Airborne Broadband Ultrasonic Tracking: Algorithms and Architectures for Mobile Devices

Mohammed Alloulah

Airborne Broadband Ultrasonic Tracking: Algorithms and Architectures for Mobile Devices

Mohammed Alloulah* et al.

Abstract—This paper describes methods for embedded, real-time airborne broadband ultrasonic tracking. The tracker has been built around the assumption of a mobile operation that is deployed ad hoc and upon demand. In order for this to happen, the embedded, real-time operation of sensor nodes has been emphasized. The efficient signalling designs that make way for multiuser, ad hoc tracking deployment are thoroughly characterized, and shown to perform close to their infrastructure-reliant counterparts. System-level parameterization of tracking is also possible subject to application needs.

We then demonstrate that real-time Doppler processing in the airborne broadband ultrasonic modality is possible, whereby velocity inference of mobile nodes is facilitated. Building on advancements from underwater acoustics research, a complex Doppler receiver has tough implications on real-time realizations. We study and characterize these requirements utilizing a high-level synthesis architectural exploration methodology. This has revealed that it is feasible to implement real-time Doppler tracking for airborne broadband ultrasound using modern reconfigurable fabrics.

Index Terms—Airborne Broadband Ultrasound Tracking

1 Introduction

The tracking of mobile users and their devices indoors is of great significance to a number of application domains. Tracking encompasses initial positioning (or localization) followed by continual monitoring of movement thereafter. Numerous examples can be found, from applications such as spatial file sharing in smart environments, computer-assisted living, robotics, all the way to full-blown motion capture. Many technologies have been shown to supply applications with tracking services. These technologies are complementary, and none possesses all desirable tracking attributes for all situations. The preference for the use of a particular tracking technology is often trumped by application-mandated considerations rather than isolated technological merits.

Widely available, pre-existing infrastructures indoors are useful for applications requiring only *coarse-grained* tracking. The most faithful example of this is perhaps WiFi fingerprinting techniques [1].

The provision of accurate, *fine-grained* tracking has been classically achieved by infrastructure-reliant systems. By heavily instrumenting a confined operational space, technologies such as ultrasound, ultrawide band (UWB), optical, magnetic, and hybrid sensing can meet the requirements of all systems [2], [3], [4], [5]; high-end and low-end alike. However, this comes at the expense of high price tag, installation overhead, calibration effort, and end-user expertise. These factors rule out the cost-effective deployment of these systems for all but the most specialized groups of users e.g. research labs, industrial plants, animation studios, etc.

Ultrasonic trackers have been shown anecdotally to be popular for the provision of tracking services indoors [6]. The low-rate of ultrasound translates into advantages in terms of accuracy, cost, and ease of installation. However, narrowband ultrasound poses restrictions on the real-world utility of these tracking systems, especially in scenarios pertaining to multiuser operation, high mobility, and rapid deployment in unprepared environments.

The use of broadband ultrasound in-air was first shown to afford a number of system-level enhancements to tracking in [7]. Since, the notion of wideband ultrasonic sensing was further explored in literature [8], which consolidates the earlier findings in [9]. However, airborne wideband ultrasonic signals present two primary challenges which have thus far limited their effective use in real-world tracking systems. These challenges fall under two categories: (1) embedded, real-time realization; and (2) proneness to aggressive Doppler distortions as a result of human-scale movements indoors. Consequently, elaborate interactive applications remain difficult to realize using airborne broadband ultrasound (ABU) for location sensing.

This paper seeks to advance the state-of-the-art in ABU sensing on two categoric frontiers of equal importance to real-world trackers: (1) amenability to embedded, real-time realization; and (2) Doppler-tolerant tracking for moving tags. In order for this

*E-mail: alloulah@outlook.com

Backdated: 31 July 2013; Version 1.0.

Manuscript had originally another co-authors in addition to the primary author currently withheld pending their approval.

to happen, the paper considers the problem at hand holistically, and jointly addresses design issues across algorithms and architectures, all targeted to mobile devices.

2 Related Work and Motivation

2.1 Ultrasonic Tracking Systems

2.1.1 Broadband

Since the early implementations of broadband ultrasound for indoor localization [9], few systems have appeared in the literature describing results in the same band of operation or further exploring signalling aspects. Work by Prieto et al. evaluates the ranging performance of CDMA acoustic signals in both still air and moving air conditions [10]. They report subcentimetre error using relatively short (32 bit) Golay codes. However, they utilized more powerful (and bulky) transducers operating in the primarily audible range of 5–25 kHz. This would have allowed for more favourable signal-to-noise ratios than are possible with compact piezofilms operating in the ultrasonic range (above 30 kHz), and it is unclear how much this SNR improvement contributed to their accuracy.

Another system described by Ureña et al. [11] replicates previous signalling [7], except that it substitutes Kasami codes for Gold ones. The system relies on the differential time-of-arrival of signals from five transmitters to estimate one receiver's location. Sporadic measurements are shown but no comprehensive characterization of accuracy within the test volume is conducted. The paper also discusses issues of embedded design, considering sequential and parallel implementations of the Kasami code correlator. The authors report the implementation of single-user, parallel correlator with eight-bit resolution which utilized 2,328 slices on a Spartan-IIE architecture. How the receiver embedded processing logic scales as the number of transmitters increases is not discussed.

Álvarez et al. recommend a limit of about 10 ms of Doppler coherence time to govern code length, in order to specifically address the air turbulence effects they observed outdoors [12]. For indoor environments turbulence is less aggressive, yet still present due to air conditioning units and building draughts. In the proposed solution for tracking in the presence of Doppler, we advocate that equalization per-chip mitigates against the symbol coherence time problem. The advantage of this method is that there are no hard limits on code length, transmission power, or transducer efficiency. More flexibility in these parameters allows the design of devices with small form factor and portability—crucial for small, battery-powered sensor nodes and wearable tags.

Yet another work by Gonzalez et al. characterizes the ranging and orientation performance of CDMA acoustic signals [8]. Two spread spectrum (SS) schemes are compared: direct sequence (DSSS) and frequency hopped (FHSS). Both schemes utilized Kasami codes, a 40 kHz carrier frequency, and a bandwidth of 15 kHz, which is considerably less than the bandwidth in our system (40 kHz). The DSSS chip rate was 10 kHz with 200 chips per symbol, while the FHSS used 2.5 kHz and 50 chips. The evaluated ranging performance reported one hundred measurements at four distances. At the maximum of which (1.24 m), the 95 percentile confidence level of the cumulative error was 4.7 mm. For an 8-element uniform circular array, the 3D orientation performance reported was obtained from few discrete locations by averaging 20 estimate at each. This resulted in: 4.5° yaw, 3° pitch, and 3.5° roll accuracies.

2.2 Smart Antenna Sensing for ABU Tracking

ABU's ability to natively sense the three physical quantities that constitute the spatio-temporal relationship of a group of co-located users—range, speed, and orientation—presents exciting new opportunities and a set of accompanying challenges. The implications of different ABU sensing topologies on PHY and MAC design are discussed next.

