be executed |V| / Nk times per frame
Post-Processing
Image

Fig. 1. Architectural diagram of a generic ultrasound imaging system, with focus on the beamformer processing the sampled RF radio-frequency signals. N is the number of transducer matrix elements, N_k is the number of voxels per cycle the BFU can produce, and $|V|$ is the number of focal points to be computed.

substantially by applying dedicated delay profiles for each image point, i.e. they "focus" at every depth and location as a function of the pulse-echo round-trip propagation time (*dynamic receive focusing*). Unfortunately, using this technique also severely increases the computation cost of image reconstruction.

This paper tackles this challenge in a general way. For convenience of illustration, we show a very generic beamforming circuit in Fig. 1, where a BeamForming Unit (BFU) takes as inputs the sampled backscattered echoes from N probe elements. The BFU consists of N_k parallel blocks, each of which capable of RX-focusing on one image point per cycle. The BFU requires multiple cycles to reconstruct a whole frame, depending on the desired output resolution and on the N_k parallelism.

For convenience, we use the following notation to describe the kernel of the beamforming algorithm:

$$s(S) = \sum_{d=1}^{N} e(D_d, t_p(O, S, D_d))w(S), \forall S \in V \qquad (1)$$

S is a point in the volume of interest V; the beamforming calculation must be repeated $\forall S \in V$. The outcome $s(S)$ is a signal that follows the reflectivity of scatterers at location S, and will eventually be used to calculate the brightness of the corresponding image pixel. N is the number of receive elements accessible in the probe, while e is the amplitude of the echo received by element $D_d (d \in 1, \dots N)$ at the time sample t_p. The value of t_p represents the propagation delay that sound waves incur from a given emission reference O (more on the choice of O to follow), to the point S, and back to the probe's destination element D_d. Finally, the echo intensities are weighted by a coefficient w that provides apodization control [31], i.e. weighs differently the echo samples to compensate the antenna-like effects that lead to sidelobes in the transducer's emission and reception radiation characteristics.

In this paper we focus exclusively on the kernel of the beamforming loop, i.e. on the identification, at runtime, of which t_p-th sample of the echo buffer $e(D_d)$ should be summed to RX-focus on each point S. This can be formalized as:

$$t_p(O, S, D_d) = \frac{|\overrightarrow{OS}| + |\overrightarrow{SD_d}|}{c} \qquad (2)$$

where c is the speed of sound in the medium, which we assume constant at 1540 m/s. This formulation holds for both 2D and 3D ultrasound imaging.

C. Zone Imaging

The imaging strategy may dictate that all points $S \in V$, or only a subset $S \in V_z \subset V$ thereof, may be computed per insonification (a strategy called "zone imaging" [32]). Zone imaging allows for an optimized choice of TX focus, and for a more conservative use of memory and computation resources. In this paper we assume that V is split in $Z \geq 1$ disjoint zones V_z $(z \in 1, \dots, Z)$ of equal size. Therefore, $Z \geq 1$ insonifications are required to capture and image the full volume of $|V| = Z \cdot |V_z|$ points.

D. Emission Reference

The point O has been defined as the "emission reference". Contrary to S and D, which have precise physical meanings, the location of this point in space can be freely chosen by the designer, provided that the implementation is consistent with the choice. Typically, O is chosen so that it makes as easy as possible to express the transmission delay towards each focal point in terms of $\overrightarrow{OS}$. For example, when emitting (non-steered) plane waves, the point O may be chosen far behind from the transducer's plane, and the transmission delay would be determined by the component of $\overrightarrow{OS}$ on the depth axis.

In this paper, we will use diverging beams for illustrative purposes. To properly compute transmission delays, in this geometry, it is a natural option to choose O in the location of the *virtual source* of such beam, i.e. at some point V_s behind the transducer. Doing so, however, means that the minimum $t_p(O, S, D)$ (for a scatterer at a point S right on the transducer's surface) is non-zero, while in physical reality, such echoes do immediately follow the emission. Therefore, the indexing into $e(D, t_p(O, S, D))$ must use a proper constant offset considering that the starting value of t_p is not 0. It is also possible to position O on the projection of V_s onto the transducer, solving the offset problem, but complicating the mathematical notation. In the following, to simplify the discussion, we will locate O at the virtual source V_s, without loss of generality.

Importantly, the imaging strategy may call for a new virtual source at each insonification to add diversity. In ultrafast imaging [33]–[35], in particular, the volume of interest can be acquired T times with different transmission origins; the beamformed results are combined to improve image quality, a process called *compounding*. We call the total set of transmission origins W, with $O \in W$. For each frame, the emission origin must be repositioned $|W| = Z \cdot T$ times.

TABLE I
SYSTEM SPECIFICATIONS

Parameter	Symbol	Value
Physical		
Speed of sound in tissue	c	1540 m/s
Transducer Head		
Transducer center frequency	f_c	4 MHz
Transducer bandwidth	B	4 MHz
Transducer matrix size	$e_x \times e_y$	100×100
Wavelength	λ	$c/f_c = 0.385$ mm
Transducer pitch		$\lambda/2$
Transducer matrix dimensions	d	$50\lambda = 19.25$ mm
Element directivity (acceptance angle)		0.707 rad
Beamformer		
Imaging volume ($\theta \times \phi \times d_p$)		$73° \times 73° \times 500\lambda$
Sampling frequency	f_s	32 MHz
Focal points (FP)		$128 \times 128 \times 1000$

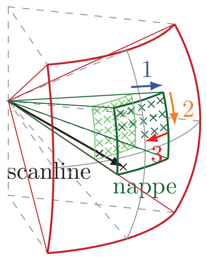


Fig. 2. Focal point calculation order in a nappe-oriented beamformer: all points at a given depth are calculated; then the depth is incremented. A scanline is also shown for comparison; a scanline-oriented beamformer will reconstruct points along the whole scanline, then move to the next.

E. System Specifications

Table I summarizes the specifications of our sample system. As a rule of thumb [36], the sampling frequency of a beamformer needs to be chosen 4 to 10 times higher than the center frequency f_c of the transducer, or two times its bandwidth with quadrature sampling [37]. We have chosen a sampling rate of $8f_c = 32$ MHz. This rate defines the resolution with which the delays need to be computed, i.e. at a granularity of about 30 ns. Furthermore, we assume that our delay computation is used in combination with a bandpass beamformer [38], which outputs complex samples at a rate of the transducer bandwidth $B = 4$ MHz. Given this rate, 1000 samples, pre-interpolation, are sufficient to fully reconstruct the output signal for the given penetration depth of 500λ. Post-interpolation, 8000 samples are computed. As mentioned above, the choice of a bandpass beamformer reduces the number of required computations. It should be noted that all resources and computation requirements mentioned in this work are derived based on the specification values listed in Table I.

F. Beamforming Order

The traditional beamforming approach reconstructs images by *scanlines*. An alternate approach reconstructs the volume one *nappe* [27], i.e. one surface with constant distance from the origin, at a time (see Fig. 2). Both approaches require the same calculations and the same amount of delay coefficients, and produce the same images. The only difference is in the ordering of the calculations: the nappe-by-nappe technique processes data in roughly the same order it is acquired from the transducer (earlier echoes processed before later echoes), contrary to the scanline-by-scanline approach, which travels back and forth time-wise. The former choice is advantageous in terms of circuit implementation, because shallower buffering is required. For this reason, without any prejudice to quality, in this paper we will outline delay calculation approaches that are optimized for a nappe-by-nappe beamformer. Since the required delay coefficients and calculation throughput are strictly the same, it is obvious that the proposed circuitry can also be used in a scanline-by-scanline beamformer, at the cost of either extra buffering of the calculated delay values, or by applying architectural tweaks.

In the following, we will discuss three key challenges related to the calculation of propagation delays: accuracy, compactness of storage, and throughput.

G. Challenge 1: Delay Calculation Accuracy in Beamforming

The accuracy of delay calculation is essential for high-resolution beamforming, because the latter relies on fine-grained time differences to locate the position of body features. Any imprecision may result in poor focus, image artifacts, aliasing, etc.

H. Challenge 2: Size of Required Delay Tables

Since t_p is used as an index into the buffer of data samples e, it must be calculated, as seen above, with a very fine grain of about 30 ns. Moreover, the values of t_p must be calculated $\forall O \times S \times D$, i.e. the number of delays per frame is $\Psi := |W| \cdot |V_z| \cdot N = T \cdot |V| \cdot N$. In a typical 2D system, reconstructing planar images of $|V| = 128 \times 1000$ focal points, using a transducer of $N = 128$ elements, this means $\Psi = 16.4 \times 10^6$ delay coefficients for each origin $O \in W$, which is acceptable for storage in a pre-computed table.

However, in a 3D system, with $|V| = 128 \times 128 \times 1000$ focal points, given a $N = 100 \times 100$-element transducer, the theoretical number of delay values to be calculated is about $\Psi = 164 \times 10^9$ per origin. With a 13-bit delay representation, 266.5 GB of storage would be required. Even when the geometry of the problem allows for simplifications due to symmetry, this is obviously challenging to either pre-compute, due to the storage requirements, or to calculate in realtime.

I. Challenge 3: Access Bandwidth to Delay Tables

Another challenge is that delay values need to be available with high throughput, in order to achieve realtime beamforming. The coefficients must be accessed once per frame, at the frame rate f_r. The rate at which the delays need to be computed is $\Psi \cdot f_r$. A 3D image, assuming $f_r = 15$ vps, requires therefore about 2.5×10^{12} delay values/s. So, if we assume that a delay value would be represented in 13 bits, 32.5 Tbits/s of bandwidth would be needed. This is obviously well outside of the capabilities of any realistic off-chip memory interface, and a better approach is called for.

III. Previous Work

Today's state-of-the-art 3D ultrasound systems perform analog beamforming in element subgroups in the transducer head to decrease the number of channels that are carried along the cable from a few thousands to a few hundred [13], [39], [40]. This is called "micro-beamforming" or "pre-beamforming", where precomputed fixed analog delays are applied to the signals received by groups of transducer elements, which are then compounded in a single analog signal [13]–[15]. Pre-beamforming reduces the channel count, which also reduces the data volume to be processed, thus making the digital beamformer orders of magnitude less complex. The ACUSON SC2000 Volume Imaging Ultrasound System computes up to 64 beams in parallel, i.e. up to 160×10^6 FP/s, using analog beamforming [41].

On the other hand, analog pre-beamforming limits the image quality, since applying a fixed delay profile to each element group is equivalent to setting a fixed focus for that group. It is desirable to have a fully-digital, maybe even software [42], beamformer to have the capability to dynamically set the focus position during receive. However, the large amount of input signals to be processed individually poses a major computation and bandwidth challenge [25]. To enable high-resolution, high-frame-rate 3D imaging, multiple scanlines need to be beamformed from a single insonification using parallel receive beamforming [41], [43], such as for ultra-fast imaging [44].

