Synthetic Aperture Radar Image Formation and Processing on an MPSoC

Stefan Wiehle, Srikanth Mandapati, Dominik Günzel, Helko Breit, and Ulrich Balss

***Abstract*— Satellite remote sensing acquisitions are usually processed after downlink to a ground station. The satellite travel time to the ground station adds to the total latency, increasing the time until a user can obtain the processing results. Performing the processing and information extraction onboard of the satellite can significantly reduce this time. In this study, synthetic aperture radar (SAR) image formation as well as ship detection and extreme weather detection were implemented in a multiprocessor system on a chip (MPSoC). Processing steps with high computational complexity were ported to run on the programmable logic (PL), achieving significant speed-up by implementing a high degree of parallelization and pipelining as well as efficient memory accesses. Steps with lower complexity run on the processing system (PS), allowing for higher flexibility and reducing the need for resources in the PL. The achieved processing times for an area covering 375 km^2 were approximately 4 s for image formation, 16 s for ship detection, and 31 s for extreme weather detection. These developments combined with new downlink concepts for low-rate information data streams show that the provision of satellite remote sensing results to end users in less than 5 min after acquisition is possible using an adequately equipped satellite.**

***Index Terms*— Constant false alarm rate (CFAR), field-programmable gate array (FPGA), multiprocessor system on a chip (MPSoC), onboard processing, sea state detection, ship detection, synthetic aperture radar (SAR).**

I. Introduction

THE number of Earth observation (EO) satellites is continuously increasing. New devices are launched containing either optical or synthetic aperture radar (SAR) sensors. Whereas formerly most missions consisted of only one satellite, multisatellite missions are becoming more widespread. Having the same orbit and imaging parameters, such missions reduce the effective revisit time and allow products to be acquired more frequently. This high acquisition frequency increases the usefulness of remote sensing data for time-critical observations, for example, to support ships at sea, warning vessels of upcoming storms, or assisting in maritime situation awareness.

Manuscript received November 19, 2021; revised February 25, 2022; accepted April 6, 2022. Date of publication April 18, 2022; date of current version May 2, 2022. This work was supported by the European Union's Horizon 2020 Research and Innovation Program under Agreement 776311. *(Corresponding author: Stefan Wiehle.)*

Stefan Wiehle and Dominik Günzel are with the German Aerospace Center (DLR), Maritime Safety and Security Lab, 28359 Bremen, Germany (e-mail: stefan.wiehle@dlr.de; dominik.guenzelg@dlr.de).

Srikanth Mandapati, Helko Breit, and Ulrich Balss are with the German Aerospace Center (DLR), Remote Sensing Technology Institute, SAR Signal Processing, 82234 Weßling, Germany (e-mail: srikantha.mandapati@dlr.de; helko.breit@dlr.de; ulrich.balss@dlr.de).

Digital Object Identifier 10.1109/TGRS.2022.3167724

However, these situations require quick delivery of the relevant information to the user. Depending on the location, a satellite might have to travel for several minutes to reach the next suitable ground station for data downlink, and then the data have to be processed and made available to the client. The EO-ALERT project [1] aims to significantly reduce this time by processing the data onboard of a satellite, then transmitting the results in the form of alerts to the user via satellite-to-satellite communications or direct downlink. The project goal is to make EO products available to the end user in less than 5 min after acquisition [enhanced near real time (NRT)].

For that purpose, a system of a total of seven multiprocessor system-on-a-chip (MPSoC) boards is envisaged. These boards cover all necessary tasks, such as communication, data encryption, and compression, as well as processing of optical and SAR acquisitions. Instead of building the outlined flight avionics, a test bench consisting of four commercial off-the-shelf (COTS) MPSoC boards serves as a demonstrator and is close to a space-ready setup regarding dimensions and energy consumption.

This article focuses on the application of a single MPSoC board for the purpose of SAR image formation and processing. The developments were conducted under the latency requirement of 210 s, which was defined in the EO-ALERT project for image formation and alert generation. The challenge of implementing SAR processing in satellite-suitable hardware has rarely been attempted thus far. An onboard SAR image formation prototype based on optical processing has been shown in [2]. A GPU-based approach was demonstrated in [3] and [4]. A pure field-programmable gate array (FPGA) or a combination of FPGA with application-specific integrated circuit (ASIC)/CPU-based implementations were also attempted for onboard image formation [5]–[11]. However, airborne SAR and spaceborne SAR platforms have different acquisition geometries, and therefore, they have different special needs. Due to the lower flight height, pulse rate and swath width and thus the data rate of airborne SAR are often lower. The curvature of the range history is stronger, and the flight path of the airplane is less stable than a satellite orbit. Thus, a processor for airborne SAR has to process less raw data by a more complex algorithm. Therefore, it is hard to compare the performance and latency times of FPGA implementations of image formation for airborne and spaceborne SAR. Joshi and Baumgartner [12] implemented an SAR ship detector using only range compression for use with DLR's airborne F-SAR and DBFSAR systems.

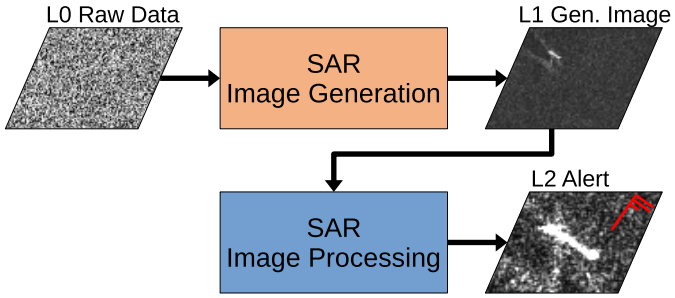

Fig. 1. High-level overview of the integrated SAR image formation and processing chain. TerraSAR-X data © DLR 2017.

For this article, adapted algorithms for SAR image formation first generate a level 1 (L1) image from level 0 (L0) raw data. Then, level 2 (L2) processing algorithms adapted from established software used at ground segments perform ship detection or wind and sea state detection. Fig. 1 provides a high-level overview of the processing chain from SAR raw data to generated products.

Preliminary results of this work were previously presented [13]–[15]. This article builds on these previous publications by presenting final results, more detailed technical descriptions, and an extensive analysis of the results. In this context, the performance of our image formation approach is compared with approaches presented by other teams in literature.

II. Hardware Overview

SAR image formation and processing were implemented on a COTS prototyping board socketed with a Xilinx Zynq UltraScale+ ZU19EG MPSoC. At the time of inception of the project, radiation-tolerant or even-hardened MPSoCs with sufficient hardware resources for implementation of the SAR and optical processing algorithms were not yet available, but this was expected to change within the near future. Because the EO-ALERT project did not include the task of building an actual payload, it was therefore decided to show the feasibility of onboard processing using COTS prototyping hardware and make transferability to different platforms a requirement for hardware and software development. Indeed, multiple options for low-Earth-orbit space-qualified MPSoCs with comparable performance to the used prototyping hardware have been released in the meantime, such as ThalesAlenia multiMIND [16] or KP Labs Leopard [17].

As shown in the schematic in Fig. 2, the Zynq UltraScale+ MPSoC combines an FPGA, the so-called programmable logic (PL), with a quad-core ARM Cortex-A53 processing system (PS), each equipped with 4 GB of DDR4 dynamic random access memory (DRAM). The PS and PL are connected via various interconnects following the Advanced eXtensible Interface (AXI) specification from the ARM Advanced Microcontroller Bus Architecture 4 (AMBA) standard. The PS runs a purpose-built embedded Linux operating system (OS) created with Xilinx PetaLinux Tools, which hosts the necessary software applications for the processing chain, and is also responsible for the task of communicating with the up-stream scheduling board in the project via Ethernet and PCI Express links. Finally, a micro-SD card is attached to

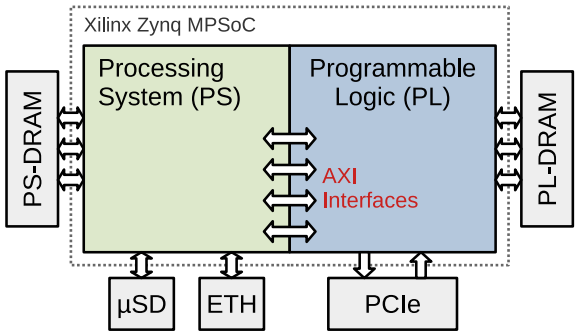

Fig. 2. Components and interfaces of the EO-ALERT prototyping board with Xilinx Zynq UltraScale+ ZU19EG MPSoC.

the PS as nonvolatile storage for static ancillary data or for debugging purposes.