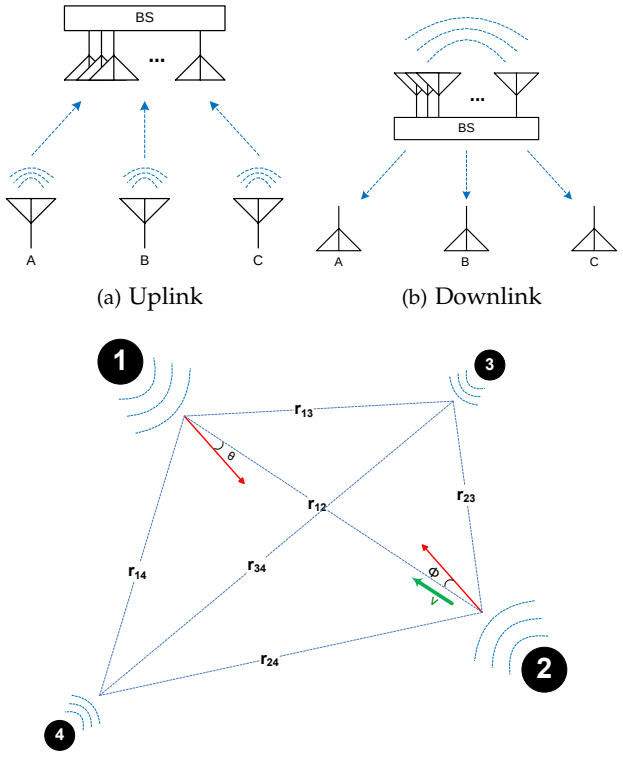
We first remark that in ABU tracking networks, the use of an RF back channel for loose synchronization is assumed. For a group of co-located users, one node will be tasked with regulating sensing through the use of a broadcast-style RF trigger, synonymous to a "tick" in simulation. This seemingly added overhead is a small price to pay for enhancing the longevity of the network and simplifying ABU transceiver requirements. Accomplishing the same task through acoustic messaging will be significantly more expensive both in terms of energy and signalling complexity. It is also assumed that nodes will report their ABU measurements over RF at a much sparser periodicity.

An ABU network consists of a number of nodes, possibly of heterogeneous architectures. A node capable of more functionalities than simple ranging will be designated as basestation (BS). In the following, we further describe the operation of possible topologies, and discuss their differing cross-layer implications.

2.2.1 Topology 1: Uplink

As shown in figure 1a, here ABU tags are transmitters and the BS is a receiver. This topology has an obvious ABU capacity limit, albeit one enhanced by spatial combining. There are three implications topology 1 has on the system-level design.

Range-velocity uncertainty. Though theoretically feasible, multicode acquisition in the dual presence of range and velocity uncertainties is hard and expensive. The relaxed scenario which is multiuser tractable is for ABU messaging to be initiated from a stationary position. One way to reconcile this constraint with usage scenarios is for a tag to request Doppler-tolerant tracking from BS, say using a push-button.



(a) Uplink (b) Downlink

(c) Hybrid P2P

Fig. 1: Smart antenna ABU networks

Higher-order motion moments. Prevalent as a result of human-scale movements, accurate Doppler-tolerant tracking is a challenge in the presence of higher-order motion moments i.e. acceleration and jerk. That is, special signal processing techniques need to be developed in order to realize ABU motion inference resilience irrespective of operational conditions.

Power Control. Power control is important owing to the pronounced distance-dependent channel variations in ABU [13]. That is, the physics of propagation across the wide band of acoustic frequencies dictates the need for extensive power control—beyond what various modulation schemes can alone alleviate. For an ad hoc ranging system, power control can only be handled by MAC as to achieve the best possible acoustic equilibrium among transmitters at different distances. Additionally, such loose power control could be implemented by the RF messages that report measurements to the end application. Doppler tracking power control, however, could be achieved to within ± 0.2 dB in similar vein to established techniques in commercial systems [14].

2.2.2 Topology 2: Downlink

Figure 1b depicts a downlink ABU topology in which tags are receivers and the BS is a transmitter. The tethered BS which has less restrictions on form factor and power affords a drastically different PHY approach. As such, the tethered BS assigns many unique spreading codes, each steered towards a particular spatial sector, sums the resultant beams, and transmits the aggregate beam continuously. The continuous stream resembles that of GPS in that it consists of *data* frames with acquisition preambles. Consequently, the need for RF synchronization is alleviated altogether. Such topology is infinitely scalable. However, this comes at the expense of equipping receiver tags in topology 2 with *binary* Doppler-tolerant reception capabilities.

Binary Doppler-Tolerant Reception. This receiver effectively doubles the computational complexity i.e. silicon area. It has been reported in the UWA literature. Dubbed the hypothesis feedback equalizer [15], it entails having two parallel adaptation branches all the time scanning for two [binary] hypotheses ahead of the end of any given symbol in order to mitigate against data transitions.

2.2.3 Topology 3: Hybrid P2P

Another hybrid topology is possible as illustrated in figure 1c. Here the emphasis is on peer-to-peer (P2P) sensing in which a combination of nodes capable of ranging, spacial functions, and Doppler processing is present. This topology relies again on an RF MAC for the successful execution of functionality. Such heterogeneous sensing will arise naturally as a result of the evolution in the network usage over time as nodes opt in and out of the service ad hoc. Thus it is most accommodative of versatile and demanding user experiences. However, this comes at the expense of supporting the reconfigurability of tracking.

Reconfigurability: The ability to adapt tracking to arising needs is a feature that could potentially enhance the user experience. This reconfigurability should at the same time facilitate the full utilization of the ABU channel. That is, the choice of signalling in ABU should lend itself to a *multiuser* operation that is both computationally tractable and accommodative of different sensing functions; be them ranging, Doppler-tolerant tracking, or beamforming for active steering or direction-of-arrival estimation.

2.3 Limitations of Ultrasonic Tracking Systems

2.3.1 Broadband

When so many systems in the literature are described by being broadband while having differing bands of operation, performance comparison becomes unreliable. However, three major observations on the state-of-the-art in broadband ultrasonic tracking hold:

1) **Unfavourable transducers:** Many of the aforementioned systems utilize bulky transmitter and receiver devices which severely limit the usage scenarios of broadband ultrasonic tracking systems. This is especially true for the targeted user-centric applications operating in a mobile fashion, which require light-weight and rugged tags. That is, the design of signalling techniques for ABU should take into account realistic, imperfect

conditions e.g. the low sensitivity of rugged, lightweight, and wideband piezo films.

2) **Decreased bandwidth:** Due to physical transduction limitations, the above systems utilized only a fraction of the bandwidth originally reported in [7]. Among other things, this influences the capacity of the channel—a property that substantially impacts the perceived quality of service (QoS). This is because both the number of coexisting users (multiple access) and their update rates are increased with the enhancing of capacity as a result of wider bandwidth.
3) **Motion-intolerance:** All reported systems fail the instance tags begin to move while emitting or receiving the wideband acoustic signals. This inability to perform sensing is due to the pronounced Doppler effect in the ABU band as a result of human-scale movements. This singular limitation severely affects human-centric tracking since stationary sensing can not be guaranteed.

2.4 Joint system-level design

Previous approaches do not consider the system-level problem at hand holistically. That is, no attention is paid to how embedded peer-to-peer sensing between mobile devices could be realized. Nor do these works address the problem of extending their choice of signalling to supply say coherent processing in situation when such sensing enriches interactive scenarios. For example extending FHSS to coherent processing for velocity estimation is, while being theoretically feasible, difficult [14, p. 57]. Hence approaches that touch on one aspect of a tracking system in isolation of equally important system-level issues are bound not to take off. Another example on this point is the use of super-resolution array processing methods that give degree-level accuracy. This level of accuracy is irrelevant when a realistic system performing peer-to-peer sensing ought to support multiuser beamforming with potentially a modest but sizeable number of users vying for the service.

With a cross-layer approach, the new notion we advocate for is involved processing for the novel ABU modality. The aim is to support a new class of emerging, infrastructureless systems that place particular emphasis on uninstrumented and unprepared environments through self-contained P2P sensing. However, the area-power-cost requirements of high-complexity wireless signal processing are best addressed through joint optimization, wherein algorithms and architectures are cross examined in order to arrive at a tradeoff particular to a set of design constraints [16], [17]. For high-complexity algorithms, joint optimization also calls for an agile electronic system-level (ESL) design methodology. Under such a methodology, algorithmic-architectural design space co-exploration is steered judiciously by the informed user while being performed automatically with guaranteed quality of results [18], [19].

This research reports on advances for ABU PHY designs targeted at multiuser ranging and Doppler-tolerant tracking. These designs are proposed while keeping a system-level view at gaze for a rapidly deployable ABU node topology in unprepared environments. In order to achieve the objective of rapid deployment, we outline our signalling approaches alongside their requisite digital architectures, demonstrating how to realize an end-to-end system.