The problem of how to compute delay coefficients to feed such beamformers at a very high throughput has been recognized as critical. For example, Sonic Millip3De [45], [46] implements ultra-fast imaging for 128 × 96 transducer elements (of which only 1024 are considered per shot) with a powerful die-stacked package. Its main bottleneck is that it requires an external DRAM memory to store beamforming delay coefficients, and several GB/s of memory bandwidth. Other works [25], [27], [47], [48] have shown that a feasible alternative is to try to compute all delay coefficients on-the-fly on-chip. Since this computation involves the evaluation of complex functions like square roots, it is mandatory to identify accurate, fast and low-area approximation circuits. A recursive and iterative method can be used to compute the square roots efficiently [49]. In [50] only every 32nd delay is truly computed and the remaining delays are interpolated. Some works and industrial products have focused on a low power, portable imager implementation [1]–[6], [51]. However, these systems either only support 2D imaging, or a very low channel-count 3D imaging with several major restrictions [52].

In this paper we explore two alternative schemes to tackle the delay generation problem. We revisit our previous works [27], [28] on beamforming architectures with increased emphasis on the delay approximation logic, showing accuracy improvements and presenting a more efficient implementation. We also build upon an alternative approach [29], based on storing a small reference delay table that serves as the basis for runtime delay calculation, improving its versatility and efficiency. We analyze and compare the merits of these architectures in terms of accuracy and throughput, evaluating their suitability for different design points of the ultrasound imaging spectrum.

IV. Delay Calculation at Runtime (TABLEFREE)

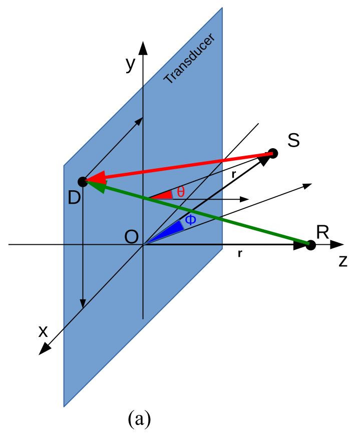
To remove the need for massive precomputed tables, delay values can be computed on-the-fly. We call this approach TABLEFREE. Based on (2), this is the problem to be solved:

$$\begin{aligned} t_p(O,S,D) &= (|\overrightarrow{OS}| + |\overrightarrow{SD}|)/c, \\ |\overrightarrow{OS}| &= \sqrt{(x_O - x_S)^2 + (y_O - y_S)^2 + (z_O - z_S)^2}, \\ |\overrightarrow{SD}| &= \sqrt{(x_S - x_D)^2 + (y_S - y_D)^2 + (z_S - z_D)^2}, \end{aligned} \tag{3}$$

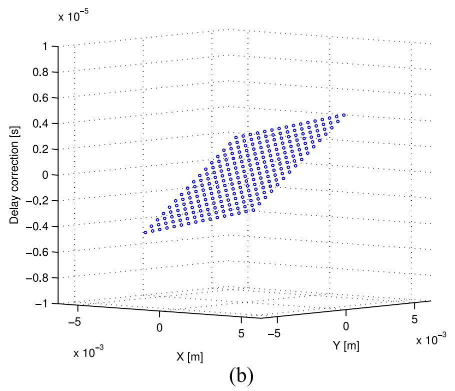
with $O = (x_O, y_O, z_O) \in W$ the insonification emission center, which is fixed over one insonification, and $D = (x_D, y_D, z_D)$ the position of one of the N different receiver elements. z_D is always 0 in our setup since we assume planar receiver arrays. In order to avoid storing all $|V| = 16.38 \times 10^6$ points $S = (x_S, y_S, z_S) \in V$, the ℓ-th point on the j-th scanline is calculated in real-time with

$$S(j,\ell) = \ell \cdot \overrightarrow{v}_j, \quad \overrightarrow{v}_j = \Delta r \cdot \begin{pmatrix} \sin(\theta_j) \\ \cos(\theta_j)\sin(\phi_j) \\ \cos(\theta_j)\cos(\phi_j) \end{pmatrix}, \tag{4}$$

where $\overrightarrow{v}_j$ points into the direction of the j-th scanline and Δr is the spacing between the points along the scanline. All scanlines originate in $(0,0,0)$. As seen before, Equation (3) and thus (4) need to be evalutated 2.5×10^{12} times per second in 3D imaging. This demands massive parallelism, but the hardware cost of a naive implementation is unacceptably high for replicating it on this scale – mostly due to the square roots involved. Much work has been devoted to approximating this delay computation. Usually, for each O, D and scanline j, the *delay profile* $t_p(O, S(j,\ell), D)[\ell]$ was approximated or computed by simpler arithmetic functions like additions and multiplications using few precomputed constants [45], [47]. These *per-channel* approaches scale very badly in terms of memory and access bandwidth for constant storage, since the number of constants depends on the product of transducer elements N and the number of scanlines, which both tend to grow quadratically with system size in 3D. In our *global* approach [27], [28], [30] we minimize memory requirements by computing the delays from very few constants describing the underlying geometry, e.g., D, O, θ_j, ϕ_j, in combination with sharing and reusing as many intermediate computation results as possible. The result of $|\overrightarrow{OS}|$ for example is fixed over one insonification, and can be reused $\forall D$, thus needing to be evaluated only $|V|$ times per frame. The computation of $|\overrightarrow{SD}|$, on the contrary, needs to be computed $|V| \cdot N$ times during the same period. Considering that N is large, ≥ 100 for 2D and even more for 3D systems, the effort to compute $|\overrightarrow{SD}|$ dominates the computation of $|\overrightarrow{OS}|$. Therefore, even though in this paper we concentrate on transmissions created by virtual sources [53] only, it clearly follows that the concepts presented can easily be adapted to other transmission strategies like plane waves, where the circuit to compute $|\overrightarrow{OS}|$ will be replaced. In Section VI-A, we elaborate in detail how to compute the delays efficiently with the TABLEFREE architecture.



(a)



(b)

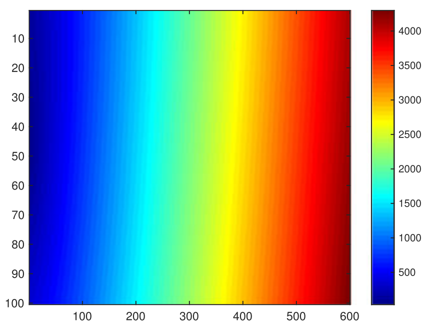

(c)

Fig. 3. (a) Propagation delays must be calculated between each S and each element D of the transducer. The reference delays are the delay values for points R on the Z axis. For a point on another line of sight, the delay can be computed from the reference delay table plus an angle-dependent offset. (b) When considering both θ and ϕ steering, the required compensation is a plane, whose inclination around the origin is a function of θ, ϕ. (c) A section of the compensated delay table for a steering angle, where the x-axis indicates the depth in time samples and the y-axis represents the probe elements in the azimuth direction. The color-map represents the two-way delay values.

V. Mixed Approach: Delay Tables plus Steering (TABLESTEER)

The TABLEFREE approach avoids entirely the usage of delay tables, but requires a large amount of circuitry instead. We propose in this section an intermediate approach, called TABLESTEER, to keep a relatively small delay table in working memory, while computing all delay values from this table with very simple mathematical operations. This table is precalculated and stored for a single line-of-sight.

A. Working Principle

1) Receive Delay: Let us assume, for the moment, that the insonification is performed as a diverging beam emitted from a virtual source $V_s \equiv O$ at the center of the transducer. The TABLESTEER approach is based on storing a *reference delay table*, containing the sum of transmit and receive delays, for the set of points along the scanline that coincides with the Z axis. For points along any other scanline, a correction should be added [see Fig. 3(b)]. This correction could be seen as "steering" the reference delay table. The steering approach is known from literature on 2D ultrasound imaging [54], and we first proposed to exploit it for 3D imaging in [29].

For 2D imaging, the reference delay table is a 2D matrix with dimensions $e_x \times d_p$; for 3D imaging, a 3D matrix with dimensions $e_x \times e_y \times d_p$, i.e. $100 \times 100 \times 1000 = 10 \times 10^6$ elements. It should be noted that not all the table elements are needed, because the probe elements have limited directivity, i.e. they cannot insonify (or receive from) scatterers steeply off-axis. Furthermore, as the sound origin O is at the transducer center as shown in Fig. 3(a), the matrix becomes symmetrical along one axis (in 2D) or even two (in 3D). A possible optimization thus is that only one quarter of the matrix (i.e. 2.5×10^6 elements in 3D) must actually be stored.

The problem to be solved now is to find the correction factor to be able to beamform a point S that is on a steered line of sight. This can be solved by referring to Fig. 3(a) and considering a point R along the reference line of sight at the same distance from the transducer's center O ($r := |\overrightarrow{RO}| = |\overrightarrow{SO}|$), where[1]:

$$O = (0,0,0); R = (0,0,r); D = (x_D, y_D, 0) \tag{5}$$

$$S = (r\sin\theta, r\sin\phi\cos\theta, r\cos\phi\cos\theta) \tag{6}$$

Note that the reference delay table holds the delay value for R. Note also that the point R is subject to the same transmit delay as the point S since they have the same distance from the emission origin O; only a receive delay difference exists. The delay for the point S can thus be expressed as the reference delay table of point R with the addition of a correction factor, as follows:

$$t_p(O,S,D) = t_p(O,R,D) + \frac{|\overrightarrow{SD}| - |\overrightarrow{RD}|}{c} \tag{7}$$

$$\begin{aligned} |\overrightarrow{SD}| &= \sqrt{(x_S - x_D)^2 + (y_S - y_D)^2 + (z_S - 0)^2} \\ &= r\sqrt{1 + \frac{x_D^2 + y_D^2}{r^2} - \frac{2x_D\sin\theta + 2y_D\sin\phi\cos\theta}{r}} \end{aligned} \tag{8}$$

$$\begin{aligned} |\overrightarrow{RD}| &= \sqrt{(0 - x_D)^2 + (0 - y_D)^2 + (z_R - 0)^2} \\ &= r\sqrt{1 + \frac{x_D^2 + y_D^2}{r^2}} \end{aligned} \tag{9}$$

The correction value we seek is thus:

$$\begin{aligned} t_p(O,S,D) - t_p(O,R,D) &= \frac{|\overrightarrow{SD}| - |\overrightarrow{RD}|}{c} \\ &= \frac{r}{c}\sqrt{1 + \frac{x_D^2 + y_D^2}{r^2} - \frac{2x_D\sin\theta + 2y_D\sin\phi\cos\theta}{r}} \\ &\quad - \frac{r}{c}\sqrt{1 + \frac{x_D^2 + y_D^2}{r^2}} \end{aligned} \tag{10}$$

[1]The coordinate expressions depend on how the volume is swept, e.g. azimuth-first or elevation-first. We assume here azimuth-first sweeping.

This cannot be further simplified, but a Taylor expansion can be used:

$$\sqrt{1+x} \approx 1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{1}{16}x^3 + ..., |x| < 1 \tag{11}$$

where the condition on x means that, if increasingly high-order polynomials are used, the expansion converges towards the root function only in the interval $-1 < x < 1$. This is a required condition for convergence of the expansion with an infinite number of terms, but is inconsequential here since we only use a first order polynomial. However, the choice of a first-order approximation *does* incur an inaccuracy, which is discussed in Section VIII-A2. By using the first-order expansion:

$$\begin{aligned} t_p(O,S,D) - t_p(O,R,D) &\approx \frac{r}{c}\left(1 + \frac{x_D^2 + y_D^2}{2r^2}\right. \\ &\quad \left. - \frac{2x_D \sin\theta + 2y_D \sin\phi\cos\theta}{2r} - 1 - \frac{x_D^2 + y_D^2}{2r^2}\right) \\ &= \frac{r}{c}\left(-\frac{x_D \sin\theta + y_D \sin\phi\cos\theta}{r}\right) \\ &= -\frac{x_D \sin\theta}{c} - \frac{y_D \sin\phi\cos\theta}{c} \end{aligned} \tag{12}$$

This correction formula is computationally efficient because it reduces complex square root calculations to just a lookup in a small table (reference delay) plus two additions. Since the possible values of ϕ, θ, x_D, y_D are discrete and few, note that the correction terms can be fully precalculated and also stored in small tables.