While the PS with its common architecture and Linux OS facilitates rapid implementation of algorithms in software, its speed is limited due to various constraints onboard of a satellite, such as power consumption or heat dissipation. The PL may be utilized to accelerate computationally intensive parts of SAR L1 and L2 processing algorithms by implementing them in dedicated hardware circuits. Such application-specific hardware processors can exploit a high degree of parallelism, for example, with heavily pipelined datapaths, to significantly accelerate performance. However, the achievable speed-up has to be carefully weighed against the more complex development of a dedicated hardware processor compared to a pure software approach. Furthermore, the extent of designs implemented in hardware is constrained by the available FPGA resources.

A crucial task at the beginning of the design phase was the hardware/software partitioning of the algorithms. An algorithm with low computational complexity but complicated decision trees is more suited to run on the PS without need of using specialized hardware. In contrast, a parallelizable algorithm with large computational complexity benefits from hardware acceleration. Table I summarizes the processing steps and the partitioning for SAR image formation, ship detection, and wind and sea state detection. Except for the parameter calculations and geo-reference grid computation, all steps of L1 image formation related to signal processing of the SAR data are implemented on the PL. In contrast, for L2, only the initial prescreening step for ship detection was selected for hardware implementation; all other steps were implemented in the PS.

Sections III and IV will explain the steps of SAR L1 and L2 processing, provide further details on how the shown partitioning was reached, and discuss extensively the hardware and software implementations.

III. Image Formation

SAR image formation reconstructs the complex reflectivity of a scene on ground from the sensor raw data acquired along the flight path. A common approach is the adoption of the matched filter concept, where the raw data are convolved with the complex conjugate and time-inverted range-variant point scatter response [18]. Most SAR focusing algorithms efficiently perform this operation in the spectral domain. Well-known and widely used spectral-domain SAR focusing algorithms are the range-Doppler algorithm (RDA), the chirp

scaling algorithm (CSA), and the omega-K algorithm (ωKA). All three algorithms have been subject to hardware-level implementation in the past. The adoption of RDA is reported in [3] and [19]–[24], CSA in [4], [9], [19], [20], and [22], and ωKA in [5], [23], and [25].

In the context of the EO-ALERT project, the StripMap (SM) imaging mode is used, since it provides a good compromise between coverage and resolution for emergency scenarios. With regard to the application in view, there are only moderate requirements on resolution (about 6–8 m) and localization accuracy (better than 10 m) of the focused SAR image. Since no new SAR instrument is designed or operated in the context of the project, TerraSAR-X single polarization SM acquisition data have been selected as a representative source of L0 raw data. The TerraSAR-X SM mode is characterized by a swath width of 30 km and an inherent resolution of about 3 m. The raw data are supplemented by the corresponding radar instrument settings as well as attitude and orbit data derived from the output of the satellite's attitude and orbit control system (AOCS).

After consideration of the limited onboard hardware resources, the desired resolution class, and the applications defined in the project, the monochromatic ωKA was selected for focusing the SAR raw data on the MPSoC. As lined out in [26] and [27], the monochromatic variant of the ωKA omits the cumbersome Stolt interpolation which eases implementation. The inherent, less accurate correction of range cell migration for wide swathes is negligible for the considerable narrow swath and aperture length of the TerraSAR-X SM mode. The ωKA and its monochromatic approximation allow the incorporation of pulse replica-based range compression without the need for additional fast Fourier transforms (FFTs), whereas the CSA being used in the TerraSAR-X ground segment requires two additional FFT for an initial replica precompression step. Due to limited main memory resources and an FFT intellectual property (IP) core 2^n length constraint, the block size for SM processing is designed to contain exactly 8192 azimuth lines and up to 32 768 range pixels represented as 8-bit/8-bit complex integer values. This results in a spatial coverage per block between 375 and 500 km^2.

The image generated on the FPGA is a multilook, slant-range-detected (MSD) σ_0-calibrated image with a resolution of approximately 6–8 m. It is accompanied with a geo-reference grid, which maps the radar time coordinates of the image to geographical coordinates. These data serve as input to subsequent image processing. For debugging and test purposes, image formation may be reconfigured to generate a single-look slant-range complex (SSC) image.

A. Hardware and Software Partitioning

The arithmetic operations within the processing steps of SAR image formation can be separated into two categories as shown in Table I: computation of focusing parameters and geolocation reference grid on the PS, and SAR signal processing of sensor data on the PL. With regard to computational effort, signal processing is by far the dominant part in the entire SAR chain. Due to moderate requirements with respect to onboard image accuracy in the project, FFTs and pixelwise filter operations can be efficiently performed using integer arithmetics. Based on the FFT accuracy analysis reported in [9], we selected a width of 16 bit for data and filter pixel representation. But in contrast to [9], we took advantage of the block floating-point option provided by the Xilinx FFT IP core. In contrast to signal processing, computation of focusing parameters and the geolocation reference grid requires less magnitudes but more complicated operations. In particular, floating-point operations are needed, so that these steps are performed in software on the PS. The hardware and software partitioning of SAR image formation with the monochromatic ωKA is depicted in Fig. 3.

TABLE I
PARTITIONING OF IMAGE FORMATION AND IMAGE PROCESSING STEPS BETWEEN PL AND PS

	Step	Partitioning
Image formation (monochromatic ω-k)	Parameter calculation	PS
	Geo-reference grid	PS
	I/Q correction	PL
	Fast Fourier transforms	PL
	Focusing filter multiplications	PL
	Detection	PL
	Multi-looking	PL
	Cache and corner turning	PL
Ship detection (based on CFAR)	CFAR pre-screening	PL
	Detection refinement	PS
	Azimuth ghost removal	PS
	Land object removal	PS
	Ship parameter extraction	PS
Wind and sea state detection (based on spectral and GLCM analysis)	Land removal	PS
	Ocean backscatter detection	PS
	Wind detection	PS
	Wave height detection	PS

Three hardware design constraints were investigated w.r.t. FFT length, cache sizes, and input-output (I/O) throughput of DRAM for achieving low latency. The designed maximum raw data block size directly determines the transform length of the FFT. The target hardware could support a 16 k $\times$ 32 k raw data block (azimuth $\times$ range), but in that case, integration of SAR image formation and subsequent image processing on the same FPGA would not be possible, as the FPGA memory resources would be almost entirely utilized by SAR image formation. For this reason, the block size was chosen to be 8 k $\times$ 32 k.

A major parameter with impact on latency is the number of raw data lines that are cached simultaneously in the FPGA from DRAM. Caching is a key concept for continuous streaming of data into FFTs and for efficient corner turning, when data coming out of the FFTs are streamed back into DRAM. In this context, the term "corner turning" refers to transposition of 2-D SAR data from range direction to azimuth direction or vice versa. This is needed due to alternating application of range and azimuth FFTs in the signal processing chain. Having caches before and after the FFTs is less complex with regard to managing the data, but fewer lines can be stored compared with a design with one combined cache, which, depending on the current processing step, serves as either input or output cache. While a combined cache could store more lines of

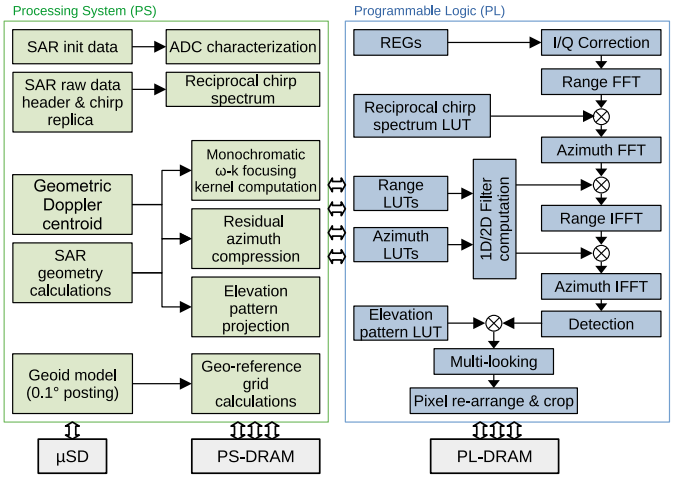

Fig. 3. Hardware/software partitioning of the monochromatic ω-k algorithm targeted for Xilinx ZU19EG MPSoC.