3 ABU Tracking

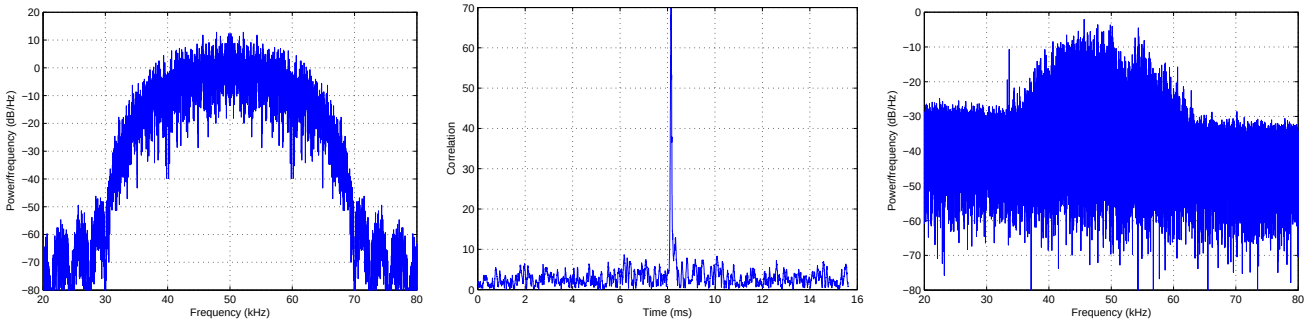
The proposed ABU tracker is comprised of a transmitter with configurable parameters and two modes of receiver detection; static and Doppler-tolerant.

3.1 Transmitter

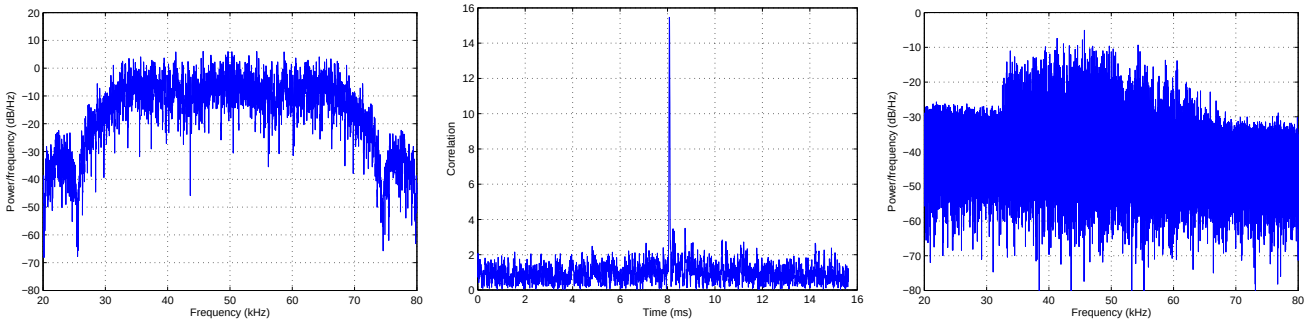
The transmitter is capable of changing its spreading rate as to increase the ABU channel spectral occupancy. Also depending on the receiver mode, it either sends a burst of one code for time-of-arrival (TOA) estimation or a continuous packet transmission for velocity estimation through the Doppler effect. A block diagram of the transmitter is shown in figure 3. The digital transmitter consists of a Gold code generation unit, a numerically-controlled oscillator (NCO), and a DSSS modulator. A mixed-signal DAC then follows whose output is in turn smoothed with an analogue reconstruction filter. Various signal scaling, conditioning, and amplification are applied before finally driving the piezo film.

In order to highlight the signalling adaptability of the transmitter, two chip rate transmission schemes are contrasted next. The nominal 20 kHz (or kChip-s/s) chip rate was doubled and a raised-cosine filter with a roll-off factor of 0.15 was used to limit the bandpass bandwidth so as to operate within the characteristics of the combined transmitter-receiver ABU wireless channel. The remaining system parameters were: 200 kHz sampling frequency for both the transmitter and the receiver, and 10x & 5x oversampling for 20 kHz & 40 kHz chip rates, respectively. Under this setup, the PSDs of the transmitter's excitation signals are illustrated in figures 2a and 2d. It is clear that the doubling of the chip rate spreads the energy of the DSSS ABU signal more around the carrier (50 kHz).

In relation to the ranging performance, a transmitter-receiver pair is positioned on-axis at a distance of 2.741 m. The time-of-flight of a code-length burst is measured under the two chip rates. A one-second continuous transmission is sampled at the receiver for the evaluation of the PSD for both chip rates as well. Figure 2 depicts the correlation and PSD obtained from the two configurations. The calculated distances from the correlation peaks of the



(a) Tx PSD$|_{f_{chip}\ =\ 20\ \text{kHz}}$ (b) Rx Corr$|_{f_{chip}\ =\ 20\ \text{kHz}}$ (c) Rx PSD$|_{f_{chip}\ =\ 20\ \text{kHz}}$



(d) Tx PSD$|_{f_{chip}\ =\ 40\ \text{kHz}}$ (e) Rx Corr$|_{f_{chip}\ =\ 40\ \text{kHz}}$ (f) Rx PSD$|_{f_{chip}\ =\ 40\ \text{kHz}}$

Fig. 2: Parseval view of the ABU DSSS system at two chip rates: 20 kHz & 40 kHz.

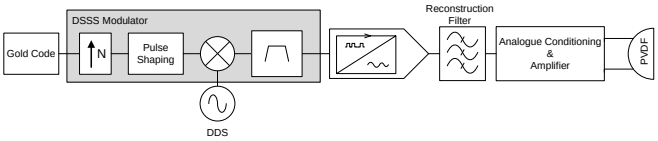


Fig. 3: Transmitter block diagram

20 kHz and 40 kHz chip rates are 2.753 m and 2.729 m, respectively. The ranging errors for both cases are on the order of 1 cm. Inspecting the corresponding correlations of figures 2b & 2e reveals that the double rate correlation gain is reduced with respect to the single rate. This reduction is around 6.5 dB. However, so does the noise of the correlation—the faster rate seems to have "dithered" the noise better. As expected from Parseval's duality principle, the 40 kHz PSD in figure 2f is also reduced similarly with respect to the 20 kHz PSD of figure 2c. It also can be seen that the double rate PSD is more spread around the 50 kHz carrier but remains buried in the noise floor of the receiver outside the intended range due to stricter roll-off factor control.

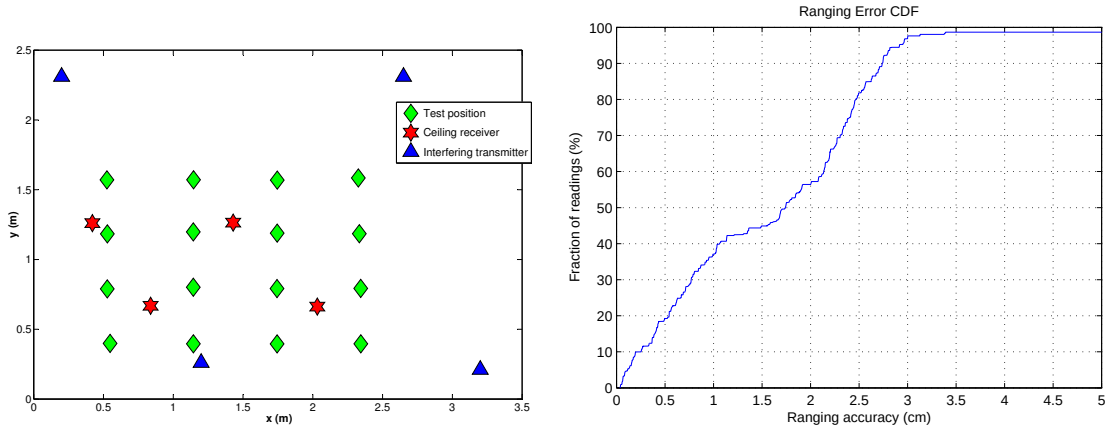
3.2 Receiver

There are two detection modes that an ABU receiver needs to support:

- *Static*: In this case, nodes are stationary most of the time and rely on TOA inference after a broadcast-style RF trigger has been sent to initiate the pseudoranging[1] interval for a group of co-located users.
- *Doppler-tolerant*: In case of severe Doppler distortion resulting from motion (e.g. caused by someone walking while wearing a node), a more sophisticated receiver mode must be devised. In order to keep the multiuser problem tractable, and depending on the tracking interval, a number of unmodulated codes are sent that would train the receiver adaptively in a mode of operation designated as *joint range-speed tracking*. Here, acquiring timing readily derives range in sub-chip resolution depending on the chip oversampling rate employed. Adaptive training also produces chip-rate phase variations as a by-product of coherent processing which can be used to estimate node velocity.