2) Transmit Delay: Note that the discussion above provides for RX delay steering, but is only correct if the transmission origin $O \equiv V_s$ is fixed at the center of the transducer. In our previous work [29] we indeed relied on this assumption. However, many ultrasound imaging techniques exploit different transmit strategies, e.g. steered plane waves, and sometimes rely on repositioning the origin freely at every insonification, e.g. in ultrafast or synthetic aperture imaging [44].

To lift this limitation, let us now assume that the virtual source V_s is anywhere behind (or on) the transducer. It can quickly be seen that the previous approach cannot be used directly, because of the need of a reference point R that is simultaneously (i) equally distant from O than S (to enable receive delay steering), and (ii) equally distant from V_s than S (to experience the same transmit delay); this is only possible when $O \equiv V_s$.

This problem can be tackled in different ways. On one hand, it is possible to devise a steering method to be applied to the transmission, too. Closed-form correction coefficients, approximate or in some cases exact, can be derived for relevant emission patterns, such as plane waves with varying orientation and diverging beams with different emission origins. In this case, the total propagation delay can be calculated as the sum of a value in a reference table, plus a transmission correction coefficient, plus a reception correction coefficient. Unfortunately, this approach requires a different correction table for every possible emission strategy, and there are emissions which are complex to compensate with a compact set of coefficients. Moreover, if approximations are involved, the accuracy of the delay calculation is degraded further.

On the other hand, the number of transmit delays that must be calculated is much smaller than that of receive delays, by a factor of $N = e_x \times e_y = 10000$ in 3D. Therefore, we believe that even if the transmit delay is computed exactly, the overhead will be small. We therefore choose to adopt the same approach as in TABLEFREE, i.e. the explicit calculation of the $|\overrightarrow{OS}|$ square root, for the transmit delays to guarantee the maximum flexibility of use of the system.

3) Accuracy Bound: Using a first-order Taylor approximation for the delay calculation introduces a potentially serious degree of inaccuracy. To control this issue, it is natural to attempt to formally bound the degree of inaccuracy. A common way to do so is by using the Lagrange bound on the Taylor remainder. Unfortunately, although a formulation of this bound can be derived, the bound diverges to infinity[2] when $r \to 0$. Therefore, a different bounding approach is required.

Note that the original function $f(x) = \sqrt{(1+x)}$ and its first-order expansion $f_1(x) = 1 + \frac{1}{2}x$ are always positive. This can be seen because the expression $(1+x)$ is the square of a distance, see (9) and (11). It can also be immediately seen that the largest approximation error occurs for $x \to \infty$, with both functions diverging to infinity, $f_1(x)$ much more quickly than $f(x)$. Therefore, the approximation can be bounded to

$$|E_{f_1}(x)| = f_1(x) - f(x) \xrightarrow{x\to\infty} f_1(x) \tag{13}$$

As mentioned previously, we have approximated two functions, $g(x)$ and $h(x)$. The error bounds for each of those approximations are:

$$|E_{h_1}(x)| \le 1 + \frac{1}{2}x_h, |E_{g_1}(x)| \le 1 + \frac{1}{2}x_g \tag{14}$$

The total error is the difference of the errors on $h(x)$ and $g(x)$, because these two functions have the same sign and the error must be calculated in the same location x. Thus:

$$\begin{aligned} |E(x)| &\le \left|(1 + \frac{1}{2}x_h) - (1 + \frac{1}{2}x_g)\right| = \frac{1}{2}|(x_h - x_g)| \\ &= \frac{1}{2}\left|\left(\frac{x_D^2 + y_D^2}{r^2} - \frac{2x_D \sin\theta + 2y_D \sin\phi\cos\theta}{r} - \frac{x_D^2 + y_D^2}{r^2}\right)\right| \\ &= \left|\frac{x_D \sin\theta + y_D \sin\phi\cos\theta}{r}\right| \end{aligned} \tag{15}$$

Looking back at (10), we can express the error in time units by multiplying by r over c:

$$|E(x)| \le \left|\frac{x_D \sin\theta + y_D \sin\phi\cos\theta}{c}\right| \tag{16}$$

Which does yield a finite bound on the Taylor expansion inaccuracy, as will be quantified in Section VIII-A.

[2]Detailed calculations are omitted for space reasons but are available on request.

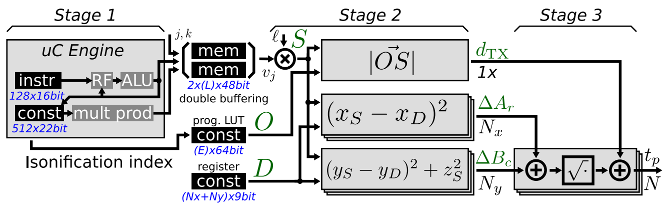


Fig. 4. Ekho delay computation: Stage 1: The programmable uC Engine computes the scanline direction vectors v_j required in one insonification and selects the insonification origin O. Stage 2: All $S \in V_z$ are computed sequentially, one per cycle, using 3 parallel DSP48 multipliers. The TX delay $|\vec{OS}|$ requires another 3 DSP48 multipliers for the squares and single square-root unit. The computation of ΔA_r and ΔB_c requires $N_x + N_y + 1$ DSP48 multipliers. Stage 3: Two adders and one square root unit are required for all N channels to finalize the computation.

VI. FPGA Architecture

A. TABLEFREE Architecture

In [27]–[29], we have presented the basic architecture to compute the two-way propagation delay $t_p(O, S, D) = (|\vec{OS}| + |\vec{SD}|)/c$ efficiently and accurately. The TABLEFREE delay computation architecture presented in this paper is based on the Ekho ASIC architecture [30], but has been heavily optimized for FPGA to support clock frequencies higher than 390 MHz. In this paper, we only give a brief overview of the Ekho architecture and highlight the relevant FPGA optimizations. Ekho follows the *global* approach introduced in Section IV: in order to completely avoid off-chip memory, it computes the delays from very few constants (less than 40 kbit), which describe the underlying geometric setup, and thus, in order to tackle the consequent computation effort, it reuses as many intermediate results as possible in combination with a smart square-root computation circuit.

In Ekho, the delays are computed in 3 stages as illustrated in Fig. 4. In Stage 1, a small programmable unit computes the direction vectors $\vec{v}_j$ of all scanlines evaluated in one insonification and stores them in a double buffer. The imaging strategy and computation order can be easily adapted by changing the program. In Stage 2, all points $S \in V_z$ are computed from these direction vectors by a scalar multiplication at a rate of one point per cycle. The TX delay $|\vec{OS}|$, which can be reused for all N channels, is also computed. Furthermore, we compute one $\Delta A_r = (\Delta x_{SD})^2$ per transducer matrix row and one $\Delta B_c = (\Delta y_{SD})^2 + (\Delta z_{SD})^2$ per column. In this stage, $3 + 3 + N_x + N_y + 1$ multiplications and one square root computation are performed. The delay computation is finalized in Stage 3, where each channel computes $t_p = \sqrt{\Delta A_r + \Delta B_c} + d_{TX}$, which only requires two additions and one square-root operation. To enable fast operation on FPGA, the Ekho design has been heavily pipelined, both the main datapaths and the programmable unit; all multiplications have been mapped into DSP slices.

In our previous work [27], [28], the square roots were computed with a piecewise linear approximation. However, this approximation required the use of a 28-bit × 28-bit multiplier, which is fine for an ASIC design, but cannot be mapped well into an FPGA providing DSP slices supporting only a limited bit-width, like 25-bit × 18-bit in the case of the Virtex-7 series DSP48E1 slice [55]. In [30], we proposed a new method based on [49], which computes the square-root iteratively and exactly. In each iteration step, an additional bit is computed. This method does not need any multipliers and can be completely unrolled and arbitrarily pipelined. It is therefore very well suited for a high-speed FPGA implementation.

B. TABLESTEER Architecture

The delay values are used as an index into an echo buffer containing slightly more than 8000 samples (from interpolation of the 1000 input samples), corresponding to a 32 MHz sampling of the two-way sound propagation time (2 × 500λ). This requires a bit-depth of 13 bits. To improve the accuracy of the sum operations, a fixed-point representation is useful. Let us assume for the moment, without loss of generality, a 18-bit design, which fits well one of Xilinx's selectable BRAM bank widths. The reference delays are always positive, thus they can be stored in 13.5 unsigned format and they can be sign-extended at the moment of applying the correction. The correction coefficients, which may be negative, must be stored with a signed 13.4 representation.

Considering the general 3D imaging case, which is the most challenging, a 10×10^6-element reference receive delay table is needed, for 180 Mb of storage. For each of the 128 × 128 steered lines of sight, the correction coefficients of (14) must be summed to the reference delays stored in the table. The former can be entirely precomputed, for a total of $100 \times 64 \times 128 + 100 \times 128 = 832 \times 10^3$ values (note that $cos\theta$ is symmetrical around 0) and thus 14.3 Mb. This amount of storage is only feasible for the latest-generation high-end FPGAs; for example, the largest Xilinx Virtex 7 carry up to 68 Mb of Block RAMs [56], while the brand new UltraScale+ architecture [57] includes up to 432 Mb of UltraRAM. To conserve area, the delay table memory should be streamed in from an external DRAM. Since delay table contents are constant during execution, this is akin to a read-only caching.