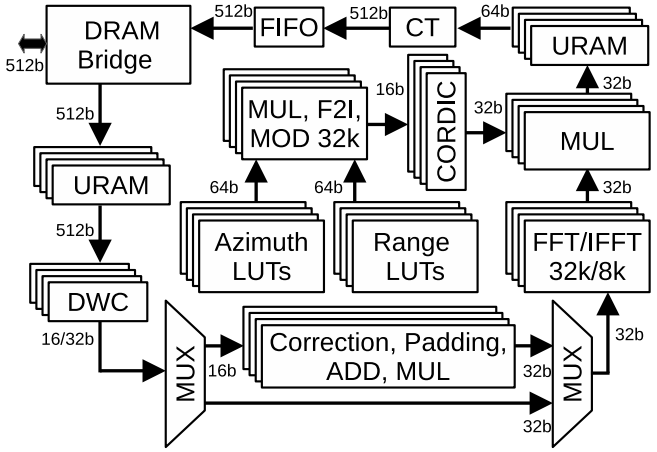

Fig. 4. Datapath of the SAR image formation module implemented in the FPGA.

data, it would require a more complicated design approach for controlling the FFT input and output and it would need more resources for handling the data. Again, due to limited resources, integration of SAR image formation and processing on the same device did not permit the implementation of such a combined cache approach.

Data I/O throughput between the hardware core and DRAM significantly impacts processing latency, because the SAR signal processing demands large amounts of data to be transferred repeatedly. For the given raw-data bit width and block size, approximately 7 GiB of data are transferred between the caches and PL-side DRAM. As the data transfer rate is governed by the data width and clock frequency of the AXI interface connecting the fabric and PL-side DRAM, a width of 512 bit was configured. It is therefore advantageous to store the SAR data in the PL-side rather than PS-side DRAM, because the interface between the PS and PL supports a maximum data width of 128 bit. In addition, the PS-side DRAM is kept free for the the PetaLinux OS and software applications.

B. Hardware Implementation

A high level of parallelization and pipelining has been achieved by implementing all the signal processing modules in hardware.

1) Signal Processing Chain: The SAR signal processing chain consists of five major blocks, which are raw data in-phase and quadrature-phase (I/Q) correction, FFT, pixelwise complex multiplications of focusing filter kernels, detection, and, finally, multilooking (cf. Fig. 3).

In the initial I/Q correction step, the digitized SAR echo data are corrected for artifacts that are present due to inevitable manufacturing tolerances in the analog-to-digital-converters (ADCs). Signal processing contains four FFT steps and begins with a range FFT, followed by complex multiplication with the reciprocal spectrum of the transmitted chirp pulse. Thereafter, an azimuth FFT transforms the SAR data from range spectral, azimuth time domain into the full 2-D spectrum, and then, the monochromatic ω-k kernel is multiplied. Next, the data are transformed into the range-Doppler domain by a range backward FFT followed by multiplication with the so-called residual azimuth compression filter. Then, azimuth backward FFT is performed to transform the focused SAR data back into 2-D time domain. Detection means the retrieval of the magnitude of the complex-valued focused SAR image and thus discarding the phase information, which is not needed for subsequent image processing, either ship detection or extreme weather detection. Multilooking is then applied on the detected data by applying a 2×2 box car filter over range and azimuth, which improves radiometric resolution at the cost of degrading the spatial resolution. Both the detection and multilooking steps reduce the data size of the focused image by a factor of 8. The detection and multilooking steps can optionally be skipped if desired by setting a register in the PL in order to obtain a single-look-complex (SLC) image after the azimuth backward FFT.

2) Datapath: The building blocks and data flow of the datapath for signal processing are shown in Fig. 4. The resources of the FPGA allow for instantiation of a maximum of four parallel datapaths, limited by the computations involving FFTs, which require a large number of block RAMs (BRAMs) for storing the results. Several components of the datapath have been realized using IP cores from Xilinx, such as FFT, RAM, coordinate rotation digital computer (CORDIC), data width converters (DWCs), and floating-point multiplication. The main advantages of using the Xilinx FFT IP core are its reconfigurability at runtime for forward and backward FFTs and the transform length of up to 65 536 points. The FFT IP core has a pipelined architecture with AXI-Stream I/O interfaces and was configured as block floating point with adaptive scaling to avoid the clipping or wrapping of integer values. The size of available UltraRAM (URAM) in the FPGA allows for 16×32 k range lines or 64×8 k azimuth lines to be cached at a time. Because of this limit, first, subblocks of SAR data are buffered from PL-DRAM to a group of cascaded URAMs forming the input cache. URAM is a dual-port on-chip memory with fixed size of $4\text{ k} \times 72$ bit, and has a single synchronous clock for both read and write operations. Next, the buffered data go through bit-width conversion and get streamed to the FFTs. The output of the FFTs is then multiplied with filter coefficients, and finally, the results are

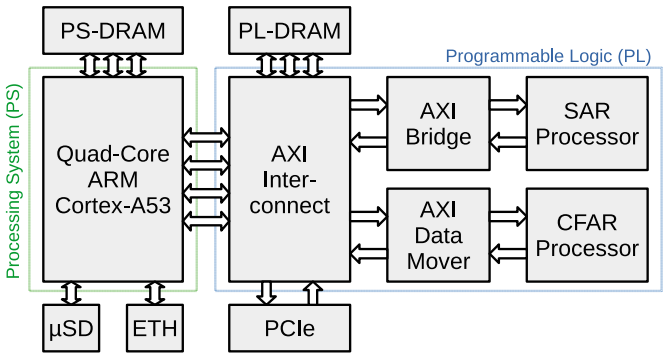

Fig. 5. Integration of the dedicated hardware accelerators for L1 image formation and L2 processing on Xilinx Zynq MPSoC. Only components and interfaces that are used by the SAR processor are shown.

written to a second group of URAMs forming the output cache. Caching is done at the beginning and end of the datapath, because it is an efficient approach to streaming the data continuously into the FFTs. Corner turning (CT) solves the issue of the data being unaligned after the previous signal processing step. Each datapath instance processes 4×32 k range lines or 16×8 k azimuth lines stored in the URAM cache in one go. In total, all four datapath instances process $2 \times 8192 \times 32$ k ranges lines and $2 \times 32768 \times 8$ k azimuth lines. Running four parallel datapaths significantly decreases the overall latency. The blocks in the datapath are controlled by a finite state machine (FSM), which is also responsible for generating the read and write addresses for the PL-DRAM.

C. Hardware Integration

The integration of the developed SAR image formation processor into the MPSoc is shown in Fig. 5. The processor has an AXI interface for data transfers to and from the PS as well as the PL-side DRAM. SAR raw data, AOCS data, and initialization data are transferred to the PL-DRAM via PCIe from the scheduling board. From there, the chirp replica, which is part of the raw data, the AOCS data, and the initialization data, are transferred to the PS-DRAM for parameter calculations. In order to store and process SAR raw data, a minimum of 2.5 GB of memory is allotted in the PL-DRAM for image formation. The purpose-built AXI DRAM bridge IP, which is part of the datapath shown in Fig. 4, connects to the PL-DRAM to transfer data between the memory-mapped devices. This DRAM bridge IP can be connected to the PS-side or PL-side DRAM via AXI and is controlled by an FSM for generating read/write addresses and data transfer sizes. All registers in the hardware core are written from the PS through an AXI interface, and lookup tables (LUTs) are loaded from the PS as well as with an AXI BRAM controller. The PS is also responsible for starting the hardware core once the calculation of processing parameters has finished, and they have been transferred to the PL. The hardware core signals the status of completion with an AXI general purpose IO (GPIO) interrupt connected to the PS.