  This mode is analogous to the rationale of Sutherland's pioneering head-mounted display system [20]. Sutherland proposed a continuous wave source transmission (i.e. standing wave) whereby the relative phase is tracked in order to estimate the distance at the receiver. Sutherland's proposal had two shortcomings. First, the relative distance is measured only within a cycle. Meaning, absolute distance is worked out by keeping track of initial distance and the number of accumulated cycles. Second, multipath receptions

1. This term is borrowed from the GPS world and is intended to reflect the uncertainty of the measurement process.



(a) Top view of deployment area

(b) Cumulative distribution of ranging errors

Fig. 4: Ranging performance for a static, embedded ABU receiver.

of the continuous wave will inevitably interfere with the direct path and degrade the measurement setup. Both issues are overcome in the ABU method proposed in this thesis. First, by making a "standing" spreading code equate to the maximum expected distance, code timing readily achieves distance disambiguation, of course provided that transmission and acquisition commence at the same time—The RF broadcast-style trigger message will guarantee this. Second, in direct sequence spread spectrum (DSSS) signalling, multipath arrivals at the receiver are resolvable, alleviating the problem in narrowband waves.

3.2.1 Static

The ranging performance of a static, embedded ABU receiver is illustrated in figure 4. The cumulative distribution of ranging errors was generated using a dataset gathered from real ABU deployment through cosimulation. As such, results are representative of real-world, resource-constrained performance.

The experimental conditions of the measurement setup are summarized here. Gold codes of length 511 were used to BPSK modulate a 50 kHz carrier, at a chip rate of 20 kHz. The signal from each receiver's analogue stage was sampled at just over 200 kHz, with twelve bits of resolution per sample.

A primary transmitter node was used to emit one hundred ranging messages, across sixteen horizontal positions (shown as diamonds in Figure 4a), at each of three different heights (10, 50 and 100 cm above the floor). To emulate the presence of co-located users, four other transmitter nodes (triangles in Figure 4a), emitted their ranging messages simultaneous to the node placed at the test positions. All transmitters emit ranging signals at the same acoustic power. Four receiver nodes (stars in Figure 4a) were mounted on the ceiling, 220 cm above the floor.

Figure 4b illustrates the multiuser performance of the co-simulated dataset. For each of the four ceiling receivers, the one hundred ranging events were processed at the forty-eight transmitter locations—a total of 19,200 measurements. The core's overall detection rate for ranging signals from the primary transmitter node is 65.4%.[2] The undetected ranging signal occurrences are commensurate with two effects previously observed with this deployment: (1) the directivity of the transducers [7, figs. 7 & 11]; and (2) transmitter near-far effects [7, sec. 4.3]. When the despreader core successfully detects events, its ranging error is better than three centimetres (2.92 cm) at the ninety-fifth percentile confidence level.

3.2.2 Doppler-tolerant

Tracker configurations. There are three configurations that demonstrated ability to track the Doppler effect in ABU: the phase-locked loop (PLL), decision-feedback equalizer (DFE) with embedded PLL, and DFE with linear interpolator (LI). The following will quickly review these configurations.

(1) PLL: In and by itself, the PLL does not constitute a reliable tracker configuration. However, it is instructive to study the architectural implications of the PLL-alone tracker since the DFE-PLL configuration does result in motion-correlated measurements. As such, cross examining the PLL and DFE-PLL tracker configurations will build an understanding on how the architectural requirements change with the inclusion of the feedforward filter (FF) which completes the implicit DFE-PLL tracker.

(2) DFE-PLL: The DFE-PLL tracker configuration provides a means for directly estimating the instantaneous frequency (IF) resulting from motion stresses in ABU. It could be of particular merit in certain situations wherein the general pattern of motion is

2. Ranging success rates higher than the reported 65% would be easily achievable using dynamic thresholding and enhanced peak detection methods [21, p. 94,109, & 152].

sought after, rather than accurately measuring velocity. That is, this configuration trades accuracy for robustness. Interestingly though, the DFE-PLL tracker is only valid for a chip oversampling rate of $N_s = 4$. This is a purely algorithmic finding which was arrived at after thorough empirical evaluation [22].

(3) DFE-LI: The DFE-LI configuration is the most accurate velocity tracker. This, however, comes at the expense of sensitivity to high order motion moments (e.g. acceleration and jerk), which are prevalent in ABU as a result of human-scale movements indoors.

Signal Design and Experimental Setup. A Gold code sequence of length 511 was BPSK-modulated onto a 50 kHz carrier at a 20 kHz chip rate. Chips were pulse-shaped using a raised-cosine filter with roll-off factor of 1. The sampling rate utilized was 160 kHz.

One dataset will be used for the characterization of the two algorithmic configurations. This dataset has typical maximum speed (about 0.8 m/sec) and contains high-order motion moments. In order to facilitate characterization, we utilize a vision-based tracking system for the generation of real-time position and speed groundtruth. A transmitter is mounted on a Lego Mindstorms robot moving forward and backward on a track while facing a stationary ABU receiver, at a distance of 2.9 m. The robot is tagged with a fiducial marker [23]. A PC-based application monitors the movement of the robot using the computer vision-based tracking. Upon motion, the application triggers an FPGA-based ABU transmitter and simultaneously commences data acquisition. The all-digital transmitter has an independent oscillator to the PC clock. The experimental setup used for data collection is shown in figure 6.

Performance. The acquired signal was down-mixed to baseband where all processing takes place. Figure **??** illustrates the performance of the DFE-LI tracking configuration. As evident from the first plot, the channel is stable and code timing is maintained throughout tracking. The second plot depicts of the evolution of the linear interpolation factor in time and the chip estimate scatter. The sensitivity of the DFE-LI tracker to acceleration and jerk was avoided initially by a priori knowledge of the Doppler stimulus. As such, the instance at which LI is activated was delayed. As shown, DFE-LI tracks Doppler very accurately during the first and most of the second motion legs. The velocity estimate for the first motion segment was measured to be 0.8047 m/s, which is virtually identical to groundtruth. However, following the decelerating motion moments in the second leg, LI diverges away from rest condition owing to high second-order motion stresses. The third plot examines this stress resulting from second-order motion moments. The IA estimator shows distinct sustained acceleration spikes in three places: at the beginning of the Doppler stimulus around 0.5 sec; around the momentary pause of the robot at roughly 3.5 sec between leg 2 & leg 3; and at the end of the test case slightly before 7 sec. These instances are in line with the acceleration groundtruth of the Doppler stimulus.

Similarly, the performance of the DFE-PLL tracking configuration is illustrated in figure **??**. Justifiably, the crude DFE-PLL tracker is noisier even though code timing stays in-lock. In the second plot the PLL phase evolution is shown alongside the chip estimate scatter, which is considerably more spread in comparison to that of the DFE-LI configuration. The third plot shows the scaled unwrapped phase of the chip estimate overlaid on the vision tracking groundtruth. The scaled instantaneous phase is evidently correlated with motion in this competitive mode, which provides a crude means for estimating velocity. This method for inferring the motion pattern could be useful in certain applications not requiring precise velocity measurements.

4 Architectures

4.1 Transmitter

The transmitter architecture is modelled in both floating-point and fixed-point arithmetic. The objective is to demonstrate that the fidelity of the proposed embedded realization is commensurate with what the airborne broadband ultrasonic channel has to offer.