We propose a refined version of the architecture proposed in [29], shown in Fig. 5. It is a *memory-centric* architecture - i.e., the heart of this block is an FPGA BRAM bank, holding a slice of reference delays. Every cycle, one reference delay is read from this BRAM, and summed to a parametric count k_x of x_D and then k_y of y_D steering coefficients. This requires $k_x + k_x \times k_y$ adders per block, of which $k_x \times k_y$ must also perform rounding to integer, generating $k_x \times k_y$ steered delays. The main improvements of this paper over [29] are (i) additional configurability, (ii) extra pipelining, (iii) optimized RTL, (iv) exact and flexible TX delay calculation. Further, we add the ability to use the k_y adders with different y_D steering coefficients over multiple cycles. The last feature allows for major area reductions of the block (for example, using only half of the k_y adders) at the price of requiring more cycles to compute the same amount of steered delays (for example, two cycles instead of one). This tradeoff will be investigated. We also now propose to keep correction coefficients off-chip and to load a small subset of them

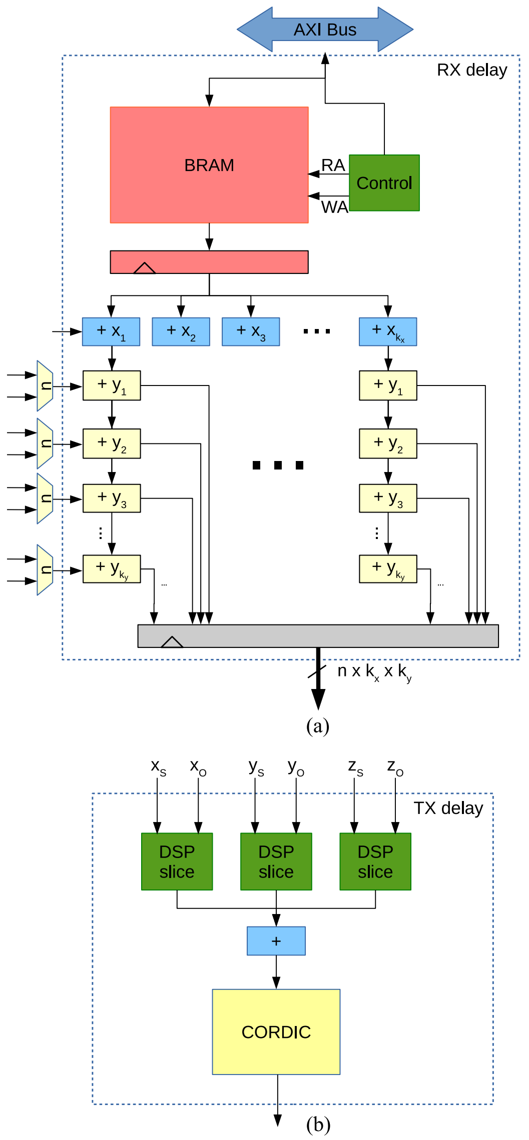


Fig. 5. Proposed architecture of the delay computation blocks. The RX delay block (a) is centered on a BRAM bank. The receive delay is computed by applying steering coefficients to a precomputed reference table. The TX delay is calculated exactly (b) as the square root of a sum of squares.

upon each insonification. Overall, these improvements enable the effective deployment of the architecture in very different scenarios, as will be seen in Section VIII.

The newly added TX delay computation is not critical in terms of resources for TABLESTEER, since it needs to be performed at a much lower throughput than RX delay computation. To minimize the design effort, in this paper we propose to use the Xilinx pre-developed CORDIC core for the calculation of the square roots, and map the coordinate squaring onto the FPGA's DSP48 slices. We chose to use the "optimum" pipelining configuration of the core; this saves area and latency in return for a lower operating frequency, that we still found to be in excess of 200 MHz. The architecture [see Fig. 5(b)] requires 12 to 14 cycles to compute the TX delay with the precision required to match respectively a 14 to 18-bit representation of the RX delays. To minimize additional computations, the delay is directly computed in samples, i.e. the input coordinates of S and O are pre-multiplied by the sampling frequency and divided by the speed of sound. Finally, the RX and TX delays are summed. Note that a single TX delay, valid for a point S, is summed to many $\overrightarrow{SD}$ RX delays.

The TABLESTEER architecture must be configured in such a way that several feasibility constraints are simultaneously fulfilled: sufficient throughput, acceptable FPGA utilization, feasible memory bandwidth. Additionally, the parameters must match a chosen insonification strategy. For instance, consider a strategy that images the volume in 64 zones, i.e. 64 insonifications per frame, each comprising 256 scanlines. This means that at most 256 unique correction coefficients are to be applied in parallel, corresponding to each scanline's intrinsic (θ_s, ϕ_s) steering. Calculating any other (θ, ϕ) correction is wasteful. If then the imaging pyramid is shrunk to 64 scanlines/insonification, even fewer correction values are needed, leading to apparently more compact logic. But to reconstruct a whole frame, 256 insonifications are now needed instead of 64, which requires to stream the reference delay table four times more often, i.e. multiplying by four times the memory bandwidth, which can become critical. Therefore, TABLESTEER is more suitable for fewer and broader insonification patterns. A full discussion of these trade-offs is omitted for brevity; in Section VIII-C we will report the most effective configurations we could find for a set of scenarios.

VII. Reference Scenarios for 2D and 3D Imaging

Although 2D and 3D ultrasound imaging differ greatly in terms of medical applications, device complexity and target costs, from a mathematical viewpoint, their processing kernels share the same problem description (1, 2). Therefore, it makes sense to consider the problem of delay computation for both within the same processing framework. We have therefore chosen to study three design points, which do not necessarily represent commonly used commercial platforms, but have instead been picked to represent different extremes of the design spectrum. These are:

1) A very low-cost, low-power 2D system, suitable for portable, battery-operated deployment. This design generates baseline-quality images, with the strict minimum of processing resources.
2) An ultrahigh-frame-rate 2D system, representative of a high-end 2D system.
3) An advanced 3D system, capable of full-resolution volume imaging when using a high-element-count matrix probe. This futuristic system is not available today, and stands for the ultimate image quality achievable with next-generation electronics.

The basic specifications of these three design points are summarized in Table II.

Since TABLESTEER can be bandwidth-limited with imaging patterns that rely on many insonifications of narrow zones, in the experiments that follow, we will divide 3D volumes in 8×8 zones for TABLESTEER, while TABLEFREE does not have this limitation and therefore we will consider 16×16

TABLE II
SYSTEM OVERVIEW

Setup	N	$\|V\|$	Z	T	Ψ	$f_r\,(target)$	$delay\ values/s$
LP2D	100	128×1000	1	1	$100 \times 128 \times 1000 = 12.8 \times 10^6$	15 Hz	192×10^6
UF2D	100	128×1000	1	16	$100 \times 128 \times 1000 \times 16 = 204.8 \times 10^6$	250 Hz	51.2×10^9
3D	10000	$128 \times 128 \times 1000$	8×8 or 16×16	1	$10000 \times 128 \times 128 \times 1000 = 164 \times 10^9$	15 Hz	2.5×10^{12}

zones. With different types of transmit beams, e.g. focused, TABLEFREE's increased flexibility may be leveraged to improve resolution slightly.

VIII. Experimental Results

In this section, we present implementation results for the two proposed delay computation architectures.

For TABLEFREE, we present different configurations in terms of the supported number of channels and extrapolate the results for various setups.

For TABLESTEER, we parameterize the design to use from 14-bit to 18-bit delay representations (-14b/-18b) (Section VI-B), assessing the accuracy vs. area tradeoff. The 14-bit configuration is tested with $k_x = 8, k_y = 8$ as well as $k_x = 16, k_y = 16$; since the latter variant generates four times more delay values per block, two comparable versions of the architecture are shown, with 200 8×8 RX blocks and 50 16×16 RX blocks. The 18-bit configuration is studied only in $k_x = 8, k_y = 16$ instances, but we further parameterize the k_y correction coefficients, by multiplexing $k_y/2$ and $k_y/4$ adders and using respectively 2 and 4 cycles to complete the computation. We estimate the necessary memory bandwidth based on the volume of data to be fetched, but including no packing and protocol overheads. For this design space exploration, we assume 64 insonifications of 256 scanlines each.

First, we will focus on the most challenging usage scenario, for 3D imaging. We will comment on how accurate the two methods are, which is key to image-reconstruction quality, by showing the Point Spread Function (PSF) contours and projection at different locations in the volume. We will also show an example image. We will then present and compare implementation results on a high-end Xilinx Virtex 7 device, XC7VX1140T, speed grade -2, to assess the utilization of resources, and thus ultimately the feasibility of the implementations. We will also evaluate the maximum achievable frequency, and thus the throughput, of the designs, to see how high frame rates can be achieved.

A. Delay Inaccuracy Quantification

1) TABLEFREE Inaccuracy: The Ekho delay computation is mathematically exact and does not use approximations. Thus, the inaccuracy of TABLEFREE is fully determined by the limited-precision computation losses. Compared to the ASIC Ekho implementation, some internal bit widths were slightly reduced in Stage 1 and 2 of the delay computation to fit the multiplications into the DSP slices and the fixed-point rounding policy was altered for error reduction. For the 3D setup the mean and maximum absolute errors compared to a double precision floating point computation are 0.296 and 1.271 samples, respectively. If we consider that the final delay value is rounded in order to select an integer sample to contribute, we find that the sample index is off at most by 1 sample and this happens in only 18% of the computations. Note that the accuracy can be arbitrarily improved by increasing the internal bit widths, or, conversely, reduced to save resources.

2) TABLESTEER Inaccuracy: The TABLESTEER approach has two causes of inaccuracy. The main one is the inaccuracy due to the algorithmic approximation in using the first-order Taylor polynomial to "steer" the reference delay table. In Section (V-A3), we demonstrated the theoretical bound of the Taylor approximation inaccuracy and we represented it by (16). To get the maximum error of the approximation theoretically, D should be at one of the four corners of the probe (i.e. $\pm x_{Dmax}$ and $\pm y_{Dmax}$) and at $\pm\theta_{\max}$ and $\pm\phi_{\max}$, as follows:

$$|E| = \frac{0.0103274}{1540} = 6.71\ \mu\text{s} \tag{17}$$

which at the target sampling frequency of 32 MHz, equals 215 signal samples. This degree of inaccuracy is unacceptable. A first mitigation factor however comes in the form of apodization; since the worst inaccuracies occur at broad angles beyond the elements' directivity, they are anyway discarded by the imaging system. This is because the apodization function is a window of weighting coefficients for the probe elements, where the elements whose directivity function makes them insensitive to given echoes are zero-weighted. Furthermore, the far-field approximation's worst errors occur only at extremely short distances from the origin and at the extreme angles of the field of view; both regions are usually the least critical for image quality [refer to Fig. 6(a)]. With a comprehensive numerical exploration in the volume of interest, considering apodization, we observed a decrease in both average and maximum absolute error. The average absolute error over the whole volume due to the algorithm itself was $44.641ns$, i.e. ≈ 1.4285 signal samples, while the maximum error equaled 3.1 μs, i.e. 99 signal samples.

The contribution of any element with a sampling error beyond 2 samples[3] is essentially noise, and thus, if the sampling error is known upfront, the element samples are better discarded than summed in. Based on this insight, as an improvement over [29], we propose to adopt a stricter apodization than usually necessary. In other words, we propose to prune the element contributions whose sampling error lies beyond 2 samples due to geometric inaccuracy. This can be seen in Fig. 6(b), which shows

[3]Based on a phase offset threshold of 90° between constructive and destructive interference, and considering that the sampling frequency is $8f_c$.