D. Software Implementation

The software part of SAR image formation comprises all geometry-related calculations based on the acquisition-specific orbit state vectors and attitude quaternions. A geoid model, globally sampled with a posting of 10 km $\times$ 10 km and stored on the internal SD card as a 9-MB file, provides heights of the sea surface with reference to the WGS84 Earth ellipsoid. On the basis of this, the following geometry-related parameters are calculated:

1) effective sensor velocities as a function of range;
2) Doppler centroid frequencies derived from the antenna pointing geometry;
3) elevation angles and projection of the antenna-pattern elevation gain profile into the slant-range geometry of the SAR image;
4) incidence angles for σ_0 calibration;
5) the SAR image heading angle;
6) a 1-km laterally spaced geo-reference grid providing latitude, longitude, and ellipsoidal height for each grid point, as well as incidence angles of the radar beam hitting the surface.

Prior to triggering signal processing in the PL, the software running in the PS computes and fills a set of 1-D LUTs residing in BRAMs of the PL. The lengths of the individual LUTs equal either the range or the azimuth size of the raw data block, in the current implementation of 32 768 and 8192, respectively. The concept of PS-side precomputed values in PL-side LUTs, which are addressed and read within the PL in synchronization with the SAR data samples streaming through the datapath, substantially reduces the complexity of the hardware circuitry.

1) replica of the transmitted chirp pulse, represented by the complex reciprocal of its spectrum;
2) Hamming window to be applied to the range spectrum;
3) range frequency-dependent terms of the phase function of the matched filter applied in the 2-D spectral domain;
4) azimuth frequency-dependent terms of this phase function;
5) range-dependent terms of the phase function of the matched filter applied in the range-Doppler domain;
6) window function consisting of the reciprocal azimuth antenna pattern of the sensor and of a Hamming window to be applied to the azimuth spectrum;
7) window function consisting of the reciprocal range antenna pattern of the sensor to be applied on the focused image.

With the exception of the complex chirp replica spectrum, all of these lookup tables contain real floating-point values. The terms required to compute the 2-D spectral phase filter are separated into two 1-D LUTs. This has the advantage that the final filter phase parameter computation adds no latency to the datapath, because it is synchronous to the FFT IP core data output.

IV. Image Processing

Once the image has been generated in the image formation phase, either ship detection or wind and sea state detection is performed. In the current operational environment on a ground station, both can be performed on the same scene consecutively, along with other detection algorithms. For the

project, however, it is assumed that only one of the tasks is relevant for the ordering client, and consequently, only one is performed for each acquisition. Both algorithms used are adapted from those currently running automatically in the NRT processing chain at DLR ground station Neustrelitz. Since these algorithms are already well described in previous publications, this section focuses on the implementation of the algorithms on the MPSoC.

A. Ship Detection

The ship detection algorithm is based on [28] and [29], where a more detailed explanation is provided. It starts with an initial prescreening, applying a constant false alarm rate (CFAR) detector on the entire image; Fig. 6 shows a simplified overview of this method. The second step reapplies CFAR only to the initially detected pixels to refine the detection and retrieve ships that might have been missed [30]. This is followed by azimuth ghost filtering, which removes aliasing echoes on the sea surface caused by ships or land objects. As these land objects might be missed when objects are filtered using a land mask, the land object removal step is performed after the ghost removal. Finally, the parameters such as length and width of the retrieved ships are extracted.

1) Hardware and Software Partitioning: As shown in Table I, the initial prescreening over the entire scene is the only step implemented in the PL. Initial tests showed that the processing time on the PS would reach about 16 min; hence, this was not possible under the project latency requirement of 210 s for the full SAR processing. The remaining four processing steps are sufficiently fast even on the PS. Although the detection refinement uses the same CFAR algorithm as the initial detection, it is only applied to the comparatively few detected pixels.

2) Hardware Implementation: The pipelined CFAR processor presented in [31] served as the basis for hardware implementation, but it was rewritten in order to allow for flexible dimensions of the CFAR background and guard windows and to enable significantly higher throughput than previously achieved. Fig. 7 shows a schematic of the datapath of the revised CFAR core, which consists of a BRAM pixel buffer connected to a 29-stage-long processing pipeline. After image formation, the L1 products reside in the PL-side DRAM, and the CFAR processor prefetches the required pixels into the buffer. It stores up to 1024 columns of the CFAR window, each containing up to 255 16-bit pixels. Therefore, with a typical background window size of 750 m, the minimum pixel size is 2.94 m, sufficient for the products generated onboard. Since the width and length of the buffer are configurable at compile time, it can be adapted to higher resolution products at the cost of increased BRAM utilization. In the previous version of the core [31], the CFAR dimensions were fixed in the pipelined datapath, so that the buffer would have to hold exactly 255 pixels in parallel irrespective of the actual size of the pixels. This may negatively impact the accuracy of detection and excluded potential targets within $(255-1)/2 = 127$ pixels of the image edge from detection altogether. The datapath was therefore improved by first adding a multiplexer for each 16-bit pixel after the buffer.

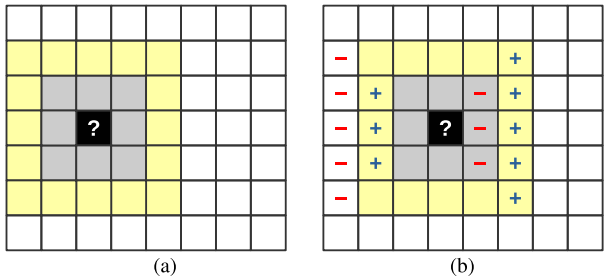

Fig. 6. Simplified illustration of the sliding CFAR window for calculation of background statistics. Background window pixels are marked in yellow and guard window pixels in gray. Only changed pixels are updated before mean calculation, which reduces the processing complexity from $O(n^2)$ to $O(n)$ with respect to the sliding window size. The real background window size is 750 m or about 200 pixels in the onboard implementation. (a) Initial CFAR window. (b) Update for next pixel.

Each multiplexer is controlled individually and set to output "0" according to the current dimensions of the CFAR background and guard. Additionally, the now variable dimensions are propagated through the pipeline stages, as they are required to calculate the mean of the background pixel intensities.

The BRAM buffer allows accessing one column of the CFAR window per clock cycle. As previously explained, four operations have been defined for the pipeline, which are addition and subtraction for a background and a guard column, respectively (cf. Fig. 6).

A fifth operation has been added, which is loading the target pixel itself from the buffer. This is required for the final step of the pipeline, where the intensity of the target pixel is compared against the calculated CFAR threshold. In a typical queue following the first in, first out (FIFO) principle, elements are discarded when read, but for the sliding CFAR window, all columns with the exception of the first one will still be needed for subsequent windows. Thus, the pixel buffer has been implemented as an indexed queue, so that an optional index may be provided to access a certain element without deleting it. After full accumulation of the first CFAR window in a row, the pipeline can therefore perform detection on one target pixel every five clock cycles regardless of the size of the CFAR dimensions, which is a major advantage over software processing. This being the case, the complexity of the CFAR algorithm is $O(1)$ with the hardware core. The results in Section V will show that this theoretical maximum was not reached due to throughput limitations from the PL-side DDR4 DRAM, but still significant speed-up was achieved.