The stimulus is baseband digital Gold code signal with infinite quantization precision. Using the fixed-point model of the transmitter, table 1 shows the obtained SNR at two chip rates. Varying the chip rate has a mild effect on the SNR of the transmitter's excitation signal. Roughly a drop of 4.2 dB accompanies the doubling of the chip rate. However, the two SNRs are significantly higher that that of the aggregate ABU channel which stands at approximately 24 dB, at one meter range [21, p. 76].

TABLE 1: Transmitter SNR

Chip Rate (kChips/s)	Roll-off (β)	Input SNR	Output SNR (dB)
20	1.0	∞	67.64
40	0.15	∞	63.47

The signal at baseband is binary, which results in efficient digital structures and similarly simplifies arithmetic operations. The silicon area of the transmitter prototype for both parameter sets analyzed earlier is given table 2. Evidently, the transmitter is compact and lends itself well to low cost, low power realizations for handheld devices.

4.2 Static receiver

An efficient core that makes acoustic simultaneous multiple access possible is shown in figure 7. The proposed core consists of the following units: a complex

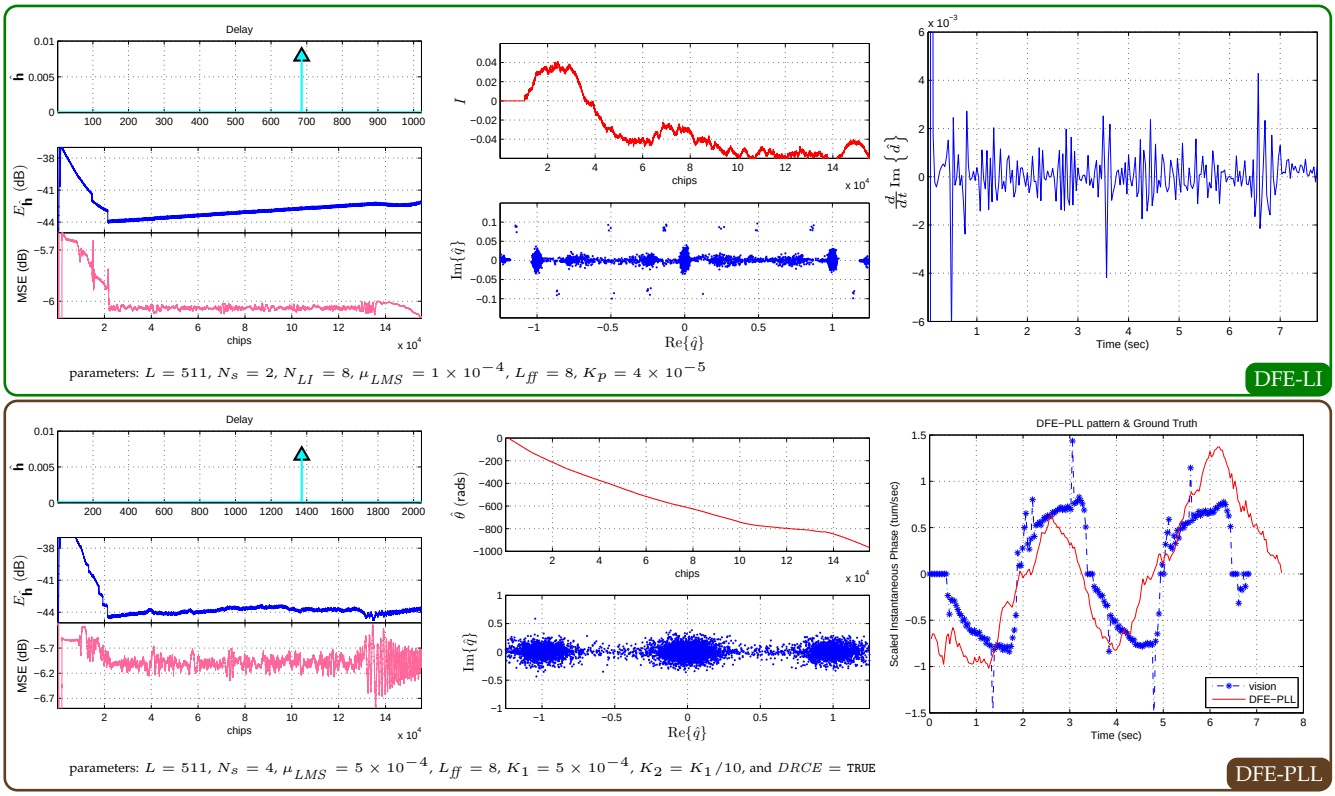
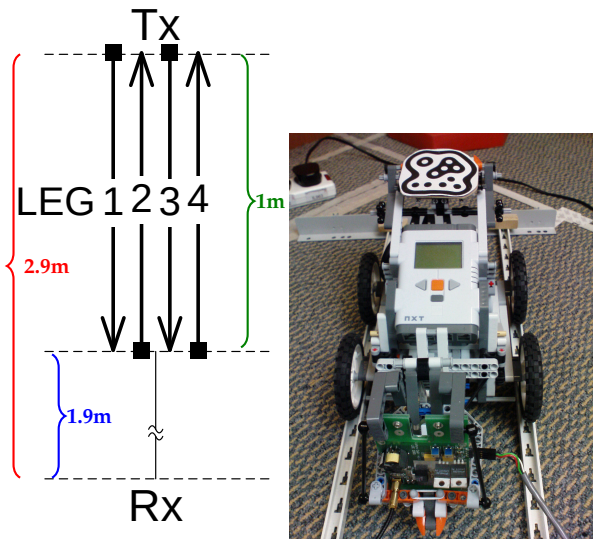


Fig. 5: Tracking performances: upper row DFE-LI and lower row DFE-PLL.



(a) (b)

Fig. 6: Experimental setup. (a) Transmitter path moving forward and backward while facing a stationary receiver. (b) Lego Mindstroms robot with mounted transmitter.

TABLE 2: Tx FPGA area requirements[a]

Resource	Used	Utilization %
DSP48s	1	3
BRAMs	5	13
Slices	308	5

[a] Device: Xilinx Virtex-4 XC4FX12

circular buffer, user code generation unit, a matched-filter bank, and a control state machine. The matched-filter bank has a number of complex fingers. The complex fingers are aimed at parallel multiuser reception as opposed to multipath. Each finger is assigned two codes in an interleaved fashion to maximize resource sharing. The Gold code is generated on-the-fly and stepped through the operation, rather than storing the long code sequences in memory.

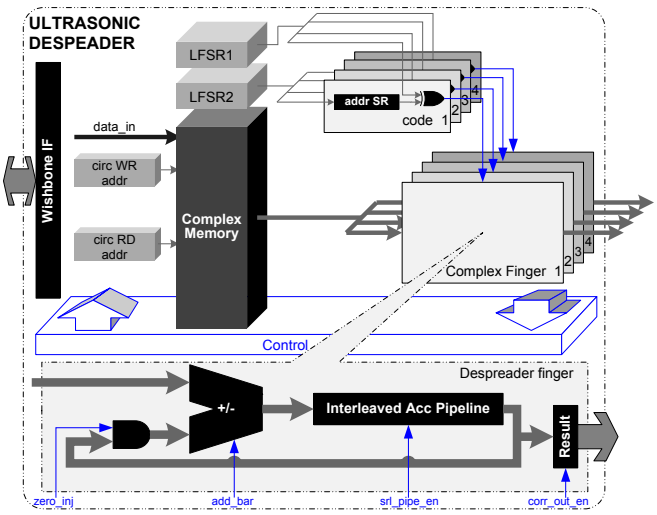


Fig. 7: Multiuser despreader block diagram

To demonstrate the design, an eight-user ultrasonic

despreader was implemented on a Xilinx Virtex-4 XC4FX12 device using RTL description. This was carried out using Xilinx's 10.3 ISE tool suite. The core operates on a 100 MHz system clock with data quantised to 16.15 signed fractional representation, and 8 guard bits. Signalling parameters utilised are a 20 kHz chipping rate, 511 bit Gold code (which leads to about 8.7 m maximum node separation in *Doppler-tolerant* mode), and four-times chip oversampling rate. Under these parameters, four fingers are required, since a maximum of two interleaved users can be accommodated for in a single complex finger. Table 3 shows the final place-and-route metrics of the four-finger, eight-user core.