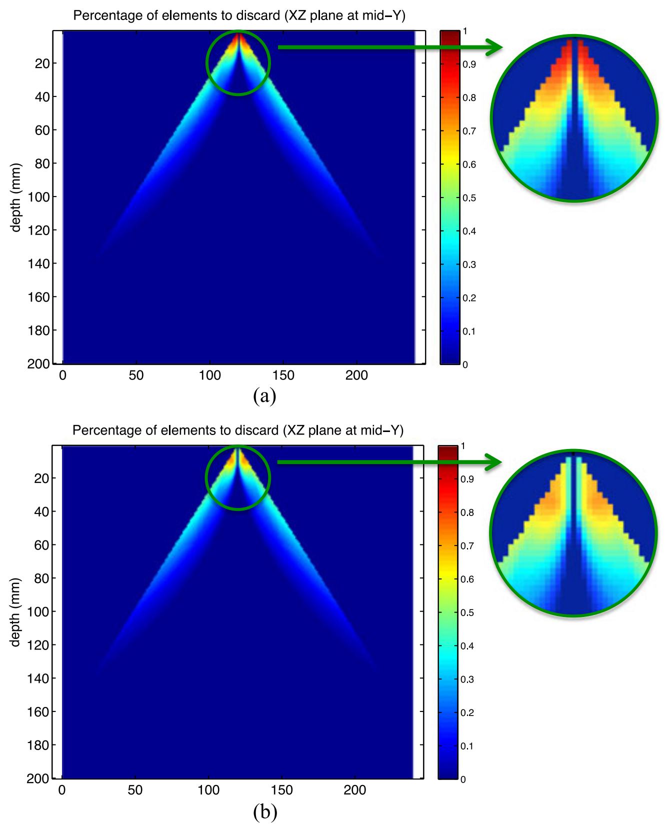


Fig. 6. Graphical depiction of the geometric approximation in TABLESTEER. (a) Geometric inaccuracy of the approach without applying apodization. The inaccuracy is significant only very close to the probe and at broad angles. (b) Geometric inaccuracy after applying apodization. The color map represents the percentage of elements that incur delay inaccuracy of more than the constructive interference threshold of 2 samples.

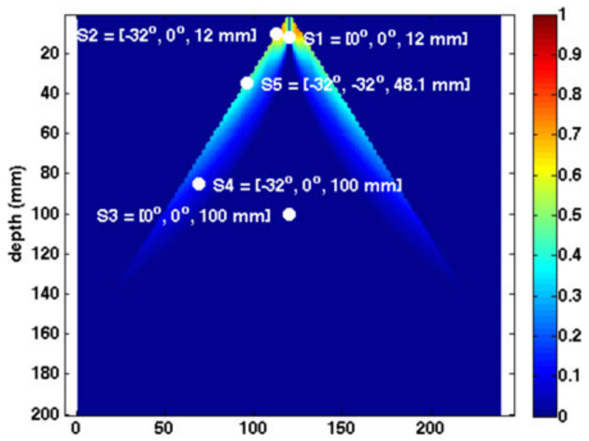


Fig. 7. Locations of scatterers being reconstructed to test PSF contours and projections (Fig. 8). The locations are overlaid on the inaccuracy map of Fig. 6(b).

the percentage of element signals that are further discarded, in addition to normal apodization, in the calculation of each focal point. We measured that, in the worst case – at 26λ depth (i.e. 1 cm) and broad angles –, the apodization must discard a further 78.8% of the matrix elements to avoid adding image noise. On average across the whole volume, the inaccuracy deriving from the geometric approximation can be essentially removed (sampling error of 2 samples or fewer) by apodizing away 18.0% of the element echoes on top of the normal directivity-related apodization patterns. As discussed before, the strictest apodization applies to focal points either very close to the transducer or at broad angles; we observed that the bulk of the focal points in the image (64.1%) require filtering away less than 20% of the element echoes.

We have assessed the approximation of the TABLESTEER delay calculation approach compared to the TABLEFREE calculation (i.e. the exact calculation) by reconstructing point scatterers at different locations in the volume. We have plotted both the Point Spread Function (PSF) contours, and the projections of those scatterers to evaluate the accuracy, the resolution, and the width of both the main and sidelobes of the proposed TABLESTEER approach. Five scatterer locations (see Fig. 7) have been chosen; a scatterer S_1 close to the probe surface and at the center of the imaging sector [see Fig. 8(a) and (f)], or very off-axis like S_2 [see Fig. 8(b) and (g)], a scatterer S_3 far from the probe surface and at the center [see Fig. 8(c) and (h)], or at a broad azimuth angle like S_4 [see Fig. 8(d) and (i)], and finally a scatterer S_5 at an intermediate depth and at broad azimuth and elevation angles [see Fig. 8(e) and (j)]. For S_1, TABLESTEER exhibits even better resolution than the reference imager which uses square roots [see Fig. 8(f) and (a)]. This counter-intuitive outcome can be explained by observing the unpredictability of delay inaccuracy artifacts. Along the central line-of-sight, where no steering occurs and the delay values are accurate, the reconstructed image is identical to the reference [central slice of Fig. 8(f) and (a)]. Away from this line (either side of 8(f) and 8(a)), the inaccuracy affects the reconstruction, yielding unpredictably slightly brighter or slightly darker voxels than the reference, which either slightly degrades or slightly improves the contrast and delineation of the feature in the central line. In this specific case, the latter phenomenon is occurring. Nonetheless, for most voxels in the volume, we tend to logically expect a degradation instead. For scatterers at the same depth and at broad angle (like S_2 in Fig. 7), which is the most critical delay calculation inaccuracy region, TABLESTEER incurs a high calculation inaccuracy. Fig. 8(g) shows that the PSF projection has a wide mainlobe which more slowly degrades to the noise floor compared to the perfect calculation. On the other hand, at far depths, either at the center azimuth angle (and/or elevation angle) or at the image edges, the proposed TABLESTEER approach yields almost a perfect match with the exact delay calculation in both PSF contours and projection [see Fig. 8(c), (h), (d), and (i)]. For intermediate depth scatterers at broad angles, the ideal delay calculation out-performed slightly the TABLESTEER calculation. This can be seen through Fig. 8(e) and (j), where the contours of the TABLESTEER are a little bit wider, and the PSF projection degrades more slowly to the noise ground level, although the mainlobe has the same width as the one of the exact calculation.

The second cause of inaccuracy, similarly to the case of TABLEFREE, is the fixed-point addition of the reference delay value with the two correction factors, and subsequent rounding to an integer index to access the data sample array. The inaccuracy due to using a fixed-point representation

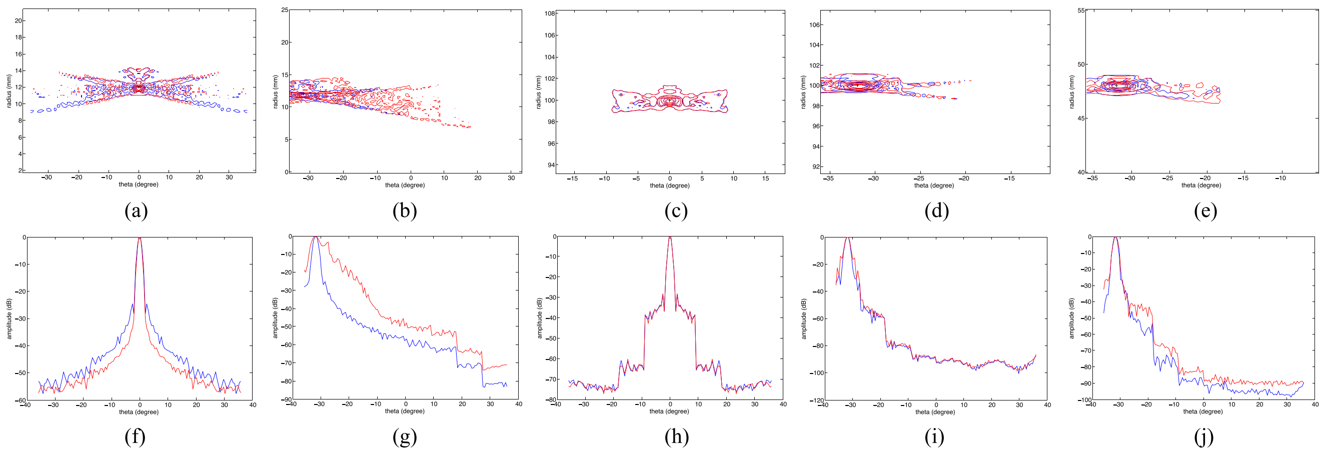


Fig. 8. Evaluation for the TABLESTEER approach based on simulating Point Spread Function (PSF) contours and their projections for different scatterer location, where (a), (f) for a scatterer S_1 at theta = 0°, phi = 0°, r = 12 mm, (b), and (g) for a scatterer S_2 at theta = −32°, phi = 0°, r = 12 mm, (c) and (h) for a scatterer S_3 at theta = 0°, phi = 0°, r = 100 mm, (d) and (i) for a scatterer S_4 at theta = −32°, phi = 0°, r = 100 mm, (e) and (j) for a scatterer S_5 at theta = −32°, phi = −32°, r = 48.1 mm. The blue curves represent the exact delay calculation (i.e. TABLEFREE), while the red ones represent the TABLESTEER approximate delay calculation.

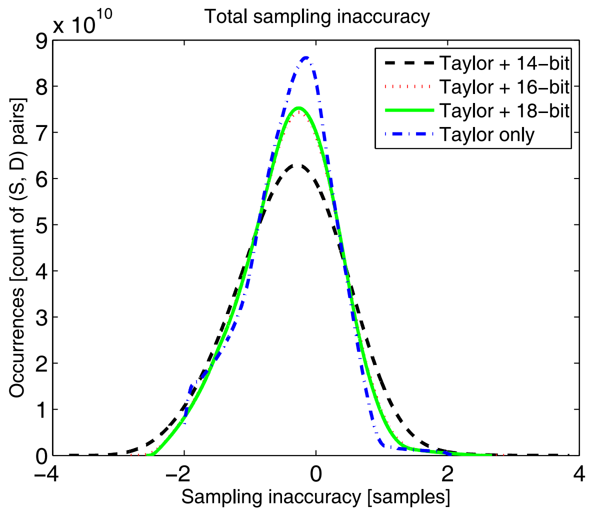


Fig. 9. Compounded probability distribution function of the various sampling errors in the TABLESTEER architecture, for varying-precision fixed-point representations.

has a uniform distribution between $[-0.5, +0.5]$ Least Significant Bits (LSBs). The final rounding of the summed value to an integer index of 13 bits incurs a further error of up to ± 0.5 samples.

Putting everything together, the total error can be seen as the sum of the Taylor error and the fixed-point representation errors, all of which can be considered as independent random variables. The probability distribution of the total error is thus the convolution of the distributions of each error. The overall outcome is plotted in Fig. 9 for different-precision fixed-point representations. The total error is dominated by the Taylor expansion error (maximum absolute error of 2.0000 samples, average 0.5824). However, using just 14 bits for the coefficient representation, the maximum absolute error increases to 3.8480 samples, with an average of 0.7246. A marked improvement can be had using a 16-bit representation (maximum absolute error = 2.9105, average = 0.6377) while an 18-bit representation offers only marginal further improvements (maximum absolute error = 2.6749, average = 0.6320).

A sample 2D image, reconstructed in Matlab with the TABLESTEER method, is shown in Fig. 10(a). The source image data is a common example from the Field II [58] distribution. The image shown is a 2D reconstruction comprising 8 sub-images (zones), each derived from a different diverging-beam insonification. It can be seen that the subject is well-delineated. A comparison with the same image reconstructed with exact delay calculation i.e. TABLEFREE [see Fig. 10(b)] shows no degradation of the image quality with very negligible differences in speckle patterns close to the probe surface.

B. FPGA Implementation Results

In Table III we report synthesis results and linear estimates for various setups for the TABLEFREE architecture. Thanks to architectural and mapping optimizations, we can fit in the given Virtex 7 FPGA 35 × 35 channels (delay computation only) at a clock rate of 392MHz. In our previous work [29] we had 1764 channels at 167MHz. This is an improvement of 63% considering the channel-throughput product.

