Design of Fully-Digital Medical Ultrasound Imaging Systems

A thesis submitted to attain the degree of DOCTOR OF SCIENCES of ETH ZURICH (Dr. sc. ETH Zurich)

presented by
PASCAL ALEXANDER HAGER
MSc ETH EEIT, ETH Zurich

born on 13.03.1990
citizen of Adelboden BE, Switzerland

accepted on the recommendation of
Prof. Dr. Luca Benini, examiner
Prof. Dr. Jean-Philippe Thiran, co-examiner

2019
Acknowledgments

The last five years have been an exciting and truly shaping journey. A doctorate really is not just about discovering new things and improving one’s skills, but also about learning how to handle the emotional roller coaster that one experiences with the successes and failures of a project that has grown so close to one’s heart over the years. Now, at the end of my doctoral thesis, I can look back with real satisfaction and appreciate the remarkable experience that has enabled me to grow in so many different ways.

For this, I would like to thank several people that I have had the pleasure to get to know over the years and who have supported me on this journey: First of all, I like to thank my supervisor Luca Benini for giving me the opportunity to pursue a Ph.D. in his group, for his guidance and support through this challenging time and for giving me the freedom to explore my own ideas as well. Secondly, I would also like to thank my co-examiner Prof. Dr. Jean-Philippe Thiran, whom I got to know before I started my doctoral thesis and with whom I held several interesting discussions over the years. Special thanks go to Norbert Felber, who already very early put the idea in my head to do a Ph.D.

I like to particularly thank my office mates in ETZ J69.2, most of which grew wings and flew out over the year as they finished their Ph.D. and continued their journey elsewhere: More precisely I thank
David Bellasi for the deep conversations and all the other questions we answered and discussed. I thank Pirmin Vogel, who was always up for a joke but with whom you could also have the most serious kind of conversations. And Micheal Gautschi, our ‘last’ Micheal, who from outside, may have appeared to be the most serious one of us, but really was not. And Florian Glaser, that may from a strictly formal point of view not belonged to our office, but still was a crucial part of us. I also thank Micheal Schaffner and Andreas Kurth, who later (re-)joined the office for the fruitful (and awarded!) collaborations we had.

I also thank Andres Gomez for tackling together with me the adventurous journey of the bi-weekly meetings. And Lukas Cavigelli, my longtime friend, who somehow managed to end up in the same place as me.

I also thank Hamed Fatemi and Jose Pineda de Gyvez from NXP. Even though our joint work did not make it in the final version of the thesis, I would not want to miss the experience to have had the chance during my Ph.D. to peek in the world of corporate research.

I also thank my collaborators on the ULTRASOUNDToGo project: Andreas Tretter, Adrien Besson, Dimitris Perdios, Stefanos Skalistis, Alena Simalatsar, Aya Ibrahim, and Federico Angiolini. Special Thanks go to Prof Giovanni De Micheli, the initiator, and leader of the NANO-TERA.CH program, which was not just a regular funding program that financed my Ph.D. position, but also provided a rich and instructive program for its Ph.D. students.

I also thank my collaborators at the Fraunhofer Institute for Biomedical Engineering: Peter-Karl Weber, Christoph Risser, Manfred Moses, Daniel Speicher, and Christoph Degel.

I also thank the various people at IIS: Special thanks to the support staff at IIS, namely Hansjoerg Gisler, Christoph Wicki and Christine Haller and the D-ITET Design center, namely Hubert Kaeslin, Beat Muheim, Frank Gurkaynak, and Alfonso Blanco.

Finally, I thank all my friends and in particular my family Christoph, Monika, Elisabeth, and Hanspeter.
Abstract

Ultrasound imaging is still one of the most used clinical imaging modalities thanks to its real-time output, its ease-of-use, its portability and low cost, and also because it is safe to use, as it operates without potentially harmful radiation. From its inception, ultrasound imaging strongly relied on hardware innovations to be the enabler for new disruptive imaging modalities and inventive system architectures. The latter is essential to building affordable products for medical service providers.

Today, the primary challenge in ultrasound system design is the accelerated pace with which new ultrasound imaging and diagnostic modes are emerging: To be competitive, a state-of-the-art imaging system today must support more than a dozen different operation modes. Also, the amount of required processing resources for the latest modalities easily exceeds several TFLOP/s. To keep up with this race requires new ultrasound system architectures, which are more flexible and powerful, and most importantly scalable.

In this thesis, we explore several hardware innovations for future ultrasound systems, with a focus on replacing analog system parts with digital solutions. First, we address the challenges around 3D imaging: We explore fully-digital beamformers for fully-sampled matrix transducers to replace existing mixed analog-digital solutions, which are less flexible. We demonstrate that a fully-digital 10’000-channel
beamformer can be implemented on a single chip, which only consumes 30.3 W. We also investigate the design of a 1024-channel multiplexer matrix probe to be used in research.

We then shift our focus to advanced 2D imaging: We explore a new system architecture, which combines the benefits of powerful and flexible software-defined ultrasound systems with the cost-efficiency of the latest off-the-shelf accelerated (CPU+GPU) computing systems. To showcase our concept, we design a unique ultrasound probe, which embeds an entire 64-channel ultrasound front-end within the probe handle, and provides access to the digitized sensor data over a 25 Gbit/s fiber-optics interface. This digital optical connector entirely replaces the expensive analog cable harness used in conventional probes. We connect our probe to an off-the-shelf PC equipped with a powerful GPU to build a fully operational ultrasound imaging system with performance comparable to medium-to-high-end traditional systems at a small fraction of the cost. Our innovative “LightProbe” architecture relies entirely on commodity hardware for processing, which can be easy-upgraded when more demanding modalities arise. With the designed system, we demonstrate that advanced compute-intensive ultrasound modalities can be implemented at low cost.
Zusammenfassung

Die Ultraschallbildgebung ist nach wie vor eine der am häufigsten verwendeten Modalitäten der klinischen Bildgebung. Sie zeichnet sich durch ihre Echtzeitausgabe, ihre einfache Handhabung, ihre Portabilität und ihre niedrigen Kosten aus, und ist zudem sicher in der Anwendung, da sie ohne potenziell schädliche Strahlung arbeitet. Von Anfang an war die Ultraschallbildgebung stark auf Hardwareinnovationen angewiesen, die neue revolutionäre Bildgebungsmodalitäten und innovative Systemarchitekturen erst möglich machten. Letzteres ist unerlässlich, um erschwingliche Produkte für medizinische Dienstleister zu entwickeln.

Die größte Herausforderung, der sich ein Entwickler von Ultraschallsystemen, heute gegenüber sieht, ist das immer schnellere Aufkommen neuer Ultraschallbildgebungs- und Diagnosemodi. Um heute ein hochmodernes Bildgebungssystem verkaufen zu können, muss es mehr als ein Dutzend verschiedener Betriebsarten unterstützen. Dazu kommt, dass die neuesten Bildgebungsverfahren mehrere TFLOP/s an Verarbeitungsressourcen benötigen. Um mit diesem Wettlauf Schritt zu halten, sind flexiblere, leistungsfähigere aber vor allem skalierende Ultraschallsystemarchitekturen erforderlich.

In dieser Arbeit werden mehrere Hardwareinnovationen für zukünftige Ultraschallsysteme untersucht, wobei der Schwerpunkt auf dem Ersatz analoger Systemkomponenten durch digitale Lösungen liegt.

# Contents

1 Introduction ............................................................................. 1
   1.1 Motivation ........................................................................ 1
   1.2 Ultrasound Imaging in a Nutshell ........................................ 3
   1.3 Current System Design Trends ........................................... 4
   1.4 Thesis Overview ............................................................... 6
       1.4.1 A Transducer Probe for 3D Imaging ......................... 8
       1.4.2 High-Performance Beamforming for 2D/3D ............. 9
       1.4.3 Ultrafast Digital Probes .......................................... 10
       1.4.4 Advanced Imaging Modalities ................................ 12
   1.5 Summary of Contributions ................................................ 13
   1.6 List of Publications ......................................................... 14

2 Ekho: A Single-Chip 3D-Beamformer ......................................... 17
   2.1 Introduction ...................................................................... 17
       2.1.1 Motivation ................................................................ 17
       2.1.2 Challenges ............................................................. 18
       2.1.3 Contributions ......................................................... 20
   2.2 Related Work .................................................................... 23
   2.3 System Level – Beamformer Design ...................................... 26
       2.3.1 Ultrasound Imaging in a Nutshell ......................... 26
       2.3.2 System Overview ..................................................... 28
       2.3.3 Beamformer Specifications ..................................... 29
## 2.3.4 The Case for Bandpass Processing

2.4 Online Delay computation

2.4.1 Hybrid Square-Root Computation (HSQRT)

2.5 Beamformer Architecture

2.5.1 Global Control and Computations

2.5.2 Beamformer Channel

2.5.3 System-Level flexibility

2.6 Error Analysis and Imaging Results

2.6.1 Methodology

2.6.2 Bandpass beamformer loss

2.6.3 Delay index accuracy

2.6.4 Fixed-point implementation loss

2.7 Pre-silicon Implementation Results

2.7.1 Estimated Area

2.7.2 Estimated Power

2.8 Comparison with Related Work

2.9 Conclusions and Outlook

2.10 Appendix - Ekho Reconfigurablilty

3 MuxHead: A 1024-element Probe for 3D Imaging

3.1 Introduction

3.1.1 Motivation

3.1.2 Contributions

3.2 System Overview

3.2.1 Multiplexer Operation

3.2.2 Multiplexer Setup

3.2.3 Circuit Details of a Channel

3.3 Implementation

3.3.1 Matrix Transducer

3.3.2 Stack Design & Probe Electronics

3.3.3 MUXHEAD Configuration and Operation

3.4 MUXHEAD Simulation

3.4.1 Setup

3.4.2 Results and Discussion

3.5 In-vitro tests

3.5.1 Setup
CONTENTS

3.5.2 Results and Discussion ........................................... 91
3.5.3 Thermal Dissipation Test ......................................... 92
3.6 Discussion and Conclusion ........................................ 94

4 LightProbe & UltraLight:
Software-Defined Ultrasound using Digital Probes 95
4.1 Introduction .......................................................... 95
  4.1.1 Motivation ..................................................... 95
  4.1.2 A Novel Ultrasound System Architecture .................. 98
  4.1.3 Contributions ................................................ 99
4.2 Related Work ........................................................ 102
  4.2.1 Taxonomy ...................................................... 102
  4.2.2 Analog Probes ............................................... 102
  4.2.3 Digital Probes ............................................... 103
  4.2.4 Real-time Raw Sample Access ............................... 104
4.3 Designing an Ultrafast Probe .................................... 105
  4.3.1 Avoid Heat from Processing ................................. 105
  4.3.2 Exploit Ultrafast Imaging to Reduce
       Energy Consumption ......................................... 106
  4.3.3 Boost Mode for High-Frame Rate Imaging ................. 107
4.4 System Architecture ............................................... 108
4.5 Host PC Implementation .......................................... 110
4.6 LightProbe Hardware Implementation ............................ 111
  4.6.1 Ultrasound Transducer Array ............................... 112
  4.6.2 Ultrasound Front-End ....................................... 113
  4.6.3 Final LightProbe Assembly ................................. 116
  4.6.4 FPGA Subsystem ............................................ 117
4.7 System Operation and Firmware .................................. 120
  4.7.1 Programming Model ......................................... 120
  4.7.2 Dynamic Power Management ................................. 121
  4.7.3 Dynamic Thermal Management .............................. 122
4.8 Methods ............................................................ 127
  4.8.1 Imaging Quality Assessment ............................... 127
  4.8.2 PRF, Frame-Rate and Power Measurements ............... 128
  4.8.3 Thermal Control ............................................ 129
4.9 Results ............................................................. 130
  4.9.1 Imaging Quality Assessment ............................... 130
  4.9.2 Achievable PRF and Frame-Rate ......................... 134
4.9.3 Power Measurements .............................................. 134
4.9.4 Thermal Control .................................................. 138
4.10 Discussion and Comparison ....................................... 142
  4.10.1 Comparison with Related Work .............................. 142
  4.10.2 Possible Extensions ........................................... 144
  4.10.3 Potential for Power and Size Reduction ................. 144
4.11 Conclusions ....................................................... 146

5 UltraLight Case-Study: High-Speed Bubble Tracking 147
  5.1 Introduction ..................................................... 147
    5.1.1 Motivation .................................................. 147
    5.1.2 Contributions ............................................... 149
  5.2 Ultrafast Bubble Tracking System .............................. 150
    5.2.1 Raw Data Acquisition .................................... 151
    5.2.2 Ultrafast B-Mode Beamforming ............................. 151
    5.2.3 Bubble Tracking ............................................. 152
  5.3 Experimental Verification ..................................... 153
    5.3.1 Simulation Experiments .................................... 153
    5.3.2 Phantom Experiment ........................................ 154
  5.4 Results .......................................................... 155
    5.4.1 Simulation .................................................. 155
    5.4.2 Phantom Experiment ........................................ 155
  5.5 Discussion ........................................................ 158
  5.6 Conclusions and Future Work ................................... 158

6 Summary and Conclusions ........................................... 159
  6.1 Overview of the main results .................................. 160
  6.2 Outlook .......................................................... 166

A PCB Gallery .......................................................... 169
  A.1 MuxHead .......................................................... 170
    A.1.1 Multiplexer Board .......................................... 170
    A.1.2 Control Board ............................................... 171
    A.1.3 Cable Connectors ........................................... 172
  A.2 LightProbe – Proof-Of-Concept ................................ 173
  A.3 LightProbe ........................................................ 174
    A.3.1 MotherBoard .................................................. 174
    A.3.2 TX Board - LM96551 .......................................... 175
Chapter 1

Introduction

1.1 Motivation

Medical ultrasound imaging is the second-most used clinical imaging modality after X-ray imaging and the most-used cross-sectional imaging technique [1]. Compared to other cross-sectional imaging techniques such as X-ray-computed tomography (CT), magnetic resonance imaging (MRI) or positron emission tomography (PET), ultrasound imaging has lower cost, is portable, and operates without potentially harmful ionizing radiation. Moreover, ultrasound imaging produces a real-time image output and thus provides a higher temporal resolution compared to other imaging modalities just producing still images.

From the beginning, the evolution of ultrasound imaging strongly relied on technology enablers provided by the semiconductor industry [2]: This started with the first real-time ultrasound systems in the 1980s made possible by microprocessors and integrated delay lines enabling electronic beam-steering, and continues until today, with software-based imaging enabled by latest graphics processing units (GPU), and miniaturization allowing to build ultra-compact hand-held imaging devices.
Even though ultrasound imaging is considered a mature technology, many innovations are still made today [3]. Moore’s Law powers many of these innovations and allows more and more sophisticated processing to be applied to the raw ultrasound data. High-frame-rate imaging [4] is one of the recent achievements only made possible with the high-performance compute resources available today. In high-frame-rate imaging, ultrasound images are captured and processed at a frame-rate of several kilohertz. This technological break-through enabled several new ultrasound imaging applications such as:

- *ultrafast contrast imaging* [5], allowing imaging below the diffraction limit using micro-bubble contrast agents,
- *shear-wave elastography* [6], enabling quantitative measurements of tissue properties,
- *vector flow imaging* [7], providing quantitative vector blood flow information with high spatial and temporal resolution,
- *functional ultrasound imaging* [8], capable of visualizing brain activity in real-time.

Another recent achievement powered by Moore’s Law is to extend conventional 2D imaging to 3D imaging [9]. In 3D imaging, a volume of the body is captured instead of only a cross-section. Even though 3D and 4D ultrasound (multiple volumes over time) are available in state-of-the-art high-end ultrasound systems, 3D/4D ultrasound is not yet widely adopted by medical practitioner outside the field of obstetrics [10, 11] and cardiology [12]. Reasons for this are the lack of substantial evidence of the benefits of 3D ultrasound over 2D ultrasound [13], the currently still inferior spatial/temporal resolution of 3D ultrasound compared to 2D ultrasound, and the high-cost in acquiring a system with 3D capability. Further hardware innovations are required to fix current resolution issues and lower the cost. Once achieved, 3D imaging may be a similar technological enabler to spark many new applications has high-frame-rate imaging did.

Continued innovation on the ultrasound hardware and system design is thus key to make new applications possible and to bring them from the lab into the health-care services such that patients can benefit.
1.2 Ultrasound Imaging in a Nutshell

During an ultrasound examination a trained operator, called Sonographer, sweeps an ultrasound probe over the body part under examination. The ultrasound imaging system provides the operator with a real-time image of a cross-section through the body below the probe. By sweeping the probe back and forth allows the operator to picture the internal three-dimensional structures in the body.

![Ultrasound Diagram]

**Figure 1.1** – A conventional ultrasound imaging system.

A typical conventional ultrasound system as depicted in Fig. 1.1 consist of an ultrasound probe, which is connected to a backend system. The backend system provides a control interface and a display to show the output image. The ultrasound probe is the part applied to the body during the investigation and embeds the electro-acoustical transducer used to send and receive ultrasound signals. The backend system contains the analog front-end electronics and the digital processing and control unit. An ultrasound image is formed in three steps:

**Signal Acquisition** The ultrasound system uses its probe to emit an acoustic pulse into the body under examination. If the pulse hits a material transition, the pulse is partially reflected due to the change in acoustic impedance. The backscattered echos are then captured again by the probe.
Beamforming From the received echo signals the system computes a *reflectivity map* indicating how much energy has been reflected from a given point in space.

Post-Processing In a post-processing step, a human-readable black & white image is formed out of the reflectivity map, where ‘white’ indicates a region of high *echogenicity* and ‘black’ a region of low echogenicity (anechoic region).

In the final image, water shows up as black, while tissue shows up as gray regions with different shades of gray depending on its echogenicity. The boundaries between bones and tissue show up as clear white lines due to the hard reflection at the bone-tissue transition. Water filled cysts show up as black regions as the enclosed water has a homogeneous acoustic impedance and thus does not scatter back the acoustic pulse.

1.3 Current System Design Trends

Today, the following trends can be observed in ultrasound imaging systems design:

**Small Portable Systems:** A digital ultrasound probe is directly connected to a smartphone, which displays the image and controls the probe using an ultrasound imaging ‘app’ [14–16]. These very compact systems can be used by the physician at the patient bedside with the same naturalness and ease as stethoscopes have been used for decades. In order to achieve compactness, the ultrasound probe integrates the entire analog front-end and performs the bulk part of the processing in the probe. This allows to directly connect the probe with a smartphone over USB or WiFi. These systems require a very high degree of integration while providing limited diagnostic functions, mostly limited by thermal constraints on the probe.

**Software-Defined Systems:** Conventional system rely on custom hardware implementations (ASIC, FPGAs) to perform the processing. Software-defined systems [17] perform the processing in software using GPUs [18–20] or on multicore (DSP) processors [21, 22]. Software-defined systems can leverage the latest
commodity hardware to reduce cost and are more flexible as new imaging functions can be implemented by simply writing the required software application.

**3D Ultrasound Systems:** 3D systems capture image volumes instead of cross sections [23,24]. This allows capturing the three-dimensional geometry of an organ in an instant without having the operator to picture the organ while sweeping the probe. This enables precise volume measurements and allows to visualize fast moving parts such as the heart valves, which are moving too fast to conceive the 3D geometry by sweeping the probe over it. However, 3D systems require an ultrasound probe with a matrix transducer containing several thousand elements [25,26] as opposed to the 2D systems, which operate with a linear transducer array with a few hundred elements. This two-order of magnitude increase in the number of front-end channels heavily complicates the system design and requires substantially increased processing resources.
1.4 Thesis Overview

In this thesis, I am going to investigate several research questions towards the design of next-generation ultrasound systems with a focus on replacing analog system parts with digital solutions.

This thesis follows a holistic approach and covers all major ultrasound system components over the entire ultrasound imaging pipeline from the transducer probe, over the front-end electronics to the processing hardware including the imaging algorithms.

An overview of the structure of the thesis is given in Fig. 1.2. The thesis extends over the three main ultrasound system design trends (mobile, software-defined, 3D) previously elaborated and over the three steps required to acquire an ultrasound image (signal acquisition, beamforming, and post-processing). In the first part, the challenges of fully-digital 3D systems were addressed (Ekho, Chapter 2) before a matrix probe was built for 3D imaging (MUXHEAD, Chapter 3). Later, the work was extended to 2D imaging by designing a digital ultrasound probe (LIGHTPROBE) to build a complete software-defined imaging system (ULTRALIGHT, Chapter 4). At last, advanced imaging modalities were implemented on our system to demonstrate that it delivers the required performance (High-Speed Bubble Tracking, Chapter 5). Fig. 1.2 also gives an overview of main on-going follow-up projects enabled by contributions of this thesis.

![Figure 1.2 – Thesis overview and follow-up projects.](image-url)
Due to the holistic approach, the contributions of this thesis enabled two ultrasound imaging system to become operational:

1. an ultrafast software-defined 2D system, and
2. an experimental 3D imaging system.

The main hardware components built for these systems during this thesis are depicted in Fig. 1.3. An exemplary output image is shown for both systems, in order to demonstrate they are both operational.

<table>
<thead>
<tr>
<th>Signal Acquisition HW</th>
<th>Processing HW</th>
<th>Output Image</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2D</strong></td>
<td><strong>LightProbe</strong></td>
<td><strong>UltraLight</strong></td>
</tr>
<tr>
<td>64-cha Digital Probe</td>
<td>Ultrafast Imaging System</td>
<td>500 fps Imaging</td>
</tr>
<tr>
<td><strong>3D</strong></td>
<td><strong>MuxHead</strong></td>
<td><strong>Ekho</strong></td>
</tr>
<tr>
<td>1024-cha Matrix Probe</td>
<td>10'000-cha Beamformer</td>
<td>3D wire phantom</td>
</tr>
</tbody>
</table>

**Figure 1.3** – The main ultrasound hardware components built.

The following sections provide for each sub-topic a summary of the prior work, the main challenges and the contributions made in this thesis. Additionally, external contributions and project involvements are declared.
1.4.1 A Transducer Probe for 3D Imaging

The breakthrough in 3D ultrasound imaging was achieved by the introduction of multidimensional transducer arrays in combination with parallel processing. The Duke T2 ultrasound imaging system [27] was the first real-time 3D system. It used a two-dimensional array transducer of 289 elements to perform a 65° pyramidal scan of $24 \times 24$ image-lines at a rate of 8 frames per second. In order to improve the quality of the image, system designer increased the number of elements in the two-dimensional array up to a point (ca. 1000) until it became unfeasible to connect every single element of the array to the back-end system due to the cabling limitations. To overcome this cabling issue and to support several thousands of elements, micro-prebeamformers were introduced into the transducer probe [25, 28]. Micro-prebeamformers are active analog circuits, which steer a patch of elements (typically 25-36 elements) to a fixed direction in order to form a single signal for the entire patch, and thereby reduce the number of required cables by more than an order of magnitude. However, this limits the synthetic steering flexibility of the transducer array in the digital back-end processing, which deteriorates the overall system resolution. Besides the reception of echoes, the transducer probe must also be able to emit acoustic pulses. For sufficient transmit beam control, each element in the transducer array must be individually controllable. State-of-the-art matrix probes incorporate the entire TX circuitry (TX-beamformer and high-voltage pulser) in the transducer probe [29].

Getting access to such a state-of-art 3D ultrasound transducer probe for research purposes is not easy. Moreover, a transducer probe with integrated analog pre-beamformers is not of interest for basic research, since access to the unaltered raw signals of each channel is desired.

In the scope of this thesis (Chapter 3), an ultrasound transducer probe for this purposes was developed based on an existing prototype in a collaborative project with Fraunhofer IBMT, Germany. The people contributing to the collaborative project at Fraunhofer where: Peter-Karl Weber (project lead at IBMT), Christoph Risser and Holger Hewener (hardware and software modifications on the DiPhAS for connecting the matrix probe), Christian Degel and Daniel Speicher (matrix transducer design), and Manfred Moses (technical assistance).

The DiPhAS imaging system with its support for matrix probes
was published with acknowledgments of the author in [20].

1.4.2 High-Performance Beamforming for 2D/3D

Being able to capture raw ultrasound data for 3D imaging is only half of the problem: One major challenge of fully-sampled beamforming for 3D ultrasound lies in the amount of processing required to reconstruct the reflectivity information from the captured signals of the transducer head. Especially demanding is managing the required pulse propagation delays for the beamforming process. Most conventional 2D systems use look-up-tables to store all delays for every channel and image point. Due to the vast amount of required delays, this is not possible for 3D systems: For an imaging system with 10 k transducer elements and 126×126×1000 image points, 159 G delay values are required. Assuming each delay is represented in 10 bit, storing 198 GB of delay information is manageable. Accessing the delays however is very challenging: For a target frame-rate of 15 Hz, the required access bandwidth is 24 Tbit/s.

The same issue occurs on a lower level for high frame rate 2D systems, which beamform only a 2D plane from a few hundreds of channels but at a very high frame-rate of several kilohertz. Since many different transmission patterns are alternated between the individual frames, many different sets of delays are required. Consequently, for an exemplary 128 channel system running at a frame-rate of 4 kHz, 2.1 G delays (128×1000×128×128) need to be accessed at a rate of 66 Gbit/s.

Due to the massive amount of data to be stored, for both 3D systems and high-frame-rate 2D systems, the delay values need to be brought to the beamformer processing unit from an off-chip high-capacity high-bandwidth memory subsystem. Transferring data at this rate will consume an enormous amount of power and deteriorate the systems compute efficiently. Thus, the delays have to be fully or partially be computed on-chip to fit within a reasonable power budget. Computing the delays on-chip is not trivial either. In order to get the required delay, the time-of-flight of the ultrasonic pulse from its emission center to the image point, and back to the receiving element needs to be computed, which involves the evaluation of several square-roots.

To alleviate the computational burden and to avoid the square-root evaluations, [30–33] propose to approximate the delay profiles,
required for each channel-scanline-pair, with a few parameters. This strategy, however, does not reduce the amount of data sufficiently to completely obviate an off-chip memory subsystem. Similar to [34], we propose to compute all delays online and on-chip, directly from the underlying geometry. The geometry can be parametrized with a few constants in the order of tens of kilobits easily, which is easily storable on an application-specific-integrated-circuit (ASIC). However, with this approach, the square-root computations can no longer be avoided and hence need to be handled efficiently in hardware.

In the scope of this thesis (Chapter 2), a scalable and reconfigurable hardware architecture for fully-digital integrated beamforming was be developed, which requires no external memory and processes the data at the information rate.

The architecture exploits several algorithmic innovations developed during the authors Master’s Thesis, which aim to reduce the computational effort for beamforming to a minimum. These algorithmic innovations are only briefly summarized in Section 2.3.4 and are also used in the ultrasound system described in Chapter 4.

This part of the work contributed in large parts to the two NanoTera.ch research projects UltrasoundToGo and BioDev.

1.4.3 Ultrafast Digital Probes

Increased digital computation capabilities at lower power consumption are also in high need for the next generation 2D ultrasound systems: In ultrafast imaging [35, 36] a 2D plane is reconstructed at a frame rate of several kilohertz, which is at the physical limit defined by the pulse propagation time. This modality opens up new diagnostic and imaging possibilities, like the sub-wavelength structural mapping and measuring the blood flow of deep vascular systems down to the capillaries by tracking micro bubbles [37], or breast tumor detection by extracting the tissues stiffness by tracking its movement upon mechanical excitation [38]. Advanced beamforming techniques are based on more accurate models for wave propagation and can thus achieve better imaging resolution [39]. However, up to now, real-time or near real-time performance of these new modalities is hardly reached, and large compute clusters are required to provide the results in a meaningful time. Accelerating these computations and ultimately
enabling them to run on lower cost hardware can be the tipping point for these modalities to break-through in everyday medical diagnostics.

Digital ultrasound probes are a promising architectural design choice to reduce system cost as they can leverage readily-available commodity hardware for processing: In contrast to their conventional analog counterpart, digital ultrasound probes integrate the analog front-end in the housing of the probe handle and provide a digital interface instead of requiring an expensive coaxial cable harness to connect. Current digital probes target the portable market and perform the bulk of the processing (beamforming) on the probe, which enables the probe to be connected to commodity devices such as tablets or smartphones running an ultrasound application to display the image and control the probe. Thermal constraints limit the number of front-end channels as well as the complexity of the processing. This prevents current digital probes from supporting advanced modalities such as Vector Flow or Elastography requiring high-frame-rate (HFR) imaging.

In the scope of this thesis (Chapter 4), we proposed a novel ultrasound system architecture, which uses an ultrafast digital probe. Our proposed ultrafast digital probe (LIGHTPROBE) is equipped with a high-speed optical link (>25 Gb/s) providing sustainable raw samples access to all channels, which allows the processing to be performed on the connected device without thermal power constraints. The new architecture allows building systems at lower cost, which at the same time provide sufficient computing capability for advanced modalities. As a proof-of-concept, we designed and built an entire ultrasound imaging system from scratch following our proposed architecture. To this end, we connected our LIGHTPROBE to a GPU-equipped PC and built the UTLALIGHT platform, the first to our knowledge ultrafast imaging system using a digital probe.

The LIGHTPROBE project was an independent follow-up project on the ULTRASOUNDToGO NANO-TERA.CH project. The NANO-TERA.CH NextStep Ph.D. tutoring program enabled the LIGHTPROBE project. The NextStep program allowed Ph.D. Students to apply for funding to finance collaborative follow-up projects. Fraunhofer IBMT contributed the piezo-electric transducer as well as initial technical guidance to the LIGHTPROBE project.
1.4.4 Advanced Imaging Modalities

As elaborated at the beginning of this chapter, several new advanced imaging modalities require an ultrasound system capable of ultrafast imaging. Our UltraLight system promises to reduce the cost of such a system to facilitate the adoption of these new modalities by the health-care services. In order to demonstrate that the UltraLight system in-fact can perform these novel advanced imaging modalities, a first advanced modality has been implemented during this thesis:

Ultrafast Bubble Tracking

Localizing and tracking of individual bubbles is an operation required for several new imaging modalities: In ultrafast ultrasound localization microscopy (uULM) individual bubbles injected into the blood flow are localized to map the vascular network below the diffraction limit. Additionally, by tracking the bubble movement over time, the streaming velocity of the fluid can be measured. For reliable localization and tracking of bubbles, an ultrafast imaging system providing a frame rate above 100 Hz over multiple seconds is required. Due to the enormous amount of data (>1 GB), data-rates (>10 Gbit/s) and the required processing performance, ultrafast imaging is currently supported only in large research systems or expensive high-end commercial systems.

As part of this thesis (Chapter 5), we demonstrate for the first time that high-speed bubble tracking can be implemented on a low-cost ultrasound system based on a digital ultrasound probe connected to a standard PC over a high-speed digital link. The ETH MSc student Pascal Armin Jud significantly contributed to this demonstration as part of his semester project, which was supervised by the author of this thesis.
1.5 Summary of Contributions

The most important contributions of this thesis are:

1. An energy-efficient full-integrated 10k channel 3D beamformer (Ekho) for medical ultrasound imaging has been designed. It demonstrates for the first time that it is possible to implement the entire 3-D delay and sum beamforming fully in digital, and on one single chip without requiring any power-hungry off-chip memories.

2. An optimized 1024-channel analog 4:1 multiplexer ultrasound probe (MuxHead) for 3D imaging has been built, and first test images were successfully acquired.

3. A 64-channel digital ultrasound probe (LightProbe) with an integrated RX/TX front-end and equipped with a high-speed optical link has been developed from scratch. It is the first ultrasound probe featuring an optical link and the first digital probe providing real-time raw sample access to the front-end data. The probe has the highest number of front-end channels (64) integrated within an ultrasound probe described in the literature and is the first digital probe supporting ultrafast imaging.

4. For the LightProbe an advanced power and thermal management system has been developed, which allows exploiting the heat capacity of the device to safely support the temporary operation of imaging modes whose thermal power dissipations exceeds the passive cooling capabilities of the device.

5. By connecting the LightProbe to a GPU-equipped PC, a complete ultrasound imaging platform (UltraLight) has been built. The platform is fully functional and provides real-time imaging using a pixel-based beamformer running on the GPU. Additionally, raw data can be captured at a kilo-hertz rate for offline processing.