Following initial multiplexing of the incoming pixels, the pipeline splits into two independent branches (see Fig. 7): the top branch calculates the squared mean of the pixel intensities, while the bottom branch calculates the mean of the squared pixel intensities. In the top branch, the pixels are directly fed into a pipelined adder tree for accumulation of the intensities. A pipelined adder tree was necessary, because addition of all 255 16-bit pixels in one clock cycle led to timing violations during physical implementation on the FPGA. The result is then added to or subtracted from the stored result as previously explained. To obtain the mean μ_b, the sum is divided by a pipelined divider and finally squared. The bottom branch works similarly, but pixels are squared first before being

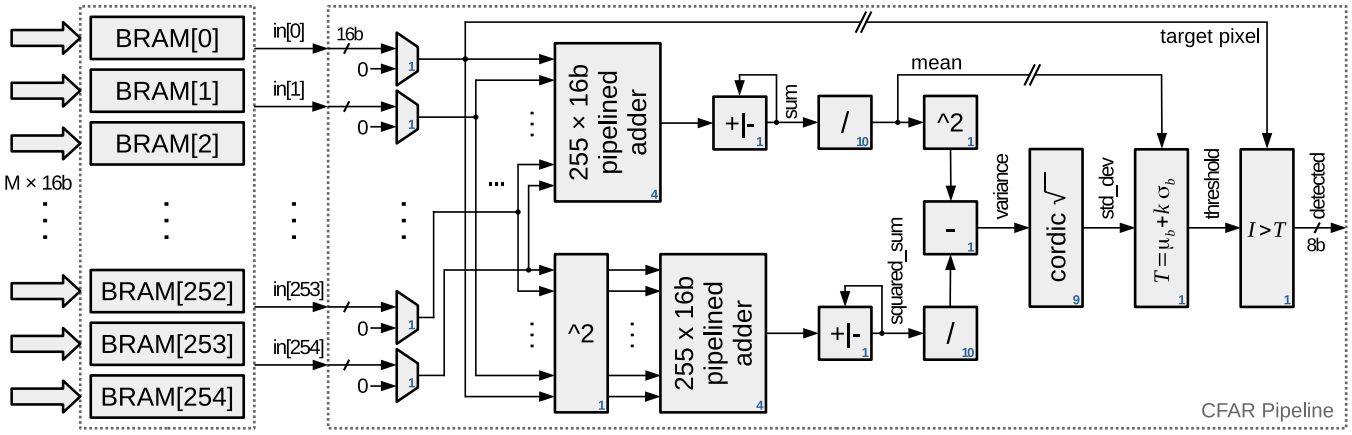

Fig. 7. Datapath of the hardware CFAR processor for ship detection. Whole columns of 16-bit image pixels are prefetched into a BRAM buffer (left) and processed in a 29-stage-long pipeline (right). The blue numbers annotate the pipeline delay in clock cycles of the respective functional blocks.

accumulated. Both branches consist of the same amount of pipeline stages, so that they can be joined again after 16 clock cycles. This is done by subtracting the squared mean (top) from the mean of squared pixels (bottom), which yields the variance of the CFAR background pixel intensities. A Xilinx CORDIC IP core is then used to obtain the square root of the variance, i.e., the standard deviation σ_b. Using the mean μ_b, standard deviation σ_b, and CFAR constant k, in the next stage, the CFAR threshold T is calculated with

$$T = \mu_\mathrm{b} + k \cdot \sigma_\mathrm{b}. \tag{1}$$

Finally, the threshold is compared against the intensity I of the target pixel. If the target exceeds the threshold, it is deemed that an object has been detected.

The output of the pipelined datapath is an 8-bit signal indicating whether the current pixel was detected or not. This is stored in an output FIFO queue, which is intermittently written to the PL-side DRAM in chunks of 64 bytes. Should the output queue fill up, the pipeline is stalled in order to prevent data loss. Likewise, the pipeline is paused if the input pixel buffer does not hold enough columns of the CFAR window. The processor is therefore robust to variations in DRAM throughput.

3) Hardware Integration: The CFAR processor was integrated in the PL of the MPSoC as shown in Fig. 5. A Xilinx AXI DataMover IP core is connected to the AXI Interconnect described in Section III-C for prefetching of the image data from the PL-side DRAM. With this IP core, data can be transferred between AXI memory-mapped and AXI-Stream domains with high throughput, which makes it suitable for feeding data into the input queue of the CFAR processor and to write the results from the output queue back into DRAM. The DataMover is controlled by an FSM in the CFAR processor itself. One full column of the CFAR window is read from DRAM per transfer. The highest possible DataMover interface width, i.e., 512 bit, was chosen to maximize throughput, so that up to 32 pixels (64 B) can be transferred each clock cycle. The CFAR processor therefore has to collect the incoming packets depending on the size of the CFAR dimensions prior to writing into the BRAM pixel buffer, which accepts a whole column at a time.

For control and status, an AXI GPIO has been integrated into the CFAR processor and connected to the PS of the MPSoC. This enables the PS to start or abort the CFAR detector by writing the respective registers, and an interrupt is raised when the current image has been processed by the hardware core. Furthermore, several AXI registers can be accessed by the PS in order to configure the core with regard to memory addresses and metadata, such as image size, CFAR window dimensions, or the CFAR constant k.

4) Software Implementation: The SAR image processing software on the ground station was written for Linux OSs and is intended to run on x86 processors traditionally found in servers, but the MPSoC used in the EO-ALERT project is equipped with an ARMv8 PS. Cross-compilation required removal of certain libraries which were not available on the target system, but were used mainly for visualization and debugging purposes. Once it was running on the ARM PS, the software was extended to support reading the project-specific binary image and auxiliary data file formats. Transfer of data between the PS-DRAM, PL-DRAM, and PCIe was enabled by including a library developed specifically for the onboard system configuration in the project. With that, the software could be adapted to reading the L1B products from the PL-side DRAM, where they are stored after the image formation phase.

When the application binary is executed for ship detection, the L1B products are first transferred to the PS, and metadata is read from the image header. From that, the parameters for CFAR calculation are derived and stored together with the metadata in the AXI registers introduced in Section III-A3 using a vendor-provided userspace I/O (UIO) library. Then, the CFAR hardware processor is started, and the program waits for the hardware interrupt. After the software receives the interrupt, it checks whether the detection was successful and transfers the binary mask from the PL- to the PS-DRAM. Ship detection then continues on the PS with data being used only from the PS-DRAM. In case any ships have been detected, so-called alerts are created at the end of the chain and transferred to the PL-DRAM, where they will be fetched from by the upstream control and scheduling subsystem. These binary messages contain parameters of the detected ship, such as its position, timestamp, length, or width. Additionally, as requested during user consultation phase in the project, a

100×100 pixel 8-bit ship thumbnail is extracted from the scene and delivered along with the parameters.

Injecting the AXI data transfers and control of the hardware core directly into the SAR software allowed keeping the existing structure of the on-ground software and therefore reduced development time.

B. Wind and Sea State Detection

Wind and sea state detection are based on [32] and [33]–[35], respectively. Again, the reader is referred to these publications for a more thorough description of the algorithms.

For weather detection, the scene is divided into a processing grid; one value for wind speed and sea state is calculated per grid cell. Here, land removal is the first step; grid cells containing more than 10% land are not considered for processing. The mean ocean backscatter is calculated in the next step. It is used as input for the wind detection, which is performed by applying the XMOD2 geophysical model function (GMF) to retrieve wind speed. This wind speed is also a parameter for the following wave height detection, which obtains features derived from a gray-level cooccurrence matrix (GLCM) analysis as well as spectral information by FFT.

In current operational use, the processing is done on a 1024×1024 pixel FFT size, approximately $1280\,\text{m} \times 1280\,\text{m}$ in TerraSAR-X SM scenes, and a grid spacing (distance between grid cell centers) of 3 km [33]. As a compromise between resolution and computational time, a grid spacing of $2\,\text{km} \times 2\,\text{km}$ is chosen for the project. This results in approximately 75 grid cells for a $375\,\text{km}^2$ scene.

Although the FFT is a function well suited for being implemented in an FPGA, no parts of the wind and sea state retrieval were ported to the PL. Testing had shown that the additional development efforts were not necessary as the latency requirements could be met by only using the PS. Furthermore, some of the PL resources were already exhausted by the image formation and CFAR hardware implementations. A thorough performance tuning was done on the code instead in order to improve performance. This included optimizing the FFT function, removing the calculation of unnecessary debugging and output parameters, and optimizing core routines for use on a reduced instruction set computer (RISC)-type system, such as the ARM CPU in the PS. As an example for the latter, multiple loops iterating over every pixel of a grid cell were combined into one, so that fetching data from memory would occur less often. On a complex instruction set computer (CISC) system, such as the Intel CPU, the operational code is running on, this would have a lower impact on performance.