It can be seen that TABLEFREE has some major advantages: it does not occupy any BRAM space, all the small memories are implemented by memory LUTs, and it does not require any off-FPGA bandwidth because all necessary coefficients are on-chip. This makes it compatible with integration in the same chip of other portions of the beamformer architecture, or of other post-beamforming functionality.

The TABLEFREE architecture is designed for high scalability thanks to its reliance on a set of parallel identical beamformer units, with very limited interaction among each other - essentially just the summing tree downstream. Therefore, to meet the requirement for more channels, it is possible to envision a design with multiple FPGAs in parallel; nine such FPGAs would accommodate the delay computation for a 100 × 100 channel

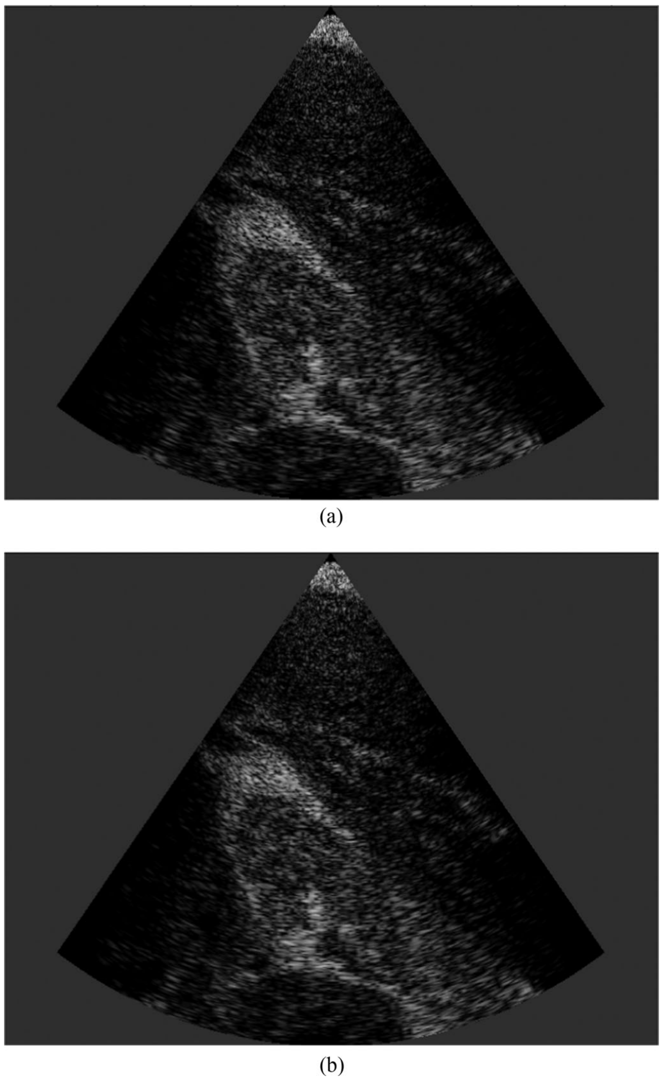


Fig. 10. 2D imaging of a kidney phantom available online on the Field II website [58]. The reconstruction is composed of 8 zones, each insonified by different diverging-beam insonifications. The imaging depth is 10 cm and the azimuth sector is 73° wide. (a) TABLESTEER method; (b) TABLEFREE method (i.e. exact calculation). On close inspection, the speckle pattern close to the transducer and at the edges of the imaging cone displays only minor differences.

imager, requiring a small number of pins and bandwidth for overall aggregation.

Synthesis results for TABLESTEER are in Table IV. TABLESTEER was optimized from the start for an FPGA implementation, and therefore makes a more balanced and efficient use of the resources of the Virtex chip. As a result, it is possible to fit the delay generation logic necessary to achieve a frame rate of 15–30 fps for 3D ultrasound in a single device, while supporting all 100 $\times$ 100 channels. The key price to pay, compared to TABLEFREE, is a loss of accuracy in the beamforming process, but mostly limited to the edges of the imaging volume.

For all configurations, we clocked the TX delay generation logic, which can run slightly in excess of 200 MHz, at half the frequency of the RX delay generation logic; this simplifies the handling of clock domain crossing when summing up TX and RX delays. Considering the latency and frequency of the TX delay logic, we found that for fixed point representations using 14, 16 and 18 bits, it was necessary to instantiate 32, 34 and 36 such blocks, respectively.

The bandwidth requirements of the architecture are challenging, although feasible; for example, GDDR5 memories with a throughput of 32 GB/s or more per single chip are now entering the market [59]. Another option is to store the reference and correction tables entirely on-chip; as seen in Section VI-B, this can be done with about 194 Mb of SRAM without exploiting any particular optimization, or about 60 Mb when considering the 4-way symmetry of the delay table. This becomes feasible with the latest Xilinx Ultrascale chips [57], that embed up to 432 Mb of on-package memory. Nonetheless, we plan to improve this property of the architecture in our future work, by exploring (i) imaging strategies with fewer insonifications and more scanlines per insonification, and (ii) interpolation strategies that leverage the slowly-changing delay behaviour due to depth-of-field considerations.

It can be seen that the idea of multiplexing a smaller number of k_y adders over multiple cycles proves counterproductive. The area of each delay calculation block is indeed sharply reduced, but since the throughput is linearly reduced, to keep constant performance, it is in fact necessary to deploy more delay blocks and global resources. This can be seen from the TABLESTEER-18b results. Similarly, it appears to be more efficient to deploy fewer delay blocks that calculate many coefficients in parallel, rather than the opposite (see the TABLESTEER-14b results). In particular, this happens because the clock frequency is not severely impacted by the extra parallelism.

It can also be seen that choosing a lower fixed-point precision (14b vs. 16b vs. 18b) allows for minor area savings, at a minor accuracy cost.

Based on these findings, we select the configuration 3D-16b-50x16x16-1X as the best instantiation of TABLESTEER for the *3D* scenario, as it achieves a good level of accuracy at a low resource utilization. The reference delay table stored in external memory is 22 MB large.

C. Adaptation of the Delay Calculations Architectures to 2D and Low-Power 2D Imaging

We now assess how suitable TABLEFREE and TABLE STEER are to other imaging setups, i.e. *UF2D* and *LP2D*.

Since the TABLEFREE inaccuracy is dominated by fixed-point losses only, no specific reevaluation is required when switching between 3D and 2D geometries. Nonetheless, we expect a slightly lower loss, since the uC-unit performs fewer computations in the 2D setup, leading to smaller cumulative errors. For the two 2D setups, the mean and maximum absolute errors in samples compared a double precision float computation are 0.288, 1.174 for *UF2D* and 0.287, 1.097 for *LP2D*. If we consider that the final delay value is rounded in order to select an integer sample, we find that the sample index is off at most by 1 sample and this happens in only 16% of the computations.

The estimated FPGA usage for the various setups is reported in the bottom rows of Table III. For the *UF2D* setup, the TABLE-FREE beamformer is configured for 100 $\times$ 1 channels, which

TABLE III
VIRTEX 7 XC7VX1140T-2 SYNTHESIS RESULTS AND ESTIMATIONS FOR THE TABLEFREE ARCHITECTURE

TABLEFREE	Supported Channels $N_x \times N_y$	Logic LUTs	Memory LUTs	Regs	BRAM	DSP	Clock	Throughput	Frame Rate
3D-10	10×10	8.6%	1.2%	5.4%	0%	0.8%	392 MHz	39.2 GDs/s	23.9 fps
3D-20	20×20	32.7%	3.1%	20.8%	0%	1.4%	392 MHz	156.8 GDs/s	23.9 fps
3D-30	30×30	72.7%	6.1%	46.3%	0%	2.0%	392 MHz	352.8 GDs/s	23.9 fps
3D-35 (est)	35×35	99.0%	8.3%	63.0%	0%	2.7%	392 MHz	480.2 GDs/s	23.9 fps
3D-100 (est)	100×100	807.8%	67.8%	514.4%	0%	22.2%	392 MHz	3.92 TDs/s	23.9 fps
UF2D	100×1	9.3%	2.0%	6.0%	0%	3.2%	392 MHz	39.2 GDs/s	191.4 fps
UF2D-$\times$ 2 (est)	100×1	18.6%	4.0%	12.0%	0%	6.4%	392 MHz	2×39.2 GDs/s	250* fps
LP2D-/100 (est)	100×1	0.5%	0.5%	0.2%	0%	0.3%	392 MHz	392 MDs/s	30.62 fps

Limited by the maximum insonification rate; the digital logic supports in theory 382.8 fps.

TABLE IV
VIRTEX 7 XC7VX1140T-2 SYNTHESIS RESULTS AND ESTIMATIONS FOR THE TABLESTEER ARCHITECTURE

TABLESTEER	Supported Channels $N_x \times N_y$	Logic LUTs	Memory LUTs	Regs	BRAM	DSP	Clock	Offchip DRAM BW **(est.)**	Throughput	Frame Rate
3D-14b-200 $\times$ 8 $\times$ 8-1X	100×100	63%	0.1%	13%	5.7%	3.2%	372 MHz	32.5 GB/s	4.8 TD/s	29.0 fps
3D-14b-50 $\times$ 16 $\times$ 16-1X	100×100	27%	0.1%	11%	1.7%	3.2%	328 MHz	28.7 GB/s	4.2 TD/s	25.6 fps
3D-16b-100 $\times$ 8 $\times$ 16-1X	100×100	66%	0.2%	13%	3.1%	3.2%	337 MHz	33.7 GB/s	4.3 TD/s	26.3 fps
3D-18b-100 $\times$ 8 $\times$ 16-1X	100×100	70%	0.4%	14%	3.0%	3.2%	343 MHz	38.7 GB/s	4.4 TD/s	26.8 fps
3D-18b-150 $\times$ 8 $\times$ 16-2X	100×100	86%	0.4%	19%	4.4%	3.2%	298 MHz	24.6 GB/s	2.9 TD/s	17.3 fps
3D-18b-300 $\times$ 8 $\times$ 16-4X (est)	100×100	191%	0.4%	35%	8.4%	3.2%	309 MHz	25.5 GB/s	3.0 TD/s	17.7 fps
3D-16b-50 $\times$ 16 $\times$ 16-1X	100×100	32%	0.2%	12%	1.7%	3.2%	299 MHz	30.0 GB/s	3.8 TD/s	23.4 fps
UF2D-16b-2 $\times$ 128 $\times$ 1-1X (est)	100×1	2%	~0%	1%	~0%	2%	300 MHz	1.2 GB/s	77 GD/s	250* fps
LP2D-14b-1 $\times$ 128 $\times$ 1-1X (est)	100×1	~0%	~0%	~0%	~0%	~0%	2 MHz**	3 MB/s	256 MD/s	20 fps

Limited by the maximum insonification rate; the digital logic supports in theory 375 fps.
**Underclocked to conserve power; the design could run 150 times faster.*

automatically removes unneeded logic, required only for 3D. On the given FPGA, our beamformer only provides a framerate of 191.4 fps for this setup, which is below the targeted 250 fps dictated by the physical insonification repetition bounds. By replicating the calculation units (note that the uC-Engine does not need to be replicated), we reach a processing capability of 382.8 fps while using less than 20% of the FPGA resources. Since I/O bandwidth and BRAMs are still completely unused, there is space to place the remaining parts of the beamformer on the same FPGA.