6. Advanced imaging modalities have been implemented on the UltraLight system to demonstrate the flexibility and power of this novel software-defined system architecture.
1.6 List of Publications

Part of the material covered in this thesis has been published in the following publications and was adapted for Chapter 2, Chapter 4 and Chapter 5:


The following publications with contributions by the author treat related topics, but are not included in this thesis:


Further publications with contributions by the author not explicitly covered by this thesis are:


Chapter 2

Ekho: A Single-Chip 3D-Beamformer

2.1 Introduction

2.1.1 Motivation

Despite the upraise of modern medical imaging methods such as X-ray computed tomography and magnetic resonance imaging, ultrasound imaging is still popular and competitive due to its low cost, portability, and safety. Up to now, most ultrasound systems used are 2D systems, which have a probe that is applied to the body under investigation and provide an image of a cross-section. A trained operator sweeps the probe to get an impression of the 3D object. In 3D imaging, a volume is captured instead of only a cross-section. Compared to conventional 2D imaging, 3D imaging has several benefits [11]: Since a full volume is captured, the operator no longer has to sweep the probe to find a good view. This significantly shortens the examination time for the patient [54]. Moreover, it allows the examiner to review the captured data offline, using several different diagnosis approaches,
such as creating any cross section (also unconventional ones impossible to create with a physical probe), render the volume with different 3D rendering modes or do precise 3D volume measurements.

Ultimately, it has been shown that 3D imaging increases the diagnostic accuracy \[10\]. Also, photorealistic 3D renderings simplify patient communication, and remote diagnosis is enabled \[55\]. Moreover, future hand-held 3D devices will allow even nonspecialist personnel to look into the body, facilitating point-of-care diagnostic \[56\].

2.1.2 Challenges

Implementing a real-time 3D ultrasound imaging system is an extremely challenging task. Traditional 2D systems acquire images sequentially line-by-line: First, an ultrasonic beam focused along a line is emitted. If the beam hits a material transition, it is partially reflected. From the received echoes, the image points (focal points) along that line (called scanline) are computed (beamformed). Since it takes a particular time for the pulse to penetrate the tissue, reach the maximal desired imaging depth and propagate back to the transducer, the maximal pulse repetition rate is limited. To compose a 3D volume, many more scanlines are required than for a 2D plane. If all those scanlines were acquired sequentially, it would take an untenable amount of time to acquire the entire volume resulting in a very low frame rate. A known solution to this problem is to send out a less focused beam or even a divergent beam \[57\] and to compute several scanlines from just one isonification. This technique is called parallel receive beamforming and used in commercial 2D and 3D systems \[58, 59\].

Today’s high-end 2D systems operate with a probe composed of an array of up to 256 electro-acoustic transducer elements, that are fully-sampled. To achieve the same lateral resolution in 3D, a matrix array with a size of $256 \times 256$ elements is required. Connecting and processing so many elements is a problem: Since every image point depends on the signal of every transducer, the processing effort is increased significantly in 3D systems compared to 2D systems. Multiplexing \[60\] and sparsely sampled arrays \[2, 61\] have been proposed to only sample a subset of all transducers at a time. However, today’s commercial high-end 3D systems rely on fully-sampled arrays to overcome the poor image quality of sub-sampled arrays, namely their inability to emit and
capture sufficient acoustic energy as well as the image clutter they cause due to increased side-lobes [25]. These systems [23, 24] use massive analog preprocessing integrated directly into the probe to reduce the number of channels significantly. In an analog pre-beamforming stage a group of neighboring transducer elements is steered to a fixed direction with the help of analog delay lines, and a single output signal per group is produced. This reduces the number of outputs from several thousand to only a few hundred and hence simplifies the cabling of the probe to the processing device and alleviates the digital processing effort [26].

Recently, several fully-digital hardware architectures for fully-sampled matrix arrays have been proposed [30,56,60,62] that integrate, instead of only an analog pre-beamforming stage, a complete digital beamformer right into the transducer head. Moving all the processing into the digital domain promises higher profit from technology scaling enabling to build more flexible systems, and scaling them up to higher complexity levels. So far, no digital processing system supports fully-sampled matrix arrays with several thousands of elements on which all channels are sampled concurrently and continuously. Such a system has to process an incoming data-stream of several Tbit/s. For an exemplary system with 100 × 100 channels sampled at 16 MHz with 16 bit, a data rate of 2.56 Tbit/s has to be processed.

However, this is not the main challenge. The main challenge of a fully-sampled, fully-digital 3D beamformer lies in managing the required pulse propagation delays for the beamforming process. Most conventional 2D systems use look-up-tables (LUT) to store all delays for every channel and focal point, but this is not possible for 3D systems: The total number of required delays is enormous. For an imaging system with 10 k transducer elements and 126 × 126 × 1000 focal points, 159 G delays are required. Storing these delays is barely manageable. If each delay is represented in 10 bit, 198 Gbyte of memory are required. The bandwidth to access these delays is critical: For a frame rate of 15 Hz the required bandwidth to access the delays is 23.8 Tbit/s, which exceeds the input data rate of the receive elements (2.56 Tbit/s) by almost an order of magnitude. Supporting such an off-chip data rate will require a lot of power and a sophisticated memory system. Therefore the delays have to be fully or partially computed on-chip to fit within a reasonable area and power budget.
2.1.3 Contributions

As part of this thesis [40, 41], we designed a scalable hardware architecture optimized for large-scale 3D beamforming that performs the entire delay computation on-chip, without the need of external memory. Similar to [34], we propose to compute all delays online and on-chip, directly from the underlying geometry, which can be parametrized with very few constants. In our architecture, we only need around 36.6 kbit of constants to describe the entire geometry, which can be easily stored on-chip. Consequently, the entire delay computation needs to be performed on-chip as well. We minimize the computation effort by sharing as many computations between the individual channels as possible.

We designed two versions of this architecture. A first version as proof of concept, and a second, fully-complete version, which is ready for silicon implementation:

1. The first architecture version [40] targets a fully-sampled $100 \times 100$ matrix array and provides enough processing power to beamform an imaging volume of $128 \times 128 \times 1000$ points at 15 Hz. We estimate the area (19.1 cm$^2$) and power (480 W) cost of this design for a mature 130 nm technology, which would allow the co-integration with the analog high-voltage front-end electronics. Since this architecture could be easily distributed on multiple chips, an implementation in this mature technology would be split over several chips.

2. For the second version [41], we decided to switch to an advanced technology (28 nm SOI), that allows the integration of the entire beamforming in a single digital chip. This digital chip can then be connected to several dedicated high-voltage frontend chips.

Moreover, we ported the delay computation part of this architecture to FPGA and compared it with another delay computation implementation by our ULTRASOUNDToGO project collaborators. This comparison is not covered explicitly by this thesis but can be found in [46, 48].

In this chapter, we focus on the second version [41] and present the first complete pre-silicon implementation in 28 nm SOI of a fully-digital delay and sum beamformer for fully-sampled transducers arrays en-
2.1. **INTRODUCTION**

Enabling 3D beamforming on a single chip. The proposed implementation requires no external memory and could be integrated right into the transducer head. Our contributions are:

- We present a complete and fully-functional single-chip beamformer implementation, which includes all processing blocks (*pre-processing, delay computation* and *delay-and-sum beamforming*) required to compute image points out of the raw digital samples captured by the analog frontend.

- We minimize the computational effort for beamforming by designing a bandpass beamformer, which operates on critically sampled signals at all stages during processing and hence avoids unnecessary computations.

- We introduce a new delay index computation method with increased accuracy and reduced circuit complexity, which computes all required delays on-chip from only a few constants.

- We integrate a microcoded control unit into our beamformer for increased flexibility and to further reduce the number of required constants for the delay computation.

- We perform a rigorous error analysis to assess the beamforming quality of our optimized bandpass beamformer.

- We quantify the overall implementation loss of our optimized digital chip implementation.

- By having an entire and complete design, we can evaluate the power/area tradeoffs between the different components.

Our beamformer computes 298.1 MFP/s and up to 40 scanlines in parallel. It requires 1.68 cm$^2$ chip area and its core power fits in a 30.3 W power budget. This is a factor of 11× improvement in area and 16× in power compared to the first version [40]. Our beamformer architecture (Ekho) is scaleable: For a 100 channel 2D setup, Ekho requires 1.68 mm$^2$ chip area and dissipates 303.4 mW, while still providing 298.1 MFP/s.

The rest of the chapter is organized as follows: After related work (Section 2.2), we give a short introduction to ultrasonic imaging
and elaborate on the bandpass processing we applied to minimize the beamforming effort (Section 2.3). Subsequently, we elaborate our online delay computation (Section 2.4) and our beamformer architecture (Section 2.5). We summarize (Section 2.6) the detailed error analysis. Finally, we present the implementation results (Section 2.7) and close with comparison (Section 2.8) and conclusions (Section 2.9).
We define a system as fully digital if the received signal of every transducer element is processed digitally. If analog processing is used to reduce the number of digital processing channels, we define the system as partially-analog. Today’s commercial 3D systems are partially-analog systems: The iE33 xMatrix Echocardiography System from Royal Philips [23] uses analog pre-beamforming [25] in the head to reduce the number of channels for 3D imaging. The latest X6-1 xMatrix transducer head [63] for this system has 9212 active elements. The piezoelectric array is directly stacked onto a micro-beamformer ASIC, which contains for each of the 9212 microchannels a high-voltage transmitter, a receive amplifier, and an analog delay line. The Acuson SC2000 3D ultrasound system from Siemens [24] performs real-time volume imaging with the 4Z1c transducer head. This actively-cooled 1.5–3.5 MHz transducer head contains a PCB-Stack with 120 ASICs to perform partial beamforming in the head to reduce the number of cables [26]. To support a high frame rate, the Acuson SC2000 performs digital parallel receive beamforming of up to 64 beams on the pre-beamformed signals at a rate of 160 MFP/s [59].

The main disadvantage of partially-analog systems using pre-beamforming is decreased flexibility: A subarray can only steer to one direction at a time and therefore only allows to compute focal points in the area to which the subarrays are currently steered. A fully-digital solution does not share this limitation but needs to acquire large amounts of digital samples. Special sampling schemes, multiplexing, and duty-cycling are used to reduce that amount: The Sonic Window from Analogic [64] is a small fully-digital handheld device using a fully-sampled 3600 element transducer matrix to visualize C-mode (2D) images of the tissue right under the skin creating the illusion of looking through the skin [56, 62]. The device only captures four samples per shot and channel to minimize the processing effort and to fit the limited power budget of a portable device. This minimalistic sampling scheme is unsuitable for high-frame-rate high-resolution 3D imaging. The Sonic Millip3De [30, 60] is a die-stacked digital-front-end accelerator for 3D synthetic aperture imaging for a 128 \times 96 transducer matrix. To reduce the processing effort, the system uses twelve-fold
multiplexing to process only a subset of 1024 channels at a time. Also, the channels are only sampled in a burst-wise fashion with a duty cycle of around 2% to reduce the amount of raw data to process. It uses pre-computed section-wise quadratic approximations to obtain the delay profiles. Per channel-scanline pair, 9 constants are needed. The constants are provided by several dedicated LPDDR2-800 memories to achieve the required access bandwidth for their specification.

If all channels are sampled continuously and concurrently as in our proposed system, the current solutions either provide modest processing performance or are gigantic systems: The *CSC2032 RX Beamformer* from Cephasonics [34] is a single-chip 32 channel digital 2D/3D beamformer that beamforms 4 scanlines in parallel. Similar to our proposed beamformer, this device computes all required delays online and on-chip from the geometric specification. Similarly, [65] presents an ASIC design of a 16 channel single-chip 2D beamformer able to compute 4 scanlines in parallel. The delays are computed with a quadratic approximation using precomputed coefficients. To provide sufficient processing power for a 3D system with thousands of channels and a decent frame rate, hundreds of these chips are required.

The *SARUS* research system [66, 67] is an extremely flexible but huge experimental ultrasound system supporting the digital acquisition of 1024 independent receive channels with a sampling rate up to 70 MHz. It contains more than 320 Gbyte of RAM and uses 320 Xilinx Virtex-4 FPGAs for processing. In its current configuration [66], the system acquires data from all 1024 channels simultaneously, but can only process 256 in real-time. In this real-time operating mode, it produces 320 MFP/s. The propagation delays are computed online on the FPGAs recursively and iteratively [31, 32]. Per channel-scanline pair, 5 parameters (100 bit) are needed [33]. The parameters are stored on dedicated SRAMs.

In ultrafast 2D imaging [4, 35] or synthetic aperture imaging [36], the entire imaging plane is beamformed in every shot from an unfocused transmission. These imaging schemes provide frame-rates of up to several kilohertz and require equally high focal point and delay computation rates as for 3D imaging. However, far fewer channels need to be processed. Since Ekho can be scaled down to support such setups, a comparison with these 2D systems is appropriate. In [68] a 128 channel synthetic aperture 2D beamformer built from 4 Virtex-5
2.2. RELATED WORK

LX330 FPGAs is presented. 16 scanlines are computed in parallel at a sampling rate of 40 MHz. This results in 640 MFP/s. The delays are computed online but only for every 32nd focal point. The remaining delays are interpolated.

Today, software-based beamformers provide sufficient processing power for 2D imaging: Graphic processing units (GPU) are used for beamforming in both commercial [69] and research systems [19]. In the later, two GTX-480 GPUs are used for 2D synthetic aperture beamforming of 32 channels at a rate of 614 MFP/s. The delays are computed online on the GPUs. In [21] 4-cores of a single 8-core TMS320C6678 KeyStone DSP are used to beamform 20 MFP/s from 64 channels. All delays are pre-computed.

FPGA, GPU, and DSP beamformers provide enough processing power for 2D or partially-analog 3D systems. For fully-digital systems, a dedicated ASIC is still required for sufficient processing power at a reasonable power budget.
2.3 System Level – Beamformer Design

2.3.1 Ultrasound Imaging in a Nutshell

Medical ultrasound imaging [2, 70] visualizes changes in acoustic impedance of the tissue by emitting a high-frequency pulse with an array of piezoelectric transducers (Fig. 2.1a, 2.1b). If the pulse hits a material transition, it is partially reflected (scattered) due to the change of acoustic impedance (Fig. 2.1c). The reflected echos are then registered by the transducer array. In a computational process called beamforming, a map of the reflectivity of the volume under investigation \( h(\vec{r}_{FP}) \) is computed from the received signals (Fig. 2.1d): In order to do that, for a set of points in the volume, called focal points, all received signals \( r_i(t) \) need to be summed up according to the time it took the pulse to travel from its emission point \( \vec{r}_{TX} \) to the focal point \( \vec{r}_{FP} \) and back to the corresponding receiving element \( \vec{r}_{RX,i} \):

\[
h(\vec{r}_{FP}) = \sum_{i=1}^{N_{RX}} w_i r_i \left( t_{peak} + \frac{|\vec{r}_{TX} - \vec{r}_{FP}| + |\vec{r}_{FP} - \vec{r}_{RX,i}|}{c} \right). \tag{2.1}
\]

The number of receive elements is denoted with \( N_{RX} \) and \( t_{peak} \) is a global time offset to align the receive signals with the peak of the impulse response \( p(t) \) of the system (see Fig. 2.1d). The signal contributions are weighted with an apodization weight\(^1\) \( w_i \) to suppress side lobes caused by the aperture shape [2]. The pulse is assumed to travel with an average speed \( c \) of 1540 m/s. In a subsequent post-processing step, the output of the beamforming \( h(\vec{r}_{FP}) \) is converted into a displayable image. This step usually involves envelop extraction of the high-frequency pulse, transformation of the coordinate system (scan conversion), compression of the dynamic range and application of resolution enhancement techniques [2].

\(^1\)Currently Ekho supports only static apodization (\( w_i \) independent of \( \vec{r}_{FP} \)).
Figure 2.1 – Principle of ultrasound imaging illustrated in 2D: a) The imaging volume is defined by a set of focal points. These are arranged along radial rays (scanlines). Changes in acoustic impedance are modeled as discrete reflectors (scatterers). b) A pulse is emitted into the volume of interest. By delaying and weighting the transmit signal of each transducer element, a virtual point source placed behind the transducers is emulated. This placement creates a divergent beam that concentrates the transmitted energy on several adjacent scanlines. c) The pulse is partially reflected when a scatterer is hit. d) For each focal point, the received signals are added up according to (2.1). A scatterer will manifest itself by causing a constructive interference on the neighboring focal points proportional to its reflectivity. $p(t)$ is the impulse response of the system, which peaks at $t_{\text{peak}}$. This delay needs to be considered as well during beamforming.
2.3.2 System Overview

As illustrated in Fig. 2.2, we separate the capture of an ultrasound image in three stages: *signal acquisition*, *beamforming* and *post-processing* – each executed by a different part of the ultrasound imaging system: The *signal acquisition* stage involves the entire process of generating and sending out a pulse and receiving the echos, including the analog to digital conversion and some basic digital preprocessing such as filtering, demodulation, and decimation. This stage requires one or several high-voltage mixed-signal ASICs connected to the transducer elements. From the resulting digital samples, the focal point values are computed in the subsequent *beamforming* step. Due to the stream-line nature of this operation, this stage is highly suited for a digital ASIC implementation. Finally, a displayable image is created in the *post-processing* stage. A processor-based system is ideal for this task, due to its flexibility and the relaxed computational requirements. In this work, we mainly focus on the beamforming stage. We also address the interfaces to the neighboring stages, since reducing the data rate on them directly reduces the power required to transfer data.

**Figure 2.2** – Overview of an ultrasound imaging system. Only the receive path is shown. The analog signals received by the transducers elements are amplified (1.1) and converted to the digital domain (1.2) by the analog frontend (AFE). Digital demodulation and decimation (1.3) are performed within the AFE to reduce the data rate to transmit the signals to the beamformer. The beamformer computes the focal points and sends them to the post-processing system.
2.3. SYSTEM LEVEL – BEAMFORMER DESIGN

2.3.3 Beamformer Specifications

The Ekho hardware architecture presented in this paper has been developed to cover many ultrasound imaging scenarios, including 2D and 3D. Many settings can be adapted by changing configuration registers or altering the executed program code. Other settings require hardware modifications, easily obtained thanks to our parametrizable architecture. In the following, we focus on a very demanding 3D configuration, as a showcase for Ekho. We also briefly cover a simpler 2D setup. Ekho’s reconfigurability is further elaborated in Section 2.5.3 and App. 2.10. The selected 3D beamformer specifications are suitable for general purpose abdominal and gynecological applications: We target a $100 \times 100 = 10,000$-element $8\text{ MHz}$ bandwidth\(^2\) phased\(^3\) matrix array, which is superior to today’s commercial high-end systems, such as the iU22 from Philips, which operates with a transducer head with 9212 elements and only a 1-6 MHz broadband frequency range \([71]\).

Our targeted imaging volume is a spherical cut (Fig. 2.3a) with $72^\circ \times 72^\circ$ opening angle with a penetration depth of $500\lambda$. We want to reconstruct this volume with a rate of $f_V = 15\text{ Hz}$ at the resolution limit of the acoustical system. Our beamformer is designed to compute the focal points at a sufficient rate to meet these targets: To sample the volume at the resolution limit of the acoustic system, the scanlines need to be placed $\Delta \omega = \lambda/(2d) = 0.01\text{ rad}$ apart \([58]\), where $d$ is the size of the aperture, i.e., the size of the transducer array in our case. Given the $72^\circ \approx 1.26\text{ rad}$ opening angle, the entire volume is spanned with $126 \times 126$ scanlines (SL) all originating from the transducer matrix center. We define the $j,k$-th scanline as the set of points given by $\vec{r}_{\text{SL},j,k}(\rho) = \rho \cdot \vec{u}_{j,k}$ with $\rho \geq 0$ and where $\vec{u}_{j,k}$ describes the direction of the scanline. The focal points are placed equidistant with distance $\Delta \rho$ on the scanlines as illustrated in Fig. 2.3b. To cover the entire penetration depth, $500\lambda/(\Delta \rho)$ focal points are required per scanline. The $\ell,j,k$-th focal point is placed at

$$\vec{r}_{\text{FP}}[\ell,j,k] = \ell \cdot \Delta \rho \cdot \vec{u}_{j,k}, \quad \vec{u}_{j,k} = \begin{pmatrix} \sin(\theta_j) \cos(\varphi_k) \\ \sin(\varphi_k) \\ \cos(\theta_j) \cos(\varphi_k) \end{pmatrix}. \quad (2.2)$$

\(^2\)We assume a 4-12 MHz broadband frequency range.

\(^3\)\(\lambda/2\) element pitch
Figure 2.3 – Beamformer Specifications (simplified): (a) Transducer matrix with the imaging volume (red) with the two opening angles $\theta_{\text{OA}}$ and $\phi_{\text{OA}}$ and the diagonal opening angle $\Phi$ (blue). The point $\vec{r}$ is expressed in spherical coordinates (black). The grid (red) illustrates the subdivision of the imaging volume in subsections. Only a subsection (green) is isonified and the corresponding scanlines (black) are beamformed. (b) The focal points are placed on scanlines defined by a vector $\vec{u}_{j,k}$. The scanlines are separated by $\Delta \theta = \Delta \phi = \Delta \omega$ in polar and elevation angle, and by $\Delta \rho$ in radial distance.
As previously elaborated, the number of isonifications per volume is physically limited and several scanlines need to be acquired concurrently. The maximal pulse repetition rate (PRI) can be estimated as $f_{\text{PRI,max}} \approx \frac{c}{(2 \cdot 500\lambda)} = \frac{f_c}{1000} = 8 \text{ kHz}$, which results in $f_{\text{PRI,max}}/f_V = 533.3$ available shots per volume. Consequently, at least $126 \cdot 126 \cdot f_V/f_{\text{PRI,max}} = 29$ scanlines need to be acquired concurrently. We subdivide the entire volume in $441 < 533.3$ sections as sketched in Fig. 2.3a such that each section is covered by $6 \cdot 6 = 36 > 29$ scanlines. The sections are acquired and beamformed sequentially, one section at a time. We create a divergent beam by placing a virtual point source behind the transducer matrix [57] to concentrates the emission in that section.

### 2.3.4 The Case for Bandpass Processing

The electro-acoustic transducers used for ultrasound imaging have a bandpass transfer function. We model this property by assuming that they block all signal energies outside a single band of interest with bandwidth $B$ centered around frequency $f_c$. For typical transducers, $B \approx f_c$ can be assumed. The received signals $r_i(t)$ inherit this bandpass property, opening up several possibilities to exploit it throughout the system (see Fig. 2.4), to increase its power efficiency. Most systems use simple analog frontends (AFE) that sample the analog signal with a sampling rate much higher than the Nyquist rate $2f_c + B$ and send the digital samples unaltered over a high-speed LVDS link to the beamformer. However, the required data rate can be minimized by performing digital demodulation and decimation within the AFE [21, 72] to obtain complex-valued digital baseband samples at a rate of $B$ (Fig. 2.4.a-b). Additionally, disturbing imperfections, like inter-channel or dynamic intra-channel offsets are effectively removed by the implicit high-pass filtering during the digital demodulation and decimation. In many beamformer designs, the focal point spacing along the scanline $\Delta \rho$ is chosen with respect to the sampling rate $f_s$ of the AFE, i.e., $\Delta \rho = c/(2f_s)$. However, the beamforming output along a scanline
Figure 2.4 – Simplified processing chain in the beamformer. An illustrative spectrum of the signal at the different processing stages is shown as well. a) The raw analog signal enters the system. The useful information (1.2) is concentrated in a frequency band of width $B$ centered around $f_c$. Signal contributions (1.1) outside this band are considered noise. b) The analog frontend (AFE) digitizes the signal and performs digital demodulation and decimation to produce critically-sampled complex-valued baseband samples. c) The baseband signal is interpolated and converted to an analytic signal before beamforming. d) The analytic beamformed signal is undersampled such that non-destructive aliasing (2.1) occurs. e-f) The signal envelope is extracted from the analytical signal.
can be approximated in the far-field\(^4\) with

\[
h(\vec{r}_{SL,j,k}(\rho)) \approx \sum_{i=1}^{N_{RX}} w_i r_i \left( t_{\text{peak}} + \frac{2\rho}{c} + \gamma_{j,k,i} \right). \tag{2.3}
\]

Assuming this approximation holds, it is apparent, that the bandpass signal property is preserved during the delay and sum beamforming operation. Analytical sampling techniques can be used to undersample \(h(\vec{r}_{SL,j,k}(\rho))\) along \(\rho\) to reduce the number of focal points that need to be computed [73]. In order to do this, all input signals \(r_i(t)\) are converted first to their analytic representation\(^5\) prior the beamforming stage. Since our beamformer receives samples that represent the baseband representation \(r_{BB,i}(t)\) of \(r_i(t)\), the analytical signal representation

\[
r_{A,i}(t) = \exp(i2\pi f_c t) r_{BB,i}(t) \tag{2.4}
\]

is obtained with a simple multiplication (Fig. 2.4.c) Again assuming that the far-field approximation holds, the analytical property is preserved during beamforming, i.e., the new complex-valued beamformer output is an analytical signal representation with \(B\) and \(f_c\) scaled by \(2/c\). Analytical signals can be uniformly undersampled with a sampling rate equal to their bandwidth without causing undesired aliasing (Fig. 2.4.d). This reduces the required spatial sampling rate from \((\Delta \rho)^{-1} \geq (2f_c + B) \cdot 2/c\) real samples to \((\Delta \rho)^{-1} = B \cdot 2/c\) complex samples.

In ultrasound imaging with typically \(f_c \approx B\), this technique reduces the number of focal points by a factor of at least 3 and consequently reduces the number of required delays by the same factor. On the other side, complex-valued signals need to be processed, and a small error will be introduced due to the assumption that the bandpass property is preserved during beamforming. We analyze in Section 2.6 how severe this error is.

In postprocessing, the signal envelope cannot be taken directly from the undersampled analytical signal. The undersampled analytical

\(^4\)\(|\vec{r}_{FP}| \gg |\vec{r}_{TX}|\) and \(|\vec{r}_{FP}| \gg |\vec{r}_{RX,\forall i}|\)

\(^5\)The analytical representation \(x_A(t)\) of a real-valued signal \(x(t)\) is defined as \(x_A(t) = x(t) + iHx(t)\), whereas \(H\) is defined such that all negative frequencies of the complex-valued signal \(x_A(t)\) are zero [74].
CHAPTER 2. EKHO - 3D BEAMFORMER

signal first has to be converted into the baseband representation

\[ h_{BB}(\vec{r}_{FP}) (\rho) = h_A(\vec{r}_{FP}) (\rho) \exp(i2\pi f_c\rho^2/c) \]  

(2.5)

from which the envelope is extracted by computing the magnitude of
the complex valued signal (Fig. 2.4.d-f). Computing the magnitude of a
complex-valued signal involves multiplying with its complex-conjugate
counterpart. This self-multiplication causes a self-convolution in the
frequency domain and hence doubles the required sampling rate to
represent this signal. In order to avoid aliasing, the beamformer output
either needs to be upsampled (interpolated) by a factor of at least 2
along the radial distance or the distances between the focal points (\Delta \rho)
need to be reduced by a factor of at least 2, such the reflectivity map
is sampled at a sufficient rate. The same applies to the polar and the
elevation angle. Our chosen \Delta \omega requires upsampling by a factor of at
least 2 to avoid aliasing in the envelope extraction step. In ultrasound
imaging, this technique is also known as IQ interpolation [58]. Choosing
less focal points at the cost of interpolation during post-processing
reduces the beamforming effort, i.e., the number of required delays, by a
factor of 8. If the RF signals along the scanlines are required for further
processing, they can be retrieved without loss from the undersampled
analytical signal by applying an appropriate interpolation filter and
taking only the real-valued signal part.

To summarize: by using bandpass processing throughout the sys-
tem, we minimize the input data rate of the beamformer and reduce
the number of focal points and delays to compute by a factor of at
least 3 in radial and 2 in both axial directions each. This results in
a total reduction by a factor of at least 12. Also, the beamformer
output data rate is reduced by a factor of 6, assuming that complex
samples require around twice as many bits compared to real samples.
Considering that a considerable part of the power is dissipated in the
IO pad drivers, the data rate reduction at the interfaces translates
directly to significant power savings.

The final beamformer specifications for our exemplary 3D system
are summarized in Tbl. 2.1. Given these specifications, the number of
focal points per second our beamformer needs to compute is more than

\[ Q_{\text{spec}} = 126^2 \times 1000 \text{FP} \times 15 \text{Hz} = 238.1 \text{MFP/s} \]  

(2.6)

Without the bandpass processing it would be 2.9 TFP/s.
Table 2.1
System Specifications

<table>
<thead>
<tr>
<th>Speed of sound in tissue</th>
<th>$c$</th>
<th>1540 m/s</th>
</tr>
</thead>
</table>

**Transducer Head:**
- Transducer center frequency: $f_c$ = 8 MHz
- Transducer bandwidth: $B$ = 8 MHz
- Transducer matrix size: $100 \times 100$
- Wavelength: $\lambda = c/f_c = 0.1925 \text{ mm}$
- Transducer pitch: $\lambda/2$
- Transducer matrix dimensions: $d = 50\lambda = 9.63 \text{ mm}$

**Beamformer:**
- Imaging Volume ($\theta_{OA} \times \varphi_{OA} \times d_{pd}$): $72^\circ \times 72^\circ \times 500\lambda$
- Calculated Volumes per Second: $f_V$ = 15 Hz
- Focal Points per Volume: $126 \times 126 \times 1000$
- Angular Focal Point Spacing: $\Delta \theta, \Delta \varphi = \lambda/(2d)$
- Radial Focal Point Spacing: $\Delta \rho = c/(2B)$
- Required Focal Point Throughput: $Q_{\text{spec}} = 238.1 \text{ MFP/s}$
- Parallel Computed Scanlines: $6 \times 6 = 36$
- Shots for one Volume: 441 Shots
- Pulse Repetition Rate: $f_{PRI}$ = 6.6 kHz
- Volume splitting: $21 \times 21$ Sections

### 2.4 Online Delay Computation

In this section, we elaborate how the effort to compute the delays is minimized: In every shot, defined by a different emission origin $\vec{r}_{TX}$, a delay $t_{\text{tot}}(\vec{r}_{TX}, \vec{r}_{FP}, \vec{r}_{RX,i})$ is needed for each computed focal point $\vec{r}_{FP}$ and receive element $\vec{r}_{RX,i}$:

$$t_{\text{tot}}(\vec{r}_{TX}, \vec{r}_{FP}, \vec{r}_{RX,i}) = t_{\text{peak}} + \frac{d_{\text{TX\rightarrow FP}}}{c} + \frac{d_{\text{FP\rightarrow RX,i}}}{c}$$

For our system with $126 \times 126 \times 1000$ focal points and 10000 channels, 158.8 G individual delays are required in total for one volume. To avoid storing and accessing these delays, we propose to compute them online. To enable an on-chip memory only solution, we assume that only the 441 positions of the emission origins $\vec{r}_{TX}$, a few (252)
constants required to compute the focal points $\vec{r}_{FP}[\ell,j,k]$ with (2.2), and the coordinates of the receive elements $\vec{r}_{RX,i}$ are stored. Note that due to the symmetry and positioning, $2 \cdot 100$ values are sufficient to store all $\vec{r}_{RX,i}$. To efficiently compute the delays from the given constants, a lot of the required computations can be shared and used to contribute to several delays: During a shot, $\vec{r}_{TX}$ is constant. This means the distance $d_{TX \rightarrow FP}$ is only computed once for each $\vec{r}_{FP}$ and all $\vec{r}_{RX,i}$. For the distance

$$d_{FP \rightarrow RX,i} = \sqrt{(x_{FP} - x_{RX,i})^2 + (y_{FP} - y_{RX,i})^2 + (z_{FP})^2}$$

the computation of $\Delta A = (\Delta x)^2$ and $\Delta B = (\Delta y)^2 + (\Delta z)^2$ can be shared among the receive elements in the same row or column. $(\Delta z)^2$ is independent of the receive element because $z_{RX,i} = 0$ holds for all elements.