Similar to ship detection, the output of the processing is an alert, which is only created and supplied for pickup on the PL-DRAM if the threshold for high wind speed, 15 m/s in the project, or wave height (4 m) is reached. The alert contains parameters such as location (given as the center of the respective grid cell), wind speed, and wave height. Unlike ship alerts, no thumbnail is included here.

TABLE II
RESOURCE UTILIZATION OF THE SAR L1 AND L2 ONBOARD PROCESSING CIRCUITS INTEGRATED INTO THE ZYNQ ULTRASCALE+ ZU19EG FPGA AT TARGET FREQUENCY OF 125 MHZ

Site Type	L1	L2	Total	Available	Utilization
Slice Registers	71K	28K	99K	1045K	9.5 %
Slice LUTs	61K	41K	102K	523K	19.5 %
Block RAM Tiles	669	145	814	984	82.7 %
UltraRAM Tiles	128	0	128	128	100.0 %
DSP Slices	317	260	577	1968	29.3 %

V. RESULTS

The developed algorithms were applied to nine scenes taken in 2016 and 2017 in the Mediterranean Sea. All scenes were acquired in TerraSAR-X SM mode. The subsections of this section describe results with regard to hardware implementation, to image and product quality, and to processing latency.

A. Hardware Results

The developed hardware processors for SAR image formation and processing have been synthesized and implemented as a combined design on the Xilinx Zynq UltraScale+ ZU19EG MPSoC using Xilinx Vivado. A target clock frequency of 125 MHz was defined for the whole circuit to match the interface frequency of the Xilinx Ethernet and GPIO modules. While the clock frequencies of individual hardware blocks could be raised by implementing clock domain crossing between different frequency domains, there was little improvement in processing latency for image formation and none for image processing, because the CFAR core is limited by throughput of the DDR4 DRAM, which runs at its own separate frequency.

The created bitstream was imported into the PetaLinux toolchain to build the OS, which includes all necessary drivers and software applications. The resulting boot files, which include the boot loader, bitstream to program the FPGA, and OS image, were transferred to the SD card attached to the MPSoC, and the board was booted.

1) Utilization: Table II shows the FPGA resource utilization of the integrated circuit for SAR L1 image formation and L2 processing. In total, the hardware design requires only 9.5% of the available slice registers and 19.5% of the LUTs. However, 82.7% of the 984 BRAM tiles and 100.0% of the 128 URAM tiles are used for temporary data storage. All of the URAM and the majority of the BRAM are taken up by L1 image formation, showing that this part of the SAR processing chain has been accelerated with priority as much as possible given the available hardware. Finally, 29.3% of the available digital signal processing (DSP) slices are used for efficient implementation of algorithmic functions such as multiplication and accumulation.

2) Throughput: The throughput of L1 processing is mainly dependent on the FFTs and the burst access to the PL-side DRAM using AXI protocol. Regarding the FFTs, the time difference between the first pixel entering the core and the last pixel coming out of the core determines the latency. Since the FFTs are configured in pipelined streaming mode, each

TABLE III
ACHIEVED THROUGHPUT OF THE CFAR HARDWARE PROCESSOR FOR SHIP DETECTION WITH DIFFERENT BACKGROUND AND GUARD WINDOW SIZES COMPARED WITH THE THEORETICAL MAXIMUM THROUGHPUT OF THE PIPELINED CFAR DATAPATH. THE CIRCUIT RAN AT 125 MHZ. A SCENE WITH A SIZE OF 3488 × 8320 PIXELS AND A PIXEL SPACING OF APPROXIMATELY 3.8 m WAS USED

Back	Guard	Latency	Throughp.	Max. core	Max.
53 px	27 px	2.42 s	1.27 GB/s	2.65 GB/s	48.0 %
81 px	41 px	2.50 s	1.88 GB/s	4.05 GB/s	46.5 %
107 px	53 px	2.57 s	2.41 GB/s	5.35 GB/s	45.1 %
159 px	81 px	2.71 s	3.41 GB/s	7.95 GB/s	42.9 %
199 px	101 px	2.82 s	4.09 GB/s	9.95 GB/s	41.1 %
239 px	119 px	2.95 s	4.71 GB/s	11.95 GB/s	39.4 %
255 px	129 px	3.01 s	4.92 GB/s	12.75 GB/s	38.6 %

line is streamed continuously from cache and processed in a pipelined manner. This reduces the time required for the next line to be fed into the core, but can only work if a sufficient number of lines can be cached inside the FPGA fast enough. With the available URAM resources, a maximum of 16 × 32 k range lines or 64 × 8 k azimuth lines may be cached at a time. Roughly, 7 GiB of data are transferred between the PL-side DRAM and cache for the selected raw data block size. By maximizing the throughput of the AXI link and with an FPGA clock running at 125 MHz, data transfers contribute about 1 s to the total latency required for L1 image formation.

For L2 ship detection, the throughput of the CFAR hardware processor can be derived from the measured latency. With the optimized CFAR algorithm explained in Section IV-A1, for each pixel of the image, approximately one column of the CFAR background window has to be loaded from the PL-side DRAM if it is assumed for simplicity that pixels near the image borders can fit a full window around them. For example, a 3488 × 8320 pixels large image with CFAR background size of 199 pixels and guard size of 101 pixels requires $3488 \cdot 8320 \cdot 199 \cdot 2\ \mathrm{B} = 11.55\ \mathrm{GB}$ of data to be transferred from the PL-side DRAM to the input buffer. Table III shows the results for different background and guard window sizes compared with the theoretical maximum throughput of the pipelined CFAR datapath in Fig. 7. This maximum implies that the input buffer always holds enough data so that the pipeline is never stalled. In that case, the pipeline would process a new target pixel every five clock cycles and thus the processing time would be constant (1.16 s) regardless of the chosen CFAR window dimensions. As shown in Table III, in reality, the latency increases linearly with the size of the CFAR background size from 2.42 s for a very small window up to 3.01 s for the maximum possible window size of 255 pixels (see Section IV-A2). This proves that the core is limited by the data throughput from the PL-DRAM to the pixel buffer in the CFAR core, as the larger the CFAR window, the more data have to be transferred. Nevertheless, enormous speed-up has been achieved over the software implementation on the ARM quad-core PS, which took 16 min, while the hardware core needed just 2.71 s with the same settings.

3) Power Consumption: Looking at the ZU19EG MPSoC, the FPGA on-chip sensors reported just 3.6 W in idle state with peaks of 6.2 W and 5.0 W during L1 monochromatic ω-k and L2 CFAR hardware processing, respectively, confirming the high efficiency of the implemented algorithms. However, this includes only the power rails of the FPGA fabric and the PL-side DRAM. For the whole SAR board, power consumption under load of about 24 W was measured, which is consistent with >20 W presented for the ZU15EG variant of the multiMIND board [16].

B. Quality Results

Porting the existing algorithms for SAR image formation and processing to the onboard system required compromises with respect to image quality and, partially, product quality. This section compares the images and products generated onboard with their counterparts created by the regular algorithms in operational use.

1) Image Formation: The accuracy of L1 image formation was analyzed in detail on two SAR acquisitions, one of a test site with corner reflectors to get the parameters of the impulse response function, and one maritime scene to test the geolocation accuracy under conditions representative of the intended application.

The selected test site in Oberpfaffenhofen, Germany, is equipped with four corner reflectors and was imaged with TerraSAR-X. The resulting scene data were processed both by the operational TerraSAR-X processor and by the EO-ALERT onboard processor. The output of the operational TerraSAR-X processor was an SSC image.