TABLE 3: FPGA resource requirements[a]

Resource	Used	Utilization %
Slices	632	11
LUTs	699	6
BRAMs	4	11
Max frequency	103.67 MHz	

[a] Device: Xilinx Virtex-4 XC4FX12

4.3 Doppler-tolerant receiver—Co-exploration

This section describes the experience of utilizing an HLS tool for C++ to RTL generation. The following presents hardware design issues that are specific to the algorithmic workload.

4.3.1 Chip Oversampling

In summing up the spirit of the proposed *chip-oriented* architecture, spreading code vectors' worth of oversampled entities are fed to the custom computational datapath. These entities can be an oversampled chip, a reconstructed oversampled chip, or a fractional channel coefficient. The algorithm consists largely of vector arithmetics at the DSSS length ($L \times N_s$) and convolution loops at the code length (L) applied to fractional entities. With 20 kHz chip rate and a nominal digital system clock of 100 MHz, the maximum clock budget per adaptation step is 5000 cycles. While this is quite long in digital terms, the number of loops that have to be performed on the DSSS-long vectors can easily exceed a chip duration. This is because a large proportion of adaptation is purely sequential in nature entailing vector data dependencies. Nonetheless, the one commonality among all computations is that operations are replicated on oversampled entities. Thus a form of data-level parallelism underlying processing is readily achievable by the concurrent manipulation of an N_s-wide stream. To this end, as will be further explained in subsequent sections, all loops are partially unrolled by N_s while memories are partitioned cyclically by the same factor as to be able to service the N_s-wide computational datapath—hence the chip-oriented architecture.

4.3.2 Parallelism

The algorithm exhibits a strong form of data-level parallelism determined by the chip oversampling rate. However, in order to tap into this underlying algorithmic uniformity, a few other considerations mandated by the processing workload have to be met, such as expression-balancing some timing-critical operations. This rules out the possibility of using a generic SIMD DSP for realizing a computationally-intensive algorithm of such specific nature, while having equally stringent application requirements on form factor and power consumption.

Task (functional) parallelism remains a challenge for HLS, mostly because of the inherent sequential semantics of high-level languages [24]. In the context of the co-exploration reported in this paper, latency is a critical requirement for the overall adaptation, as is often the case for real-time transceiver designs. Thus all functions are hardware-inlined in order to remove function hierarchy and eliminate calling overhead. Consequently, the functional parallelism problem for HLS is reduced to scheduling, which can be inferred by data dependencies analysis.

Instruction parallelism is automatically handled by HLS. Since the early introduction of HLS as a viable design methodology, more credible automation can now extract instruction parallelisms seamlessly, and can be further aided by the judicious intervention of an expert user through code transformations [25]. This allows for added parallelism subject to broader design space investigation for various trade-offs (such as area, speed, etc) by means of which the final solution is steered.

For the real-time adaptive application at hand, boosted levels of instruction parallelism have been further extracted by pipelining the numerous loops that constitute the algorithm. In addition, the original software code underwent a major refinement with respect to instruction parallelism. This involved visualizing a classic pipeline in mind and rewriting the untimed, sequential C++ code as to reflect a pipelined operation with prefetch, execute, writeback if applicable, and update. This is important so that the pipeline initiation interval (II) remains at one even though the pipeline depth itself could be as high as 30 in certain loops, hence resulting in no pipeline stalls. In such a case, after a small initial ramp-up latency (e.g. 30 clock cycles), the throughput of a loop would be maximized at one operation per clock cycle.

Another remark pertaining to parallelism is the following. In the object-oriented design, global parameters were loaded in variables local to loops or functions prior to proceeding with computations. This is necessary for maximum throughput in these rather long loops. Failure to do so results in long pipeline stalls since the tool effectively translates every occurrence of a global parameter into a function call, which greatly degrades operation scheduling.

4.3.3 Memory

Partitioning. Exploiting the underlying data-level parallelism necessitates a multiport memory architecture as to increase data throughput. In order to read out a complete oversampled chip from the signal vectors in one clock cycle, memories were cyclically partitioned by N_s. This is done without introducing any modifications to the code by applying high-level directives. Cyclic partitioning of a signal vector results in a number of equivalent smaller vectors whose entries are the parent's entries interleaved across the now increased memory ports by the required factor. This uniform partitioning is applied to all DSSS vectors; circular and linear. Furthermore, both the Gold code sequence and its estimate are similarly partitioned. This is done for a different reason as will be discussed in the following section. Finally, since the feedforward filter is targeted towards chips, it is much shorter and as such does not require partitioning.

Consistent addressing. It is required that the memory access pattern for various vectors be consistent throughout the algorithm. This entailed refining the original code such that addressing these memories becomes unified as either high-to-low or low-to-high. This aids streaming of data from one function/loop to another and is of particular importance when vector data dependencies occur. Abiding by this consistent access pattern ensures that a datum entry can be immediately streamed to where it is next required as opposed to having to wait on the whole vector operation.

Contention.

(1) access stall: Various functions combine to make a heavy use of the two circular signal vectors; namely, the signal and average signal vectors. Aggravated by vector data dependencies at times, this gives rise to access bottlenecks due to multiplexing and the limited memory ports available coupled with operation rescheduling and/or retiming, which the tool also automatically performs part of a global optimization problem. These access bottlenecks resulted in pipeline initiation interval (II) of value 2 for two loops in the adaptive algorithm; effectively, doubling their latencies. Initially, this was mitigated against by fully partitioning the two circular signal vectors into registers. In the fully partitioned architecture, the circular vectors have as many ports as entries. This is obviously of tremendous generic fabric cost not to mention the very long decode logic delays on these rather deep DSSS-length vectors. Synthesis had proceeded successfully initially. It was not until placing-and-routing the design that failure to meet timing objectives under the fully partitioning high-level directive surfaced. Therefore, memory access bottlenecks and subsequently pipeline stalls on the two circular signal vectors had to be tolerated for these two loops and were deemed unavoidable as opposed to going fully partitioned. It is interesting to note that this is only applicable to the $N_s = 4$ case—the $N_s = 2$ case did not result in access stalls on these circular vectors.

In order to compensate for these bottlenecks, parallel gain has to be extracted from somewhere else. Bearing in mind the extremely low hardware cost of the last loop `despreading_loop`—i.e. binary coefficients to put it in signal processing terms—it was decided to partially unroll the loop by the same factor that occurs throughout all loop unrolling directives; N_s.

(2) writeback dependencies: The sequence of computations involving the channel estimate vector amounts to four loops; namely, updating, finding maximum coefficient, truncating if applicable, and calculating power. In order to achieve maximum throughput on this subset of algorithmic operations, the following techniques were utilized resulting in modifications to the original code. Firstly, a temporary channel estimate vector was allocated and passed to a first of two channel-related functions which manually merges updating and searching for the maximum coefficient. The updated channel estimate was written to the temporary vector as to mitigate against writeback dependencies that would have otherwise arisen in the pipelined loop. Expression balancing is also applied on the update loop. Simultaneously on the fly, a fast complex magnitude estimator [26] is used for the max search. Listing 1 demonstrates the expression-balanced, pipelined loop. The second loop then takes on the tasks of truncating the temporary channel vector into the original channel vector while calculating the channel power at the same time.

Listing 1 illustrates the various concepts discussed thus far. First, the channel and interference-free signal are fetched from memory into pipleline registers. Second, multiplications are performed and stored in intermediate registers. Third, the channel is updated and the result is written back to the temporary channel vector in order to avoid writeback dependencies. Fourth, at the same time, the magnitude of the current channel coefficient is computed and the result is written back to the channel magnitude vector. Finally, the current magnitude is tested against the maximum coefficient, which is updated correspondingly if applicable.