<table>
<thead>
<tr>
<th>Table 2.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required computations to calculate all delays required for the evaluation of a single focal point</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Quantity</th>
<th>$+/-$</th>
<th>$\times$</th>
<th>$\sqrt{}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\vec{r}_{FP}[\ell,j,k]$</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>$d_{TX \rightarrow FP}$</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>$\Delta A = (\Delta x)^2$</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>$\Delta B = (\Delta y)^2 + (\Delta z)^2$</td>
<td>$2 \cdot 100 + 1$</td>
<td>100 + 1</td>
<td>0</td>
</tr>
<tr>
<td>$\sqrt{\Delta A + \Delta B + d_{TX \rightarrow FP}}$</td>
<td>$2 \cdot 10000$</td>
<td>-</td>
<td>10000</td>
</tr>
</tbody>
</table>

Each focal point $\vec{r}_{FP}[\ell,j,k] = \ell \cdot \Delta \rho \cdot \vec{u}_{j,k}$ is computed only once per volume since a different section of focal points is evaluated in each shot. To do this efficiently, the $x$, $y$, $z$ components of all 36 vectors $\Delta \rho \cdot \vec{u}_{j,k}$ used in one shot are precomputed on-chip with the help of a small processor. A small program executed on this processor also controls which scanlines, i.e., which vectors, have to be computed in a specific shot. Note that if this information were stored naively in a table, this table would need 441 entries, one for each shot, and each entry would contain $6 \cdot 6 = 36$ indexes of at least 14 bit each to point to one of the $126 \cdot 126$ scanlines. This would require another 222 kbit of
memory, which we efficiently avoid with our small processor. Internally we neither compute distances \( d \) nor times \( t \). To avoid the division by \( c \) and the deduction of a specific sample index to select from the input stream, all values are mapped to the delay index domain, i.e., \( d \mapsto d \cdot f_D / c \) and \( t \mapsto t \cdot f_D \), where \((f_D)^{-1}\) is the resolution in time with which the delays are computed, and the samples are selected.

Tbl. 2.2 lists the required operations to compute all delays to evaluate a single focal point. The operations required to precompute \( \Delta \rho \cdot \vec{u}_{j,k} \) are neglected. It is apparent that the 10000 square-root evaluations are by far most critical. To realize an online on-chip computation, it is imperative to have a hardware-efficient square-root computation algorithm.

### 2.4.1 Hybrid Square-Root Computation (HSQRT)

We have developed a novel very efficient square-root computation algorithm: In general, the focal points are evaluated sequentially and in a specific order. This implies that there is some correlation in the result of two consecutive square-root computations. We propose a new method to compute the square-root that exploits this correlation to efficiently choose a smart starting point for the following standard iterative bit-by-bit calculation. To model this correlation, we assume that the absolute difference between two consecutive results is bounded by \( \delta \), i.e., \( \left| \sqrt{x_n} - \sqrt{x_{n-1}} \right| < \delta \). The iterative bit-by-bit calculation is based on [31–33]. Our method operates on fixed-point number representations and computes \( y = \sqrt{x} \) up to \( B_I \) integer bits and \( B_F \) fractional bits. The computation is exact in the sense that, assuming a quantized fixed-point input \( x \), the quantized\(^6\) exact result \( y = Q_{B_I,B_F} \{ \sqrt{x} \} \) is indistinguishable from our result \( \hat{y} = \text{HSQRT} \sqrt{x} \). A piece-wise constant approximation of the square-root (Fig. 2.5) delivers the starting point for the bit-by-bit method. The sections are placed equidistant in \( y \) with distance \( 2^{N_b} \geq \delta \) and are defined by their starting point \( s_i \) and their value \( c_i = \sqrt{s_i} \). \( N_b \) is a design parameter of our method, which will be discussed later. By storing the index \( k \) of the section used in the previous computation, we can determine the current section by only comparing \( x \) with \( s_k \) and \( s_{k+1} \): If \( x \) is smaller

\(^6\)Rounding to the next value, not truncation.
than $s_k$ the index is decremented, if it is larger than $s_{k+1}$ the index is incremented, otherwise the index is kept\(^7\). This section tracking algorithm implements a mapping function $f : X \rightarrow K$ that returns the corresponding section index $k$ for a given input value $x$. This mapping function can be expanded to support a wider set of applications\(^8\) as we will elaborate later. From $k$, a piece-wise constant approximation $\hat{y}_0 = c_k$ is obtained. Note that since $c_k = k \cdot 2^{N_b}$, this approximation returns a preliminary result of the $B_I - N_b$ most significant bits of the final result $\hat{y} = \sqrt{x}$. In the subsequent $m = N_b + B_F$ steps of the bit-by-bit calculation, the remaining bits are computed. In each step, depending on the sign of $D_i = (\hat{y}_{i-1} + 2^{N_b-i})^2 - x$, the value $(2^{N_b-i})$ corresponding to the currently determined bit is added or not to $\hat{y}_{i-1}$ to get $\hat{y}_i$. To avoid the squaring in each step, not only $\hat{y}_i$ but also the difference

$$\Delta_i = \hat{y}_i^2 - x, \quad \Delta_0 = s_k - x$$

(2.9)

is propagated through the iteration. Using $\Delta_i$, the $D_i$ can be computed using shift and add operations only, which enables a very efficient

\(^7\)At the cost of more comparisons, changes of $x$ exceeding more the one section can be supported to allow $2^{N_b} < \delta$.

\(^8\)If larger jumps than $\delta$ occur in a sporadic but deterministic manner, additional logic can handle these special cases.
implementation in hardware:

\[ D_i = (\hat{y}_{i-1} + 2^j)^2 - x, \quad j = N_b - i \]  
\[ = \hat{y}_{i-1}^2 - x + 2 \cdot \hat{y}_{i-1} \cdot 2^j + 2^{2j} \]  
\[ = \Delta_{i-1} + 2^{j+1} \hat{y}_{i-1} + 2^{2j}. \]  

After \( m \) steps we obtain \( \hat{y}_m \). The error \( \sqrt{x - \hat{y}_m} \) is confined in the interval \([0, 2^{-B_F})\) and has a non-zero mean. The error interval is moved to \([-2^{-(B_F-1)}, 2^{-(B_F-1)})\) by adding \( 2^{-B_F} \) to \( \hat{y}_m \) if \( D' = (\hat{y}_m + 2^{-B_F-1})^2 - x \geq 0 \). This makes the result indistinguishable from the true value \( \sqrt{x} \) quantized to \( B_F \) fractional bits. Our modified bit-by-bit calculation method based on [31–33] is show in Algo. 1.

1: function ISQRT(\( \Delta_0 = s_k - x, \quad \hat{y}_0 = c_k \))
2: for \( i \leftarrow 1, m \) do
3: \( j \leftarrow N_b - i \)
4: \( D_i \leftarrow \Delta_{i-1} + 2^{j+1} \hat{y}_{i-1} + 2^{2j} \)
5: if \( D_i < 0 \) then
6: \( \hat{y}_i \leftarrow \hat{y}_{i-1} + 2^j \)
7: \( \Delta_i \leftarrow D_i \)
8: else
9: \( \hat{y}_i \leftarrow \hat{y}_{i-1} \)
10: \( \Delta_i \leftarrow \Delta_{i-1} \)
11: end if
12: end for
13: Rounding Correction:
14: \( j \leftarrow -B_F - 1 \)
15: \( D' \leftarrow \Delta_m + 2^{j+1} \hat{y}_m + 2^{2j} \)
16: if \( D' < 0 \) then
17: \( \hat{y} \leftarrow \hat{y}_m + 2^{-B_F} \)
18: else
19: \( \hat{y} \leftarrow \hat{y}_m \)
20: end if
21: end function

**Algorithm 1** – Modified ISQRT Algorithm
2.5 Beamformer Architecture

Fig. 2.6a illustrates the top-level diagram of the beamformer (BF) ASIC architecture. This architecture is an improvement of [40, 75]. In every shot, all focal points within a section are beamformed. The computation order of these points has an impact on the data and compute coherency that can be exploited: Within the section, we compute the focal points on a per-nappe basis as shown in Fig. 2.6b. We use the term nappe to describe the set of all focal points equidistant from the transducer array center, i.e., \(|\vec{r}_{FP}| = \text{const} \). On each nappe, the focal points are computed sequentially, line-by-line. This order assures that the assumed correlation exploited in our HSQRT algorithms holds. Also, the computation of the nappe is aligned to the pulse propagation such that only a small interval of the received signal needs to be buffered for the computation. If the focal points were computed on a per-scanline basis, the entire signals would have to be stored. The received signal of each transducer is buffered and processed by an assigned beamformer channel (BFC). Every BFC selects and contributes a value to the adder tree to form the currently computed focal point. The computed focal points are then sent to the post-processing system. The global control and the shared computations are performed in a separate unit.

2.5.1 Global Control and Computations

The global control and computation unit controls the operation of the entire beamformer. It also performs all shared delay index computations, i.e., it computes \(\Delta A, \Delta B\) and \(d_{TX\rightarrow FP}\). As part of this task it incorporates a small programmable unit (uC-Engine) that plans the order in which the sections are computed and precomputes the corresponding vectors. The bit-widths were chosen such that the final delay index is off by at most 1 interpolated sample with a probability < 30%. This stringent quality metric assures that our architecture can cope with many imaging scenarios without requiring any re-evaluation, unlike a metric that only considers the final output image in a particular application.

The dataflow of the global control and computation unit is depicted in Fig. 2.7: The System Control handles the top level control of the
2.5. BEAMFORMER ARCHITECTURE

Figure 2.6 – Top level architecture of the our Beamformer ASIC. (a) A beamformer channel (BFC) (1.1) is assigned to each transducer element and selects the value to contribute to the currently computed focal point. The global control and computation unit (1.2) performs the shared computations and contains the programmable unit for the scanline geometry computations. An adder tree (1.3) sums up the outputs of the all BFCs forming the final focal point. (b) Computation order of the focal points within a subsection (green).

beamformer. It handles the startup procedure, starts the beamforming process when the isonification begins and reports the completion of the beamforming computations. During startup and when the computation of a new shot starts, it activates the uC-Engine, which then computes the required vectors for the next shot. The vectors are double buffered, such that one bank is available for the uC-Engine to store its result and the other one can be accessed to compute the coordinates of the focal points required for the currently computed shot. Since one focal point is computed per cycle, a second low-latency Fast Control unit is employed to control the computations.

The uC-Engine is a straightforward instruction based computation unit. Its main goal is to provide the possibility to describe the higher-level control flow of the beamformer with a program to avoid overspecific control units and huge constant tables that instruct which operation, i.e., which scanline, is computed in which shot and in which order (see Lst. 2.1). The 16 bit instructions are stored in a small dedicated memory. The uC-Engine has two data paths: One main 8 bit
path for general purpose programming, with a small 16-word register field, that is partially mapped to the outer control units to influence the control flow. A small arithmetic logic unit supports additions and comparisons to allow the programming of a simple control flow with conditional branches and address computations. The second data path is a tailored 19-22 bit data path to compute the vector values. The trigonometric functions are implemented as a configurable 22 bit look-up-table. The read address for the LUT is provided directly by the ALU output of the first datapath. Currently, we support $2^8 = 256$ LUT values. For our current setup we store all 126, $\sin \theta_j$ and $\cos \theta_j$ values\(^9\) $(2 \cdot 126 = 252)$ in the LUT. We do not exploit the sine and cosine symmetries to reduce the number of required constant. The LUT output is fed to an instruction-controlled multiply-store unit to compute the vector values. The computed vector values can then be written out to the scanline vector bank using a special instruction. The write address is again provided directly by the ALU of the first data path. The programmable uC-Engine can be easily extended and allows flexible in-system reconfiguration of the beamformer.

In total, 36.6 kbit of constants are required to compute the delays: $2 \cdot 126 \cdot 22$ bit for the sine and cosine LUT-entires, $441 \cdot 64$ bit to store information related to the shots like the $\vec{r}_{TX}$ positions, $200 \cdot 9$ bit for the $\vec{r}_{RX}$ and $64 \cdot 16$ bit to store the instructions for the uC-Engine.

### 2.5.2 Beamformer Channel

The architecture of the beamformer channel (BFC) is shown in Fig. 2.8. The task of the BFC is to convert the incoming baseband samples into the analytical representation by a complex modulation and return the value that corresponds to the delay associated with this channel. A $2 \times 16$ bit data-path\(^{10}\) is used for the complex signals, which is typically supported by an ultrasound AFE with integrated demodulation and decimation [72]. For sufficient beamforming quality, the delays are computed with sub-sample precision, and the corresponding values need to be interpolated. We compute the delays with a precision that relates to a sampling rate of $8f_c$, which lies in the suggested span

---

\(^9\)Since $\theta_k = \varphi_j \forall j = k$, $2 \cdot 126$ values are sufficient.

\(^{10}\)An aggressively optimized 3D system may be able to operate with a lower bit-width (Section 2.6). Our conservative choice enables wider compatibility.
2.5. BEAMFORMER ARCHITECTURE

Figure 2.7 – Global Control and Computation Unit: Several control units are implemented hierarchically: A System Control unit communicates with the outside world. A programmable uC-Engine controls which scanline is computed in which shot and order. It does so by precomputing the 36 scanline orientations $v_{j,k} = \Delta \rho \cdot \mathbf{u}_{j,k}$ used in one shot and storing them in a double buffered memory. A Fast Control unit organizes the one-per-clock-cycle computations, i.e., controls $j, k$ and $\ell$. The coordinate of the currently computed focal point $\mathbf{r}_{FP}$ is obtained by multiplying the iterating $v_{j,k}$ with $\ell$. The $\mathbf{r}_{TX}$, which is fixed for one shot, is provided by a programmable look-up table, implemented as a single port memory. The forward delay $d_{TX\rightarrow FP}$ is only computed once. The shared computations for the backward delay ($\Delta A_r$, $\Delta B_c$) are performed $100 \times$ each; once for each row and column, respectively.

of $4f_c$ to $10f_c$ [73, 76]. The computed sample is multiplied with a configurable static weight (apodized) and then passed on to the adder tree, which is implemented as a binary tree to minimize latency.

Interpolation & Modulation

Our target sample selection precision is $8f_c$. The input sampling rate is $B$. Hence the required interpolation factor is $Q = 8f_c/B = 8$. To have sufficient interpolation quality we consider the $M = 6$ neighboring samples, which corresponds to a filter order of $M \cdot Q = 48$ with respect to the output. Since our beamformer computes PBF = 36 scanlines in parallel from one shot using the same input data, and the focal point density along a line $(\Delta \rho)^{-1} = B \cdot 2/c$ in space corresponds to the sampling rate $B$ of the baseband samples, each interpolated and modulated sample is used $PBF/Q = 4.5$ times on average. To avoid unnecessary computations, we pre-compute all interpolated and modulated samples and store them in a ring buffer.
The execution order of the complex modulation and the interpolation can be exchanged: If the interpolation is performed before the modulation, \( M \) real-complex multiplication are required per complex output sample for the interpolation and 1 complex-complex multiplication is needed for the modulation. This equals to \( 2M + 4 \) real-real multiplications. If the modulation is performed first, the modulation is not required in our case \((f_c = B, \exp(i2\pi f_c k/B) = 1, \forall k)\) but then a complex-valued bandpass interpolation filter is required, which needs \( M \) complex-complex multiplications per complex output sample. This equals to \( 4M \) real-real multiplications. To reduce the number of multiplications, we perform the interpolation before the modulation. The interpolation is implemented with a polyphase filter, and the modulation uses a LUT to store the complex carrier coefficients.

Note that other systems [33] compute the interpolated value on the fly from the two neighboring samples. Different from those systems, we critically sample the input signal. Hence, just considering the two neighboring samples will not provide sufficient interpolation quality. Therefore, more samples (\( M \)) have to be considered, which means that “on-the-fly computation” not only comes at a higher computational cost but also a drastically increased bandwidth to the BFC buffer.

### Buffer Size and Arrangement

The pre-interpolated, pre-modulated analytical samples are stored in a buffer. The computation of the focal points is aligned to the propagation of the ultrasonic pulse, i.e., all focal points on the \( \ell \)-th nappe \((|\vec{r}_{FP}| = d_\ell = \ell \cdot \Delta \rho)\) are computed within \( 2\Delta \rho/c \) seconds. To compute the focal points on the \( \ell \)-th nappe, the samples corresponding to the interval

\[
\left[ \min_{|\vec{r}_{FP}|=d_\ell} d_{tot}, \max_{|\vec{r}_{FP}|=d_\ell} d_{tot} \right]/c
\]

(2.13)

with \( d_{tot} = |\vec{r}_{TX} - \vec{r}_{FP}| + |\vec{r}_{FP} - \vec{r}_{RX,i}| \) are needed in the buffer. As shown in Fig. 2.8b, the required samples lay within a slowly shifting time span around

\[
t_{ref}(d_\ell) = d_{ref}(d_\ell)/c = (2d_\ell + d_{TX})/c
\]

(2.14)
2.5. BEAMFORMER ARCHITECTURE

![Beamformer Channel (BFC) Architecture Diagram]

a) BFC Architecture

![Buffer Size Plot]

b) Required Samples in Buffer

Figure 2.8 — Beamformer Channel (BFC) architecture: (a) The BFC Architecture. (b) The plot shows qualitatively the required time span of samples in gray needed in the buffer for beamforming depending on the depth $d_\ell$ of the currently computed nappe. Before the computation can start, the buffer needs to be filled with samples covering the timespan $t_{\text{prefill}}$.

with $d_{\text{TX}} = |\vec{r}_{\text{TX}}|$. Using $d_{\text{ref}}(d_\ell)$, the interval of required samples can be expressed as

$$[d_{\text{ref}} - \Delta d_{\text{min}}(d_{\text{ref}}), d_{\text{ref}} + \Delta d_{\text{max}}(d_{\text{ref}})]/c \ .$$

(2.15)

In our setup,

$$\max_{d_{\text{ref}}} \Delta d_{\text{min}}(d_{\text{ref}}) \leq \hat{d} \quad \text{and}$$

$$\max_{d_{\text{ref}}} \Delta d_{\text{max}}(d_{\text{ref}}) \leq \sin(\Phi/2)\hat{d} + d_{\text{TX}}(1 - \cos(\Phi)) \ .$$

(2.16)

(2.17)

both hold for all receive channels, where $\hat{d} = \sqrt{2}d/2$ is the maximum distance of a receive element from the transducer matrix center point, $d$ is the width and height of the transducer matrix and $\Phi = 2 \arccos \left(\cos^2(\theta_{OA}/2)\right)$ is the diagonal opening angle (see Fig. 2.3a and Tbl. 2.1). Considering that we only compute a subset of all focal
points on a nappe per shot defined by a specific virtual source placement \((d_{\text{TX}} \leq \tan(\Phi_{\text{section}}/2) d/2)\), we can tighten the second bound to

\[
\max_{d_{\text{ref}}} \Delta d_{\text{max}}(d_{\text{ref}}) < \sin(\Phi/2)\hat{d} + d_{\text{TX}}(1 - \cos(\Phi_{\text{section}})) \quad (2.18)
\]

\[
< \sin(\Phi/2)\hat{d} + d/2. \quad (2.19)
\]

This results in a required buffer size of

\[
(\hat{d} + \sin(\Phi/2)\hat{d} + d/2)Qf_c/c = 697.1 \text{ samples.} \quad (2.20)
\]

Note, that the buffer size of every BFC could be adapted to the assigned receive element \(i\) by choosing \(\hat{d}_i = |\vec{r}_{\text{RX},i}|\). We did not do this in this design. Also, note that these bounds are not tight. Having an analytical expression for the required buffer size enables easy hardware reconfiguration for specification changes. We have chosen a buffer size of 1024 samples mainly to support also other placements of the virtual sources resulting in different constraints and more importantly to avoid a costly modulo computation to get from the sample selection index provided by the delay computation to a ring buffer address. The buffer is implemented with a 1024 × 32 bit single port memory. Read and write cycles are alternated with 5/6 read duty cycle. At the beginning of the shot, the buffer needs to be prefilled with \(\Delta_{\text{max}}\) samples covering the timespan \(t_{\text{prefill}}\) as illustrated in Fig. 2.8b.

**Local Delay Index Calculation**

In each BFC the delay computation is finalized by the *local delay index computation* (LDIC) unit. This unit computes

\[
d_{\text{tot},i} = \sqrt{\Delta A + \Delta B + d_{\text{TX} \rightarrow \text{FP}}}. \quad (2.21)
\]

The square-root computation is performed by our HSQRT algorithm, which exploits the correlation between two subsequent computed values to efficiently find the starting point for the subsequent iterative computation. The HSQRT implementation shown in Fig. 2.9a is performed in two stages: Stage I realizes the mapping function \(f : X \rightarrow K\) and Stage II implements the ISQRT computation.

To cover the entire imaging volume, \(B_I = 13\) integer bit are computed. One fractional bit \((B_F = 1)\) is computed to reduce fixed-point
2.5. BEAMFORMER ARCHITECTURE

![Diagram](image.png)

**Figure 2.9** – Local Delay Index Calculation (LDIC): (a) HSQRT circuit: The input $x$ (27 bit) is compared with $s_{k'}$ and $s_{k'+1}$ with $k'$ the section index from the previous square-root calculation. Depending on the comparison result, the section index for the current calculation $k$ is determined and the corresponding $s_k$ and $c_k$ are fed as a starting value to the ISQRT pipeline. Note that by definition, the three LUTs providing the $s_i$ values (twice for comparison and once for the starting value) implement nothing else than a squaring function and a fixed bit-shift, i.e., $s_i = (i \cdot 2^{N_b})^2 = i^2 \cdot 2^{2N_b}$. The LUT providing $c_i$ is nothing but a hard-wired bit-shift, i.e., $c_i = i \cdot 2^{N_b}$. The iterative square-root computation is completely unrolled in the hardware implementation. (b) shows the $i$-th ISQRT stage with $j = N_b - i$.

calculation errors in the subsequent addition with $d_{TX\rightarrow FP}$. This restricts the parameter $N_b$ to the set $\{-1, \ldots, 13\}$. By choosing this parameter, the circuit complexity can be traded off between Stage I and Stage II: $N_b = -1$ relates to a LUT-only solution without the Stage II and $N_b = 13$ to a fully-iterative implementation requiring no Stage I. Also, to assure that our proposed mapping function works, the correlation constraint $2^{N_b} \geq \delta$ has to be fulfilled. A small $N_b$ will cause large LUTs in Stage I, but requires very few iterative stages in Stage II. A large $N_b$ reduces the LUT sizes in Stage I, but will require more iterative stages in Stage II. The main concern with more
iterative stages are the required pipelining registers to keep the longest path in the ISQRT from getting critical.

Our evaluations have shown, that for an implementation in the mature UMC 130 nm technology [77] or on an FPGA, these pipelining registers are dominant. In these cases, the minimal area is achieved with an $N_b$ around 8. For this choice, the correlation constraint is violated in a sporadic but deterministic manner when larger jumps occur as the focal point computation jumps to the next line or nappe. These cases can be easily handled by extending our mapping function $f : X \rightarrow K$ such that the section index $k$ is stored at the beginning of a line or nappe and reloaded when the computation jumps to a new nappe or line. Having this additional circuit, the change between two subsequent computations can be bounded with $\delta = \max(\Delta \rho, \Delta \omega \cdot d_{pd}) \cdot f_D / c$, which holds during normal focal point computation advancement. In the advanced technology used for this implementation, the pipelining register price is very low, such that the minimal area is achieved with $N_b = 12$, which would not require any advanced mapping function that can compensate for jumps violating the correlation constraint. The cost for this advanced mapping function is very low since most of it is handled by the fast control unit in the global computation and calculation unit. We kept this logic for Ekho’s FPGA compatibility. Note that these jumps could also be solved by increasing the number of comparators in the mapping function to correctly handle jumps larger than $2^{N_b}$. However, this would also mean that more LUTs are required and, also that, this is less scalable because the extended jump distance cannot be easily estimated, since it heavily depends on the size and aspect ratio of the subsections of the imaging volume computed per shot.

For the first computed focal point, the section index needs to be known. In the worst scenario, a starting index for each scanline-receive-element pair is required. For 10k channels and 441 shots, this would mean another 4410000 constants, because a different set of scanlines is computed in each shot. However, this can be avoided for most cases: One solution is to let the mapping function converge to the correct section by itself by feeding it with the input of the first focal point. This can be done during the buffer pre-fill time. The solution we used for this design is to choose $2^{N_b} > \hat{d} \cdot f_D / c + \Delta \rho$, such that the very first focal point always lies within the first section so we can initialize
2.5. BEAMFORMER ARCHITECTURE

the mapping functions of all BFCs with 0 in every shot.

The ISQRT hardware implementation is straightforward and illustrated in Fig. 2.9b. The circuit is designed such that the bit-widths are minimized in every stage. Note that in each stage the bit-width to represent $\hat{y}_i$ increases by one, and the bit-width to represent $\Delta_i$ decreases by one. Further, the addition of $2^j$ and the mux in the $\hat{y}$ path do not really exist, in fact, $\hat{y}_i$ is only concatenated with the inverted sign bit of $D_i$.

2.5.3 System-Level flexibility

Our beamformer architecture can be easily adapted for different applications (see also App. 2.10): Changing the number of channels just implies changing the number of BFC channels. The focal point computation rate can be increased by replicating parts of the global control and computation unit and the buffer and LDIC parts in the BFCs. Note that our beamformer can be converted to a 256 channel 2D beamformer by just reducing the number of BFCs and changing the software. Different isonification strategies are easily supported, i.e., to support plane wave imaging, only the $d_{TX\rightarrow FP}$ computation part in the global control and computation unit needs to be replaced. Keep in mind that the timespan available in the buffer might need to be adapted. A further extension allows placing the scanlines in an arbitrary fashion in space (not restricted to be originating from the transducer center). Such geometries are supported with minor changes in the global control and computation unit. However, the buffer sizes need to be adapted accordingly. Furthermore, it requires to use the iterative-only design of the HSQRT implementation, since no computation correlation can be assumed anymore. Note that with an analog pre-beamforming stage, this high system flexibility cannot be reached: The subarrays can only be steered to a single direction at once, thus limiting the space in which focal points can be computed from the pre-beamformed signals.
2.6 Error Analysis and Imaging Results

Since Ekho is only a beamformer and not a complete ultrasound imaging system, the error analysis is focused on showing that an implementation of Ekho does not insert considerable implementation loss to a perfect reference beamformer evaluating Eq. (2.1) precisely. This is the standard methodology to assure that only the beamformer implementation is evaluated and not the entire imaging system, which mainly governs the overall image quality. Ekho has 3 main error sources:

- **Bandpass beamforming**: As elaborated in Section 2.3, the beamformer output is only approximately a bandpass signal, and thus analytical undersampling introduces errors.

- **Delay index computation**: The delay index quantization and its fixed-point precision calculation will result in non-ideal sampling instants of the receive signals.

- **Fixed-point computations**: The receive signals are processed, i.e., interpolated, modulated, apodized, added, with fixed-point precision. The rounding to a certain precision introduces an additional quantization error.

2.6.1 Methodology

To assess the bandpass beamforming error and the fixed-point computations error we resort to a 2D beamformer with a linear transducer array. This was done for several reasons: First, we want our beamformer to be applicable to 2D systems as well. In 2D systems, the SNR of the receive signals is better due to the larger transducer element size [26]. Therefore the beamformer must operate with higher bit-width in 2D systems in order not to introduce any significant errors. Also, in the 2D case, only 100 channels are added up to form one focal point. The adding up of 10’000 channels in the 3D case will more likely average out randomly introduced computation errors. Assessing the quality in 2D will assure that our beamformer provides sufficient quality also for this scenario. However, only assessing the 2D case, will not validate the required precision in the adder tree for 3D. Nevertheless, since the adder tree covers only 1.3% of the area for $N_{RX} = 100$, its precision
2.6. ERROR ANALYSIS AND IMAGING RESULTS

can be increased with minimal impact. Furthermore, the simulation
time for a 2D system is orders of magnitudes lower than to simulate
the complete ultrasonic system for the 3D case.

Our 2D system has inherited the parameters of our 3D system
listed in Tbl. 2.1. Instead of a matrix array, only a linear transducer
array with 100 elements is considered. The imaging volume is reduced
to a plane $(72^\circ \times 500\lambda)$, and instead of $21 \times 21$ shots (each shot used
to compute $6 \times 6$ scanlines) only $21$ shots are performed, and in each
shot, $6$ scanlines are computed. The physical part of the ultrasonic
system is simulated with Field II [78,79]. The double-precision golden
model and the bit-true model of the RTL implementation are run in
MATLAB 8.3 2014a by MathWorks. The bandpass property of the
transducer is modeled by setting the impulse response to those of a
180 order FIR bandpass filter with cutoff frequencies at $f_c \pm 0.8 \cdot B/2$.
The factor 0.8 is to assure that our definition of bandwidth is satisfied,
i.e., no signal energy remains outside $B$. The volume is excited with
a 4-period sine pulse shaped with a Hanning window. The virtual
sources for each shot are placed at $-6.28\, cm \angle \theta_{C,i}$ with $\theta_{C,i}$ the center
angle of the computed section in that shot. To the simulated signals,
a $1/t^2$ time gain compensation profile is applied. For beamforming,
a static Hanning apodization is applied. In the post-processing, the
focal points are interpolated, scan converted and log-compressed.

2.6.2 Bandpass beamformer loss

In order to assess the errors introduced by the analytical undersamp-
ing, we compare the beamforming results of a bandpass beamformer
with $(\Delta \rho)^{-1} = B \cdot 2/c$ and $\Delta \omega$ scanline separation, with an lowpass
beamformer with $(\Delta \rho)^{-1} \geq (2f_c + B) \cdot 2/c$ and $\Delta \omega/2$ scanline sepa-
ration. The output of both beamformers, i.e., the focal points values
are shown in the first and last column of Fig. 2.10. Each pixel rep-
resents a focal point. Note that the lowpass beamformer requires $6$
times more focal points than the bandpass beamformer in the 2D
case. The two middle rows in Fig. 2.10 show a preliminary image
after focal point interpolation\textsuperscript{11} and envelop extraction. Note that
there is no noticeable difference in the major features, except for some

\textsuperscript{11}Interpolation only applied to the bandpass BF output.
differences in the background noise: We use the Peak Signal-to-Noise Ratio (PSNR) to quantify the difference. The PSNR between two images \((x_{i,j}, y_{i,j})\) is computed with:

\[
\text{PSNR}_{\text{dB}} = 20 \log_{10} S - 10 \log_{10} \left( \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} (x_{i,j} - y_{i,j})^2 \right) \tag{2.22}
\]

with \(S\) the maximal occurring value. Note that the beamforming process introduces speckle noise, visible in Fig. 2.10b, and that a difference in noise should not be considered as an error in beamforming. With a PSNR of 45.0 dB on the linear image, we conclude that bandpass beamforming only creates minor image deteriorations and can be safely applied.

### 2.6.3 Delay index accuracy

The delay index computation is assessed for the 3D system specified in Tbl. 2.1 by comparing the true delay with the output of our hardware implementation: Remember that we compute the delays in the delay index domain, i.e., \(t \mapsto t \cdot f_D\), in which the final value \(k = t \cdot f_D\) has to be rounded to an integer value to select from the buffer the interpolated sample closest to the true delay value. We compare the true delay without any quantization \(k_{\text{tot,true}} = t_{\text{tot}} f_D / c\) with the quantized true value \(Q(k_{\text{tot,true}}) = \text{round}(t_{\text{tot}} f_D / c)\) and the output of our hardware implementation \(k_{\text{tot,HW}} = \text{round}(k_{\text{TX→FP,HW}} + k_{\text{FP→RX,HW}})\).

To reduce the simulation effort, 100 receive channels were chosen randomly from all 10k channels. The delay computation was simulated for all 441 shot, i.e., for all \(126 \times 126 \times 1000\) focal points. Three figures of merit were chosen: The mean absolute error, the max absolute error, and the error variance. The errors are reported in Tbl. 2.3. We report the delay computation accuracy for the forward \((k_{\text{TX→FP}})\) and backward \((k_{\text{FP→RX}})\) as well as the total \((k_{\text{tot}} = k_{\text{TX→FP}} + k_{\text{FP→RX}})\) delay.