For the *LP2D* setup, the TABLEFREE beamformer is configured for 1 channel only. By time-sharing the computation unit of one single channel, we can compute the delays for all 100 channels, while still exceeding the target frame rate by a factor of 2. Thus, to save power, the clock frequency could be halved.

TABLESTEER is also suitable for 2D ultrasound imaging, as can be seen by imagining e.g. $\phi = 0$ in (14). In this case, the y_D dimension disappears and the required processing becomes trivial. Referring to Fig. 5, there is no need for the k_y adders. For the *LP2D* case, the FPGA resource occupation becomes negligible, and so are the external memory footprint and bandwidth. In fact, in this configuration, we suggest (not shown in the Table) preloading all the reference delay table and the correction coefficients (about 200 kB of data), doing completely away with the memory interface. Even so, the FPGA resource utilization is around 3%.

The *UF2D* case is particularly interesting. Since 16 different TX beams are emitted, TABLESTEER requires 16 reference delay tables, or about 3 MB of values. These values are accessed at a very high rate (250 fps, each frame based on $T = 16$, so 4 kHz). Here, both options are possible: either an architecture with fully on-chip reference tables (roughly 3.3 MB of data, filling up about 50% of the FPGA BRAMs) (not shown in the Table) or off-chip streaming, consuming 1.2 GB/s of bandwidth. To meet the extremely fast rate of imaging, at least 2 delay blocks must be instantiated to keep up with the throughput.

IX. CONCLUSIONS

Receive beamforming is the most critical stage of ultrasound image reconstruction, and is critical for both portable imagers - where power consumption must be kept at a minimum - and next-generation 3D devices, since current electronics do not allow imaging at the full resolution capabilities of modern matrix arrays at real-time frame rates.

In this paper, we have described two techniques and implementations, named TABLEFREE and TABLESTEER, to tackle the kernel of the beamforming algorithm, i.e. the generation of delay values. The two techniques have different strong suits; TABLEFREE concentrates on accuracy and does away with an external memory interface altogether, while TABLESTEER uses approximations to reduce circuitry drastically, albeit at a cost of reduced scan-volume and resolution near its edges.

When evaluated against the backdrop of a high-end FPGA, both yield good performance results. TABLESTEER can process enough delay values to keep up with a 100 × 100-element transducer using only a fraction of a single FPGA, although the memory bandwidth is more critical, leading to constraints on the number of zones per frame. TABLEFREE supports only up to 35 × 35-element transducers in this configuration, but is self-contained within the FPGA. Either achievement is unprecedented since all current academic or industrial projects rely either on pre-beamforming which reduces quality, or on massive arrays of processing chips. Both architectures show even more promise in view of the latest generation of Xilinx FPGAs with more logic cells and UltraRAM blocks [57], and TABLEFREE has already been considered for a dedicated ASIC implementation [28]. Both techniques also demonstrate excellent downward scalability, and can fulfill the needs of even ultrafast 2D imaging with a small fraction of FPGA resources, which is indicative of very low-power operation potential.

As a next step, we plan on studying both architectures within a full on-FPGA beamformer, evaluating the resource trade-offs with other portions of the beamformer. We are also planning a more detailed estimation of power consumption. For the TABLESTEER architecture, an optimization of the memory bandwidth requirements is planned.

Acknowledgment

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

References

[1] GE Healthcare, “Vivid i,” 2015. [Online]. Available: http://www3.gehealthcare.com/en/products/categories/ultrasound/vivid/vi vid_i

[2] GE Healthcare, “Vscan portfolio,” 2015. [Online]. Available: http://www3.gehealthcare.com/en/products/categories/ultrasound/vscan_po rtfolio

[3] MobiSante, “Mobisante ultrasound,” 2015. [Online]. Available: http://www.mobisante.com/

[4] “Lumify,” Philips Healthcare, 2016. [Online]. Available: https://www.lumify.philips.com/web/.

[5] H. Hewener and S. Tretbar, “Mobile ultrafast ultrasound imaging system based on smartphone and tablet devices,” in *Proc. 2015 IEEE Int. Ultrason. Symp.*, Nov. 2015, pp. 1–4.

[6] J. Kang *et al.*, “A system-on-chip solution for point-of-care ultrasound imaging systems: Architecture and ASIC implementation,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 10, no. 2, pp. 412–423, Apr. 2016.

[7] Philips Healthcare, “HD15 purewave ultrasound system,” 2015. [Online]. Available: http://www.medical.philips.com/main/products/ultrasound/systems/hd15/

[8] Philips Healthcare, “EPIQ 7 ultrasound system,” 2015. [Online]. Available: http://www.medical.philips.com/main/products/ultrasound/systems/epiq7/

[9] J. Powers and F. Kremkau, “Medical ultrasound systems,” *Interface Focus*, vol. 1, no. 4, pp. 477–489, Aug. 2011.

[10] Philips Electronics N.V., “iE33 xMATRIX echocardiography system,” 2017. [Online]. Available: www.healthcare.philips.com

[11] GE Healthcare, “Voluson E10,” 2017. [Online]. Available: www.gehealthcare.com

[12] Philips Electronics N.V., “Philips iU22 ultrasound with xMATRIX system specifications,” 2012. [Online]. Available: www.healthcare.philips.com

[13] B. Savord and R. Solomon, “Fully sampled matrix transducer for real time 3D ultrasonic imaging,” in *Proc. IEEE Symp. Ultrasonics*, 2003, vol. 1, pp. 945–953.

[14] J.-Y. Um *et al.*, “A single-chip 32-channel analog beamformer with 4-ns delay resolution and 768-ns maximum delay range for ultrasound medical imaging with a linear array transducer,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 9, no. 1, pp. 138–151, Feb. 2015.

[15] H. gil Kang *et al.*, “Column-based micro-beamformer for improved 2D beamforming using a matrix array transducer,” in *Proc. 2015 Biomed. Circuits Syst. Conf.*, 2015, pp. 1–4.

[16] T. M. Carpenter, M. W. Rashid, M. Ghovanloo, D. Cowell, S. Freear, and F. L. Degertekin, “Time-division multiplexing for cable reduction in ultrasound imaging catheters,” in *Proc. 2015 Biomed. Circuits Syst. Conf.*, 2015, pp. 1–4.

[17] J. T. Yen, J. P. Steinberg, and S. W. Smith, “Sparse 2-D array design for real time rectilinear volumetric imaging,” *IEEE Trans. Ultrason., Ferroelect., Freq. Control*, vol. 47, no. 1, pp. 93–110, Jan. 2000.

[18] A. Austeng and S. Holm, “Sparse 2-d arrays for 3-d phased array imaging-design methods,” *IEEE Trans. Ultrason., Ferroelectr., Freq. Control.*, vol. 49, no. 8, pp. 1073–1086, Aug. 2002.

[19] GE Healthcare, “Voluson E8 expert,” 2017. [Online]. Available: www.gehealthcare.com

[20] Y.-F. Li and P.-C. Li, “Software beamforming: Comparison between a phased array and synthetic transmit aperture,” *Ultrason. Imag.*, vol. 33, no. 2, pp. 109–118, Apr. 2011.

[21] B. Yiu, I. Tsang, and A. Yu, “Gpu-based beamformer: Fast realization of plane wave compounding and synthetic aperture imaging,” *IEEE Trans. Ultrason., Ferroelect., Freq. Control*, vol. 58, no. 8, pp. 1698–1705, Aug. 2011.

[22] J. Ma, K. Karadayi, M. Ali, and Y. Kim, “Ultrasound phase rotation beamforming on multi-core DSP,” *Ultrasonics*, vol. 54, no. 1, pp. 99–105, Jan. 2014.

[23] C. Lee, H.-Y. Sohn, D.-H. Han, and T.-K. Song, “Real-time implementation of the echo signal processing and digital scan conversion for medical ultrasound imaging with a single tms320c6416 DSP,” in *Proc. SPIE - The Int. Soc. Opt. Eng.*, vol. 6920, Mar. 2008, Art. no. 692004.

[24] D. N. Truong and B. M. Baas, “Massively parallel processor array for mid-/back-end ultrasound signal processing,” in *Proc. 2010 Biomed. Circuits Syst. Conf.*, 2010, pp. 274–277.

[25] J. Jensen *et al.*, “Sarus: A synthetic aperture real-time ultrasound system,” *IEEE Trans. Ultrason., Ferroelect., Freq. Control*, vol. 60, no. 9, pp. 1838–1852, Sep. 2013.

[26] E. Boni *et al.*, “ULA-OP 256: A 256-channel open scanner for development and real-time implementation of new ultrasound methods,” *IEEE Trans. Ultrasonics, Ferroelect., Freq. Control*, vol. 63, no. 10, pp. 1488–1495, Oct. 2016.

[27] P. Vogel, A. Bartolini, and L. Benini, “Efficient parallel beamforming for 3D ultrasound imaging,” in *Proc. 24th Ed. Great Lakes Symp. VLSI*, 2014, pp. 175–180.

[28] P. A. Hager, P. Vogel, A. Bartolini, and L. Benini, “Assessing the area/power/performance tradeoffs for an integrated fully-digital, large-scale 3D-ultrasound beamformer,” in *Proc. 2014 Biomed. Circuits Syst. Conf.*, 2014, pp. 228–231.

[29] A. Ibrahim *et al.*, “Tackling the bottleneck of delay tables in 3d ultrasound imaging,” in *Proc. 2015 Design Autom. Test Eur. Conf.*, Mar. 2015, pp. 1683–1688.

[30] P. A. Hager, A. Bartolini, and L. Benini, “Ekho: A fully-digital integrated 3d beamformer for medical ultrasound imaging,” *IEEE Trans. Very Large Scale Integr.*, vol. 24, no. 5, pp. 1936–1949, May 2015.

[31] K. Thomenius, “Evolution of ultrasound beamformers,” in *Proc. IEEE Ultrason. Symp.*, Nov. 1996, vol. 2, pp. 1615–1622.

[32] G. Mclaughlin, “Zone sonography: What it is and how it’s different,” ZONARE Medical Systems, Inc., Tech. Rep., 2012.

[33] D. Shattuck, M. Weinshenker, S. Smith, and von Ramm OT, “Explososcan: a parallel processing technique for high speed ultrasound imaging with linear phased arrays,” *J. Acoust. Soc. Amer.*, vol. 75, no. 4, pp. 1273–1282, Apr. 1984.

[34] J. Bercoff, “Ultrafast ultrasound imaging,” in *Ultrasound Imaging - Medical Applications*, I. V. Minin and O. V. Minin, Eds. Rijeka, Croatia: InTech, Aug. 2011.

[35] M. Tanter and M. Fink, “Ultrafast imaging in biomedical ultrasound,” *IEEE Trans. Ultrason., Ferroelect., Freq. Control*, vol. 61, no. 1, pp. 102–119, Jan. 2014.

[36] B. Steinberg, “Digital beamforming in ultrasound,” *IEEE Trans. Ultrason., Ferroelect., Freq. Control*, vol. 39, no. 6, pp. 716–721, Nov. 1992.

[37] E. A. Howard *et al.*, *Combined Two-Dimensional Tissue/Flow Imaging*. Boston, MA, USA: Springer, 1980, pp. 533–544.