(3) feedforward filter adaptation: The complex feedforward filter is implemented as an FIR filter with a RAM-based circular buffer. This is more economical than a register file realization especially given the real and imaginary parts of the signal and coefficient buffers; which means quadrupling the generic fabric utilization for such a realization. As a result of the relatively short delayline of the N_s-order filter, adaptation was implemented efficiently incurring only twice the filter's length in clock cycles latency. The two delaylines are loaded into local arrays reversing the

```
1  update_n_magx_loop : for (int m = 0; m < cdma.SIZE(); m
       ++)
2  {
3    //! fetch
4    Yi1 = cdma.ds.h_hat_vect[m];
5    Yi2 = cdma.ds.v0_hat_vect[m];
6    //! multiply
7    h = this->cmplx_mult(Yi1, Xr1);
8    v0 = this->cmplx_mult(Yi2, Xr2);
9    hm = h + ( q_kTc ? -v0 : v0 );
10   //! writeback
11   h_hat_vect_temp[m] = hm;
12   hMag = fxMath.template cmplx_mag_eqrppl<
         h_hat_mag_type, typename D::ch_type, ap_ufixed
         <18,1> >(hm);
13   acqParams.h_hat_mag_vect[m] = hMag;
14   if ( hMag > magxCh )
15   {
16     magxCh = hMag;
17     chDelay = m;
18   }
19 }
```

Listing 1: Merged channel update and maximum coefficient search.

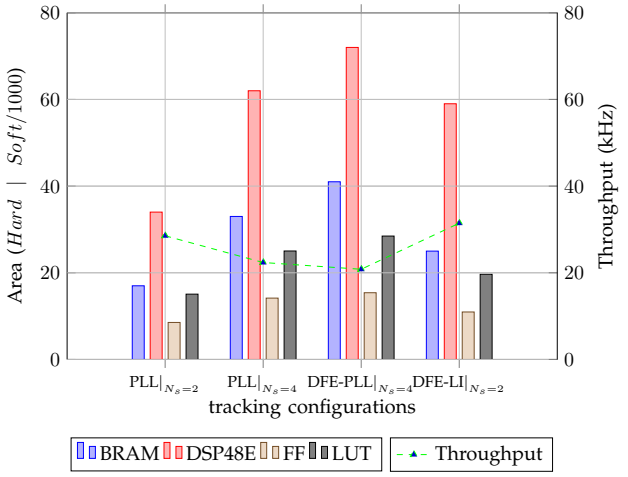


Fig. 8: HLS metrics comparison between PLL, DFE-PLL, and DFE-LI

signal vector. Indexing the delaylines has to be done in a consistent manner with the actual filtering subroutine as elaborated on above. The local arrays are then used with the adaptation factor to update the original coefficient vector while always pipelining and expression balancing for maximum throughput.

4.3.4 Loops

Various loop-related arrangements have already been alluded to in prior discussion. Nonetheless, for the sake of a cohesive treatment of hardware concepts, the transformations pertaining to loops will be enumerated again under this heading. First, channel-related loops are manually merged due to the reasons discussed above. Apart from this manual step, subsequent operations are performed automatically through the use of high-level directives. Second, unrolling is applied on virtually all loops occurring in the algorithm. Nested loops are fully unrolled. The remaining loops in which intensive processing takes place are partially unrolled by N_s. Third, pipelining directives are utilized on the partially unrolled loops for maximal parallelism.

4.3.5 Co-exploration results

Having described the concepts surrounding the algorithm implementation, the results will be supplied next. The results are based on HLS synthesis by Xilinx's AutoESL[3] version 2010.a.2. The device targeted is the Xilinx Virtex5 XC5VSX94T, package FF1136, speed grade -1. The basic synthesis commands are clock period of 10 ns with 1.25 ns uncertainty. The back-end tools are the Xilinx ISE 12.2 suite.

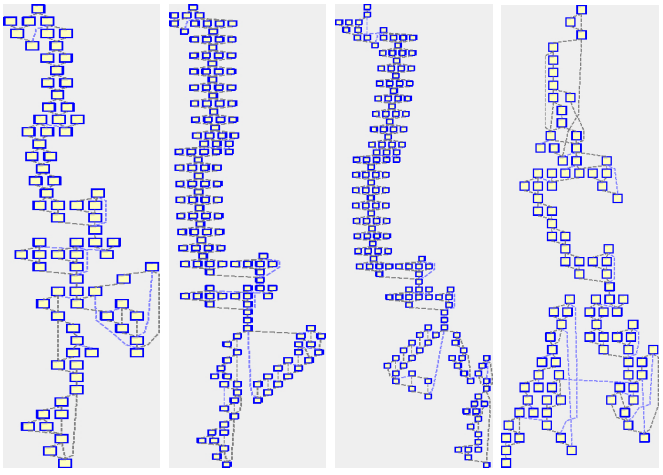
The IO used in the RTL implementation settings are an input complex chip of type RAM and an output flag of type WIRE connected in a slave bus setup. The flag is tied to the boolean value indicating successful acquisition, which is updated every adaptation step.

Figure 8 shows the characterization of area and throughput as reported from synthesis results. Place-and-route (PAR) was used to sanity-check these results. However, PAR results are not be reported. Besides the long completion overhead, this is also because the downstream tools will prune and optimize away all unnecessary logic that is not tied to the final output assignment. The output assignment is currently a single boolean flag indicating [continual] acquisition. A production-level implementation has to communicate out more algorithmic primitives such as chip estimates for estimating velocity, timing for range[4], channel status for fidelity, etc. Note that when requiring more IO assignments, no additional latency will be incurred since external communication can be overlapped with computations [27] using non-blocking reads/writes [28]. It is also worth nothing that in the context of automated architectural exploration, prior work in literature oftentimes reports on execution cycles and area estimates from HLS only in avoiding the lengthy overhead of synthesis and PAR [29], [30].

The rapid co-exploration allowed us to gain in-depth insights into the effect the essential algorithmic parameters have on the chip-oriented architecture. Perhaps most interesting of all was the almost non-effect halving the spreading code (L) had on BRAM usage. This is due to the sequential nature of most of the operations in the algorithm. In the proposed architecture, data is streamed from the global data plane to where it is needed. This streaming is achieved by high throughput FIFOs which are implemented using BRAMs. The number of required FIFOs is algorithm-dependent as opposed to code length-dependent. As expected, the major effect of halving the code length

3. AutoPilot formerly of AutoESL

4. at least once initially



(a) PLL$|_{N_s=2}$ (b) PLL$|_{N_s=4}$ (c) DFE-PLL$|_{N_s=4}$ (d) DFE-LI$|_{N_s=2}$

Fig. 9: CDFG of tracking configurations

on the chip-oriented architecture is seen in the doubling of the computational throughput. On the other hand, halving the chip oversampling rate results in mild throughput enhancement. In order to reap area savings as we decrease code length, modifications to the high-level directives of the chip-oriented architecture have to be made: cyclic memory partitioning and loop unrolling can be relaxed by a factor of two, in order to balance attainable throughput with the requisite area.

Figure 9 shows the control data flow graph (CDFG) for all tracker configurations. The constraining effect of the chip-oriented directives is most evident in the PLL and DFE-PLL configurations at $Ns = 4$—a horizontal increase in the scope of concurrency can be immediately spotted. The DFE-LI tracker, notably, has very favourable data dependencies with two distinct execution branches of minimal coupling towards the latter part of adaptation.

5 Modifications Needed for a Production System

The set of designs reported in this paper should be viewed as a comprehensive feasibility study on the practical deployability of peer-to-peer, ad hoc ABU sensing. In order to realize a production-ready system, a number of further enhancements and capabilities would be necessary to engineer.

The co-exploration of Doppler tracker configurations can be further optimized in terms of fixed precision datatypes. The Doppler effect is non-stationary and a manual fixed-point precision optimization has to be pursued iteratively. This is formally referred to as *simulation driven analysis* in [31]. Thus, the numerical optimization is laborious and should be deferred until a genuine commitment for building a real-time ABU motion tracker is reached. Note that the numerical design optimization in NASA's Apollo mission consumed around 30% of total development resources [32].