#### Accuracy of forward/backward delay computation

Both the forward \((k_{\text{TX→FP}})\) as well the backward \((k_{\text{FP→RX}})\) delay are computed with 1 bit fractional precision. The mean absolute error and the variance of the error (see Tbl. 2.3 and Fig. 2.11) match the
2.6. ERROR ANALYSIS AND IMAGING RESULTS

Figure 2.10 – Beamforming quality comparison: All images show absolute values. Fig. 2.10a show the linear output and Fig. 2.10b the same outputs log compressed to 80 dB. BP FP shows the output of the bandpass beamformer \((\Delta \rho)^{-1} = 2B/c\) and RF FP the output of the RF beamformer \((\Delta \rho)^{-1} = 2(2f_c + B)/c\). Each pixel represents the value of one focal point. BF post and RF post show a preliminary image after interpolation and envelop detection. Note they only show differences in the background speckle noise but for BF post 6 times less focal points were required to produce the image. The PSNR between the two preliminary images are 45.0 dB for the linear image and 23.4 dB for the log-compressed image respectively. Note that the later value highly depends on the applied log-compression.
Figure 2.10 – The log compressed version of Fig. 2.10a.
2.6. ERROR ANALYSIS AND IMAGING RESULTS

Table 2.3
Delay index accuracy: Simulation results (upper part) and expected quantization errors (lower part).

| Comparison           | mean $|e|$ | max $|e|$ | var $e$ |
|----------------------|-------|--------|--------|
| $k_{TX\rightarrow FP, HW}$ vs $k_{TX\rightarrow FP, true}$ | 0.111 | 0.432 | 0.018 |
| $k_{FP\rightarrow RX, HW}$ vs $k_{FP\rightarrow RX, true}$ | 0.128 | 0.385 | 0.022 |
| $k_{tot, HW}$ vs $k_{tot, true}$ | 0.344 | 1.284 | 0.104 |
| $k_{tot, HW}$ vs $Q(k_{tot, true})$ | 0.273 | 1 | 0.199 |
| $k_{FP\rightarrow RX}$ vs $Q(k_{FP\rightarrow RX, true})$ | 0.125 | 0.25 | $(2^{-1})^2/12 = 0.021$ |
| $k_{tot, true}$ vs $Q(k_{tot, true})$ | 0.25 | 0.5 | $(2^0)^2/12 = 0.083$ |

expected value from quantization quite well (0.111, 0.128 vs 0.125) and (0.018, 0.022 vs 0.021). This implies that our computed value considering all the fixpoint computations including the square root computation has the same error statistics as if the value were computed with double point precision and quantized afterward. Therefore we conclude that we have no implementation loss.

**Total delay computation error**

The total delay is the sum of the forward and backward delay ($k_{tot} = k_{TX\rightarrow FP} + k_{FP\rightarrow RX}$). Since this operation is performed in fixed-point precision and the results have to be rounded to an integer sample selection index, an additional error is introduced. If we compare the true value with our hardware implementation, we get a mean error of 0.344 samples, and if we compare it with the quantized true value, we see that our computation is never off by more than one sample.

**Periodic errors**

As elaborated in [80], periodic errors of the focusing delays over the receive elements cause undesired side-lobes. The introduction of periodic errors is assessed as in [33] by computing the power spectral density (PSD) of the backward delay computation error over all receive elements in a row for a single scanline and averaging the PSD over all points along the scanline. As shown in Fig. 2.12, the introduced error is approximately white with a small (−8 dB) mean
offset. The noise level matches the expected one from quantization at $10 \log (2^{-1})^2/12 = -16.8$ dB. Since there are no peaks at non-zero frequency exceeding the quantization noise floor, we conclude that there are no periodic errors introduced and thus no undesired sidelobes created by the delay calculation.

To summarize: Our forward and backward delay computation to 1 bit fractional precision is considered exact in the sense that the introduced errors are indistinguishable from the quantization error of the output value. The total delay computation is off by at most one sample with 27.3% probability.

### 2.6.4 Fixed-point implementation loss

The overall implementation loss was assessed by comparing the Point Spread Function (PSF) between our fixed-point precision hardware implementation and a double-precision reference beamformer. Both
2.6. ERROR ANALYSIS AND IMAGING RESULTS

![Error Power Spectral Density over RX–Channels](image)

**Figure 2.12** – Power spectral density of the error $k_{FP\to RX,\text{true}} - k_{FP\to RX,\text{HW}}$ (blue) of the scanline $\theta_j = \varphi_k = 0.5750$ over all 100 receive elements in one row at $\vec{r}_{RX,i} = (x_{RX,i}, 4.81 \times 10^{-5}, 0)$ averaged over all points along the scanline. The red line indicates the expected error for the given quantization.

operated on the same input data with the same section-wise imaging strategy and were performed with the 2D setup elaborated before. The contour plot of the PSF (Fig. 2.14) shows no significant difference until the noise floor is hit at $-60$ dB. The projected PSF reveals a slightly increased side-lobe at $\pm 3.3^\circ$ of 7 dB. Furthermore, we assess the 2D image of a synthetic kidney (Fig. 2.13): The PSNR of the image generated by our implementation is 37.9 dB with respect to the reference image. This relates to a hardly perceptible implementation loss.
Figure 2.13 — Image comparison between the double-precision reference beamformer with $f_D = 120$ MHz and our hardware implementation. An artificial kidney scan was simulated with field II. A kidney scattering map (kidney example) available on [81] was used to create an artificial phantom consisting out of 400'000 scatteres. The image is log-compressed and shows 60 dB.
2.6. ERROR ANALYSIS AND IMAGING RESULTS

Figure 2.14 — Simulated point spread function (PSF) of a single scatterer positioned at (0, 0, 48.1 mm) indicated by the black plus. The top contour plot shows the $\rho, \theta$ cross-section of the PSF (lines at $-5, -10, -20, -30, -40, -60$ dB) and the bottom plot a RMS projection on the angle thereof. The two double-precision golden model with $f_D = 120$ MHz (solid) and $f_D = 64$ MHz (dotted) delay resolution are shown in blue. Our hardware implementation is shown in red. The black crosses in the bottom plot indicate the center angles of the 21 shots. In each shot 6 scanlines were computed.


2.7 Pre-silicon Implementation Results

The beamformer was implemented for the STMicroelectronics 28 nm SOI process and synthesized\textsuperscript{12} for a 400 MHz clock frequency. The core supply voltage is 0.6 V and the supply voltages for the memory periphery and array are 0.7 V and 0.8 V respectively. It operates with a clock frequency of $f_{\text{clk}} = 384$ MHz for the following reasons: First, it is a multiple of the rate $B$ with which the input samples arrive at the beamformer. It is further equal to the product of $Q \cdot M \cdot B$ such that two multipliers are sufficient for the interpolation circuit. The data rate of the interpolated samples is then $f_{\text{clk}}/M$, which results in a BFC buffer write/read ratio of $1:5$. This relates to an instantaneous focal point computation rate of $Q_{\text{inst}} = f_{\text{clk}} \cdot 5/6 = 320$ MFP/s. One nappe needs to be computed in $2\Delta\rho/c = 1/B$ seconds. Therefore $f_{\text{clk}}/B = 48$ cycles are available per nappe, of which only in 40 cycles a sample can be read from the single-port buffer. Thus 40 > 36 scanlines can be computed in parallel. Considering the maximal pulse repetition rate (PRI) of the system ($f_{\text{PRI,max}} = 7.59$ kHz) and the time to pre-fill the buffers with $\Delta_{\text{max}}$ samples, the beamformer can compute $Q_{\text{eff}} = (1-6\Delta_{\text{max}} f_{\text{PRI,max}}/f_{\text{clk}})Q_{\text{inst}} = 298.1$ MFP/s in average, which exceeds our minimal specifications of $Q_{\text{spec}} = 238.1$ MFP/s, required to meet the target specifications.

2.7.1 Estimated Area

The beamformer was synthesized for a subset of all channels, i.e., $N_{\text{RX}} = \{4, 9, 16, ..., 100\}$ receive channels to determine how the area of the different units grows with $N_{\text{RX}}$. In a second step, we used this knowledge to extrapolate the results to $N_{\text{RX}} = 10000$ channels. Note that these channel-subset beamformers are fully functional for our intended setup: Several instances of them can be distributed across several chips, and their outputs could be summed up in the post-processing step, resulting in an indistinguishable output compared to the single

\textsuperscript{12}The synthesis and the back-annotated power estimations were performed with Design Compiler 2013 by Synopsys. The RTL and gate-level simulations were performed with ModelSim 10.0d by Mentor Graphics. The floorplan was created with Cadence Encounter 12.0. MATLAB 8.3 2014a by MathWorks was used to run the golden model to verify the hardware implementation.
chip implementation. The synthesis results are shown in Fig. 2.15.

![Cell Area](image)

**Figure 2.15** – Post-synthesis area results for BFs with different number of BFC instances ($N_{RX}$). Note that the BFCs grow dominant in area with increasing channel count.

The total area of all BFCs, the Adder Tree (AT) and the BF Config (which contains storage elements for the static apodization weight and the positions of the receive elements) grow linear with $N_{RX}$. The fast computing part of the global control and compute unit (GC compute), i.e., the part that computes $\Delta A$ and $\Delta B$, grows with the square-root of $N_{RX}$. The remaining parts of the global control and compute unit, e.g., the programmable unit and all the remaining constant memories remain constant, since they do not depend on $N_{RX}$. The numerical results and gate equivalents (NAND2X3) for 100 channels are listed in Tbl. 2.4 ($A_{100}$). Since the overall beamformer area grows linearly with $N_{RX}$, we can upper bound the required area by

$$A(N_{RX}) \leq A_{100} \frac{N_{RX}}{100}, \quad N_{RX} > 100. \quad (2.23)$$

With $A_{100} = 1.68 \text{ mm}^2$, this result in a total area of $A(10 \text{ k}) \approx 1.68 \text{ cm}^2$. Note that the area required to store the constants for the delay index computation grows irrelevant. So does the area for the uC-Engine. Further note that the area required for the non-BFC parts covers only 5.6% for 100 channels. This means there is not much area loss if the BF would be distributed over several chips and a single chip implementation is not required to profit from the shared computations.

A lot of related work [30,33] focuses on minimizing the computation effort per channel. However, as we have just demonstrated, if computations can be shared between channels, they rapidly grow irrelevant.
CHAPTER 2. EKHO - 3D BEAMFORMER

Table 2.4
Area and Power Estimations for a Beamformer with $N_{RX} = 100$.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Area $[\mu m^2]$</th>
<th>GE</th>
<th>Power $[mW]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF (total)</td>
<td>$A_{100} = 1.675 \text{ M}$</td>
<td>3.421 \text{ M}</td>
<td>$P_{100} = 303.4$</td>
</tr>
<tr>
<td>BFCs</td>
<td>1.582 \text{ M}</td>
<td>3.231 \text{ M}</td>
<td>264.561</td>
</tr>
<tr>
<td>Adder Tree</td>
<td>22.678 \text{ k}</td>
<td>46.320 \text{ k}</td>
<td>4.211</td>
</tr>
<tr>
<td>BF config</td>
<td>10.227 \text{ k}</td>
<td>20.888 \text{ k}</td>
<td>1.427</td>
</tr>
<tr>
<td>GC (with SHT mem)</td>
<td>37.596 \text{ k}</td>
<td>76.789 \text{ k}</td>
<td>8.897</td>
</tr>
<tr>
<td>GC compute</td>
<td>21.939 \text{ k}</td>
<td>44.810 \text{ k}</td>
<td>3.863</td>
</tr>
</tbody>
</table>

The only thing that matters for the delay computations is the number of computations performed individually within each channel. This strategy is also valid for 2D beamformers with only 100-300 channels. The BFCs dominates the circuit complexity. Fig. 2.16 illustrates how the BFC is composed of its sub-blocks. The interpolation and modulation part depends on the transducer bandwidth since this defines the data rate to be processed in this part. In the first architecture version [40], this was less dominant because we considered a transducer head with half the bandwidth, i.e., 4 MHz instead of 8 MHz. The per channel part of the delay index computation only covers 9.5% and has around the same size as the two multipliers required for the apodization. The buffer is the most dominant part.

![Figure 2.16](image-url) – Estimated Area and Power distribution in within one BFC.

We show the proposed chip-level floorplan in Fig. 2.17a and the floorplan of a $5 \times 5$ channel BFC cluster in Fig. 2.17b. The BFC cluster can be placed and routed separately and instantiated in a grid over the entire BF chip. The input pads can be realized as micro-pads placed on top of the BFC cluster. The shared computations are performed in the center part of the chip and distributed outwards.
2.7. PRE-SILICON IMPLEMENTATION RESULTS

Figure 2.17 – Floorplan of the single-chip beamformer: (a) shows the chip-level overview with 10x10 BFC clusters, each cluster (1.1) contains 100 channels. The global compute and calculation unit (2.1) and the adder tree (green, 2.2) along with the other auxiliary circuits are placed plus-shaped (light blue) in the middle [area exagerated]. The shared computations are performed along the branches of the GC unit and distributed by row (blue, 1.2) and column (red, 1.2) to the BFC clusters. To avoid long path propagation delays to distribute the shared computations along the chip, we successively delay the computations in the BFC clusters by inserting pipelining registers at the yellow borders (3.1). This latency needs to be compensated in the adder tree. (b) one BFC cluster with $5 \times 5$ channels.

2.7.2 Estimated Power

We report the post-synthesis power simulation results in Tbl. 2.4 ($P_{100}$). The power numbers for a $N_{RX} = 100$ channel beamformer were obtained as follows: A $N_{RX} = 4$ channel beamformer was simulated, and for all circuit blocks the power dissipation was computed. For each block, the power was extrapolated to $N_{RX} = 100$ according to the increase in area of that block. The reported numbers include the clock tree but not the power dissipated in the I/O pads. Again we upper bound the power dissipation with

$$P(N_{RX}) \leq P_{100} \frac{N_{RX}}{100}, \quad N_{RX} > 100.$$  (2.24)
With $P_{100} = 303.4\,\text{mW}$, we obtain a total core power dissipation of $P(10\,\text{k}) \approx 30.3\,\text{W}$. The power dissipated for DRAM memory accesses is equally important: With our on-chip delay computation we avoid accessing delays from an off-chip memory: 500\,W would be required to access all delays from an LPDDR3 memory assuming an access datarate of 23.8\,Tbit/s and an energy usage of 20\,pJ/bit for the memory module and PHY\textsuperscript{13}.

\textsuperscript{13}Using the Micron System Power Calculator [82] and [83].
2.8 Comparison with Related Work

The processing effort for a beamformer depends on the number of focal points it produces per second $Q_{out}$ and over how many channels $C$ it needs to sum up contributions. Accordingly, we define that $C$ beamforming operations are required to produce one focal point. In Tbl. 2.5 we compare the processing throughput $\Phi = C \cdot Q_{out}$ expressed in beamforming operations per second (BOPS) for different beamformers reported in literature. Also, we report the power dissipation $P_{\text{diss}}$ and power efficiency in BOPS per Watt. Our achieved processing power and power efficiency are unmatched by any other system to the best of our knowledge. This is reached not only because of the advanced technology, but also thanks to our approach to reducing the delay computation effort jointly over all channels and avoiding off-chip accesses, instead of optimizing only on a per-channel basis as in related work. Furthermore, due to the bandpass processing, unnecessary oversampling is avoided, and a larger volume is covered with the same number of focal points. However, this advantageous feature of our design is not considered in the power efficiency metric. Sonic Millip3De [30], the only other fully-integrated solution, is the only system that processes more channels. It uses an entirely different imaging strategy than we do: Instead of concentrating the transmission into a subvolume and doing many shots at a high rate (6.6 kHz) while acquiring all channels, this system performs only very few shots with a completely un-concentrated transmission at a very low rate (192 Hz) and acquires only 1/12 of all its channels at a time. Therefore, the processed input data rate ($4096 \times 1024 \times 192 \times 1$ Hz $= 0.81$ GS/s) is orders of magnitude smaller than what our system processes ($8$ MS/s $\times 10000 = 80$ GS/s). To compensate for the little energy received from those unconcentrated transmissions, 16 shots are performed and summed up. To get the delays, two piece-wise quadratic approximation sections are used per scanline receive element pair for each transmission. To achieve the required access bandwidth of 6.2 Gbyte/s and storage capacity for the constants, 6 2 Gbit x16 LPDDR2-800 memories are used. Even by comparing with their power estimate for an 11 nm technology node, we are $2.3 \times$ more power efficient in a 28 nm technology node. This is achieved, by completely avoiding external memory accesses as well
as our optimized beamformer architecture. SARUS exceeds our focal point computation rate but processes fewer channels. The delay index computation method [33] needs 5 parameters requiring 100 bit of storage per channel and scanline. Considering our specifications with $126 \times 126 \times 10000 = 163 \text{M}$ channel-scanline pairs, this method requires 1.98 Gbyte of memory. We only need 36.6 kbit. Compared to the first architecture version [40], we have improved the area and power consumption by $11 \times$ and $16 \times$ respectively.

**Table 2.5**

Comparison with related work. Top 3D, bottom 2D systems.

<table>
<thead>
<tr>
<th>System</th>
<th>$C$ [cha]</th>
<th>$Q_{out}$ [MFP/s]</th>
<th>$\Phi$ [GBOPS]</th>
<th>$P_{diss}$ [W]</th>
<th>$\Phi/P_{diss}$</th>
<th>Tech.</th>
</tr>
</thead>
<tbody>
<tr>
<td>[30] Millip3De</td>
<td>12288</td>
<td>10.2$^a$</td>
<td>126</td>
<td>2.94$^b$</td>
<td>42.9</td>
<td>11</td>
</tr>
<tr>
<td>[66] SARUS</td>
<td>256$^c$</td>
<td>320</td>
<td>82</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>v1 [40]</td>
<td>10000</td>
<td>269</td>
<td>2693</td>
<td>480</td>
<td>5.6</td>
<td>130</td>
</tr>
<tr>
<td>v2 [41]</td>
<td>10000</td>
<td><strong>298</strong></td>
<td><strong>2981</strong></td>
<td><strong>30.3</strong></td>
<td><strong>98.4</strong></td>
<td><strong>28</strong></td>
</tr>
<tr>
<td>[65] ASIC</td>
<td>16</td>
<td>40</td>
<td>0.64</td>
<td>n/a</td>
<td>n/a</td>
<td>180</td>
</tr>
<tr>
<td>[19] dual GPU$^d$</td>
<td>32</td>
<td>614</td>
<td>20</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>[68] 4 FPGA$^e$</td>
<td>128</td>
<td>640</td>
<td>82</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>[21] DSP$^f$</td>
<td>64</td>
<td>20</td>
<td>1.28</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>v2 for 2D [41]</td>
<td>100</td>
<td>298</td>
<td>29.81</td>
<td>0.303</td>
<td>98.4</td>
<td>28</td>
</tr>
</tbody>
</table>

$^a$ A volume with $4096 \times 50 \times 50$ focal points is computed at 1 Hz.

$^b$ Reported power (11 nm), without transducer and ADC.

$^c$ 1024 channels are supported but only 256 for real time processing.

$^d$ For beamforming: two GTX-480 GPUs.

$^e$ Four Virtex-5 LX330.

$^f$ 4-cores of a 8-core TI TMS320C6678 KeyStone DSP.

Compared to state-of-the-art 2D systems our $N_{RX} = 100$ beamformer provides a processing throughput that exceeds a dual GTX-480\textsuperscript{14} GPUs setup [19] by almost 50% and is with 303 mW core power dissipation suited for mobile applications. Compared to the quad Virtex-5 LX330 FPGA solution [68] our integrated approach provides only $2.75 \times$ less processing throughput at a much smaller form factor and power budget. The 8-core TI Keystone DSP based solution [21] targets low-power, low-cost ultrasound devices. However, since our

\textsuperscript{14} One GTX-480 can be expected to consume 200 – 300 W under full load.
2.8. COMPARISON WITH RELATED WORK

processing throughput is $23 \times$ higher and the DSP will most likely consume an order of magnitude more power, it will not compete with the power efficiency of our integrated approach. Compared to the ASIC beamformer [65] including an on-chip delay computation based on a per-channel quadratic approximation, we are about $10 \times$ more area efficient\textsuperscript{15} in terms of technology normalized GBOPS/mm$^2$.

\textsuperscript{15}Circuit area (180 nm) without apodization weight computation: 14.4 mm$^2$. Comparison: $(29.81/1.68)/(0.64/14.4) \cdot (28/180)^2 = 9.7$. 
2.9 Conclusions and Outlook

In this work, we have presented a pre-silicon implementation of the first fully-digital single-chip beamformer for 3D ultrasound. The beamformer requires no external memory, uses only 1.68 cm$^2$ chip area and dissipates 30.3 W. This is enabled by processing the data at the information rate using bandpass processing and reducing the computation effort for the beamforming delays. The later is achieved by a clever sharing of computations in combination with a programmable unit. This unit additionally enables software system configuration. Our parametrizable beamformer can be used for high-performance 2D imaging as well, at a very low power budget of 303 mW.

In the future, our single-chip beamformer can be integrated directly in the transducer head, enabling a new class of fully-digital 2D and 3D ultrasound systems. These systems will provide more flexibility in a smaller form factor. Our chip can replace the analog pre-beamforming stage of today’s 3D systems, which will ultimately allow building high-performance portable 3D ultrasound systems.
2.10 Appendix - Ekho Reconfigurability

The Ekho architecture is highly parameterizable: The system configuration is specified in a file similar to Tbl. 2.1 and the entire hardware including the software configuration files are composed semi-automatically. If the specifications are outside the parametrization domain of the hardware, an error is reported. The center frequency $f_c$ and bandwidth $B$ can be modified easily, but so far not without minor manual hardware changes. Reductions by integer multiples could be done conceptually without hardware changes, but the required modifications in the control circuits of the interpolation and modulation circuit are not yet in-system configurable. Furthermore, the coefficient LUTs for the interpolation and modulation are currently hardwired but could be made configurable with minor overhead. The number of receive elements and their placement is handled by the parametric hardware description. Currently, planar rectangular phased arrays with any number of channels and aspect ratio are supported. This includes 1D arrays for 2D imaging systems. The imaging volume can be expanded at the cost of frame rate or resolution. Both can be done by simple reprogramming without hardware modifications. The volume computation rate (frame rate) can be increased by reducing the numbers of focal point per volume (software) or by increasing the focal point computation rate. The focal point computation rate can be increased by replication. More complex scanline placements and different isoionification schemes can be adopted by expanding the global control and computation unit.

To show how easily Ekho can be reconfigured, we briefly outline how the 3D beamformer elaborated in the paper was reconfigured into the 2D beamformer used in Section 2.6: There are both hardware and software modification required. Afterward, further configuration options for Ekho are elaborated.

2.10.1 Ekho for 2D

Hardware modifications

For the 2D setup, a linear array of transducers is sufficient. Therefore, we changed the transducer matrix configuration in the parametric
CHAPTER 2. EKHO - 3D BEAMFORMER

architecture from $N_{cha,x} = 100, N_{cha,y} = 100$ to $N_{cha,x} = 100, N_{cha,y} = 1$. This automatically composed a 100 channel beamformer.

**Configuration/software modifications**

To adopt the new geometry, the configuration files to be loaded into the beamformer had to be updated: These contain the element positions and apodization weights for all channels and the positions of the virtual sources used in each shot. Furthermore, a few control registers had to be configured differently in the global control unit. Finally, the software for uC-Engine was adapted for 2D imaging and loaded into the beamformer. Lst. 2.1 shows a code snippet used for 3D imaging.

```plaintext
[...]
mpredi 0 # load dr (LUT pos 0)
add wy0 cnty tmp # compute cos(wy) index
mprodadd cosptr tmp # multiply with cos(wy)
add wx0 cntx tmp # compute cos(wx) index
mprodadd cosptr tmp # multiply with cos(wx)
mwriteclr PBFc 2 # store result in buffer
addi PBFc 1 PBFc # incr PBFc
eq PBFc SPBF tmp # if PBFc >= SPBF
jpc tmp SHOTDONE # -> shot done
addi cntx 1 cntx # incr cntx
ex cntx Swx tmp # if cntx >= Swx
jpc tmp RESETCNTX # -> next line
jp LP1 # else -> continue
RESETCNTX:
clr cntx # cntx = 0
addi cnty 1 cnty # incr cnty
jp LP1 # continue
[...]
```

**Listing 2.1** – uC-Engine code snippet: Line 2-7: Computation of the z-coordinate ($\Delta \rho \cos(\theta_j) \cos(\varphi_k)$) of the scanline vector $\vec{v}_{j,k}$. Line 8-10: Check if all scanline vectors for one shot have been computed. Line 11-18: Counter logic to compute the scanline vectors within a sub-section in a predefined order. Since the vectors are written linearly into the double-buffer, the computation order will be maintained, when the shot is beamformed.
2.10. APPENDIX - EKHO RECONFIGURABILITY

Performance

Since we modified the hardware only in the number of channels, the instantaneous focal point computation rate $Q_{\text{inst}} = 320 \text{ MFP/s}$ is preserved, and still 40 scanlines can be computed parallel. Again the effective performance $Q_{\text{eff}}$ will be a bit lower, due to the time to pre-fill the buffers with the data from the next shot.

Changing the Imaging Strategy

In Section 2.6 a fine-grain section-wise imaging strategy is used. Other strategies can be implemented without hardware changes: By decreasing the field of view to $\theta_{\text{OA}} = 68.6^\circ$ ($\varphi_{\text{OA}} = 0^\circ$), 120 scanlines are required to cover the imaging plane. This allows a coarse-grain section-wise imaging strategy with only three shots, which pushes the frame rate to approximately $f_{\text{PRI,max}}/3 \approx 2.7 \text{ kHz}$.

2.10.2 Further Configuration Options

If fully coherent data from one time instant is desired, the entire plane, i.e., all 120 scanlines, should be computed in parallel from one single shot. This is achieved by two possible modifications: a) If the target frame rate remains $2.7 \text{ kHz}$, the acquisition system can capture one shot and idle during the time it would take for the other two shots. The beamformer is continuously processing and thus provides already enough processing power. However, the computation is no longer aligned with the propagation of the ultrasonic pulse: During an acquisition burst, new data is produced faster than it is processed. Thus more buffering is required. One can either increase the size of the BFC buffer or add an additional buffer before the interpolation stage. The later requires less capacity. Additionally, to support 120 scanlines, the size of the double buffers for the scanline vectors has to be increased accordingly. b) If the entire plane has to be computed in each shot captured with $f_{\text{PRI,max}} = 8 \text{ kHz}$, the processing power is no longer sufficient, and the beamformer needs to be replicated partially such that the simultaneous computation of three focal points is supported, i.e., $Q_{\text{inst}} = 960 \text{ MFP/s}$ is achieved. This is easily done by replicating threefold the shared computation part of the global computation unit, the adder tree, as well as the BFC buffers and the LDIC units.
Chapter 3

MuxHead: A 1024-element Probe for 3D Imaging

3.1 Introduction

3.1.1 Motivation

2D systems use an ultrasound probe with a linear array of transducer with up to 300 elements [71] to capture an image with good resolution and field-of-view. 3D systems require a matrix of transducer elements to capture the required signals for 3D volume reconstruction. To provide a field-of-view and resolution comparable to 2D systems, the matrix must contain thousands of elements (32 × 32 and more).

In most 2D probes, each transducer element is directly connected over a micro-coaxial cable to the backend system. For 3D matrix probes with more than a thousand transducer elements, connecting every element individually over a cable to the backend system is unfeasible, since it would require an intangible amount of micro-coaxial cables. The resulting cable harness would be very expensive and bulky, and very hard to handle by the ultrasound operator. Thus
any ultrasound probe for 3D imaging must perform some sort of pre-processing within the probe to simplify the cabling design. Commercial matrix transducer probes such as the X6-1 by Philips [63] with 9212 elements thus implement analog-prebeamforming [25] in the probe to reduce the number of cables. Analog-prebeamforming\(^1\) combines the receive signals of a patch of transducer elements to one channel. This technique has two main disadvantages:

- First, the actual raw signal of each and every element is lost and with it part of the information acquired by the probe.
- Second, a significant amount of active analog electronics is required to be integrated into the probe, to implement the analog delay and sum operation. These active components dissipate a lot of power, which can heat the probe beyond acceptable temperatures.

Commercial probes, such as the 4z1c matrix probe by Siemens [26] are thus actively-cooled through the cable.

While these commercial matrix probes are highly advanced and enable useful modalities for sonographers, they are of limited use for researchers interested in having access to the raw signal data from the individual ultrasound transducers. Moreover, these proprietary probes can only be operated with the completely closed commercial systems they were designed for. To facilitate research on 3D ultrasound, an open matrix probe is desired, which can be connected to existing ultrasound research systems and provides open access to the raw signal from all transducer elements.

Current state-of-art ultrasound research systems [84,85] provide 256 ultrasound channels. Connecting matrix probes with 1024 and more elements is thus not easily possible. To solve this issue, researchers either built their own high-channel-count ultrasound system [66], or they connected and synchronized several existing 256-channel systems [86,87] to obtain a total channel count of 1024 to connect all elements of a matrix probe for 3D imaging. Both options come at a considerable time and cost effort. From a cost and usability point of view, it is desirable to have a matrix probe that can be connected to a single already existing research system.

\(^1\)See Section 2.1 for details
3.1. INTRODUCTION

3.1.2 Contributions

As part of this thesis, we designed in collaboration with the Fraunhofer Institute for Biomedical Engineering (IBMT) a $32 \times 32$ ultrasound transducer matrix probe with an integrated 4:1 multiplexing and amplification stage. The probe can be connected over a 256-micro-coaxial cable harness to an existing 256-channel research ultrasound system that provides raw RF data access. The design is based on an early prototype by IBMT. In summary, our contributions are:

1. We present the MuxHead, an open 1024-element handheld matrix probe for 3D imaging.
2. Compared to the early prototype, we completely redesigned the hardware of the probe to achieve a handheld form-factor and to connect the MuxHead to a research system.
3. We performed a complete 3D imaging simulation of the MuxHead to find an imaging setup for the in-vitro test.
4. We programmed the required processing pipeline to compute volumes out of the captured raw data and visualize them.
5. We show in-vitro test results to demonstrate that the MuxHead can successfully acquire 3D volumes.

Our optimized PCB stack for the MuxHead requires only a volume of $30 \times 33 \times 112 \text{ mm}^3$, which fits in a casing for handheld operation. The designed communication protocol between the MuxHead and the research system provides reliable operation with precise and fast switching between the multiplexing configurations. With a first imaging test, we demonstrate that the MuxHead can successfully acquire and resolve individual nylon wires of a wire phantom in water by using plane-wave imaging.

The rest of the chapter is organized as follows: Next, we describe the overall system architecture (Section 3.2) of the MuxHead connected to a research system, before the implementation details (Section 3.3) of the MuxHead are elaborated. Then we explain the simulation setup and the imaging strategy (Section 3.4), which we have selected for the in-vitro test (Section 3.5), which is presented afterwards. We then close with discussions and conclusion (Section 3.6).
3.2 System Overview

The overall system architecture of the 3D ultrasound research system is depicted in Fig. 3.1: A $32 \times 32$ 1024-element matrix transducer probe is connected to a 256-channel ultrasound research system. The probe embeds a 4:1 multiplexing stage, which allows connecting a subset or all of the 1024 elements to the 256 system channels. Given that the signal output of a single matrix element is rather weak due to its small size, the probe also provides an amplifier for every system channel for cable impedance matching and to boost the signal before it is sent over the coaxial cable harness. For fast and reliable control of the multiplexer stage, the probe embeds a microcontroller. The microcontroller is connected with the research system over several auxiliary signals used for synchronization and control as well as an $I^2C$ bus for configuration.

![Diagram of the system architecture]

**Figure 3.1** – System Overview of our 3D ultrasound system: It consists out of our MuxHead connected to a 256-channel research system. The MuxHead operates with a $32 \times 32$ transducer matrix and provide an integrated 4:1 multiplexing and receive amplification stage. The probe is connected with the system over a 256-channel micro-coaxial cable harness.

3.2.1 Multiplexer Operation

After power up and configuration over $I^2C$, the MuxHead observes the auxiliary signals provided by the connected research system to change the configuration of the multiplexing stage in the right moment. During operation, the probe enumerates through different multiplexer
configurations, which are programmed over the I²C bus. Besides changing the multiplexer configuration between individual acquisitions, the MUXHEAD also allows different multiplexer configurations for the transmit and receive phase within one acquisition: The MUXHEAD can switch instantaneously from the transmit to the receive configuration to be immediately ready to receive the backscatter signals after the transmit pulse has been sent.