A point target analysis was performed on the basis of the four corner reflectors both for SSC images and for the MSD image. Table IV lists the parameters of the impulse response function. The 3 dB width in azimuth and range of the onboard products is increased because of the reduced processing bandwidth in the EO-ALERT processor. There is an inverse proportionality between the chosen processing bandwidth and the resulting peak width. The azimuth processing bandwidth of the EO-ALERT processor amounts to 2 kHz in azimuth and 82.5 MHz in range (compared with 2.765 kHz and 100 MHz in the operational TerraSAR-X processor).

Hence, an increase in the 3 dB width by 38% in azimuth and by 21% in range is expectable and provides an explanation for almost all of the observed peak broadening. As a consequence of the peak broadening, peak power also differs between operational and onboard processing. Peak phase and peak position were compared on the basis of single corner reflectors, and the found differences were averaged (as an average position or phase of all corner reflectors does not hold any meaningful information). On average, the peak phase differs in the range of one degree, while the peak positions show only slight differences in the order of a few centimeters. Peak-to-sidelobe ratio (PSLR) and integrated sidelobe ratio (ISLR) of the SSC images are similar, whereas PSLR and ISLR of the MSD image are degraded as a consequence of the detection step. Regarding average signal energy, both SSCs are within 0.2 dB m^2 of each other, indicating correct radiometric calibration of the EO-ALERT processor. The same is true for the main lobe energy.

TABLE IV

RESULTS OF THE POINT TARGET ANALYSIS ON CORNER REFLECTORS AT THE DLR TEST SITE IN OBERPFAFFENHOFEN, GERMANY. THE MEASURED POINT TARGET PARAMETERS FROM THE TERRASAR-X IMAGE AND FROM THE EO-ALERT IMAGE ARE COMPARED AGAINST EACH OTHER

Parameter	TerraSAR-X SSC	EO-ALERT SSC	EO-ALERT MSD	Difference SSCs	Unit
Azimuth 3 dB width	2.97	4.41	5.45	1.44 ± 0.22	m
Range 3 dB width	1.77	2.20	4.04	0.42 ± 0.06	m
Peak power (cal)	47.62	44.92	42.14	-2.70 ± 0.29	dB
Peak phase	*	*	*	-1.11 ± 2.56	deg
Peak azimuth position	*	*	*	0.04 ± 0.25	m
Peak range position	*	*	*	-0.03 ± 0.08	m
Azimuth PSLR	-30.91	-29.96	-9.70	0.96 ± 0.73	dB
Range PSLR	-30.30	-29.77	-9.87	0.53 ± 0.53	dB
2-D ISLR	-16.36	-17.42	-1.99	-1.06 ± 0.73	dB
Clutter power (cal)	-8.27	-8.79	7.08	-0.52 ± 0.15	dB
Signal energy (cal)	55.22	55.02	57.82	-0.19 ± 0.10	$\mathrm{dB\,m^2}$
Main lobe energy (cal)	55.12	54.94	55.62	-0.17 ± 0.09	$\mathrm{dB\,m^2}$

(*=averaging of this parameter does not lead to a meaningful value)

TABLE V

RESULTS OF THE POINT TARGET ANALYSIS PERFORMED ON A SCENE IN THE GULF OF NAPLES. THE DEVIATIONS OF THE POSITIONS OF SHIPS IN THE EO-ALERT IMAGE RELATIVE TO THE RESPECTIVE POSITIONS IN THE TERRASAR-X IMAGE ARE DETERMINED

Target	Azimuth	Range	Latitude	Longitude	Height	Unit
Ship 1	-1.73	1.32	-1.01	3.39	0.24	m
Ship 2	0.32	0.03	0.39	0.54	0.28	m
Ship 3	1.14	-0.61	0.74	-1.38	0.28	m
Average $\pm$ Stddev	-0.09 $\pm$ 1.48	0.25 $\pm$ 0.98	0.03 $\pm$ -1.01	0.85 $\pm$ 2.40	0.26 $\pm$ 0.03	m

The geolocation accuracy of the generated image is measured w.r.t. the position of some bright scatterers onboard of ships which are observed in an example scene acquired in the Gulf of Naples. The radar coordinates (range and azimuth) of the onboard generated image are compared with the coordinates in the original TerraSAR-X product, which is well known for its high localization accuracy at centimeter level [36], [37]. Thereafter, on the basis of the same points, the accuracy of the geolocation grid is cross checked against the one of the TerraSAR-X products. By visual inspection, three ships were clearly noticeable in the example scene. At each of the ships, one dominant scatterer was selected for the measurement.

The results are presented in Table V. On average, the difference between the measured coordinates from the TerraSAR-X and EO-ALERT images is at decimeter level w.r.t. azimuth and range. The standard deviation of the difference is below 2 m. Thus, the localization accuracy excels the project requirement of 10 m. Furthermore, the same test site features the high contrast scenery of Naples and its port. The hardware-focused image proves, as depicted in Fig. 8, that the block floating-point FFT operations do not cause visible artifacts such as smear out of bright targets into the darker image areas.

2) Ship Detection: The ship detection algorithm proved robust to the changes in image quality caused by onboard processing. Differences in detection results occur due to the coarser product resolution. From the 21 ships with automatic identification system (AIS) data included in all test scenes, three were rather small with 20, 20, and 21 m, respectively. These three ships were not detected in the onboard algorithm. The next larger ship was above 60 m and was successfully detected, as were all remaining 17 ships on the test scenes.

Fig. 8. Hardware-processed TerraSAR-X image detail depicting parts of the city of Naples and its port. The 10 km × 5 km detail is extracted from a focused azimuth block covering 32 km in range and 13 km in azimuth direction.

Fig. 9 shows thumbnails of different-size ships, as they are included with the alerts from onboard processing (left) with the same thumbnail extracted from on-ground TerraSAR-X Multi-look ground range-detected (MGD) imagery (right). Onboard thumbnail dimensions are 100 px × 100 px with pixel size of approx. 3.80 m, so that each thumbnail covers an area of 380 m × 380 m. The MGD thumbnails have the same coverage in meters, but pixel size is smaller at 2.75 m. Thus, they contain 138 px × 138 px. The shown ships with lengths 70, 115, and 180 m were detected by both the on-ground and onboard processor using their respective image products; however, the 20-m-long ship could not be detected by the onboard processor as previously mentioned.

With the onboard system, more false positives are detected on water originating from land structures. This is a direct result of the reduced scene size in azimuth: while a full TerraSAR-X SM scene has a length of 60 km, the onboard scenes have a reduced length of 12.5 km, and coastal areas were selected