Analog circuitries that facilitate power control would be needed for both static and Doppler-tolerant tracking as discussed earlier. In this paper, no attempt was made at realizing any form of power control. The keen reader is referred to [13] for a thorough analysis on the distance-dependent variations in the ABU band and a rationalization for ABU power control.

The reconfiguration of tracking is to be supported by a mixture of algorithmic parametrization and hardware reconfigurability. Further significant developments are needed to realize a configurable ABU tracking system that can dynamically cater for changing end-user needs. Customizable internet devices are part of an increasing trend to leverage finer-grained hardware reconfiguration in support of network adaptability to dynamic conditions.

6 Conclusion

This paper has developed the novel modality of airborne broadband ultrasound (ABU) in support of ad hoc, real-time, and mobile tracking applications.

References

[1] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2000, pp. 775 –784.

[2] "InterSense, Inc," http://www.intersense.com/, 06 June 2011.

[3] "Ascension Technology Corporation," http://www.ascension-tech.com/, 06 June 2011.

[4] "Ubisense Limited," http://www.ubisense.net/, 06 June 2011.

[5] R. Fontana, E. Richley, and J. Barney, "Commercialization of an ultra wideband precision asset location system," *Ultra Wideband Systems and Technologies, 2003 IEEE Conference on*, pp. 369–373, 16-19 Nov. 2003.

[6] M. McCarthy, P. Duff, H. L. Muller, and C. Randell, "Accessible Ultrasonic Positioning," *IEEE Pervasive Computing*, vol. 5, no. 4, pp. 86–93, 2006.

[7] M. Hazas and A. Hopper, "Broadband ultrasonic location systems for improved indoor positioning," *IEEE Transactions on Mobile Computing*, vol. 5, no. 5, pp. 536–547, May 2006.

[8] J. Gonzalez and C. Bleakley, "High-Precision Robust Broadband Ultrasonic Location and Orientation Estimation," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 3, no. 5, pp. 832–844, Oct. 2009.

[9] M. Hazas and A. Ward, "A novel broadband ultrasonic location system," in *Proc. of UbiComp*. Springer-Verlag, 2002, pp. 264–280.

[10] J. Prieto, A. Jiménez, J. Guevara, J. Ealo, F. Seco, J. Roa, and F. Ramos, "Performance evaluation of 3D-LOCUS advanced acoustic LPS," *Instrumentation and Measurement, IEEE Transactions on*, vol. 58, no. 8, pp. 2385 –2395, 2009.

[11] J. Ureña, A. Hernández, A. Jiménez, J. M. Villadangos, M. Mazo, J. C. García, J. J. García, F. J. Álvarez, C. de Marziani, M. C. Pérez, J. A. Jiménez, A. R. Jiménez, and F. Seco, "Advanced sensorial system for an acoustic LPS," *Microprocessors and Microsystems*, vol. 31, no. 6, pp. 393 – 401, 2007.

[12] F. J. Álvarez, J. Ureña, M. Mazo, A. Hernández, J. J. García, and C. de Marziani, "High reliability outdoor sonar prototype based on efficient signal coding," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 53, no. 10, pp. 1862 –1872, Oct. 2006.

[13] M. Alloulah, M. Hazas, and M. Stojanovic, "Airborne Broadband Ultrasound: Wireless Channel and Motion Tracking Algorithms," Tech. Rep., 2012, submitted to JASA.

[14] J. K. Holmes, *Spread Spectrum Systems for GNSS and Wireless Communications*. Norwood, MA, USA: Artech House, Inc., 2007.

[15] M. Stojanovic and L. Freitag, "Multichannel Detection for Wideband Underwater Acoustic CDMA Communications," *IEEE Journal of Oceanic Engineering*, vol. 31, no. 3, pp. 685–695, July 2006.

[16] T. Austin, D. Blaauw, S. Mahlke, T. Mudge, C. Chakrabarti, and W. Wolf, "Mobile supercomputers," *Computer*, vol. 37, no. 5, pp. 81–83, May 2004.

[17] Arvind, R. Nikhil, D. Rosenband, and N. Dave, "High-level synthesis: an essential ingredient for designing complex ASICs," in *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on*, nov. 2004, pp. 775–782.

[18] J. Cong, G. Reinman, A. Bui, and V. Sarkar, "Customizable domain-specific computing," *Design Test of Computers, IEEE*, vol. 28, no. 2, pp. 6–15, march-april 2011.

[19] Y. Guo, D. McCain, J. R. Cavallaro, and A. Takach, "Rapid industrial prototyping and SoC design of 3G/4G wireless systems using an HLS methodology," *EURASIP J. Embedded Syst.*, vol. 2006, pp. 18–18, January 2006. [Online]. Available: http://dx.doi.org/10.1155/ES/2006/14952

[20] I. E. Sutherland, "A head-mounted three dimensional display," in *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, ser. AFIPS '68 (Fall, part I). New York, NY, USA: ACM, 1968, pp. 757–764. [Online]. Available: http://doi.acm.org/10.1145/1476589.1476686

[21] M. Hazas, "Indoor Location Systems," PhD in Location Sensing, Laboratory for Communication Engineering – University of Cambridge, Digital Technology Group, William Gates Building, 15 JJ Thomson Avenue, Cambridge, CB3 0FD, 2002.

[22] M. Alloulah, "Real-Time Tracking for Airborne Broadband Ultrasound," PhD Dissertation, School of Computing and Communications, Lancaster University, LA1 4WA, 2011.

[23] reacTIVision 1.4, http://reactivision.sourceforge.net/.

[24] A. Papakonstantinou, Y. Liang, J. A. Stratton, K. Gururaj, D. Chen, W.-M. W. Hwu, and J. Cong, "Multilevel granularity parallelism synthesis on FPGAs," in *Field-Programmable Custom Computing Machines (FCCM), 2011 IEEE 19th Annual International Symposium on*, may 2011, pp. 178–185.

[25] B. Buyukkurt, J. Cortes, J. Villarreal, and W. A. Najjar, "Impact of high-level transformations within the ROCCC framework," *ACM Trans. Archit. Code Optim.*, vol. 7, pp. 17:1–17:36, December 2010. [Online]. Available: http://doi.acm.org/10.1145/1880043.1880044

[26] M. Allie and R. Lyons, "High-Speed Square Root Algorithms," in *Streamlining Digital Signal Processing*, R. Lyons, Ed. John Wiley & Sons, Inc., 2007, ch. 16, pp. 165–172.

[27] J. Cong and Y. Zou, "FPGA-Based Hardware Acceleration of Lithographic Aerial Image Simulation," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 2, no. 17, pp. 17:1–17:29, Sept. 2009.

[28] M. Fingeroff, *High-Level Synthesis Blue Book*. Xlibris Corporation, 21 May 2010.

[29] H. Ziegler and M. Hall, "Evaluating heuristics in automatically mapping multi-loop applications to FPGAs," in *Proceedings of the 2005 ACM/SIGDA 13th international symposium on Field-programmable gate arrays*, ser. FPGA '05. New York, NY, USA: ACM, 2005, pp. 184–195. [Online]. Available: http://doi.acm.org/10.1145/1046192.1046216

[30] B. So, P. Diniz, and M. Hall, "Using estimates from behavioral synthesis tools in compiler-directed design space exploration," in *Design Automation Conference, 2003. Proceedings*, june 2003, pp. 514–519.

[31] C. Bouganis and G. Constantinides, "Synthesis of DSP Algorithms from Infinite Precision Specifications," in *High-Level Synthesis From Algorithm to Digital Circuit*, P. Coussy and A. M. (Eds.), Eds. Springer-Verlag, 2008, ch. 11.

[32] H. Kreide and D. Lambert, http://history.nasa.gov/computers/Ch4-2.html, h. Kreide and D.W. Lambert, "Computation: Aerospace Computers in Aircraft, Missiles and Spacecraft," Space/Aeronaut., 42, 78 (1964); see also N.H. Herman and U.S. Lingon, "Mariner 4 Timing and Sequencing," Astronaut. Aeronaut., 43 (October 1965).