### 3.2.2 Multiplexer Setup

The $32 \times 32$ transducer matrix is divided in 256 $2 \times 2$ patches of neighboring elements as illustrated in Fig. 3.2. Every patch has a fixed assignment to a system channel. All four transducer elements in the patch have their own programmable switch to connect the element to the assigned system channel.

![Matrix transducer multiplexer assignment](image)

**Figure 3.2** – Matrix transducer multiplexer assignment: All 1024 elements (E0 to E1023) are assigned to a system channel (S-0 to S-255) in $2 \times 2$ patches of four transducer elements.

This setup provides high-flexibility as it allows the user to connect none, just one, any subset or all elements of the patch to the system channel. If multiple elements are connected simultaneously to the
system channel, they act as one larger transducer element: During transmit, all elements will jointly emit the same acoustic wave. During receive, all their receive signals will be added up because piezo-electric transducers can be modeled as a voltage source.

### 3.2.3 Circuit Details of a Channel

A simplified circuit diagram of a single channel is depicted in Fig. 3.3. We show channel S-0, but all 256 channels are built exactly the same way. The circuit diagram shows the piezoelectric elements allocated to the channel, the multiplexing stage as well as the amplification stage, the micro coaxial cable and a typical RX/TX frontend circuit, as it can be found in most ultrasound systems.

![Circuit diagram of a single system channel (S-0)](image)

**Figure 3.3** – Circuit diagram of a single system channel (S-0), showing the multiplexing and amplification stage integrated in the MUXHEAD, as well as the cable and the TX/RX frontend in the connecting system.

Four piezoelectric elements (E0, E1, E32 and E33) are connected over four switches to the system channel (S-0). Given that a piezoelectric transducer can be modeled as a capacitive load, a bleed resistor per element provides a discharge path to eliminate residual voltage build up.

To avoid having separate RX and TX wiring from the system to the probe, the RX and TX path share the same coaxial wire. Otherwise, twice the amount of micro-coaxial cables (512 instead of 256) would be needed to connect the probe. Sharing the cable makes the amplification
stage more complicated: During the transmit phase, up to ±100 V pulses are sent by the system through the cable to stimulate the piezoelectric elements for acoustic emission. The sensitive receive amplifier needs to be protected from this high-voltage (HV) pulses. During transmit, the HV pulses pass through the anti-parallel diode pair to reach the multiplexing stage. To protect the amplifier input, it is AC-coupled to the multiplexing stage and has an HV protection circuit to limit the input voltage. The amplifier output also has an HV protection and automatically disconnects from the shared channel, when a voltage above a threshold (2.7 V) is observed, in order not to drive back a signal during transmit. The amplifier not only boosts the signal (by 6 dB) during the receive phase, it also matches the output impedance of the piezoelectric element (a few kΩ) to the cable impedance (typ. 65 Ω).
3.3 Implementation

The MuxHead probe consist of the following four main parts: the matrix transducer, the probe electronics, which are distributed over a stack of five PCBs, the cable harness and a 3D-printed housing. A photograph of the MuxHead is shown in Fig. 3.4.

![Photograph of the MuxHead with the housing lid removed. – Photo courtesy of Fraunhofer Institute for Biomedical Engineering (IBMT)](image)

The main design goal for the MuxHead was to fit all the required electronics in the smallest possible volume to enable a handheld form-factor. Given that our probe features 1024 transducer elements in total and requires 256 instances of the channel circuit described before, a highly space-optimized implementation is critical to achieving a manageable overall probe size that allows comfortable holding of the probe.

3.3.1 Matrix Transducer

The $32 \times 32$ matrix transducer was designed and manufactured by Fraunhofer IBMT. The transducer has a center bandwidth of 4 MHz and an element pitch of 300 µm. Every row of elements provides its own flexible PCB connector. The flex PCBs have been specifically designed to connect to our PCB stack and to minimize the routing overhead on the PCBs, as explained later.
3.3. IMPLEMENTATION

3.3.2 Stack Design & Probe Electronics

We achieve a form factor of $30 \times 33 \times 112 \text{mm}^3$ for the probe electronics with a highly space-optimized PCBs stack. Given the tremendous amount of inputs (1024) and outputs (256), the overall design had to be optimized to ease the signal routing on the PCBs.

The PCB stack as shown in Fig. 3.5 consist out of four identical *Multiplexer Boards* (Mux Board) and one *Control Board*. The stack connects to the matrix transducer through 32 flex PCBs. The stack provides four cable connectors (one per Mux Board) to connect to the cable going to the backend system. Each connector carries 64 system channels. An additional auxiliary connector provides the power and control signals.

![Diagram of PCB stack](image)

**Figure 3.5** — Photograph of the PCB stack with no cables connected. A single flex PCB used to connect to the transducer matrix to the MuxBoard, is visible in the lower left corner.

A simplified block diagram of the PCB implementation is shown in Fig. 3.6: The 32-row matrix is split in 16 double-rows. For each matrix double-row, which covers 16 $2 \times 2$ patches, a 64-element multiplexing and amplification stage is implemented: The multiplexing is realized.
with two 32-channel 2:1 integrated HV switch chips (HV9201, Microchip Technology) providing integrated bleed resistors. The switch state is controlled through a latchable shift-register. The amplification stage is implemented with two octal operational amplifiers (MAX4805, Maxim Integrated), providing HV in/output protection and an automatic HV switch to disconnect the amplifier output. This 64-element multiplexing and amplification stage is implemented 16 times in total with four instances on each of the four Mux Boards.

The control board PCB hosts the microcontroller and a temperature sensor. The temperature sensor allows checking whether the electronics integrated into the probe produce excessive heat. The microcontroller (PIC32MX440F128, Microchip Technology) interfaces the multiplexer shift-register lines over a high-speed 16-bit parallel master port for fast programming of the switches. The amplification stage can be manually deactivated when not needed. The connecting backend system communicates with the microcontroller over I2C for configuration and uses two signals SYNC and CLR for control during imaging.

The control board PCB hosts the microcontroller and a temperature sensor. The temperature sensor allows checking whether the electronics integrated into the probe produce excessive heat. The microcontroller (PIC32MX440F128, Microchip Technology) interfaces the multiplexer shift-register lines over a high-speed 16-bit parallel master port for fast programming of the switches. The amplification stage can be manually deactivated when not needed. The connecting backend system communicates with the microcontroller over I2C for configuration and uses two signals SYNC and CLR for control during imaging.

Figure 3.6 — Simplified block diagram of the implementation of the multiplexing/amplification stage in the MuxHead including the microcontroller subsystem to control the operation.
In total, the MuxHead needs seven external power supplies to operate the multiplexer and amplifiers: ±2 V and ±5 V for bias and amplifier supply, 3.3 V for the digital part, and the transmit voltage supplies $V_{pp}$, $V_{nn}$ for the level-shifters of the high-voltage switches in the multiplexers. All voltages are supplied by the connecting ultrasound system through the cable.

Fig. 3.7 shows a side view of the PCB stack with the Control Board on top and four Mux Boards plugged as a stack from below. Each Mux Board carries 4 multiplexing and amplification stages and has its own cable connector for 4 x 16 system channels. Power and control are provided from the Control board through the stack connectors to the entire PCB stack.

Figure 3.7 – Side view of the PCB stack, which consists out of four Mux Boards and one Control Board. Four Multiplexing and Amplification stages (see Fig. 3.6) are placed per Mux Board (two on each side). Signal routing is reduced by staggered placement of the flex PCB connectors. The analog signals are routed within the Mux Board (orange, pink). Power (red) and control (blue) is routed through the stack connectors to reach the entire PCB stack.

To reduce the wire routing within the PCB, the flex PCBs from the transducer connect right next to their corresponding mux stage. To achieve this, flex PCBs with different length, orientation and offset have been designed and connected to the matrix transducer in a repeating pattern. Fig. 3.8 shows the flex PCB design in detail.
3.3.3 MuxHead Configuration and Operation

To configure the MUXHEAD over I²C, we implemented a configuration tool running on the connecting research system. With this tool, multiplexer configurations can be created and programmed into the MUXHEAD: With the current MUXHEAD firmware, the probe can store eight switch configurations. Switch configuration sequences with a length of up to 32 can be composed out of these eight configurations. This programming model reduces the memory requirements as switch configurations are often used multiple times in a sequence. As an example, in the default sequence (TX, RX1, TX, RX2, TX, RX3, TX, RX4) the same transmit configuration (TX, all elements active) is used four times – each time with a different element of the $2 \times 2$ patch active during receive phase (RX1-4).
3.3. IMPLEMENTATION

During continuous imaging operation, the connecting system uses the SYNC signal to trigger the switching to the next configuration in the sequence. The CLR signal resets the sequence counter and is used to align the sequencing in the MUXHEAD with the imaging system.

The specific control operation and its timing are shown in Fig. 3.9. This control operation has been specifically designed to avoid any control signal corruption from the HV transmit pulses and to mitigate noise injection on the receive signals. To achieve this, the system synchronizes with the MUXHEAD only between the TX-RX cycles, i.e., when no HV pulses are emitted and no backscattered signals received.

![Diagram of MUXHEAD control operation and timing]

**Figure 3.9** – MUXHEAD control operation and timing: TOP: Overall operation with SYNC signal, operation phases (Config, TX, RX) and analog signal present on the system channel (red: HV pulse, green: received echo). BOTTOM: Close-up on the internal control timing: After detecting a rising edge on SYNC (1) the transmit configuration is shifted and loaded into the switches (2), followed by shifting the receive configuration in. Now the system is ready for TX (3). After a configurable delay (4) the receive configuration is activated (5) to receive the echos.
Upon detection of a rising edge of SYNC the MuxHead plays back an entire TX-RX cycle: First, the microcontroller shifts the transmit configuration into the switches and activates it. Then, the receive configuration is pre-loaded into the shift registers of the switches. This configuration phase takes a fixed amount of time $t_c$. Now, the MuxHead is ready for transmitting (TX), and the connecting system can send the HV pulses to the MuxHead to emit an acoustic wave.

When a programmable delay ($t_{TR}$) since the sync event expires, the MuxHead switches from the transmit to the receive configuration. Since the receive configuration is already preloaded in the shift-register and only needs to be latched into the switches, this switching occurs instantly. Now, the MuxHead is in receive (RX) mode and is amplifying and forwarding the received echos to the connecting system. The MuxHead waits in the receive configuration until the next rising edge of SYNC is detected, which starts the next TX-RX cycle in the sequence. If the CLR signal is high during the rising edge of SYNC, the sequencing reset.

Note that triggering the switching from the TX to RX configuration with an expiring timer on the MuxHead is much more reliable and precise than observing a control signal sent by the research system due to the following reason: Given that the control signals share the same cable as the system channels, heavy pulsed noise is injected on the control lines during the TX phase, when the HV pulses are sent. These injected pulses can prematurely trigger the MuxHead to change its multiplexer configuration.

Similarly, the I²C interface is not used during imaging, as digital communication is either corrupted by the HV pulses or the data transfer may add noise on the susceptible receive signals.
3.4 MuxHead Simulation

Before the in-vitro tests with the actual device, we performed a complete 3D imaging simulation of the MuxHead to find a useful imaging setup and to verify our processing pipeline that computes volumes out of the captured raw data.

3.4.1 Setup

The ultrasound simulation of the MuxHead is performed using Field II \([78, 79]\). We simulate all 1024 transducer elements with a simple 3D point-scatter phantom. The point-scatters are placed on a 3D grid in the volume below the transducer. For the transmit scheme we use 9 plane waves with an inclination distributed equally between \(\pm 20^\circ\) with respect to the \(y\)-plane. For the transmit pulse, a 4 period 4 MHz sine is used. The setup is depicted in Fig. 3.10.

The processing pipeline applies a square TGC compensation before performing standard bandpass DAS-beamforming using a static 2D-Hanning apodization over the matrix transducer (see Section 2.3). For each of the nine transmit events, i.e. plane-wave emissions, a low-resolution Cartesian volume is computed. The nine volumes are then coherently compounded to obtain the high-resolution volume, on which the signal envelop is extracted before log-compression.

3.4.2 Results and Discussion

In Fig. 3.10, the computed 3D volume is visualized as a 3D contour plot as well as a ray-trace rendering. The contour plot shows the signal amplitude (reflectivity) on several planes through the volume. The translucent ray-trace rendering visualizes the volume with the same camera orientation as the other two plots in this figure. Fig. 3.11 show several cuts through the volume. All visualizations reveal that the scatters are well resolved on the \(yz\)-plane at \(x = 0\). The further the scatters are away from this plane, the weaker their signal and resolution are.

This is expected, as we only steer the transmit plane-wave along the \(yz\)-plane at \(x = 0\) (see the setup in Fig. 3.10). Considering the dimensions of the matrix transducer, no signal is emitted in the regions further apart from this plane. Hence the scatters cannot be
seen. Furthermore, because there is no angular diversity from different plane-wave emissions in the x-axes direction, the resolutions of the scatters are as low as when just using one single plane-wave emission with $0^\circ$ inclination straight down (see xz-plane at $x = 0$ in Fig. 3.11).

![Figure 3.10 – MuxHead simulation with 9 coherently added plane waves:](image)

**Figure 3.10 – MuxHead simulation with 9 coherently added plane waves:**

**Left:** Setup with matrix transducer (black square) and scatteres (red crosses) in volume distributed on a 3D grid. The lines (black) indicate the vectorial direction of the 9 plane-waves. **Middle:** Contour plots through x-0, y-0 and z-{−1, −2.5, −4} planes (see also Fig. 3.11). **Left:** Translucent ray-trace volume rendering with 15 dB dynamic range and the y-axes colored from red (negative) to blue (positive).

Due to this imaging setup, the yz-cut-plane at $x = 0$ depicted in Fig. 3.11 can also be considered as a normal 2D plane-wave image, but with one crucial difference: By using a matrix transducer, *dynamic elevational focus* can be applied\(^2\). In 2D imaging, the elevational focusing is what defines that one actually gets a cross-section through the body under investigation and not a superimposed signal of the entire volume below the transducer. In normal 2D imaging, the focusing on the elevational plane is achieved passively by depositing an acoustic lens on the transducer, which provides a fixed focal-length focus on the elevational plane perpendicular to the transducer surface. Matrix transducers allow dynamic focusing at all focal-depths resulting in better image quality.

\(^{2}\)Dynamic elevational focus is also applied in 1.25D and 1.5D commercial system, where multi-row array transducers are used to improve the 2D image.
Figure 3.11 – MUXHEAD simulation: Cuts through the volume along xz/yz-planes through origin as well as cuts along the xy-plane at depths 1.75 cm and 2.5 cm. Not all point scatters (red crosses) are visible due to the chosen transmit strategy.
3.5 In-vitro tests

After having defined a first simple TX setup and verified with simulations that it could produce useful images with the MUXHEAD, we setup a first in-vitro test.

3.5.1 Setup

The overall in-vitro test setup is shown in Fig. 3.12: The MUXHEAD is connected to the 256-channel DiPhAS research system [20, 85] developed and provided by IBMT. For the transmit settings we use the setup described in the previous section. During receive, the raw signals from the MUXHEAD are sampled with 80 MS/s and stored on disk, before they are processed offline with our processing pipeline.

![Image](image.png)

**Figure 3.12** – MUXHEAD in-vitro test setup showing the overall test setup with the water tank phantom and the MUXHEAD connected to the DiPhAS research system.

The phantom setup is shown in Fig. 3.13. For the phantom, we use a water bath containing a highly-reflective metal cylinder with a nylon wire phantom placed on top of it. The MUXHEAD was positioned such that the top surface of the metal cylinder is at a depth of 4.5 cm in the water. The nylon wire phantom was rotated around the z-axis by 45° such that the nylon wires pass diagonally through the volume (see Fig. 3.13b).
3.5. **IN-VITRO TESTS**

3.5.2 **Results and Discussion**

In Fig. 3.14, the computed 3D volume from the in-vitro test is visualized as a 3D contour plot as well as a ray-trace rendering. The figure also includes a plot of the geometric setup, which shows the position and orientation of the transducer, the plane-wave emission directions and the location of the metal surface at 4.5 cm depth. The 3D contour plot visualizes cross sections at the depths of the nylon wires (2.5 cm and 3.5 cm) as well as the metal surface of the cylinder (4.5 cm). In the translucent ray-trace rendering, the metal surface is visible on the bottom, and seven individual nylon wires can be made out in the center part. The weak signal on the top part is image clutter. Fig. 3.15 shows several cuts through the computed volume.

As elaborated in the previous section, the chosen transmit setup produces a normal 2D plane-wave image at the $yz$-plane at $z = 0$. On this cut (see Fig. 3.15), the metal surface can be seen on the bottom as well as five nylon wires in the middle section. On the top, there is signal clutter caused by the switching from transmit to receive configuration followed by the activation of the receive amplifier. The later manifested as a dot artifact below the center of the matrix at depth 0.8 cm.

---

**Figure 3.13** – **MuxHead** in-vitro phantom setup. a) shows the metal cylinder with a nylon wire phantom placed on top of it. b) shows a closeup of the exact phantom positioning for the output shown in Fig. 3.14 and Fig. 3.15.
Overall, the main features of the test setup are well visible. Right below the transducer matrix, the resolution of the nylon wires matches the results from the simulation qualitatively. As already observed in the simulation, the signal response is weaker, the further it is away from the center region. As explained before, the resolution on the x-axes is again inferior due to lack of transmit steering in this direction.

With this first and elementary test, we have verified that 3D volumes can be successfully captured with the MUXHEAD.

### 3.5.3 Thermal Dissipation Test

To verify that the power dissipation from the electronics in the MUXHEAD does not excessively heat the probe, we configured the DiPHAS system to perform TX-RX cycles with a repetition rate of 10 kHz continuously. We measured the temperature within the MUXHEAD using the integrated sensor. The internal temperature settled at 34.5°C, which results in plenty of safety margin to the 43°C allowed by medical regulations for surfaces in contact with human skin.
Figure 3.15 – MuxHead in-vitro measurement: Cuts through the volume along xz/yz-planes through origin as well as cuts along the xy-plane at depths 2.5 cm and 3.5 cm with 15 dB dynamic range.
3.6 Discussion and Conclusion

With the MuxHead, we have implemented a $32 \times 32$ ultrasound matrix probe for 3D imaging. The MuxHead suits the needs for researchers as it provides full access to the raw analog ultrasound signals from all its 1024 transducer elements. The MuxHead integrates a $4 : 1$ analog multiplexing stage to simplify the cabling and to reduce the number of required channels in the connecting system to 256. This ultimately allows connecting the MuxHead to 256-channel research systems such as the DiPHAS, which are readily available on the market.

We successfully connected the MuxHead to the DiPHAS research system and demonstrated its operation with a first simple 3D imaging setup.
Chapter 4

LightProbe & UltraLight: Software-Defined Ultrasound using Digital Probes

4.1 Introduction

4.1.1 Motivation

Traditional ultrasound imaging systems operate with a passive transducer probe, where the piezo elements in the probe are connected directly over an analog coaxial cable harness to a backend system. The backend system contains the analog front-end to send ultrasound pulses and to receive echos, as well as the digital processing unit to compute the output image. Moreover, the backend system provides a control interface for the sonographer and a display to observe the imaging output in real-time.

Today, two trends in ultrasound system design can be observed: On one hand, towards software-defined ultrasound imaging, where the entire signal processing is performed in software [17]. In the past, general
PROCESSING UNITS DID NOT PROVIDE ENOUGH COMPUTE POWER AND CUSTOM HARDWARE SOLUTIONS WERE REQUIRED TO COMPUTE THE IMAGE FROM THE RAW PROBE SIGNALS. TODAY, THE IMAGE FORMATION IS INCREASINGLY PERFORMED IN SOFTWARE ON GRAPHICS PROCESSING UNITS (GPUS) [18–20] AND MULTICORE PROCESSORS [21,22,47]. THESE NEW SYSTEM ARCHITECTURES HAVE NOT ONLY ENABLED NEW IMAGING MODALITIES (ULTRAFAST IMAGING [4,88], VECTOR FLOW IMAGING [89,90], ELASTOGRAPHY [91]), BUT HAVE ALSO REDUCED SYSTEM COST, AS NO ULTRASOUND-SPECIFIC HARDWARE IS NEEDED FOR PROCESSING.

THE AIXPLORER (SUPERSONIC IMAGING) OR THE DIOPHAS (FRAUNHOFER IBMT) ARE TWO EXAMPLES OF SUCH SOFTWARE-DEFINED SYSTEMS. BOTH SYSTEMS ADOPT THE ULTRAFAST SYSTEM ARCHITECTURE PARADIGM [88,92], WHICH PROPOSES THAT THE RAW DIGITIZED DATA SHOULD BE DIRECTLY SENT TO A GPU FOR PROCESSING.

THE REQUIRED HIGH-BANDWIDTH LINK TO TRANSFER RAW DATA FROM THE ANALOG-TO-DIGITAL CONVERTERS (ADCs) INTO THE MEMORY OF THE GPU IS ONE OF THE MAIN CHALLENGES FOR SOFTWARE-DEFINED SYSTEMS: A 256-CHANNEL SYSTEM SAMPLING ALL CHANNELS SIMULTANEOUSLY WITH 12 B AND 30 MS/s PRODUCES RAW DATA AT A RATE OF ALMOST 100 Gb/s.

THE SECOND SYSTEM TREND IS TOWARDS DIGITAL ULTRASOUND PROBES: THESE PROBES NO LONGER NEED AN EXPENSIVE AND BULKY ANALOG CABLE TO CONNECT, BUT PROVIDE A WIRED OR WIRELESS DIGITAL INTERFACE SUCH AS USB OR WI-FI TO CONNECT TO A COMMODITY DEVICE SUCH AS A SMARTPHONE, TABLET OR PC, WHICH RUNS AN ULTRASOUND SOFTWARE APPLICATION TO CONTROL THE PROBE AND TO DISPLAY THE IMAGE. TO PROVIDE THIS CONNECTIVITY, THE DIGITAL PROBE EMBEDS THE ANALOG FRONT-END IN THE PROBE HOUSING AND PERFORMS AT LEAST PART IF NOT ALL OF THE DIGITAL PROCESSING ON THE PROBE [14–16]. ON-PROBE PROCESSING IS INEVITABLE AS ALREADY A 16-CHANNEL PROBE SAMPLING WITH 12 B AT 30 MS/s PRODUCES DATA AT A RATE OF 5.8 Gb/s, WHICH ALREADY EXCEEDS WHAT STANDARD LINKS, SUCH AS USB 2.0 (MAX. 0.48 Gb/s), USB 3.0: (5 Gb/s) OR WI-FI 802.11n (MAX. 0.6 Gb/s) CAN PROVIDE.

DIGITAL PROBES CONNECTED TO A SMARTPHONE OR TABLET RUNNING AN ULTRASOUND APP ARE WIDELY USED TO BUILD PORTABLE SYSTEMS SUCH AS THE MOBIUS PE SYSTEM (MOBISANTE, REDMOND, USA) [93], THE PHILIPS VISIQ & LUMIFY (PHILIPS HEALTHCARE, NL) [16,94], THE CLARIUS HANDHELD WIRELESS SCANNERS (CLARIUS, CAN) [95] OR THE IQ (BUTTERFLY NETWORK, USA).

WHILE THESE DIGITAL PROBES PROVIDE ALREADY GOOD IMAGING QUALITY FOR STANDARD B-MODE, THEY DO NOT (YET) SUPPORT THE MORE DEMANDING MODALITIES SUCH AS ULTRAFAST IMAGING, VECTOR FLOW OR ELASTOGRAPHY.
The main reason for this is that digital probes are thermally-limited devices [15], meaning that the amount of power that can be dissipated within the probe is limited, as otherwise, the probe would become too hot. The surface temperature of a device in direct contact with the patient must be kept below 43°C to comply with medical safety regulations (IEC 60601-1 [96]). This thermal limitation restricts the number of receive channels and the complexity of the processing feasible to implement on a digital probe: Even tough the power consumption per receive channel has decreased significantly in the past years, from 100 mW down to 40 mW [29, 97], a 64-channel probe still consumes 2.6 W in the receive path alone. Thus, digital probes typically operate with no more than 16 receive channels (amplifier, ADC) and use time-multiplexing [14] or analog pre-beamforming [98] to support larger transducer arrays and to have enough thermal budget left for on-probe processing. However, to support ultrafast modalities, all transducer channels should be sampled simultaneously and more complex processing is required. Fitting both into the thermal budget of a probe cannot easily be achieved.
4.1.2 A Novel Ultrasound System Architecture

To resolve this issue, we propose a novel ultrasound system architecture, which relies on an *ultrafast digital probe* that samples all channels simultaneously and uses a *high-speed digital link* to transfer the raw channel data to an off-the-shelf PC subsystem, where the processing is performed without thermal power constraints.

This architecture combines the flexibility of *software-defined systems* and the cost-efficiency of *digital probes*. We expect such an architecture to come with significant cost savings as it requires only a minimal amount of ultrasound-specific hardware, i.e., just the probe and a refurbished housing for the PC with a console keyboard. Also, the expensive coaxial analog cable harness is no longer needed, which itself has a significant contribution to the price of the system. While it has been demonstrated that tablets and smartphones [99] could provide sufficient compute power on their embedded GPUs for software beamforming, these devices today do not yet provide sufficient external I/O bandwidth to sustainably receive the raw data. We, therefore, focus on a PC-based processing system for the time being.

The challenge to build an ultrafast digital probe is to include a high number of front-end channels (64 or more) and a high-speed (>10Gb/s) link into the probe within thermal limits.

In this thesis, we demonstrate that this challenge can be addressed by combining the following three approaches elaborated in detail in Section 4.3:

1. First, we equip the probe with a fiber optical link to provide sufficient link bandwidth (>25Gb/s).
2. Second, we reduce the average power consumption for normal B-mode imaging by exploiting the ultrafast imaging capabilities of the probe.
3. Third, we develop a thermal management solution that enables high-performance operation modes with a power dissipation exceeding the passive cooling capabilities of the probe.
4.1.3 Contributions

As part of this thesis [42–44], we designed a complete and fully-functional ultrasound imaging system to demonstrate the feasibility of our newly proposed ultrafast digital probe system architecture. The system has been designed and published in several steps:

1. In our first work [42], we have introduced the LightProbe concept, i.e., using an optical link to connect a programmable digital probe to remove the bandwidth bottleneck to enable low-cost ultrasound systems with increased capabilities. An early system prototype with a reduced number of channels was used to estimate the power consumption in different operation modes to assess the feasibility from a power dissipation point-of-view.

2. In our second work [43], we have presented the UltraLight system, which is a complete software-defined ultrasound imaging system consisting of our digital transducer probe, called LightProbe, connected to a standard PC equipped with a GPU. The system allows rendering B-mode images on the GPU from the raw digital samples captured by the probe.

3. In our latest work [44], we focused on the final LightProbe design, discussed in detail design decisions and hardware details, performed extensive quality and performance evaluation on different imaging methods and introduced the thermal management needed to operate such a device.

In this chapter, we focus on the results from our latest work [44] as well as the details on the overall UltraLight system.

In summary, our contributions are:

1. We present UltraLight, to the best of our knowledge, the first ultrafast imaging system based only on a 64-channel digital probe and off-the-shelf system components.

2. We present LightProbe, a fully-functional hand-held digital transducer probe. The final version of the probe uses a 64-element 4 MHz linear phased-array and integrates a 64-channel 100 Vpp reconfigurable TX/RX front-end including analog-to-digital conversion (up to 32.5 MS/s @ 12 bit). The
CHAPTER 4. LIGHTPROBE & ULTRALIGHT

probe integrates a Xilinx Artix 7 FPGA that provides a high degree of in-probe configurability to support different transmit and pre-processing schemes. To provide real-time raw data access, the probe features an optical link interface achieving data transmission rates of up to 26.4 Gb/s.

3. We demonstrate that, even with only passive cooling, a digital probe can support continuous plane-wave imaging with a frame rate of 47 Hz indefinitely.

4. We also demonstrate how high-frame-rate imaging (200-500 fps) can be safely supported in a digital probe, despite high peak power consumption (10.7 W) and high link bandwidth requirements (15.36 Gb/s).

5. We implement the required interfaces and software applications to connect the LIGHTPROBE to a standard PC and demonstrate full real-time system operation of various conventional and ultrafast imaging modes.

The ULTRALIGHT system was used in various projects including the high-speed bubble tracking demonstration in Chapter 5. Other notable projects using the ULTRALIGHT system not explicitly covered in this thesis were:

**On-head Data Compression:** Performing data compression within the probe reduces the load of the digital link, which reduces the power dissipation and lowers the IO requirements for the connecting device. Decimation & demodulation was implemented on the LIGHTPROBE, and it was verified that the image suffers no quality impact. Alternative compression schemes based on compressive sensing were explored in collaboration with Adrien Besson and Dimitris Perdios from LTS5 at EPFL.

**LIGHTPROBE for Mobile Imaging:** Mobile imaging with the LIGHTPROBE has been explored in two student projects carried out by Matthias Brägger, where he extended the LIGHTPROBE hardware with a WiFi radio module and explored novel imaging techniques to reduce the data load for the radio link as well as the power consumption in the ultrasound front-end. Real-time imaging at 1 fps could be demonstrated over a low-rate WiFi link.
4.1. INTRODUCTION

Vector Flow Imaging: Ramon Aerne demonstrated in his Master Thesis that vector flow imaging could be implemented on the ULTRALIGHT system. He implemented the required algorithms on the system and performed in-vitro experiments to show that vector flow can be measured.

The rest of the chapter is organized as follows: After related work (Section 4.2), we elaborate the main design decisions (Section 4.3) before we outline the system architecture (Section 4.4) and implementation (Section 4.6). We show extensive measurements (Section 4.8-4.9) to demonstrate our system works, produces excellent images and does not overheat. Finally, we discuss our the results (Section 4.10) and close with conclusions.
4.2 Related Work

4.2.1 Taxonomy

We define an ultrasound transducer probe as digital, if the analog signals received by the transducer are converted to the digital domain within the probe, and either the raw or processed digital signals are output by the probe. In contrast to it, we call a probe analog if it outputs analog signals. We further differentiate between active and passive analog probes:

4.2.2 Analog Probes

Passive analog probes contain no active electronics at all and just consist of the piezoelectric transducer array, where each element is individually connected directly to the analog cable. Given a 256-element transducer array, 256 analog cables are required to connect the probe, which requires an expensive micro-coaxial cable harness along with a sizeable high-pin-count plug. To reduce the number of cables, some analog probes embody integrated high-voltage multiplexer, that enables to connect only a subset of the entire array at a time to the cable [100, 101]. A 192-element transducer probe with integrated 3:1 multiplexer for example, only requires 64 analog cables to connect. But it also allows just to use one-third of the array at a time. We call such probes active analog as they contain some kind of active electronics. Active analog probes come in many flavors and they may even contain the entire TX frontend [102], such that the full array can be used during transmit while having connected the probe to the backend system using integrated multiplexer. Some active analog probes feature an analog pre-beamforming stage [25, 103, 104], that allows to steer a group of neighboring element to a fixed direction by using programmable analog delay lines in order to produce a single output signal per group and to reduce the number of required analog cables, which is especially important to make matrix probes for 3D imaging feasible that feature close to 10’000 elements [63]. Another approach reducing the number of cables is high-frequency per-sample time-division multiplexing [105, 106]. However, this method requires a high-quality, high-bandwidth analog cable as the signals are modulated
on analog pulses with a frequency of several hundred MHz. Chapter 2 and Chapter 3 provide more information on this topic.

While analog probes increasingly got more sophisticated, they still have to be connected to a dedicated backend system providing the required analog-to-digital conversion stage and rely on an expensive micro-coaxial cable harness to do so. Also, as the number of cables in the harness is increasing to support more and more elements in the array, the ergonomics for the operators suffers as the cable gets heavier and stiffer.