[38] R. G. Pridham and R. A. Mucci, "Digital interpolation beamforming for low-pass and bandpass signals," *Proc. IEEE*, vol. 67, no. 6, pp. 904–919, Jun. 1979.

[39] G. Frey and R. Chiao, "4Z1c real-time volume imaging transducer," *Siemens Healthcare Sector, White Paper*, 2008.

[40] J. Um *et al.*, "An analog-digital hybrid RX beamformer chip with non-uniform sampling for ultrasound medical imaging with 2D CMUT array," *IEEE Trans. Biomed. Circuits Syst.*, vol. 8, no. 6, pp. 799–809, Dec. 2014.

[41] K. Üstüner, "High information rate volumetric ultrasound imaging," *Siemens Healthcare sector, White paper*, 2008.

[42] R. E. Daigle, "Ultrasound imaging system with pixel oriented processing," US Patent 8 287 456, Oct. 16 2012.

[43] T. G. Bjåstad, "High frame rate ultrasound imaging using parallel beamforming," Ph.D. dissertation, Norwegian University of Science and Technology, Trondheim, Norway, 2009.

[44] J. A. Jensen, S. I. Nikolov, K. L. Gammelmark, and M. H. Pedersen, "Synthetic aperture ultrasound imaging," *Ultrasonics*, vol. 44, pp. e5–e15, 2006.

[45] R. Sampson, M. Yang, S. Wei, C. Chakrabarti, and T. Wenisch, "Sonic Millip3De: A massively parallel 3D-stacked accelerator for 3D ultrasound," in *Proc. IEEE 19th Int. Symp. High Performance Comput. Archit.,* Feb. 2013, pp. 318–329.

[46] R. Sampson *et al.*, "FPGA implementation of low-power 3D ultrasound beamformer," in *Proc. 2015 IEEE Int. Ultrason. Symp.*, Nov. 2015, pp. 1–4.

[47] S. I. Nikolov, J. A. Jensen, and B. Tomov, "Recursive delay calculation unit for parametric beamformer," in *Medical Imaging*. Int. Soc. Opt. Photon., 2006, pp. 61470D–61470D.

[48] Cephasonics Receive RX Beamformer, CSC2032, 2016. [Online]. Available: http://www.cephasonics.com

[49] S. Nikolov, J. Jensen, and B. Tomov, "Fast parametric beamformer for synthetic aperture imaging," *IEEE Trans. Ultrason., Ferroelectr., Freq. Control.*, vol. 55, no. 8, pp. 1755–1767, Aug. 2008.

[50] J. Park *et al.*, "Efficient implementation of a real-time dynamic synthetic aperture beamformer," in *Proc. IEEE Symp. Ultrason.*, Oct. 2012, pp. 2250–2253.

[51] BK Ultrasound, "Sonix tablet," 2016. [Online]. Available: http://bkultrasound.com/products/sonix/systems/sonixtabletq-ultrasound-system

[52] Y.-J. Kim *et al.*, "A single-chip 64-channel ultrasound RX-beamformer including analog front-end and an LUT for non-uniform ADC-sample-clock generation," *IEEE Trans. Biomed. Circuits Syst.*, vol. 11, no. 1, pp. 87–97, Feb. 2017.

[53] C. Sumi and S. Uga, "Effective ultrasonic virtual sources which can be positioned independently of physical aperture focus positions," *Rep. Med. Imag.*, vol. 2010, no. 3, pp. 45–59, 2010.

[54] S. Holm, "Digital beamforming in ultrasound imaging," in *Proc. Nordic Signal Process. Symp.*, Jun. 1994.

[55] Xilinx Inc., "7 series DSP48E1 slice," 2014. [Online]. Available: http://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf

[56] Xilinx Inc., "Virtex-7 FPGA family," 2014, [Online]. Available: www.xilinx.com/products/silicon-devices/fpga/virtex-7.html

[57] Xilinx Inc., "Ultrascale architecture," 2014. [Online]. Available: www.xilinx.com/products/technology/ultrascale.html

[58] "Field ii simulation program: Calculation of b-mode image of synthetic kidney," Field II, Apr. 2012. [Online]. Available: http://field-ii.dk/?examples/kidney_example/kidney_example.html

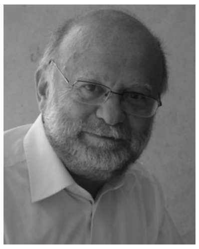
[59] Samsung, "Samsung electronics starts mass producing industrys first 8-gigabit graphics DRAM (GDDR5)," 2015. [Online]. Available: http://www.samsung.com/semiconductor/insights/news/13921

Aya Ibrahim received the B.Sc. and M.Sc. degrees with distinction in systems and biomedical engineering from Cairo University, Egypt, in 2010 and 2013, respectively. She is currently working toward the Ph.D. degree in the Integrated Systems Laboratory (LSI) of the Swiss Federal Institute of Technology, Lausanne, Switzerland. Her research interest includes digital image and signal processing, medical imaging, and low-power HW/SW codesign. She received the Bronze Leaf Award for her paper on the PRIME 2016 Conference.

Pascal A. Hager received the M.Sc. degree with distinction in electrical engineering and information technology from ETH Zurich, Zurich, Switzerland in 2014, where he is currently working toward the Ph.D. degree. Since then, he is with the Integrated Systems Laboratory, ETH Zurich. His research interests include medical imaging, digital signal processing and low-power integrated circuit design. He received the Best Paper Award at the 2013 IEEE VLSI-SoC Conference.

Andrea Bartolini (M'13) received the Ph.D. degree in electrical engineering from the University of Bologna, Bologna, Italy, in 2011. He is currently a Post-Doctoral Researcher in the Integrated Systems Laboratory, ETH Zurich, Zurich, Switzerland. He also holds a Post-Doctorate Position in the Department of Electrical, Electronic and Information Engineering Guglielmo Marconi, University of Bologna. His current research interests include green computing and dynamic resource management ranging from embedded to large-scale high performance computing systems with special emphasis on thermal and power-aware HW/SW codesign techniques.

Federico Angiolini received the M.S. degree (*summa cum laude*) in electrical engineering from the University of Bologna, Bologna, Italy, in 2003, and the Ph.D. degree from the Department of Electronics and Computer Science, University of Bologna, in 2008. His initial research interests included memory hierarchies, multiprocessor-embedded systems, and networks-on-chip, and resulted in him cofounding and being CEO of iNoCs Structured Interconnects, based in Lausanne, CH. Since 2013, he has been in the Swiss Federal Polytechnical School of Lausanne, where he is working on medical imaging and drug delivery platforms.



Marcel Arditi (M'82) was born in Alexandria, Egypt, in 1951. He received the graduated degree in 1975, the M.Sc. degree in physics from the University of Geneva, Geneva, Switzerland, and the Ph.D. degree from the Department of Medical Biophysics, University of Toronto, Toronto, ON, Canada, specializing in the use of annular arrays in medical ultrasound imaging, in 1982. From 1982 to 1985, he was with SRI International in Menlo Park, CA, USA, where he conducted research for various industrial and government organizations, still in the ultrasound imaging field. In 1985, he joined Battelle-Europe in Geneva, Switzerland, and managed contract research projects for corporations in the optical and automotive industries, in fields related to image and signal processing. Between 1994 and 2014, he was a Senior Scientist with Bracco Suisse SA, also in Geneva; he contributed to the acoustical specifications and characterization of microbubble-based contrast agents for medical ultrasound and developed software for quantifying blood perfusion with contrast-enhanced ultrasound. Since 2015, she is with EPFL in Lausanne, Switzerland, on novel beamforming strategies for medical ultrasound.

Jean-Philippe Thiran was born in Namur, Belgium. He received the Electrical Engineering degree and the Ph.D. degree from the Universit catholique de Louvain, Louvain-la-Neuve, Belgium, in 1993 and 1997, respectively. He joined the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in February 1998 as a Senior Lecturer. He is currently an Associate Professor in Signal Processing and Director of the Signal Processing Lab 5 at EPFL. He also holds a 20% Associate Professor position in the Department of Radiology of the University Hospital Center and University of Lausanne. His current scientific interests include image analysis and computer vision, medical imaging and multimodal signal/image processing, with application to brain connectivity analysis, remote sensing, human-computer interactions, etc. He is author or coauthor of several book chapters, of 150 journal papers, and of more than 200 conference papers in image processing and analysis. He holds seven international patents. Until 2015, he was an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING. Among other duties, he was also the technical Co-Chair of the 2015 IEEE International Conference on Image Processing. He was a member of the Machine Learning for Signal Processing Technical Committee from 2008 to 2010 and of the Image, Video, and Multidimensional Signal Processing Technical Committee from 2009 to 2014 of the IEEE Signal Processing Society.

Luca Benini received the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 1997. He is the Chair of digital circuits and systems at ETHZ and is a Full Professor at the Universita di Bologna, Bologna, Italy. His research interests include energy-efficient system design for embedded and high-performance computing. He is also active in the area of energy-efficient smart sensors and ultra-low power VLSI design. He has published more than 800 papers in peer-reviewed international journals and conferences, four books, and several book chapters. He is a Fellow of the ACM and a member of the Academia Europaea. He received the 2016 IEEE CAS Mac Van Valkenburg award.

Giovanni De Micheli (F'94) received the Nuclear Engineer degree from the Politecnico di Milano, Milano, Italy, in 1979, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California at Berkeley, Berkeley, CA, USA, in 1980 and 1983, respectively. He is a Professor and the Director of the Institute of Electrical Engineering and of the Integrated Systems Centre, EPFL, Switzerland. He is the Program Leader of the Nano-Tera.ch program. Previously, he was a Professor of Electrical Engineering at Stanford University, Stanford, CA, USA. His research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis for emerging technologies, networks on chips, and 3-D integration. He is also interested in heterogeneous platform design including electrical components and biosensors, as well as in data processing of biomedical information. He is the author of Synthesis and Optimization of Digital Circuits book (McGraw-Hill, 1994), and co-author and/or co-editor of eight other books and of more than 750 technical articles. He has an H-index of 92 according to Google Scholar. He is a Fellow of ACM and a member of the Academia Europaea. He is member of the Scientific Advisory Board of IMEC (Leuven, B), CfAED (Dresden, D), and STMicroelectronics. He received the 2016 IEEE/CS Harry Goode award for seminal contributions to design and design tools of Networks on Chips, the 2016 EDAA Lifetime Achievement Award, the 2012 IEEE/CAS Mac Van Valkenburg award for contributions to theory, practice and experimentation in design methods and tools, and the 2003 IEEE Emanuel Piore Award for contributions to computer-aided synthesis of digital systems. He also received the Golden Jubilee Medal for outstanding contributions to the IEEE CAS Society in 2000, the D. Pederson Award for the best paper on the IEEE Transactions on CAD/ICAS in 1987, and several Best Paper Awards, including DAC (1983 and 1993), DATE (2005), Nanoarch (2010 and 2012), and Mobihealth (2016). He has been serving IEEE in several capacities, namely: Division 1 Director (2008-9), co-founder and President Elect of the IEEE Council on EDA (2005–2007), President of the IEEE CAS Society (2003), Editor in Chief of the IEEE Transactions on CAD/ICAS (1997–2001). He has been Chair of several conferences, including Memocode (2014) DATE (2010), pHealth (2006), VLSI SOC (2006), DAC (2000), and ICCD (1989).