### 4.2.3 Digital Probes

*Digital* probes output digital signals. Thus, they no longer need a bulky coaxial analog cable harness. However, they must provide a large bandwidth digital link. If the link provides insufficient bandwidth to output the raw digital sample stream directly, some sort of preprocessing in the probe is required to reduce the amount of data to be transferred. We call probes able to output raw samples *ultrafast digital* probes and probes deploying on-probe processing *compressing digital* probes.

Several compressing digital probes are described in the literature: Performing the entire beamforming process on the probe reduces the amount of data to transfer. [14] presents an ultrasound transducer probe with an integrated 16-channel frontend equipped with a Xilinx Spartan 6 FPGA for on-head beamforming and USB 3.0 to connect with a smartphone. The head has a size of $180 \times 55 \times 5 \text{mm}^3$ and consumes 8.16 W, which heats the system internally to 82.3°C and the transducer surface to 35°C. The MobiUS PE System (MobiSante, Redmond, USA) and Philips Lumify (Philips Healthcare, NL) are two commercial transducer probe with an integrated frontend that can be connected to a PC or tablets over USB 2.0. Due to the limited USB link bandwidth, data must be preprocessed in the probe. However, as the thermal budget in the probe is limited, only simple processing is permitted on-probe, which allow only to implement basic imaging modalities. The battery-powered handheld Clarius Handheld Wireless Scanners (Clarius, CAN) connects to tablets or smartphones over a Wi-Fi 802.11n [95], which is even slower than USB 2.0. The ACUSON Freestyle (Siemens Healthcare, GER) is another commercial wireless scanner using a proprietary wireless link.
Recently, synthetic aperture sequential beamforming (SASB) [107] has been proposed as a method to reduce data bandwidth required for a digital probe [15]. SASB is a two-stage beamforming method, which consists of a first stage low-complexity on-probe beamformer, which combines the data from all channels to a single channel followed by a second stage beamformer in the connecting device, that combines the data from multiple acquisitions to form an image. While it has been shown that SASB produces better images than conventional methods [108], it does not reach the quality of synthetic transmit aperture (STA) imaging [36], where data from all channels is combined over multiple acquisitions in a single beamforming step.

To fully support ultrafast imaging, the raw data of all channels must be accessible in real-time to the processing unit, which is only provided by direct digital probes.

4.2.4 Real-time Raw Sample Access

Real-time access to the raw sample streams, as we propose with our ultrafast digital probe, is currently only provided by high-end systems targeting research: The Vantage system (Verasonics) available in different configurations from 64 to 256 channels consists of a box housing the frontend electronics, which is connected to a workstation over 8 PCI express 3.0 lanes [84]. This link provides data rates up to 6.6GB/s (52.8Gb/s). If this link is insufficient to stream the data, it provides local buffer memory of 64MB/channel. Similarly, DiPhAs (Fraunhofer IBMT) [20] provides a 256 channel frontend connected to a GPU-enabled PC over PCIe all integrated into the same casing. The 256-channel ultrasound advanced open platform (ULAOP-256, University of Florence) [109] is a modular system featuring both FPGAs (Altera, ARRIA V GX) and DSPs (Texas Instruments, 320C6678) for massive configurable on-system processing. The platform can be connected to a host PC over USB 3.0. The synthetic aperture research ultrasound system (SARUS) [66,67] is a huge but very flexible 1024-channel experimental ultrasound system that features 320 Xilinx Virtex-4 FPGAs for processing and more than 320 GB of RAM. It can sample all of its 1024 channels simultaneously with a sampling rate up to 70 MHz, but can only process 256 in real-time.
4.3 Designing an Ultrafast Probe

4.3.1 Avoid Heat from Processing

High-frame-rate (HRF) imaging is extremely compute-intensive. [4] estimates several TFLOP/s are required for gray-scale compounded plane-wave ultrafast imaging. The first problem to solve when designing a digital probe for ultrafast imaging is to move the power-hungry processing from the probe into the PC, where there are no thermal power constraints. Current state-of-the-art GPUs like the Nvidia Tesla V100, provide sufficient processing power (15 TFLOP/s) and I/O bandwidth (1.2 Tb/s) for this task.

However, this requires the digital raw data from all channels to be transferred from the probe to the PC: A 64-channel probe sampling with 12 b at 32.5 MS/s produces a data stream of 24.96 Gb/s. This rate is not yet supported with current common interfaces like USB 3.1 (10 Gb/s) and USB 3.2 (20 Gb/s). USB Type C allows simplex datarates up to 4x10 Gb/s, along with an auxiliary USB 2.0 and supply power. However, currently available copper-based USB Type C cables providing these rates are restricted to less than 1 m length, which would make the handling of the probe uncomfortable for the user. Higher data-rates are possible with optical links, which can be easily scaled to even higher data-rates to support even more channels. Optical links also impose no practical limits on the length. Optical point-to-points links providing 40 Gb/s or even 100 Gb/s are widely used in data centers today, as they provide higher throughput at lower energy consumption compared to copper-based solutions [110]. We thus decided to equip our probe with a link based on the Quad Small Form-factor Pluggable (QSFP) standard, which allows data rates of up to 40 Gb/s over an optical fiber, requires as little as 1 W, and is readily available at low cost (100-200 $ USD).

Being able to stream raw data from the probe also allows buffering entire acquisitions sequences (several GB) for offline processing, as high-bandwidth high-capacity memory is easily provided in the connecting PC system. Providing this buffering in the space and power constraints of the probe is not easily possible. Offline processing is interesting for applications, where real-time output is not needed, i.e. for Vector Flow Imaging, where the output can be computed offline before a
slow-motion playback is provided to the operator.

### 4.3.2 Exploit Ultrafast Imaging to Reduce Energy Consumption

Plane-wave imaging allows to obtain a comparable B-mode image [111] with much fewer shots than what is required for conventional line-by-line imaging: [112] reports that a good image can already be produced with 21 plane-waves. Considering plane-wave imaging with a pulse repetition frequency (PRF) of 5 kHz, a complete frame can be captured in a short pulse burst taking only 4.2 ms. Assuming a target frame-rate of 25 Hz, the front-end can be turned off for almost 90% of the time. On top of this, also less energy is required for transmit due to the lower number of transmissions.

![Figure 4.1](image_url)

**Figure 4.1** – Ultrasound front-end power consumption for B-Mode imaging at 25 Hz: Conventional imaging may use the whole frame time period to scan the entire field-of-view (FoV). Ultrafast methods capture the entire FoV with much fewer shots. This drastically reduces the required energy to capture a frame as less energy-intensive transmissions are required and the front-end can be powered down for a large share of the time.

Line-by-line imaging requires many more shots to scan the entire field-of-view (FoV) sequentially. As this scanning process may take as long as the entire frame period, the front-end may not be turned off. As illustrated in Fig. 4.1, performing B-mode imaging with plane-wave isonification allows to significantly reduce the average front-end power consumption. This relaxes the thermal budget of the probe and allows
4.3. DESIGNING AN ULTRAFAST PROBE

placing more channels in the digital probe. To materialize these power savings, the front-end hardware of the probe must support power down modes with fast transition times and low power sleep states. Our probe is optimized for this (Section 4.6.2) and equipped with a dynamic power management system (Section 4.7.2), which activates the best possible power mode at all times. The achieved energy savings and quality impact will be quantified in Section 4.9.

4.3.3 Boost Mode for High-Frame Rate Imaging

Performing normal rate B-Mode imaging with ultrafast methods has allowed us to reduce the average power consumption of the front-end sufficiently to equip the probe with a front-end that allows simultaneous sampling of all transducer channels. This enables high-frame-rate (HFR) imaging required for Vector Flow or Elastography. However, operating the probe continuously in HFR imaging will result in a power consumption which no longer allows keeping the probe’s surface temperature within regulatory requirements with passive cooling only.

On the other hand, many HFR applications not necessarily require continuous operation. A vector flow slow motion video or elastography map only requires a brief acquisition sequence taking no longer than a few seconds to capture. One option is to operate the probe in a dual-mode operation, where the probe is operated by default in normal-rate B-mode imaging. With this mode, the right position of the probe can be found before activating an advanced mode requiring HFR imaging. By dynamically controlling the imaging parameters (e.g., frame-rate) of the normal mode the probe can be kept at a temperature, which allows that the HFR boost-mode to be activated for a certain duration and periodicity indefinitely, without ever having to turn the system off for cooling.
4.4 System Architecture

Fig. 4.2 shows the system level architecture of our ULTRALIGHT imaging system: It consists of our digital transducer probe, called LIGHTPROBE, connected to a host PC. While the probe is mainly responsible for the raw signal acquisition, including the ultrasound isonification, analog-to-digital conversion, and data transmission, the host PC runs the ultrasound imaging application, which controls the probe and performs the processing.

Figure 4.2 – ULTRALIGHT System Block Diagram: Our digital transducer probe, called LIGHTPROBE, connects to a host system over a unidirectional digital optical link. Control (USB 2.0) and power (max. 15 W) are provided over a single USB Type C. The PC is equipped with an adapter to move the raw data received over the optical link into the main memory.

Tbl. 4.1 summarizes the hardware specifications of the probe. The probe integrates the piezo-electric transducer array along with a 64-channel ultrasound front-end. It features a Field-Programmable-Gate-Array (FPGA) for control, data aggregation, and processing as well as multiple communication interfaces and a power supply module. The probe connects to the host PC over a unidirectional optical link for raw data transfer, and a USB Type C cable to provide power (max. 15 W) and a communication channel to control the probe. On the PC, an adapter-card provides the optical link connector.
4.4. SYSTEM ARCHITECTURE

Table 4.1
LIGHTPROBE Hardware Specifications

| System                                      | - 64 RX/TX channels with size of $25 \times 45 \times 220 \text{mm}^3$  
|                                             | - Xilinx Artix-7 FPGA (XC7A200T-2), 1 GB DDR3 Memory               |
| Interfaces                                  | - up to 26.4 Gb/s optical interface over QSFP                   
|                                             | - USB Type C (Power max. 15 W, USB 2.0 for control, UART)       
|                                             | - Wireless LAN IEEE 802.11b/g/n (not used in this work)          |
| TX stage                                    | - per channel configurable bipolar 64 pulse sequences           
|                                             | - 100 Vpp, 0.625 to 20 MHz pulse frequency                      
|                                             | - Delay Range 102.4 $\mu$s, Delay Resolution 0.78 ns           |
| RX stage                                    | - Bandwidth 14 MHz (AAF), 12 b up to 32.5 MS/s ADC              
|                                             | - digitally controlled, variable gain amplification: -5 to 31 dB |

The USB connector or the integrated WLAN module also allows connecting the probe to portable devices such as tablets or smartphones. For these cases, the required data bandwidth on the digital link has to be reduced with on-probe processing, which is not discussed in this work. Fig. 4.3 summarizes the possible use-cases for the LIGHTPROBE.

![Figure 4.3 – Possible LIGHTPROBE Application: Since the LIGHTPROBE contains all relevant hardware for ultrasound imaging it can be connected to a large range of devices enabling many different kind of application scenarios.](image-url)
4.5 Host PC Implementation

To demonstrate real-time software-defined imaging, the LightProbe is connected to a host PC providing sufficient processing resources: Besides raw processing power, sufficient interface and bus bandwidth is required to sink the data streamed from the probe and forward it to the processing units. For our UltraLight system, we use a standard PC with a Quad-Core Intel Xeon 3.5 GHz Processor and 64 GB DDR4 memory. The PC is equipped with an NVIDIA GTX1080 Graphics Controller with 8 GB RAM.

![UltraLight system in operation](image)

**Figure 4.4** – The UltraLight system in operation. The pictures shows the host PC and the LightProbe in an earlier state without the housing.

To interface the optical high-speed link with the PC, we implemented a QSFP to PCIe3.0 link adapter using a Xilinx Kintex Ultrasync KCU105 development board. We used this board since it was readily available and only a small fraction of its resources was utilized. In practice, the adapter can be implemented much simpler and cheaper as elaborated in detail in Section 4.10.2.

A picture of the overall UltraLight system is shown in Fig. 4.4.
4.6 LightProbe Hardware Implementation

The main goal when designing LightProbe was to realize a probe that could be considered truly handheld. To achieve a very compact design, the LightProbe is implemented with a specially designed PCB stack, which is shown in Fig. 4.5. The stack has a size of $220 \times 25 \times 45 \text{mm}^3$, consists of two 32-channel TX boards, a high-voltage supply module, an interface board, and a motherboard. The latter carries an FPGA module and provides the RX stage and the optical connectors. See Fig. 4.6 for pictures of the individual PCBs. The following subsections describe the different subsystems of the probe in detail.

Figure 4.5 – The LightProbe PCB stack with the piezoelectric transducer.
4.6.1 Ultrasound Transducer Array

The LIGHTPROBE is currently equipped with a custom 64-element 4 MHz linear phased-array transducer fabricated by Fraunhofer IBMT (Germany). More details about the transducer can be found in [43]. Since the transducer is connected to the front-end with a plug, it can be easily replaced. We support transducer arrays with up to 64-elements without the need for multiplexing. To support transducers with up to 256 elements the PCB stack could easily be extended with a 4:1 multiplexer board.
4.6.2 Ultrasound Front-End

The ultrasound front-end supports 64 channels and is built from discrete components. A detailed block diagram of the front-end is shown in Fig. 4.7. It contains the hardware to emit ultrasonic pulses, as well as to amplify and digitize the received echoes: The pulse waveforms are generated by a \textit{TX-Beamformer} (TXBF) and converted to high-voltage signals required to excite the piezoelectric transducer elements by a \textit{HV-Pulser}. On the receive path, a \textit{TX/RX-Switch} (T/R) blocks the high-voltage pulses to protect the sensitive analog receive front-end (AFE), which contains the amplifiers and the analog-to-digital converters (ADC). The components for the front-end are chosen to provide low power consumption, good and fast power down modes, and enable a compact implementation.

For the TX pulser, an integrated 8-channel chip (HV7350, Microchip) is used, which employs a special direct-coupling topology for the gate drivers such that no additional supplies besides the positive and negative transmit voltage (±50V) and a control supply are required to power the drivers. Leakage currents through the high-voltage MOSFETs can significantly contribute to the heat generation as even small currents may cause a large dissipation due to the large current drop (100V). We chose this chip due to its fast (<500 µs) power down mode reducing the MOSFET leakage below 10 µA (<1mW).

Dedicated 8-channel TXBF chips (LM96570, TI) are used that create the pulser MOSFET control signals. The TXBF supports per-channel configurable bipolar pulse sequences with a frequency of 0.625 to 20 MHz over a delay range of 102.4 µs with a resolution of 780 ps. Using TXBF chips does not allow completely arbitrary waveform generation, but simplifies the transmit path design compared to implementing the TXBF on the FPGA, which would require more I/O pins on the FPGA and more PCB routing space to connect all HV MOSFET switch signals to the FPGA. To support 64 channels, eight TXBF and pulser chips are placed on two 32-channel TX boards plugged to the motherboard from both sides.

The RX stage digitizes the received echoes. The ultrasound echoes cover a large dynamic range of over 100 dB, due to the attenuation of the acoustic pulse as it is traveling through tissue. In practice, the full dynamic range is not needed all time, and an ADC with
lower dynamic range can be used in combination with a variable gain amplifier (VGA), which adjusts its amplification over time while capturing data. We selected an integrated analog front-end (AFE) chip (AFE5851, TI), which provides both a VGA (-5 to 31 dB) and an ADC (12b up to 32.5 MS/s). The variable gain amplification profile is digitally programmed, and its playback is triggered by the FPGA. The chip supports 16 channels at a very low power consumption of just 39 mW per channel. To achieve this power efficiency, two channels share a 65 MS/s ADC in a time-multiplexed manner, which results in

**Figure 4.7** — Detailed top-level block diagram of the **LightProbe**, showing how the various subsystems are connected within the probe.
a half-sample period time-offset between the odd and even channels. This offset is compensated in the FPGA on the probe, such that the connecting system receives fully aligned samples. We selected this component due to its low active power consumption and because it supports two power-down modes (64mW and 5mW per chip), which can be excited very quickly (<50µs and <200µs). Four AFE chips are used to provide 64 channels.

The AFE inputs are protected from the HV TX pulses with eight 8-channel T/R switches (TX810, TI) that clamp the input signal. The digital samples from the AFE are transferred to the FPGA (6.24 Gb/s per AFE chip) over a high-speed serial interface. All front-end chips are kept in sync with the required reference clocks and configured over a Serial Peripheral Interface (SPI) bus operating with up to 75 Mb/s. A custom hardware SPI accelerator in the FPGA enables fast reconfiguration of the TX delays in the TXBFs between shots.
4.6.3 Final LightProbe Assembly

Fig. 4.8 shows the final LightProbe assembly, with the probe in its milled aluminum housing and the two cables required to connect the probe to a PC.

Figure 4.8 – The LightProbe in its milled aluminum housing. Two cables (USB and QSFP) are required to connect the probe to the PC for ultrafast imaging: The USB for power and control, and the optical fiber (QSFP) for data.
4.6.4 FPGA Subsystem

The FPGA aggregates the data from the ADCs and implements the high-speed link to the host system. A microprocessor instantiated on the FPGA runs the firmware to communicate with the host system and to control the probe.

The FPGA is provided on a compact (4×5 cm²) commercial module (TE0712, Trenz Electronics). It features a Xilinx Artix-7 FPGA (XC7A200T-2FBG484C) with four 6.6 Gb/s transceivers to connect the optical interface. The module has 8 Gb of DDR3 memory and 256 Mb Quad-SPI Flash for configuration and operation. The chosen FPGA has plenty of spare resources to implement on-probe buffering or processing to connect as well with portable devices. We selected this module to speed up the implementation time of our probe and give us flexibility for future on-probe processing. Thus it is not particularly optimized for low power consumption.

![LightProbe FPGA Top-level diagram](image)

**Figure 4.9** — LightProbe FPGA Top-level diagram: It features the control infrastructure (upper half) and the receive data-path (lower half). The control infrastructure is built around a soft-microcontroller (MicroBlaze) equipped with the required peripherals. A hardware finite-state-machine (LP FSM) orchestrates the overall signal acquisition by sending the required synchronization signals (TX-, TGC-, RX-trigger) with precise timing. The main data-path contains the AFE interface (AFE Link), which produces per acquisition a raw data packet containing the samples of all channels. The data-packet may be processed in the processing domain on the probe, before it is forwarded to the optical interface block (Aurora Link).
The FPGA design (Fig. 4.9) contains a Xilinx MicroBlaze soft-core microcontroller with 256 kB memory equipped with the required peripherals (timer, interrupt controller, I/O peripherals). A custom hardware finite-state-machine (LP FSM) is used to orchestrate the overall ultrasound acquisition depending on the settings requested by the connecting system. This state-machine allows precise PRF control and assures that the right number of samples are captured and that the acquisition is in sync with the transmit pulse and the TCG profile.

The receive data-path on the FPGA from the AFE interface to the optical link contains three stages:

1. First **AFE Link** abstracts the high-speed data-streams from the AFEs and interacts with the LP FSM to produce a data packet, which contains the raw samples from all channels over a configurable time-period properly aligned to the transmit event.

2. Second, the packet is forwarded to the *processing domain*, where any potential preprocessing block can easily be instantiated and connected. Even though we postulate and allow a complete software-defined system architecture, we still allow performing hardware preprocessing in the FPGA on the probe. In the current design, we placed a delay adjustment (DelAdj) block that applies a $f_s/2$ delay filter to the odd channels to compensate for the time-interleaved sampling of odd and even channels of our AFE chip. Additional processing blocks can be easily added.

3. Third, the processed packet is forwarded to the **Aurora Link** which sends it to the PC using the Xilinx Aurora 8b/10b protocol taking care of DC-free coding and link synchronization. Due to the coding loss, only 80% of the link bandwidth is used for payload data. On the PC, the packet is received, and its content is written into main memory using a direct-memory-access (DMA) transfer.

The maximal data rate of our optical link is 26.4 Gb/s given by the maximal transducer speed of the FPGA transceivers ($4 \times 6.6$ Gb/s). Given our 4 MHz transducer we operate the ADC with a 20 MS/s sampling rate producing a peak data rate of only 15.36 Gb/s. Con-
sidering coding loss and protocol overhead, the link is run at reduced speed (6.25 GHz) providing a raw bandwidth of 25 Gb/s over the four fibers in our cable.

Note that there are plenty of spare resources available on the FPGA (FF, LUT, DSP <30%) to extend the data-pipeline with additional processing. The FPGA utilization is reported in Table 4.2.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Used</th>
<th>Available</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUT</td>
<td>28'226</td>
<td>133'800</td>
<td>21.1%</td>
</tr>
<tr>
<td>LUTRAM</td>
<td>8'602</td>
<td>46'200</td>
<td>18.6%</td>
</tr>
<tr>
<td>FF</td>
<td>34'329</td>
<td>267'600</td>
<td>12.8%</td>
</tr>
<tr>
<td>BRAM</td>
<td>97</td>
<td>365</td>
<td>26.6%</td>
</tr>
<tr>
<td>DSP</td>
<td>64</td>
<td>740</td>
<td>8.6%</td>
</tr>
<tr>
<td>IO</td>
<td>152</td>
<td>285</td>
<td>53.3%</td>
</tr>
<tr>
<td>GT</td>
<td>4</td>
<td>4</td>
<td>100.0%</td>
</tr>
</tbody>
</table>
CHAPTER 4. LIGHTPROBE & ULTRALIGHT

4.7 System Operation and Firmware

In this section, we first outline how an ultrasound imaging application running on the PC interacts with the LIGHTPROBE and what operations are performed to trigger an ultrasound acquisition until the captured raw data is available on the main memory of PC for processing. In subsections that follow, we then describe the programming model and provide some insight into the power and thermal management systems implemented on the probe.

After powering up and self-check, the LIGHTPROBE waits for an application to be launched on the PC. When an application is started, the LIGHTPROBE has to be first configured over USB for the desired imaging mode by setting the transmit and receive parameters. The optical link is enabled, and the QSFP-to-PCIe adapter is initialized. The ultrasound application can now use the probe for imaging. Upon issuing a trigger signal, the LIGHTPROBE emits the ultrasonic pulses, captures the echoes and sends the data to the host PC. The LIGHTPROBE then waits for the next trigger signal or to be reconfigured for another scenario.

To ease ultrasound application development, we provide a Python library that abstracts away the driver interactions. After having configured the probe, with a simple `data = libLP.getNext()` call, the required commands to trigger an acquisition are sent to the probe, the data is transferred, and a handle to the captured raw data is returned.

4.7.1 Programming Model

The LIGHTPROBE provides a flexible model to program acquisition sequences (Fig. 4.10). The basic building block of a sequence is a shot. A shot contains the transmit event, i.e., the emission of an ultrasound wave, followed by the receive phase, i.e., the period during which the echoes are captured. A TX profile defines the parameters (delays, pulse shape) of the emitted wave. Multiple shots can be combined to a burst. Within a burst, every shot can use a different TX profile.

Since changing the TX profile requires to reconfigure the TXBF chips, a short re-configuration phase is inserted. Depending on the triggering settings, shots, burst or multiple bursts can be launched
with a single trigger command issued by the ultrasound application running on the host PC.

![Diagram](image)

**Figure 4.10** – Acquisition Sequence Programming (LEFT): A burst contains multiple shots. A shot is a TX event followed by a RX phase.

### 4.7.2 Dynamic Power Management

As elaborated in Section 4.3, fast and good power down modes are required to benefit from the power savings possible by imaging strategies that allow duty-cycling of the front-end.

To keep the probe as cool as possible all unneeded modules must be prevented from generating heat. In a complex system as the LIGHT-PROBE, where individual supplies can be turned off, and many chips support many different power modes, the size of the combined state space of feasible system power states explodes. We selected the most useful in terms of transition time and the power savings they provide:

- **ready** (RDY): the entire front-end is powered up, the RX path is sampling, and the TX path is ready to emit pulses. The probe has to be in this state to acquire raw data.

- **power down** (PDN): the front-end power supplies are shut down. Recover operation from this mode takes 350 ms due to the time required to recharge the supply domains.

- **receive power down** (RX-PDN): the power supplies remain on, but the TR switches, the TX pulsers, and the AFE are put in power down, which also disables the AFE-FPGA link. Recovering from this mode requires to re-synchronize the AFE-FPGA links, which takes 25 ms.
**receive standby** (RX-STBY) the TR switches, the TX pulsers, as well as the ADC, are in standby. The AFE-FPGA link is only partially powered down to keep the synchronization, which enables to resume operation in only 600 µs.

The **dynamic power management**, which is part of the firmware, automatically activates the power down state with the highest possible power savings to keep the average power consumption as low as possible (Fig. 4.11, Right). To do this, it first computes the length of the idle period ($t_{idle}$) between two bursts from the current imaging settings (FPS, PRF, RX period) and then selects the best possible power mode to enter given its transition time and overhead. The selected mode is activated immediately after the last receive phase of the burst. A timer is set to schedule the wakeup, such that the probe will be back in RDY shortly before the next shot starts.

**Figure 4.11** – Dynamic power management: If the idle period ($t_{idle}$) between two bursts is long enough for a power state transition (yellow, blue), the corresponding power mode is activated to reduce the power dissipation during the idle period.

### 4.7.3 Dynamic Thermal Management

Digital ultrasound probes, like other devices embedding active electronics, are thermally limited systems, which means for some operating modes they can produce more heat than can be dissipated. Continued operation within these operating modes can cause the device to heat up to a point where it exceeds prescribed thermal limits for safe operation.
Some ultrasound probes use active cooling. Air cooling is not a viable option since air inlets and moving rotors would impede proper disinfection of the probe. One solution is to use liquid cooling through the cable, which has been implemented in probes for 3D imaging [26]. However, to keep complexity and cost within reasonable bounds, passive cooling is the only option.

As it will take some time for the probe to heat up, one possible solution is to support some imaging modes only for a limited period of time until the probe has reached a critical temperature and needs to be put down for cooling. This mode of operation is already known from commercial digital probes for the portable market. Many systems do not allow uninterrupted operation and require the probe to be periodically turned off for cooling. For example, the Sonon 300 series Wi-Fi transducer probes (Healcerion, KOR) supporting B- and Color mode imaging are specified for a maximal 10 min operation with 10 min resting time [113]. Other probes [95] can be equipped with an external fan for longer scanning time or provide a docking station to cool probes back down to their operable temperature range. In our view, this provides not a good user experience to the operator, as the examination may need to be interrupted. Sustainable operation is thus preferable.

Providing sustainable operation with passive cooling only, requires the average dissipated power to stay below a certain level to assure that the surface temperature never exceeds 43°C to comply with medical safety regulations (IEC 60601-1) [96].

One way to ensure this, is to restrict the probe to modes, which when operated continuously in worst-case conditions (high ambient temperature) never heat the probe beyond 43°C. This puts overly strict restrictions on the system, as it is seldom in worst-case conditions and some modes do not need to be operated continuously. Another approach is to adapt the operation mode (or its parameter) depending on the current temperature of the probe.

The firmware of LIGHTPROBE features dynamic thermal management to enable this kind of thermal aware operation: The probe is equipped with multiple temperature sensors that are used as an input for our thermal control algorithms. As highlighted in Fig. 4.6 temperature sensors are placed at critical locations close to all major heat sources, which are the FPGA, the optical transceiver, the AFE chips, the TX pulsers, and the HV supplies. For each temperature sensor
we define a maximal tolerable value $T_{\text{max},i}$, such that when reached, the outside surface temperature does not exceed regulatory limits. We define the *thermal margin* $T_m$ as the minimal margin over all sensors. The firmware computes $T_{\text{max},i}$ every second:

$$T_m = \min_i T_{\text{max},i} - T_i \quad (4.1)$$

The milled aluminum housing of the LightProbe (see Fig. 4.12) features thermal contact areas to the FPGA and PCB stack to ease heat spreading, and provides sufficient heat capacity to temporarily sink the heat during high-performance modes without increasing the surface temperature significantly.

![Figure 4.12](image_url) — The open milled aluminum LightProbe housing. The housing provides on both sides heat contact areas to ease heat dissipation from the PCB stack to the housing surface. On the contact areas the gray anodization has been milled away.
In this work, we implemented two thermal controllers:

- First, a **Thermal-Aware-Performance (TAP)** controller that adapts imaging settings (the frame-rate in this work) to keep the probe below a given temperature.

- Second, a **Boost-Mode (BM)** controller that allows dual-mode operation of a continuously operable *default* mode (B-mode) and a *boost* mode (2 seconds of 210 Hz imaging) with a power consumption exceeding the sustainable cooling capabilities. The boost mode can be activated on-demand with a defined periodicity (every 5 seconds) to provide a consistent Quality of Service.

The first controller (TAP) sets the frame-rate depending on $T_m$ by inserting a delay $d_{th}(T_m)$ between the triggering of the bursts. The dynamic power management (Section 4.7.2) takes care of activating the best power down mode between bursts. Designing the controller response $d_{th}(T_m)$ is not straightforward as the controlled variable, i.e., the temperature, reacts with a delay to control inputs, i.e., framerate changes. Controlling a system with a delayed response using a standard P(ID)-controller is susceptible to instability. Instability can be avoided by setting the slope of the linear control response very flat. However, the slope cannot be set arbitrarily flat as, at a certain temperature, a specific frame-rate is required to avoid the system from overheating. In our case, controller and thermal stability was not achievable with a linear control response.

We thus designed $d_{th}(T_m)$ as a one-sided quadratic function (Fig. 4.13), which has a low response when sufficient thermal margin is available.

![Non-Linear Control Response](image-url)
(stability) and a progressive response on decreasing thermal margin (overheating avoidance).

\[ d_{th}(T_m) = \begin{cases} \alpha T_m^2 + \beta T_m + \gamma, & \text{if } T_m \leq \tau \\ 0, & \text{otherwise} \end{cases} \quad (4.2) \]

If the probe is cool enough \((T_m > \tau = 10^\circ C)\), the thermal management does not interfere, and the probe runs at the configured frame-rate. For smooth frame-rate changes and to remove sensor noise, a first-order IIR low-pass filter is applied to \(d_{th}\). Currently, TAP only sets the frame-rate. Other settings could be controlled as well. We use this controller to provide normal B-mode imaging at best possible performance (frame-rate) in all temperature conditions.

The second controller (BM) uses a TAP controller with a modified set-point \((\tau \text{ set to } 15^\circ C)\) to let the probe cool down during default mode to build up additional thermal margin for the boost mode. A LED informs the user when the system is thermally ready \((T_m > T_m, \text{bst rdy} = 10^\circ C)\) to switch to the boost mode. The user can do so by pressing a button on the probe. By choosing the set-point of the default mode (using \(\tau\)) and the thermal margin for activation \((T_m, \text{bst rdy})\) appropriately depending on the heat output of the boost mode, it can be ensured that the boost mode can be activated with a lower bounded duty cycle, without ever overheating the system.
4.8 Methods

4.8.1 Imaging Quality Assessment

To assess the imaging quality of the LightProbe, we implemented several imaging strategies (both conventional and ultrafast) and quantified the achieved contrast and resolution. For the resolution and contrast measurements, a CIRS 054GS phantom (0.5 dB/cm-MHz) is used. If not stated otherwise a 4 MHz 2 period [+1,-1,+1,-1] transmit pulse is used. All channels were sampled with a sampling frequency of 20 MHz, and the RF data was processed offline.

For all strategies, the following processing steps were performed: The raw RF data was bandpass filtered and converted into analytical signals. Dynamic receive beamforming was performed on a polar grid with a dynamic Hanning apodization. The beamforming grid composes out of 91 radial lines with 0.86° axial spacing and 192.5μm radial point spacing and covers a field-of-view of 78° and a depth of 0-19.3 cm. After beamforming we performed 2D scanline interpolation, envelop extraction and scan-conversion. The quality metrics were extracted, and the images were log-compressed for displaying.

The following imaging strategies were evaluated:

**TXF1** Conventional line-by-line imaging with dynamic receive focusing on 91 scanlines (0.86° spacing, 78° field-of-view) and a fixed transmit focus at 5 cm.

**TXF2** Same as TXF1 but with sequential multi-zone transmit focusing to 5 and 10 cm. The scanlines from the two acquisitions with different transmit focus were linearly blended over the depth.

**STA64** synthetic transmit aperture imaging [36] with 64 sequential single element isonifications. 64 low-resolution images were coherently summed to one high-resolution image (HRS) before envelop extraction.

**PW31** coherent-compounded plane-wave imaging using 31 waves with angles from -15° to +15° with 1° increments.

**VS13** synthetic aperture imaging with 13 virtual sources [114, 115] placed behind the transducer on a 90° 6.3 mm radius arc. The
13 LRS images were coherently combined.

**VS13-CE** same as VS13, but extended with coded excitation [116, 117] using two 48 bipolar pulse sequences emitted with 8.88 MHz. The two acquisitions were decoded and combined before fed to the normal processing pipeline.

For every strategy, we quantify the *resolution* at different depths with the full-width at half-maximum (FWHM) of the vertical point targets in the phantom. The theoretical FWHM in degrees of the system assuming full use of the aperture for transmit and receive is according to [2] \( \phi_{th} = \frac{1.206 \lambda}{a \cdot 360/2\pi} = 2.11^\circ \) with \( a \) the transducer aperture size (1.26 cm) and \( \lambda \) the wavelength (0.385 mm).

*Contrast* is quantified using the anechoic cylinders in the phantom using the contrast ratio (CR) [2] and the contrast-to-noise-ratio (CNR) [118]:

\[
CR = \frac{\mu_{\text{out}} - \mu_{\text{in}}}{\mu_{\text{out}} + \mu_{\text{in}}}, \quad \text{CNR}_{\text{dB}} = 20 \log_{10} \frac{\mu_{\text{in}} - \mu_{\text{out}}}{\sqrt{\sigma_{\text{in}}^2 + \sigma_{\text{out}}^2}/2},
\]

(4.3)

with \( \mu \) the mean and \( \sigma^2 \) the variance of the B-mode image within and around the cyst.

The *signal-to-noise ratio (SNR)* as a function of depth is computed along the center line in the B-Mode picture before log-compression using mean and standard deviation values obtained from 100 consecutively captured images of a CIRS 040GSE phantom using VS13:

\[
\text{SNR}_{\text{dB}}(z) = 20 \log_{10}(\mu(z)/\sigma(z)).
\]

(4.4)

### 4.8.2 PRF, Frame-Rate and Power Measurements

The maximal achievable pulse repetition frequency (PRF) is physically limited by the time it takes the acoustic pulse to reach the desired imaging depth and for the echoes to return to the transducer. In a real system, the achievable PRF is further limited by configuration overheads between acquisitions and the data-sinking capabilities of the receive path. Since the digital link of LIGHTPROBE allows to stream the received data off the probe continuously, we can operate the system perpetually with a very high PRF as opposed to systems,
4.8. METHODS

which have to periodically stall their operation to deplete internal buffers over a low-rate link.

For the power and PRF measurements, we configured the probe to take 2048 samples per shot, which takes 102.4 µs with a sampling rate of 20 MHz. Thus, a maximum PRF of 9.8 kHz can be reached in theory. To measure the consumed power, the probe was supplied with a Keysight N6705B DC Power Analyzer instead of the PC using USB.

4.8.3 Thermal Control

Both thermal-aware controllers where implemented and executed on the LIGHTPROBE, while the power consumption and achieved frame-rate was captured. Profiling starts from a cold system (21°C ambient temperature). The probe was configured for the PW31 imaging scheme with the 31 acquisitions performed in 6.5 kHz PRF burst. To obtain more conservative results, we chose PW31 over VS13 as it requires more shots per frame compared to the powered-optimized VS13 operating with 13 shots only. For normal B-mode imaging, the free-running frame-rate is set to 210 Hz. The thermal controller will reduce the frame-rate to a steady-state value that is thermally sustainable. This high free-running frame-rate is chosen to force reasonably fast convergence to the steady-state value. The boost-mode is configured as a 2 second period of acquiring PW31 frames at a rate of 210 Hz. This setting is reasonable to capture a flow sequence, which can be replayed at lower frame-rate. Controller input and output variables are extracted over a debug interface.
4.9 Results

4.9.1 Imaging Quality Assessment

Fig. 4.14-4.16 show the imaging output for all strategies. We show a wide-angle image of the CIRS phantom down to 9 cm with the vertical point targets aligned on the center axis of the transducer. The lateral (axial) resolution is quantified for every point target by measuring the FWHM in degrees. We plot the depth resolution profile and report the average resolution over the depth 2-9 cm.

The system resolution is almost perfectly achieved for STA64 and PW31 over the entire depth, which is expected for these strategies. For TXF1 and TXF2 this is only the case at their transmit focal depths (5 and 10 cm), which is expected as well. VS13 achieves a slightly worse resolution but also needs by far the least amount of shots to obtain an image. The coded-excitation in VS13 impairs the resolution only marginally.

For contrast, we show the image section around the anechoic cylinder at 4 cm depth and report the CR and CNR using the regions indicated in the image. As expected STA64 provides a poor contrast (CNR: 3.42 dB) due to the little amount of acoustic energy emitted into the tissue by using only a single transducer element for transmission. Using all elements for transmit yields a better CNR for the other synthetic aperture based methods PW31 and VS13 (CNR: 6.86 dB and 7.03 dB). As expected, the best contrast is achieved with conventional sequential scanning (TXF1, TXF2) which uses transmit focusing (CNR: 7.60 dB and 7.58 dB). Coded excitation boosts the penetration depth and increases the VS13 contrast at 4 cm depth from 7.03 dB to 7.17 dB and at depth 7 cm from 3.56 dB to 6.63 dB (see Fig. 4.16), but requires twice the number of shots and has a more energy intensive TX pulse.

This evaluation confirms that with ultrafast methods (PW31 & VS13) almost the same imaging quality can be achieved for still images as with standard methods. Compared to TXF2, the ultrafast methods PW31 and VS13 require 6× and 14× fewer shots, while dropping the average resolution less than 0.3° (VS13) and the contrast less than 0.72 dB (PW31).
4.9. RESULTS

<table>
<thead>
<tr>
<th>TXF1 (1 Zone)</th>
<th>TXF2 (2 Zones)</th>
</tr>
</thead>
<tbody>
<tr>
<td># Shots: 91</td>
<td># Shots: 2 × 91</td>
</tr>
</tbody>
</table>

**Figure 4.14** — Imaging quality of the LightProbe for different transmit strategies (TXF1, TXF2): For each strategy, the lateral (axial) -6 dB point spread is measured at multiple depths (red circles), as well as the contrast [CR] and contrast-to-noise [CNR] ratio of an anechoic region. For the lateral resolution we report the average angle spread for the points at 2-9 cm depth. The resolution profile shows the theoretical system FWHM resolution (red line) in degrees $\phi_{th}$. For each strategy the number of required acquisition per frame (shots) is listed.
Figure 4.15 — Imaging quality of the LightProbe for different transmit strategies (STA64, PW31). See Fig. 4.14 for detailed plot description.
4.9. RESULTS

<table>
<thead>
<tr>
<th></th>
<th>VS13</th>
<th>VS13 with CE</th>
</tr>
</thead>
<tbody>
<tr>
<td># Shots:</td>
<td>13</td>
<td>2 × 13</td>
</tr>
<tr>
<td>Resolution [50 dB]</td>
<td>![Resolution images]</td>
<td>![Resolution images]</td>
</tr>
<tr>
<td></td>
<td>avg: 2.7°</td>
<td>avg: 2.8°</td>
</tr>
<tr>
<td>Contrast [40 dB]</td>
<td>CR: 0.73 CNR: 7.03 dB</td>
<td>CR: 0.75 CNR: 7.17 dB</td>
</tr>
<tr>
<td></td>
<td>![Contrast images]</td>
<td>![Contrast images]</td>
</tr>
<tr>
<td>Contrast deep [40 dB]</td>
<td>CR: 0.46 CNR: 3.56 dB</td>
<td>CR: 0.73 CNR: 6.63 dB</td>
</tr>
<tr>
<td></td>
<td>![Contrast deep images]</td>
<td>![Contrast deep images]</td>
</tr>
</tbody>
</table>

**Figure 4.16** – Imaging quality of the LIGHTPROBE for different transmit strategies (VS13, VS13 with CE). See Fig. 4.14 for detailed plot description.
4.9.2 Achievable PRF and Frame-Rate

In the standard case, where the TX profile needs to be updated between the shots, all delays (1.8 kb) are programmed into the TXBF chips in 25 µs with our SPI-HW-accelerator. In this case, the LightProbe achieves a PRF of 6.5 kHz.

The maximal frame-rates we can achieve for the various imaging strategies with a 6.5 kHz PRF are listed in Tbl. 4.3.

<table>
<thead>
<tr>
<th>MODE</th>
<th># Shots per Image</th>
<th>max. FPS [Hz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>TXF2</td>
<td>2 × 91</td>
<td>36</td>
</tr>
<tr>
<td>TXF1</td>
<td>91</td>
<td>71</td>
</tr>
<tr>
<td>PW31</td>
<td>31</td>
<td>210</td>
</tr>
<tr>
<td>VS13</td>
<td>13</td>
<td>500</td>
</tr>
<tr>
<td>VS13 CE†</td>
<td>2 × 13</td>
<td>250</td>
</tr>
<tr>
<td>QS64†</td>
<td>2 × 13</td>
<td>102</td>
</tr>
</tbody>
</table>

† estimated values

With all strategies, we reach a frame-rate above 30 Hz. With VS13, we can achieve 500 Hz. The high-frame-rate operation is demonstrated with videos\(^1\). In the next chapter (Chapter 5), we will use the LightProbe to track trajectories of individual gas bubbles at 200 fps.

For these very high frame-rates, we buffer the raw data on the PC’s main memory and interrupt the acquisitions after a few seconds to store the data on disk to process it later offline. Note that these high frame-rates are sustainable by state-of-the-art GPU-based beamformers [18, 19].

4.9.3 Power Measurements

Fig. 4.17 shows a detailed power breakup of the LightProbe for different power state and operation modes: If the probe is in RDY state (8.1 W) and not emitting pulses, more than half of the power

\(^1\)http://hdl.handle.net/20.500.11850/283465
Figure 4.17 — LightProbe Power Measurements: LEFT: LightProbe power consumption (measured) in various operation mode and power states (see text), showing the power contributions from the Ultrasound front-end, Optical Link (including transceiver in FPGA) and FPGA-Subsystem. MIDDLE LEFT: More detailed power analysis of the ultrasound front-end showing the power distribution in the four main power modes. MIDDLE RIGHT: Detailed power breakup of the AFE in RDY and the optical link when on. TOP RIGHT: Potential power improvements for RDY (estimated), see Section 4.10.3 for details.
(4.4 W) is spent in the ultrasound front-end. The most significant contribution to the front-end power is the AFE (2.45 W). Thus the AFE should be turned off whenever possible. If the PRF and frame-rate settings allow, the front-end can enter the RX-STBY or RX-PDN power mode in between acquisitions consuming only 1.9 W or 1.26 W. In idle periods >350 ms, the front-end can be powered down (PDN) consuming as little as 25 mW. If the probe is performing high-frame-rate ultrafast imaging (VS13, 500 Hz), the probe power consumption rises to over 10 W due to the additional power spent for emitting pulses. If the probe is performing normal-rate B-mode imaging (VS13, 30 Hz) the power consumption decreases below the idle level (RDY) as more power is saved with power management than used for transmission.

We report the front-end power consumption separately from the FPGA subsystem and the optical link. Note that we focused on this work on reducing the consumption of the front-end as it cannot be scaled down as easily as the digital counterparts. Currently, 3.64 W are spent for the FPGA subsystem and the optical link independent of the operating mode. As elaborated in Section 4.10.3 these parts can be further optimized.

Tbl. 4.4 reports the ultrasound front-end power consumption for various imaging modes (TXF2, TXF1, PW31, VS13) running at their maximal possible frame-rate as well as rates for B-mode imaging (30 and 50 Hz). Even though the modes provide substantially different maximal frame-rates (36-500 Hz), all modes consume around 7.1 W at their max-rate, as raw data is acquired with the same 6.5 kHz PRF. VS13 provides a 14× higher frame-rate compared to TXF2 at the same power consumption.

<table>
<thead>
<tr>
<th>MODE</th>
<th>max. FPS (6.5kHz)</th>
<th>Avg Pwr max. FPS</th>
<th>Avg Pwr 50 Hz</th>
<th>Avg Pwr 30 Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>TXF2</td>
<td>36 Hz</td>
<td>7.1 W</td>
<td>n/a</td>
<td>6.3 W</td>
</tr>
<tr>
<td>TXF1</td>
<td>71 Hz</td>
<td>7.1 W</td>
<td>5.5 W</td>
<td>3.9 W</td>
</tr>
<tr>
<td>PW31</td>
<td>210 Hz</td>
<td>7.1 W</td>
<td>3.0 W</td>
<td>2.4 W</td>
</tr>
<tr>
<td>VS13</td>
<td>500 Hz</td>
<td>7.1 W</td>
<td>2.2 W</td>
<td>2.0 W</td>
</tr>
</tbody>
</table>
4.9. RESULTS

Fig. 4.18 shows the measured ultrasound front-end power consumption for various modes and frame-rates (30 Hz to max), with and without dynamic power management activated. If it is deactivated and no power down modes are utilized, the front-end power consists of a static contribution (RDY power, 4.4 W) and a contribution growing proportionally with the frame-rate accounting for the power dissipated for ultrasound pulse emissions (TX). With dynamic power management, the average power consumption can be substantially reduced when there is sufficient idle time ($t_{idle}$) between frames to put the front-end into a power-down mode. At 30 Hz power savings of -6% (TXF2), -29% (TXF1), -50% (PW31) and -57% (VS13) can be achieved. For normal B-mode imaging a frame-rate of 30 Hz is sufficient. If ultrafast imaging (PW31, VS13) is used at this low rate, the average power consumption is drastically reduced (-62% 2.4 W PW31, -68% 2.0 W VS13) compared to conventional imaging (TXF2, 6.3 W) as the front-end can be powered down longer between frames. Note that without dynamic power management, the savings with ultrafast imaging would be less than half, only -29% in PW31 and -32% in VS13. This shows that dynamic power management is crucial to realize the potential savings ultrafast method offer.
4.9.4 Thermal Control

Fig. 4.19 shows the TAP controller performance starting from a cold system (21°C) until the system reaches a steady-state.

![Figure 4.19](image)

**Figure 4.19** – Measured performance of the LIGHTPROBE using the Thermal-Aware-Performance (TAP) controller: The top plots shows the average power consumption of the front-end (blue) along with the instantaneous frame-rate (FPS, red). The lower plot shots the thermal margin $T_m$ in °C (red) and the manipulated variable $d_{th}$ of the controller (blue). Controlling the frame-rate in combination with activating the provided power down mode allows the system to reach a steady state, which still provides a frame-rate of 48 Hz.

At first, the system is running with the configured free-running frame-rate (210 Hz) for 3 min, heating up until the thermal margin ($T_m$) is reduced to 10°C and the controller throttles the systems frame-rate by adding a delay ($d_{th}$) in between capturing frames. Immediately after throttling starts, the added delay $d_{th}$ is long enough (>0.8 ms) to activate RX-STBY between frames, which reduces the average power consumption. The system continues to slowly heat up for the next 15 minutes until it reaches a steady state with $T_m$ >5°C and a frame-rate of 47 Hz (1.5 kHz PRF in average). If the system would further heat up, e.g., when a more power intensive transmit pulse is used, $d_{th}$ would be further increased until at 20 ms there is enough time to activate RX-PDN.

Fig. 4.20 (LEFT) shows a thermal camera (FLUKE Ti95) image of the housing of the probe showing that the outside temperature stays below 43°C. Note that the transducer tip is currently the hottest...
part of the probe. This is only the case if the tip is not touching any surface (skin or phantom) serving as a cooling path. Note that the transducer temperature can be easily reduced by providing a heat conductance path to the metal housing, which is currently not given as the connecting parts (Fig. 4.8) are made out of ABS plastic.

![Thermal camera picture of the LightProbe in thermal regulation](image)

**Figure 4.20** – Thermal Performance: **LEFT**: Thermal camera picture of the LightProbe in thermal regulation. The housing temperature is < 37°C and the transducer 41°C, both are not exceeding 43°C. **RIGHT**: Noise measurement depending on depth and temperature.

Fig. 4.20 (RIGHT) shows the SNR of a B-mode image (VS13) along the center-line as a function of depth at different internal probe temperatures at the AFE chips. Over our temperature range (30-50°C) the noise performance is constant.

While simple controllers, such as the TAP, successfully avoid overheating, they do not provide a consistent QoS, since after providing the peak performance for a few minutes, they throttle the system and stay in throttled mode indefinitely, unless the probe is turned off for cooling.

Fig. 4.21 shows the system performance using the *boost-mode* (BM) controller, which supports dual-mode imaging with a frame-rate controlled *default* mode (D-Mode) and a Boost-Mode that can be activated on-demand. After startup the D-Mode provides again peak performance for a few minutes (20-100s) before it is throttled. Activating the Boost-Mode at a high rate (150-300s in Fig. 4.21) accelerates the heating process. Note that now the frame-rate of D-Mode (D-FPS) drops sufficiently low ($d_{th} > 20$ ms) to activate the second power down mode RX-PDN in between frames. At one point $T_{m}$ reaches $T_{m, bst rdy}$.
The background color of the bottom plot indicates whether the mode can be activated (green) or not (red). For 2 s, the high performance mode can be temporarily activated upon user request when \( T_m > T_m^{\text{bst rdy}} = 10^\circ C \). The top plots show the average power consumption of the front-end (blue) and the lower plot shows the thermal margin \( T_m \) in °C (red) and the manipulated variable \( d_{th} \) of the controller (blue). Initially, the system operates in the default imaging mode with a thermally-adapted frame-rate (D-FPS, red) and a boost mode (green) with a FPS of 210Hz (PRF 6.5 KHz) and the manipulated variable \( d_{th} \) of the controller (blue). Initially, the system operates in the default imaging mode and the background color of the bottom plot indicates whether the mode can be activated (green) or not (red).
= 10° C (@ 280s) and the user is no longer allowed to activate Boost-Mode for a few seconds until the system has cooled a bit. When Boost-Mode is not activated for a while, the system cools, and D-FPS is increased again (@ 350s). We demonstrate a consistent Boost-Mode launch rate (once every 5 seconds) by keeping the activation button pressed (400-450s). During these 50s 11 Boost-Mode bursts are launched. Even now, D-Mode still provides a frame-rate >6 Hz.

With the BM controller, we demonstrate that we can ensure the temporary use of a high-performance mode dissipating more energy than what is thermally sustainable in continuous operation, while still providing a consistent QoS for the operator.
4.10 Discussion and Comparison

Our power results in Tbl. 4.4 confirm that digital probes can significantly reduce (more than -60%) the average front-end power consumption during normal rate (30 Hz) B-mode imaging by implementing ultrafast imaging methods in combination with duty-cycling of the front-end. In all normal-rate B-mode scenarios (VS13, PW31: 30-50 Hz), we can apply duty-cycling to reduce the overall front-end power consumption to even lower values (2.0-3.0 W) than the idle consumption of an always-on front-end (RDY, 4.4 W).

This is a very fundamental observation as digital probes with many channels (>32) are often regarded infeasible as an always-on front-end is assumed and the consumption of the ADC is simply added up, without considering the additional power saving potential that using more ADCs can provide.

4.10.1 Comparison with Related Work

In Tbl. 4.5 we compare the LightProbe with both systems and concepts reported in the literature as well as commercial systems:

With the LightProbe we can support the same imaging modalities (B/M-Mode, Color Doppler) as current digital probes for portable imaging do [16, 93–95, 113, 119–121]. On top of this, we support modalities that are typical of high-end systems, which we demonstrated so far with the high-speed-imaging example. LightProbe provides real-time raw sample access. This feature is currently only supported by top-of-the-line software-based commercial and research systems [17, 20, 66, 84], but without the need for any external device that houses the ultrasound front-end and provides the cable connector to connect the probe. The LightProbe provides true continuous RF access (≥240 Mb/s per channel)\(^2\), which is comparable to the capabilities of the 256-vantage system (206 Mb/s per channel) given its 6.6 GB/s sustained data access [84]. Even though LightProbe supports only one-fourth of the number of channels of these high-end systems, we are able to do so at a lower cost, power budget, and smaller form factor.

\(^2\)The 240 Mb/s relate to the currently set sampling rate of 20 MS/s.
### Table 4.5
Comparison with Related Work

<table>
<thead>
<tr>
<th>System</th>
<th>ADC(^a)</th>
<th>Link</th>
<th>Mode</th>
<th>Connects to</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobiUS PE [93]</td>
<td>?</td>
<td>USB</td>
<td>B</td>
<td>PC</td>
</tr>
<tr>
<td>Usan [119]</td>
<td>8</td>
<td>USB</td>
<td>B</td>
<td>Dedicated Tablet</td>
</tr>
<tr>
<td>Visiq [16]</td>
<td>?</td>
<td>USB</td>
<td>B/M, Doppler</td>
<td>Dedicated Tablet</td>
</tr>
<tr>
<td>Lumify [94]</td>
<td>?</td>
<td>USB</td>
<td>B/M, Color Doppler</td>
<td>Smartphone/Tablet</td>
</tr>
<tr>
<td>iViz [120]</td>
<td>?</td>
<td>Cable</td>
<td>B, Color Doppler</td>
<td>Dedicated Tablet</td>
</tr>
<tr>
<td>Clarius [95]</td>
<td>?</td>
<td>Wi-Fi</td>
<td>B, Color Doppler</td>
<td>Smartphone/Tablet</td>
</tr>
<tr>
<td>Sonon300c [113]</td>
<td>?</td>
<td>Wi-Fi</td>
<td>B</td>
<td>Smartphone</td>
</tr>
<tr>
<td>Freestyle [121]</td>
<td>?</td>
<td>UWB</td>
<td>B, Color Doppler</td>
<td>Dedicated Device</td>
</tr>
</tbody>
</table>

| [15]            | 64\(^b\)  | Wi-Fi| SASB                  | -                      |
| [122]           | 192\(^c\) | Wi-Fi| SASB (Vector Flow)    | Tablet                 |
| [14]            | 16        | USB  | 3.0 B/C (MUX)         | Smartphone             |
| [123]           | 1         | USB  | 2.0 B (MSAS)          | PC                     |
| [124]           | 1         | Wi-Fi| B/3D                  | Display                |

| LightProbe      | 64        | Optical\(^d\)| any (RF access) | PC\(^d\) |
| DiPhAs [20]     | 256       | PCIe        | any (RF access) | PC       |
| Vantage [84]    | 256       | PCIe        | any (RF access) | PC       |

\(^a\) Number of ADC in commercial system rarely made public.
\(^b\) System-level design study for a wireless probe.
\(^c\) No probe hardware. Probe emulated with SARUS for demonstration.
\(^d\) Smartphone & tablet possible over USB/Wi-Fi.

Supporting ultrafast imaging at high frame-rates with a digital probe requires an elaborated power and thermal management as the peak power consumption in the probe may exceed 10 W (VS13, 500 Hz). The thermal control management allows using these modes with a defined periodicity safely.

Compared to the 16-channel (128 with multiplexing) digital ultrasound probe [14] equipped with USB 3.0 to connect to smartphones, our 64-channel LightProbe provides 4× more channels and even has a 30% lower overall probe power consumption (5.84 W vs. 8.16 W) for B-mode imaging (VS13 50 Hz). The probe mentioned above has a similar size (180×55×35 mm\(^3\)) and performs the beamforming on the probe (FPGA) and the postprocessing on the smartphone’s GPU. Under operation, the system heats up internally to 82.3° C and the transducer to 35° C, allowing sustainable B-Mode operation as the
LightProbe does. Note that many digital probes not even supporting ultrafast imaging do not allow uninterrupted operation and require the probe to be periodically turned off for cooling. For example, the Sonon 300 series Wi-Fi probes (Healcerion, KOR) are specified for a maximal 10 min operation with 10 min resting time [113]. Clarius probes [95] can be equipped with an external fan for longer scanning time or provide a docking station to cool probes back down to their operable temperature range.

### 4.10.2 Possible Extensions

Note that the LightProbe hardware presented in this work has the required interfaces (USB, Wi-Fi) to connect to tablets and smartphones as well. Given the reduced data bandwidth of these interfaces, additional buffering or processing (beamforming) may be implemented on the LightProbe’s FPGA.

Implementing conventional digital beamforming (not ultrafast) is not critical from a power point of view: As shown in [41], powerful integrated digital beamformer (100-channel, 300 M focal points per second) consume a few hundreds of mW, which is very little compared to the more than 2.0 W used in the front-end.

The adapter that allows connecting the optical link of the LightProbe to the PC is currently implemented with a commercial FPGA Development Board (Xilinx KCU105, 3000$) plugged into a PCIe slot. This solution can easily be replaced with a more cost-efficient solution: Changing the protocol of the fiber link from Aurora to Ethernet allows using an off-the-shelf server-grade QSFP Ethernet network card on the PC side. This brings the adapter cost down to 300-400$.

### 4.10.3 Potential for Power and Size Reduction

LightProbe is a prototype to demonstrate the capabilities an ultrafast digital probe may provide and to show how the main system-level challenges (interface speed, power/thermal management) can be solved. The current probe is built entirely with off-the-shelf components readily available. Thus there is a large improvement potential to reduce the size and power consumption if state-of-the-art components were used as described in recent literature:
While the current AFE is already very efficient (38.3 mW/cha @ 20 MS/s, measured), state-of-the-art literature systems can reach 38.6 mW/cha @ 40 MS/s [97], which is a 2× energy efficiency increase in mW/cha per MS/s. These power figures are already reached by recent (2017) 32-channel AFEs (AFE58JD32, TI) consuming 42 mW/ch @ 40 MS/s. Our optical link uses a module consuming 0.87 W @ 25 Gb/s. State-of-the-art modules like the ECUO Optical Firefly (Samtec) using VCSEL array technology can transfer 168 Gb/s using <1 W (TX) with a footprint of 2 cm², resulting in a 6× higher link power efficiency and 7× size reduction. Similarly, the latest 16 nm Xilinx UltraScale+ FPGAs provide high-speed transceivers with twice the power efficiency (50 Gb/s @ 0.5 W) and realize digital functionality with a 2.4× performance increase per Watt over the older 28 nm FPGA generation currently used in LightProbe. We estimate 1.2 W can be saved on the FPGA subsystem. By replacing the active T/R switches with passive ones (e.g., MD0101, Microchip), another 0.73 W could be saved. The current HV/LV-supplies were highly optimized for footprint (±50 V in 9 cm²) at the cost of good conversion efficiencies under low load. Currently, around half of the front-end power during power down modes (RX-STBY and RX-PDN) is lost in the converters, as these supplies were designed for high-efficiency at their nominal load.

If all these changes were implemented, we estimate that the total consumption in RDY can be reduced by 2.4× from 8.08 W to 3.3 W (see Fig. 4.17 Right). Considering this 2.4× possible reduction, we expect even HFR imaging (VS13, 500 Hz) can be operated with a total probe consumption of around 6 W. The power consumption and size may be reduced even further if the FPGA is replaced by an application-specific-integrated-circuit (ASIC) that co-integrates part of the transmit and receive path with the transceivers for the optical link.

The LightProbe is larger compared to commercial probes. Given that the PCBs are only sparsely populated (Fig. 4.6) and the previously mentioned improvements, we expect that a simple PCB re-layout would reduce the probe length by 2×. By switching to the latest fully-integrated TX chips such as the STHV1600 (STM) the TX boards are no longer needed as TXBF, Pulser and T/R can be implemented with 4 chips using the same space only the T/R switches need now. This halves the probe thickness, resulting in a size comparable with current probes.
4.11 Conclusions

We have presented a 64-channel digital ultrasound transducer probe (LightProbe), which confines the entire ultrasound front-end in the probe handle and outputs the raw data samples in real-time over a high-speed optical link. Moreover, we have presented an ultrasound imaging system (UltraLight) built around our 64-channel digital probe connected to a desktop PC performing real-time software-defined imaging on the GPU.

The LightProbe not only supports ultrafast imaging, but it also exploits its ultrafast capabilities to reduce the power consumption during normal-rate B-mode imaging with smart duty-cycling of the front-end. A novel thermal management approach allows the LightProbe to exploit the thermal capacitance of the casing to support the intermittent operation of high-performance modes whose power dissipation exceeds the sustainable passive-cooling capabilities. Our thermal management ensures periodic activation of these modes without ever having to turn off the probe for cooling, providing a consistent quality of service to the operator.

Our work shows that using digital probes instead of analog probes is a feasible system architecture option for future cart-based software-defined ultrasound systems, as they allow to build powerful and cost-efficient systems thanks to the minimal amount of ultrasound specific hardware required. Also, our approach to increase the number of receive channels to support ultrafast imaging to reduce the average front-end power consumption may be leveraged in future mobile systems to support more modalities.
Chapter 5

UltraLight Case-Study: High-Speed Bubble Tracking

5.1 Introduction

5.1.1 Motivation

Conventional medical ultrasound imaging is fundamentally limited by diffraction: Clinical system achieve a resolution of 100 µm to 1 mm depending on the operation frequency. In the past years, several super-resolution techniques for ultrasound imaging were proposed to overcome these resolutions limits [125]: Ultrasound localization microscopy (ULM) [37] is a super-resolution technique that allows mapping the vascular system by pinpointing and tracking tiny gas microbubbles injected into the blood flow.

Despite their small size of a few micrometers diameter, microbubbles appear as detectable blurred spots in the ultrasound image as they generate a large backscatter response due to the high acoustic impedance difference of the gas in the bubble compared to the surrounding blood or tissue. Following and tracking the center of each
bubble allows to map vessels precisely and beyond the resolution limit of the system. As the bubbles float in the bloodstream, they can reach speeds of several cm/s. To reliably track the bubbles at these speeds and to increase the tracking accuracy, capturing the ultrasound images at a frame rate above 100 Hz over many seconds is required.

In [37] ultrafast ultrasound localization microscopy (uULM) for deep super-resolution vascular imaging is introduced and demonstrated on a rat brain in an in-vivo experiment: Within 150 seconds, 75’000 ultrasound images were captured with a rate of 500 frames per second. By tracking bubbles over multiple frames, microvessels with less than 10 µm in diameter could be mapped.

Since then micro bubble tracking algorithms have been improved in tracking robustness [126] and better trajectory reconstruction when a high number of bubbles are present per frame [127, 128].

However, up to now, uULM has only been demonstrated on large and expensive ultrasound research systems, due to the required ultra-fast imaging capabilities. Our goal is to show that high-speed bubble tracking can be implemented on a low-cost ultrasound system.

In the previous Chapter 4 [42–44], we have presented the Ultra-Light system, a low-cost ultrafast-capable imaging platform based on a digital ultrasound probe connected to an off-the-shelf GPU-equipped PC for software-defined imaging. The platform features a 4 MHz 64-channel digital ultrasound probe, which integrates the analog front-end and connects to the PC over a 25 Gbit/s fiber optical link. The link allows transferring the raw RF samples from all channels to the PC in real-time. The system can perform imaging at kHz rate and thanks to the raw channel data access, allows implementing any imaging modality in software, by just writing the required “ultrasound application” to run on the PC using commodity hardware such as high-performance processors and GPUs.
5.1. INTRODUCTION

5.1.2 Contributions

As part of this thesis [45], we demonstrate for the first time high-speed ultrasound bubble tracking with a low-cost software-defined ultrasound system. With our system, we can precisely track gas bubbles with a frame-rate of 200 Hz. Before implementation, we simulated the imaging system and computed a median bubble localization accuracy of 5.1 µm and 11.7 µm in lateral and axial direction respectively, with a correct bubble detection and step tracking rate of 95.8% and 99.1%. In the experimental setup we were able to track 376 gas bubbles over 3-237 frames over a length of 0.03-29.4 mm with velocities up to 49.6 mm/s. In summary, our contributions are:

- An implementation of an ultrasound bubble tracking system on the ULTRALIGHT ultrafast imaging platform.

- A detailed bubble tracking performance evaluation, by feeding the system with simulated raw data, where bubble positions and movement are precisely known.

- An experimental demonstration, where we demonstrate gas bubble tracking in carbonated water at 200 Hz.

The rest of the chapter is organized as follows: First, we describe the overall system architecture (Section 5.2), including the data capturing process and the implemented bubble tracking algorithm. Then, we describe our experimental verification setups (Section 5.3), consisting of a simulation experiment and a phantom experiment to assess real-world performance. Finally, we present the results (Section 5.4) and close with discussion (Section 5.5) and conclusions (Section 5.6).

The ETH MSc student Pascal Armin Jud contributed the implementation of the tracking algorithm and its evaluation to this chapter as part of his semester project, which was supervised by the author of this thesis.
5.2 Ultrafast Bubble Tracking System

Fig. 5.1 shows the system overview of our low-cost bubble tracking system implemented on the UltraLight system (Chapter 4): The UltraLight system consists of our digital transducer probe connected to a host PC over an optical link. The probe incorporates the piezo-electric transducer array along with a 64-channel transmit/receive front-end and an FPGA for data serialization and control. For the host PC, an off-the-shelf PC is used (Quad-Core Intel Xeon 3.5 GHz with 64 Gbyte DDR4 RAM). The PC is equipped with a fiber optics PCIe 3.0 adapter card to interface with the probe, and an NVIDIA GTX1080 Graphics Card with 8 Gbyte RAM for processing.

Ultrafast raw ultrasound RF data is acquired with our digital probe and sent to the host PC. On the PC, the raw data is beamformed on a GPU and fed to our bubble tracking pipeline, which outputs bubble trajectories to be further used for super-resolution imaging. In the following, each step is explained in detail.
5.2. ULTRAFAST BUBBLE TRACKING SYSTEM

5.2.1 Raw Data Acquisition

Our probe operates with a 4 MHz 64-channel phased array. Standard plane-wave ultrafast imaging with a phased-array would result in a very narrow field of view. We thus use a synthetic aperture imaging method (VS13) with 13 virtual sources (VS) placed behind the aperture on a 90° arc with 6.3 mm radius to create diverging waves. A 4 MHz 2-period pulse is used for transmit. Per transmit-receive cycle, 2500 samples are taken simultaneously from all 64 channels with 12 bit at 20 MS/s. The 13 acquisitions to compose a frame are captured in a short burst with a pulse repetition rate of 3 kHz to reduce motion artifacts. Frames are captured with a controlled frame rate of 200 Hz. At this rate, all bubbles moving less than 20 cm/s only move less than 1 mm between frames. The raw RF data size of a single frame is 3.1 Mbyte, which results in a raw data stream of 5 Gbit/s that has to be streamed from the probe to the PC. The raw data stream is sent over the fiber optics link to the PC where the PCIe 3.0 adapter card receives the raw data and transfers it to the PC’s main memory. By default, the system captures and displays B-mode images at normal frame rate (20 fps) for proper probe positioning. Upon user request, a 2 second (400 frames) bubble tracking sequence is captured with prior settings.

5.2.2 Ultrafast B-Mode Beamforming

A sequence of 400 B-mode frames is computed from the raw data. To do so, the raw RF signals of the 13 VS acquisitions required to compose a frame are transferred to the GPU, where they are filtered, beamformed and coherently compounded to form a single B-Mode gray-scale frame on a high-pixel-density Cartesian grid (48 µm/px axial, 50 µm/px lateral). Compared to our systems theoretical resolution (193 µm axial, 2.11° lateral), this pixel-density corresponds to a 4× oversampling in the axial direction and assures 4× lateral oversampling after a depth of 5.5 mm. The oversampling assures that no relevant information is lost that could help to later exactly pinpoint the bubbles in the B-mode image. The overall data-size of the high-pixel-density B-mode sequence is 13 Gbyte.
5.2.3 Bubble Tracking

Our *bubble tracking pipeline* implemented in MATLAB loads the 400 frame B-mode sequence, first removes static background, then detects and localizes bubbles in each frame, and finally fuses the bubble positions to trajectories:

**Background Suppression**

The static background is suppressed with a temporal average filter: First, the background is computed as the average B-mode image over the sequence, which is then subtracted from each frame.

**Bubble Detection and Localization**

Our bubble detection approach is based on [127]: To detect bubbles in a frame, the background-suppressed image is convoluted with a 2D Gaussian kernel approximating the point-spectral-function (PSF) of a bubble with expected size. Local maxima in the filtered image exceeding a threshold are considered bubble positions. An initial bubble position estimate is set to the center coordinate of the pixel in input B-mode image that corresponds to the local maxima. These initial positions are refined with a local spline interpolation of the B-mode image by $8 \times$ to obtain the position with a grid-accuracy of ca. 6 µm.

**Bubble Trajectory Formation**

Bubble trajectories are formed by tracking individual bubbles over multiple subsequent frames. We implement the approach proposed in [128], which exploits the fact that given the high-frame-rate, bubbles are most likely to appear in frame $n$ close to the position they were in frame $n - 1$. By pairing bubbles between consecutive frames, they are tracked through the entire sequence of frames. For increased tracking robustness we adopted the partial assignment algorithm for bipartite graph pairing that enforces a mutual minimal paring distance proposed by [128].

Our bubble tracking pipeline produces an array of trajectories: Each trajectory consists of a start frame number and a list of coordinates for all consecutive frames the bubble could be tracked until it was lost. We
post-process the trajectories, i.e., remove short trajectories (tracking over less than 3 frames) and compute the bubble speed for each step in the trajectory. The trajectories could now be further processed to form a super-resolution image or a vector flow image. Our system visualizes the bubble trajectories in a slow-motion playback of the 200 fps imaging sequence with the trajectories plotted over the B-mode image.

5.3 Experimental Verification

5.3.1 Simulation Experiments

The performance of our bubble tracking system was assessed with two simulations: A first simulation with randomly moving bubbles to quantify the general detection/tracking performance, and a second simulation of an artificial vessel tree inspired by [127] to assess the discrimination of vessels with an inter-vessel distance below the resolution of our system.

For the first simulation, a region of $10 \times 10 \text{ cm}$ was simulated with 50 bubbles moving along straight trajectories through a homogeneous speckle background. The bubble starting point and direction was chosen randomly within the simulated area. The speed was set randomly between $(0-13.3 \text{ mm/s})$.

For the vessel discrimination study, single bubbles moving through a vessel structure were simulated. A $10 \times 10 \text{ cm}$ speckle region was used as background. The vessels structure was placed at $5 \text{ cm}$ depth and consisted out of two parallel $5 \text{ mm}$ long vessels with a $3 \text{ mm}$ long reconvergent bifurcation in the middle. The distance between the main vessels and the bifurcated vessels was $125 \mu\text{m}$ and $62.5 \mu\text{m}$ respectively. To assess vessel discrimination along both axes, the vessel structure was simulated with lateral and axial placement orientation with respect to the transducer. 1'000 frames were simulated.

Field II [78, 79] was used to simulate our probe and to obtain the raw RF data the probe would capture and send to the PC. The simulated raw data was fed to the processing pipeline (beamformer, bubble tracking) described before in Section 5.2.
5.3.2 Phantom Experiment

The real-world performance of our bubble tracking system was evaluated in a phantom experiment: Fig. 5.2 sketches the setup. A water tank of 10 cm height was filled with carbonated water. Table salt was added to stimulate the creation of small gas bubbles. The diameter of the small ascending gas bubbles was measured to be approximately 200 µm.

Figure 5.2 – Sketch of the setup and close-up picture of the water tank showing the small (200 µm diameter) gas bubbles rising up (highlighted with red circle).
5.4 Results

5.4.1 Simulation

In the simulation of the randomly moving bubbles, our bubble tracking system was able to track the bubbles with a bubble detection rate (ratio of the number of detected bubbles to the number of set bubbles) of 95.8%. The median spatial error of the detected bubble positions was 11.7 $\mu$m (mean 17.18 $\mu$m) and 5.1 $\mu$m (mean 25.4 $\mu$m) in axial and lateral direction respectively. 99.1% of the bubble transitions from one frame to the next could be properly tracked. The average absolute speed error was 0.235 mm/s.

The results of the vessel discrimination simulation are shown in Fig. 5.3: We show a maximum intensity persistence (MIP) image [129] of the background-suppressed B-mode sequence overlaid with the found bubbles (left) and bubble trajectories (middle), as well as cross sections (right) through the normalized MIP image and the normalized bubble trajectories density through that cross sections (accumulated over a 1 mm neighborhood). The MIP image representing conventional diffraction-limited imaging does not allow to discriminate the vessel structure. The persistence image (see lateral) only shows an unrecognizable brighter region around the vessel structure. The drawn bubble trajectories, however, allow clear discrimination of the four vessels, including the bifurcated section, which is separated by only 65.5 $\mu$m. If the vessel separation is reduced to 31.25 $\mu$m, the vessels can no longer fully be discriminated in the lateral case.

5.4.2 Phantom Experiment

In the phantom experiment, our bubble tracking system was able to track 376 gas bubbles over the 400 frame 200 fps sequence. Bubbles were tracked over 3-237 frames (median 7) and a trajectory length of 0.03-29.4 mm with a top speed of 49.6 mm/s. Our system can visualize the found trajectories by rendering a looped playback of the captured B-mode sequence with overlaid bubble trajectories. The animation of this experiment can be downloaded here\(^1\). Fig. 5.4 shows frame 100/400 of

\(^1\)http://hdl.handle.net/20.500.11850/293097
Figure 5.3 – Results of the vessel discrimination simulation with the vessel structure placed with lateral and axial orientation. From top to bottom: Detected bubbles positions (blue) and trajectories (red). In the background the MIP image (20 dB). On the right, cross-section view through the MIP image (black) at $x = 0$ mm and $z = 50.5$ mm and the normalized bubble trajectory density (red) passing through that cross section (super-resolution image).
this animation. Bubbles found in the current frame are indicated with red circles. All trajectories found up to the current frame are drawn as lines. The lines are colored according to the instantaneous speed of the bubble. The experimental tracking precision is quantified by comparing all trajectories longer than 5 mm (64 trajectories) compared to a smoothed spline-fitted version thereof. The low mean absolute path difference (1.7 µm axial and 4 µm lateral) indicate a high system tracking precision with little positioning noise. Also, note the high qualitative tracking fidelity with which the gas bubbles are tracked as they accelerate on their curvy ascent.

![Bubble Tracking: Frame 100/400](image)

**Figure 5.4** – Output frame 100/400 of our bubble tracking system during the phantom experiment. Bubbles detected in the current frame are marked with red circles. Trajectories found up to the current frame are drawn as lines. The color of the line indicates the instantaneous speed of the bubble.

After capturing the sequence (2 s) and running the high-pixel-density beamforming on the GPU (30 s), about 3 minutes are required to run the bubble tracking algorithm in MATLAB before the system outputs the slow-motion playback.
CHAPTER 5. BUBBLE TRACKING

5.5 Discussion

The bubble localization accuracy of our system using simulated RF data is the order of 10-30 µm, which is significantly below the diffraction resolution of our system (ca. 200 µm axial, 920 µm lateral at 2.5 cm depth). With the vessel discrimination simulation, we were able to show that we can discriminate structures with a separation of 62.5 µm. Smaller lateral structures (31.25 µm) could no longer be discriminated, which matches the determined mean lateral spatial error of 25.4 µm. Compared to [127], our simulated vessel separation is slightly better (62.5 vs. 100 µm) despite our much lower transducer bandwidth (4 MHz vs. 30 MHz), which we explain with the different system noise model used.

Currently, our phantom experiment is performed with 200 µm gas bubbles in carbonated water. While we demonstrate precise tracking (<5 µm trajectory noise) at high speed (up to 50 mm/s), future work is required to assess the tracking performance of smaller bubbles (1-10 µm). Overall, our results provide a proof of concept that ultrafast bubble tracking can be implemented on low-cost software-defined ultrasound systems operating with digital probes. We show that the capturing rate (5 Gbit/s) and amount (>1 Gbyte) of raw RF data including the processing can be handled by such ultrasound systems.

5.6 Conclusions and Future Work

In this paper, we have presented an ultrasound bubble tracking system, which can precisely track gas bubbles at high speed. The system was built entirely by writing the required ‘application’ for our software-defined ultrasound system. Our system can be built at very low cost compared to other ultrafast-capable systems currently used for bubble tracking. Apart from the digital probe, which can be manufactured for a few k§, our system relies entirely on off-the-shelf system components (PC, GPU), which keeps the overall cost down.
Chapter 6

Summary and Conclusions

Hardware innovations have been and keep being the primary enabler for novel medical ultrasound modalities: At first, development systems with increased hardware functionality are required to explore and investigate disruptive ideas. Once the research community has managed to demonstrate that these ideas work in the lab, a second round of hardware innovations is required to design system prototypes that can later be engineered into products, that bring these ideas out into the world to the medical service providers.

In this thesis, we investigated several new hardware innovations to drive this process forward. We started by addressing the various challenges around 3D ultrasound imaging: First, we have designed with the MUXHEAD a matrix ultrasound probe, which provides access to the analog signal of all transducer elements. Having access to such a probe is crucial for researchers exploring and designing imaging algorithms for 3D imaging. Second, we demonstrated with the fully-integrated beamformer EKHO that digital beamforming is getting feasible for very large transducer matrices with up to 10’000 elements. Providing cost-, power- and space-efficient compute elements able to stem the tremendous computational load required for 3D imaging is most vital
for 3D ultrasound to leave its niche application and bring its inherent benefits to other modalities.

We then shifted our focus to high-performance 2D ultrasound imaging: We explored with the LightProbe a novel 2D ultrasound system architecture, which combines the benefits of the latest powerful software-defined ultrasound systems with the cost-efficiency and ease-of-use of the latest ultra-portable systems. The LightProbe is a unique ultrasound probe embedding the entire ultrasound front-end within the probe handle and providing real-time access to the raw digitized sensor data over a high-speed fiber-optics interface. With the LightProbe, we then built our own, fully operational ultrasound imaging system, which we call UltraLight. The UltraLight system relies entirely on commodity hardware for processing. With the UltraLight system, we could demonstrate that advanced modalities such as high-frame-rate imaging, do not require expensive equipment, but can be implemented as well on very cost-efficient hardware.

6.1 Overview of the main results

The main results and contributions can be summarized as follows:

Reducing IO with Bandpass Data Representation

In an ultrasound imaging system, large amounts of data need to be moved at high speed between components. These data movements significantly contribute to the complexity and power dissipation of the system. First, digitized raw data must be transferred from the analog-to-digital converters to the beamforming hardware. In a 256-channel 2D system, data-rates of 100 Gbit/s and more are required for this link. In a fully-sampled 10’000-channel 3D system data-rates of several Tbit/s may be required. Second, the beamformed image data must be moved out of the beamforming hardware to the post-processing unit. In a conventional 2D imaging system, which acquires the image line-by-line, this link is uncritical as the required bandwidth is in the order of a few Gbit/s. In today’s parallel 2D and 3D systems, which acquire many lines or even the entire image at very high frame-rate, the required bandwidth for the beamformed data is in the order of...
100 Gbit/s as well. Compressing the transferred signals is thus very desirable to simplify the system design and save power. Fortunately, ultrasound raw data, as well as the beamformer output data, are bandpass signals. This bandpass property allows compressing the signals by changing the data representation from the RF domain to the bandpass domain without impairing the image quality.

The Ekho beamformer architecture presented in this thesis has been designed to minimize I/O by leveraging this approach: The raw input data is accessed in its critically-sampled bandpass representation. This is considered as a lossless compression of the raw data with a fixed guaranteed compression factor. Typically a compression factor of $2-3 \times$ can be achieved. Higher compression-rates are possible if the raw signal was heavily oversampled. Similarly, the beamformed output volume is critically bandpass sampled to reduce the output bandwidth without impairing the image quality. Compared to a Nyquist sampled volume, the amount of data is reduced by $12 \times$ for 3D imaging and by $6 \times$ for 2D imaging.

The bandpass signal representation is used as well in the UltraLight system to speed-up the data transfer to the GPU beamformer, and it can be activated on the LIGHTPROBE to compress the data on the fiber-optics link.

**Minimizing the Beamforming Effort**

The beamforming compute complexity grows proportionally to the number of pixel/voxels computed. Beamforming images at high-rate or beamforming volumes thus demand for much more compute resources than conventional 2D imaging.

Luckily, switching to a bandpass signal representation also helps to reduce the beamforming effort as the computation of pixel/voxels with redundant information is avoided. This requires a modified beamformer, which directly outputs bandpass data. Consequently, the bandpass representation does not only reduce the output rate by $12 \times$ ($6 \times$ for 2D); it also reduces the beamforming effort.

We demonstrated and evaluated this bandpass beamforming approach in detail on the Ekho 3D beamformer, and used it again later to boost the performance of the ULTRALIGHT system.
Avoiding Precomputed and Externally-Stored Delay Tables

Beamforming for 3D ultrasound is extremely compute-intensive – especially when considering fully-digital beamforming on fully-sampled transducer matrices: several Tbit/s of raw sensor data has to be combined with Gbytes of delay information to compute a volumetric image.

We have already elaborated on how to reduce IO bandwidth. However, reducing raw data input and image output streams has only limited impact if the delay information required for beamforming has to be loaded from external memory: In a typical 3D system, for 2.5 Tbit/s of input data, 24 Tbit/s delay data is required.

In this thesis, we showed that a naive implementation, which loads all delay information from off-chip memory, would consume 500W in the DDR interface alone. We thus designed for Ekho a new delay index computation method with increased accuracy and reduced circuit complexity, which computes all required delays on-chip from only a few constants. To even further reduce the number of required constants for the delay computation, we integrated a microcode control unit into our beamformer, which allows flexible programming of a range of 3D imaging strategies. Overall, the combined methods reduced the memory requirement from Gbytes to below 50 kbit, such that no external delay memory is needed anymore.

Pushing the Frontiers of Fully-Digital 3D Beamforming

By combining all the approaches as mentioned above, we could demonstrate with Ekho that fully-digital beamforming of large matrices is feasible: Overall, our single-chip 28 nm 10’000-channel beamformer implementation, which includes all processing blocks required to compute image points out of the raw digital samples, consumes 30.3 W and achieves a state-of-the-art beamforming efficiency of 98.4 GBOPS/W.

The implementation may also be scaled down for more efficient 2D imaging: A 100-channel 2D-version of Ekho consumes only 0.3 W.

Replacing the Analog Probe Cable with a Fiber-Optics Link

Conventional ultrasound probes connect to the backend system over a coaxial cable harness, which composes out of hundreds of individual micro-coaxial cables that are attached to the piezoelectric transducer
elements in the probe. The cable harness should remain flexible and light for ergonomics, which makes it expensive and relevant for system cost. The cable cost and size is increasingly critical for 3D systems using matrix probes with several thousands of elements.

In this thesis, we explored the replacement of this analog cable by a fiber-optics link for 2D systems. We could demonstrate that a fiber-optics link provides sufficient bandwidth to transport the raw ultrasound signal from the probe to the connecting system. The implemented 25 Gbit/s link solution based on QSFP is low cost (100-200 $ USD) and can be easily scaled up to even higher data rates. Compared to an analog cable harness, the fiber is thinner and lighter, resulting in increased ergonomics, has no length limitations and is immune to electrical interference, which may simplify the use of ultrasound in dual-modality applications with MRI.

**Exploit Ultrafast Imaging to Save Power in the Front-End**

In ultrafast imaging, the entire image is acquired with a single ultrasound wave emission. Multiple emissions may be combined for better image contrast and resolution. Ultrafast imaging enables frame-rates of several kHz, which is substantially higher to what conventional line-by-line systems provide. Conventional line-by-line systems may perform hundreds of wave emissions until sufficient image lines are acquired to compose an image. So far ultrafast imaging has been mainly considered as an enabler for high-frame-rate imaging, and it took the research community a while until an image quality comparable to conventional line-by-line imaging was achieved.

With the ultrasound system we have built, we could demonstrate that the energy required to acquire the raw data for a single image is approximately proportional to the number of ultrasound wave emissions. This relation implies that ultrafast imaging inherently requires less energy to acquire an image of the same quality as conventional methods.

This observation allowed us to explore in this thesis ultrafast imaging as a method to save power: We could show with our LightProbe that we can reduce the ultrasound front-end power consumption by 68% to 2.0 W by using ultrafast methods for 30 Hz imaging compared to a conventional line-by-line method consuming 6.3 W. The image quality was only marginal impaired.
Digital Ultrasound Probes are not Limited by Peak Power

Prior to this work, digital ultrasound probes with many front-end channels (64 and more) were considered infeasible. A simple calculation was used to argue this: A decent ultrasound receive-channel consumes 100 mW under operation. Placing 64 front-end channels in a probe would thus result in a power dissipation of 6.4 W for the receive path alone. Additional substantial power will be dissipated in the transmit path needed to emit ultrasound waves. Dissipating the resulting overall power over the housing of the probe without letting the surface temperature exceeding the allowed 43° C for medical devices is considered unrealistic.

However, the calculation mentioned above assumes continues operation of the front-end electronics, which is no longer required when ultrafast imaging methods are used for normal-rate (30 Hz) imaging and the front-end can be heavily duty-cycled. This duty-cycling results in a significant reduction of the power dissipated in average, which is relevant for thermal considerations.

With the LIGHTPROBE we could demonstrate that even with only passive cooling, a 64-channel digital probe can support continuous plane-wave imaging with a frame rate of 47 Hz indefinitely.

We also showed how high-frame-rate imaging (200-500 fps) could be supported safely in a digital probe, despite high peak-power consumption (10.7 W). Continuously dissipating such a high power would exceed the sustainable passive-cooling capabilities of the probe, given prescribed surface temperature regulations. We introduce a novel thermal management approach to allow the LIGHTPROBE to exploits its thermal capacitance to support an intermittent operation of such high-performance modes without ever letting the surface temperature exceed regulatory limits. In order to provide a consistent quality of service to the operator, our thermal management ensures periodic activation of these modes without ever having to turn the probe off for cooling.

Digital Probes are a Viable Architecture Option

Overall our work on digital ultrasound probes proves that such probes are a viable architecture option for future ultrasound systems:

Our ULTRALIGHT system demonstrates that digital probes allow
building high-performance imaging systems without performance compromises using only a minimal amount of ultrasound specific hardware, which can be confined within the probe handle. As the LightProbe is highly configurable and all processing is performed in software, new imaging modalities can be easily implemented by writing the required "ultrasound app" for our system. Conclusively, we could show

1. that using digital probes instead of analog probes is a feasible system-architecture option for future cart-based software-defined ultrasound systems, as they allow to build powerful and cost-efficient systems due to the minimal amount of ultrasound specific hardware required, and

2. that increasing the number of receive channels in a digital probe to support ultrafast modalities, can reduce the average energy consumption of the probe, even though it increases the peak power consumption. This design concept can also be leveraged to build future portable systems supporting more modalities.
6.2 Outlook

In this work, we investigated several new approaches to improve current ultrasound systems by replacing analog components with digital solutions. However, certain aspects still leave room for further improvements, which will be explained in the following.

While we have shown that fully-digital 3D beamforming is getting feasible for fully-sampled matrix transducer, we did not address the challenges on how to design the very high-channel-count analog front-end for such a probe. Having such a front-end is particularly interesting when considering that the digital beamformer could be directly integrated into the transducer probe. Such an architecture would enable a new class of fully-digital 3D ultrasound systems, which provide more flexibility in a smaller form-factor. Indeed, our findings from the LIGHTPROBE project on how to reduce front-end power can be applied to matrix probes as well.

Also, it is not yet entirely clear if the raw signal of every element in a fully-sampled matrix array is required for decent-quality 3D imaging. In the past years, row-column-addressed fully-sampled arrays [130] have emerged as a viable alternative. These arrays can short-circuit all elements in a row or column and thus only need $N$ front-end channels instead of $N^2$ to operate a $N \times N$ matrix. However, beamforming effort stays high for these probes as still, a volumetric image has to be computed. Ekho could be adapted to support such matrix transducers.

Further increasing the number of front-end channels is also the next intermediate step for digital probes. Our current implementation supports 64-channels. This number is sufficient for most available phase array transducers. However linear and curved transducers typically feature 128, 192 or even 256 elements and would profit from more channels to avoid multiplexing.

Also, currently our digital probe is built entirely from standard components including an FPGA. Increasing the degree of integration by designing a custom ASIC, can further reduce size and power consumption for digital probes especially since high-bandwidth inter-chip communication can be avoided altogether.

Moreover, additional work is required to ease the programming of software-defined ultrasound systems. As elaborated, state-of-the-art
systems have to deal with data rates in the order of 100 Gbit/s. Writing optimized code for these rates is challenging. To relieve the general-purpose compute units (e.g., GPUs) from simple stream-like processing, one could consider extending the high-rate interface adapters with simple, programmable stream-processors for light-weight operations. These stream-processors could be used to perform typical ultrasound preprocessing operations, such as signal filtering and correlation, such that data arrives already pre-processed in memory. Similar approaches are already deployed in 100G network adapters for packet filtering and protocol handling.
Appendix A

PCB Gallery

This appendix list the most relevant PCBs that have been created for this thesis.
A.1 MuxHead

A.1.1 Multiplexer Board

**Photograph**

**Layout**

<table>
<thead>
<tr>
<th>Project:</th>
<th>MUXHEAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>Multiplexer Board for the MUXHEAD: Contains a 256-to-64 multiplexing stage and 64 receive amplifiers to boost the signals and match the cable impedance.</td>
</tr>
<tr>
<td>Designers:</td>
<td>Pascal Alexander Hager (Initial Design) Active Technologies (DFM Revision)</td>
</tr>
<tr>
<td>Year:</td>
<td>2015</td>
</tr>
<tr>
<td>Size:</td>
<td>33 × 110 mm</td>
</tr>
<tr>
<td># Layers:</td>
<td>10</td>
</tr>
<tr>
<td># Power Supplies:</td>
<td>7</td>
</tr>
<tr>
<td># Component:</td>
<td>179</td>
</tr>
<tr>
<td># Nets:</td>
<td>492</td>
</tr>
<tr>
<td># Vias:</td>
<td>1389</td>
</tr>
</tbody>
</table>
A.1.2 Control Board

**Project:** MUXHEAD

**Description:** Controller board for the MUXHEAD: Communicates with the connected ultrasound system connected and controls the multiplexer switches.

**Designer:** Pascal Alexander Hager

**Year:** 2015

**Size:** 33 × 60 mm

- **# Layers:** 4
- **# Power Supplies:** 8
- **# Component:** 64
- **# Nets:** 80
- **# Vias:** 181
A.1.3 Cable Connectors

Micro-Coaxial Cable Adapter

Power & Communication Connector

Project: MUXHEAD
Description: Cable connector boards: The micro-coaxial cable adapter PCB provides the landing pads for the micro-coax cores and their common shielding.

Designer: Pascal Alexander Hager
Year: 2015
### A.2  LightProbe – Proof-Of-Concept

#### Photograph

<table>
<thead>
<tr>
<th>Project:</th>
<th>LIGHTPROBE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>Intermediate Demonstration System: A reduced channel count test system of the LIGHTPROBE hardware. This 32-channel system was built to evaluate the hardware components before the more compact PCB-stack was designed.</td>
</tr>
<tr>
<td>Designer:</td>
<td>Pascal Alexander Hager</td>
</tr>
<tr>
<td>Year:</td>
<td>2016</td>
</tr>
<tr>
<td>Size:</td>
<td>163 × 120 mm</td>
</tr>
<tr>
<td># Layers:</td>
<td>6</td>
</tr>
<tr>
<td># Power Supplies:</td>
<td>10</td>
</tr>
<tr>
<td># Component:</td>
<td>622</td>
</tr>
<tr>
<td># Nets:</td>
<td>561</td>
</tr>
<tr>
<td># Vias:</td>
<td>1459</td>
</tr>
</tbody>
</table>
A.3 LightProbe

A.3.1 MotherBoard

Project: LightProbe
Description: LightProbe Motherboard: The motherboard of the final LightProbe design featuring the 64-channel ultrasound receive frontend, the fiber optics link interface, the clock management and several power supplies.

Designers: Pascal Alexander Hager (Schematic)  
Alfonso Blanco (Layout)
Year: 2016

Size: $183 \times 43$ mm

# Layers: 6
# Power Supplies: 10
# Component: 493
# Nets: 543
# Vias: 1443
A.3. LIGHTPROBE

A.3.2 TX Board - LM96551

Photograph

Project: LIGHTPROBE
Description: Transmit Stage Board: The 32-channel 100 Vpp transmit stage featuring the integrated transmit beamformer LM96570 and the high-voltage pulser IC LM96551.

Designer: Pascal Alexander Hager
Year: 2016

Size: $71 \times 40$ mm

# Layers: 6
# Power Supplies: 8
# Component: 210
# Nets: 170
# Vias: 712
A.3.3 TX Board - HV7350

**Project:** LightProbe

**Description:** Transmit Stage Board: The 32-channel 100 Vpp transmit stage featuring the integrated transmit beamformer LM96570 and the high-voltage pulser IC HV7350. Compared to the LM96551, the HV7350 pulser requires fewer power supplies (-3) and has lower leakage currents.

**Designer:** Pascal Alexander Hager

**Year:** 2016

**Size:** $71 \times 40$ mm

**# Layers:** 6

**# Power Supplies:** 5

**# Component:** 202

**# Nets:** 170

**# Vias:** 631
### A.3.4 Power Supply Board - v1

**Description:** Power Supply Module v1: Power supply module for the LightProbe creating out of a single external 5V supply all voltages required for the ultrasound frontend (1.8 V, ±5 V, ±10 V, ±50 V, -55 V). Supports both transmit board variants.

**Designers:** Pascal Alexander Hager (Schematic)  
Alfonso Blanco (Layout)

**Year:** 2017

<table>
<thead>
<tr>
<th>Project</th>
<th>LightProbe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Power Supply Module v1: Power supply module for the LightProbe creating out of a single external 5V supply all voltages required for the ultrasound frontend (1.8 V, ±5 V, ±10 V, ±50 V, -55 V). Supports both transmit board variants.</td>
</tr>
</tbody>
</table>
| Designers   | Pascal Alexander Hager (Schematic)  
Alfonso Blanco (Layout) |
| Year        | 2017                     |
| Size        | 71 × 40 mm               |
| # Layers    | 6                        |
| # Power Supplies | 9                        |
| # Component | 257                      |
| # Nets      | 142                      |
| # Vias      | 385                      |
A.3.5 Power Supply Board - v2

Photograph

Project: LightProbe
Description: Power Supply Module v2: Revision of the power supply module for the LightProbe with improved inrush current handling and EMI. Creating out of a single external 5V supply all voltages required for the ultrasound frontend (1.8 V, ±5 V, ±50 V). Only supports the HV7350 transmit board variant.

Designers: Pascal Alexander Hager (Schematic)
Alfonso Blanco (Layout)
Year: 2017

Size: 71 × 40 mm
# Layers: 6
# Power Supplies: 6
# Component: 203
# Nets: 110
# Vias: 380
A.3.6 USB Type C and WiFi Interface Board

Project: LightProbe
Description: Interface Board: Extends the LightProbe with a USB Type C connector for USB power supply (5 V, max. 3 A) and for the USB 2.0 control interface. The interface board also features a Wireless LAN IEEE 802.11b/g/n module for wireless probe connectivity.

Designer: Matthias Brøgger (Student Project)
Year: 2017

Size: 81 × 43 mm  
# Layers: 4  
# Power Supplies: 2  
# Component: 49  
# Nets: 44  
# Vias: 232


[23] Koninklijke Philips Electronics N. V., “iE33 xMATRIX.”
[34] Chephasioncs, “RX Beamformer CSC2032.”


[64] bkultrasound (analogic), “Sonic Window.”


[69] Supersonic Imagine, “Aixplorer.”


[77] United Microelectronics Corporation, “UMC L130 PROCESS.”


Curriculum Vitae

Pascal Alexander Hager was born on the 13th of March in Oberdiessbach, Switzerland, in 1990. He received both his M.Sc. degree with distinction in electrical engineering and information technology from ETH Zurich, Switzerland in 2014. Since then, he has been with the Integrated Systems Laboratory (IIS), ETH Zurich, where he is currently pursuing his Ph.D. degree under the supervision of Prof. Dr. Luca Benini. His research interests include medical ultrasound imaging, digital signal processing and low-power integrated circuit design. Mr. Hager received the Best Paper Award at the IEEE VLSI-SoC Conference in 2013 and the IEEE ESTIMedia Symposium in 2016, as well as the Best Poster Award at the Nano-Tera Annual Meeting in 2015.