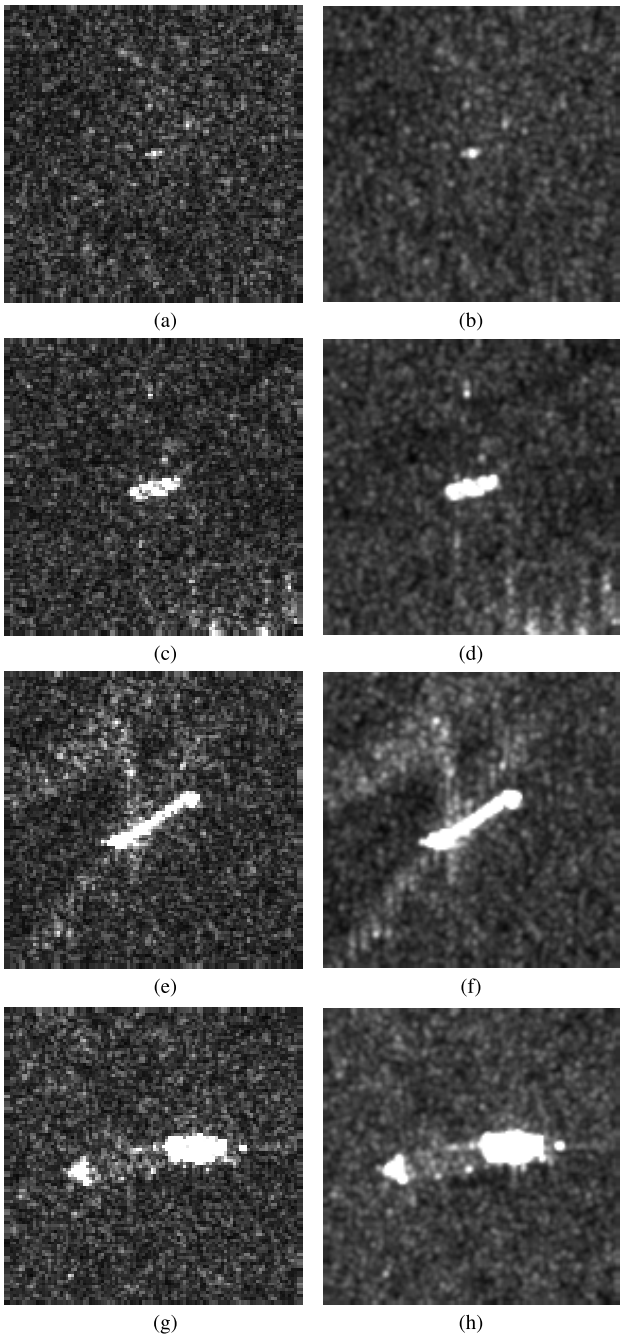

Fig. 9. Ship example thumbnails of different sizes for a scene acquired on February 2, 2017 at the coast of Naples, Italy. Onboard generated products are shown in the left column, with on-ground TerraSAR-X MGD imagery in the right column for comparison. The latter reveals smoother appearance and better speckle reduction due to its more elaborate highly accurate multilook processing. (a) 20 m length, onboard. (b) 20 m length, MGD. (c) 70 m length, onboard. (d) 70 m length, MGD. (e) 115 m length, onboard. (f) 115 m length, MGD. (g) 180 m length, onboard. (h) 180 m length, MGD.

from the full TerraSAR-X scene. As a result, the onboard scenes do not contain enough land to include the source of azimuth ambiguities appearing on the water. Hence, these cannot be filtered out by the algorithm. This is of course only a problem near land and does not affect scenes on open water with no land nearby. The operational processing has generally the same problem; however, it is not as frequent as the full TerraSAR-X SM scene, which often contains large enough sections of land.

A fourth point, which holds true also for the extreme weather scenario, is the land mask used. The onboard system uses the global self-consistent, hierarchical, high-resolution shoreline database (GSHHS) land mask in full resolution (about 60 m posting, about 80 MB file size) as a compromise between resolution and storage space; the operational system uses a land-water mask derived from OpenStreetMap data (resolves features down to about 5 m, about 1 GB file size) which also includes many harbor structures not resolved in GSHHS. Errors in the land masking might lead to real ships on water being filtered out, or false positives when objects on land are not removed.

3) Extreme Weather: An example of extreme weather processing is shown in Fig. 10. While ship detection relies mostly on the CFAR algorithm, which is rather robust to image quality, especially the sea state algorithm using GLCM and spectral analysis is expected to be more sensitive to the minor quality degradation of the onboard scenes. Also, the operational processing chain in the ground station can compute the wind direction from models even before the scheduled acquisition take place and provide a wind field for the scene at the start of near real-time processing. While the support for such precomputed wind fields is available also to the onboard algorithm, the required preparation time and data transfer do not make this feasible for a quick-react onboard system as envisaged in EO-ALERT. Hence, the onboard algorithm is always run with a fixed input wind direction.

A direct comparison of detected wind speeds and wave heights is not possible due to different integration areas, where always 1024×1024 pixels are taken, which results in a larger area onboard than on the original MGD products due to the higher pixel sizes. Also, the scene centers are not exactly aligned due to the different sizes and boundaries of the scenes. Fig. 11 shows a comparison of the detected wind speeds U10 and wave heights Hs including all values retrieved from the scenes in Table VI. The apparent differences are caused partly by the different radiometric accuracies of the onboard product as discussed in Section V-B1, and also by different integration areas as stated earlier. Additionally, the sea state algorithm was trained on hundreds of original TerraSAR-X SM scenes with reference to buoy data [33], and it is very sensitive to changes in product quality and resolution. Hence, for operational use, the algorithm would need to be retrained with a large number of onboard processed scenes; this high effort was not foreseen for the demonstrational character of this project.

For wind, a general tendency toward too high values is seen in Fig. 11, especially for low wind speeds < 5 m/s, but a minor overestimation remains also for higher wind speeds. The root-mean-square deviation (RMSD) is 0.7 m/s.

Sea state shows a similar tendency toward slight overestimation for low wave heights, but tends to be underestimated for higher wave heights. These discrepancies are likely caused by the missing retraining. The RMSD for sea state is 0.2 m.

C. Latency Results

Minimizing the processing latency was a core development goal in the EO-ALERT project. The total system latency goal,

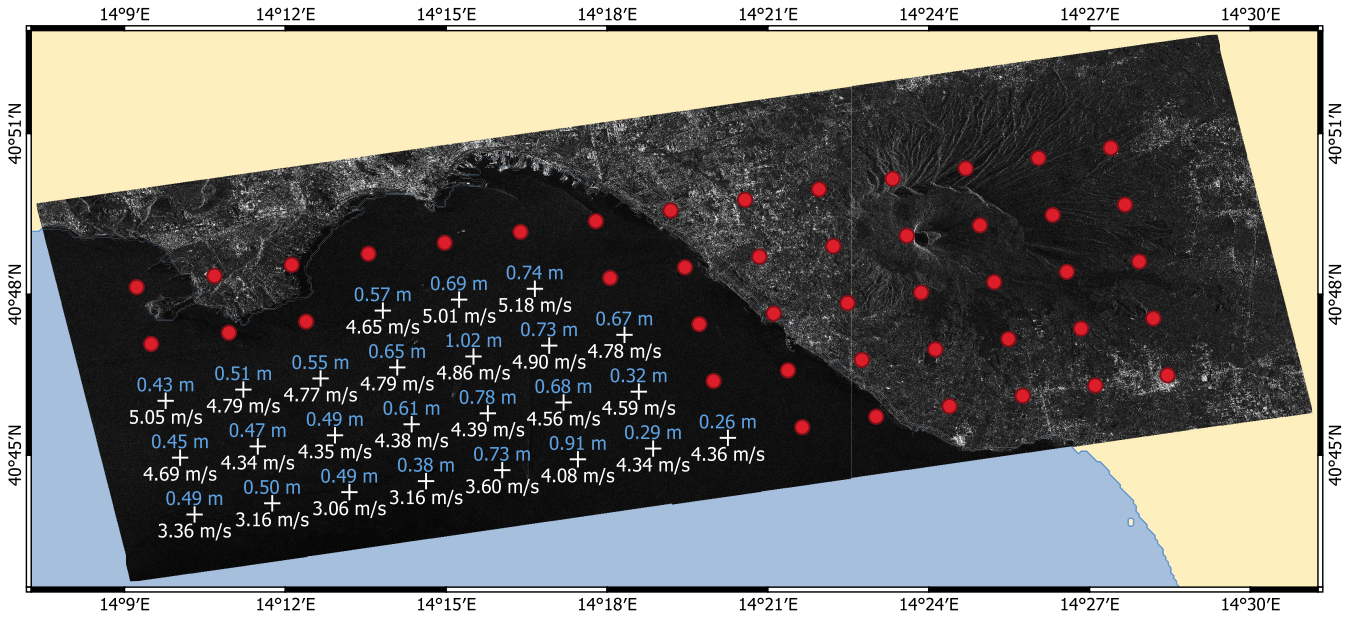

Fig. 10. Wind and sea state processing for a scene around Naples, Italy (scene #7 in Table VI). Red points show invalid grid cells on land, for which no calculation is performed. White crosses mark valid grid cells with the respective results for wind speed U10 (white) and mean wave height Hs (blue). TerraSAR-X data © DLR 2017.

TABLE VI
SCENE PARAMETERS FOR THE NINE SELECTED TEST ACQUISITIONS AND RESULTING PROCESSING LATENCIES

Number	Date	Masked Land [%]	Image Formation [s]	Ship detection [s]	Wind and sea state detection [s]
1	2016–11–22	1.6	3.7	12.2	37.4
2	2016–11–22	0.9	3.7	12.3	38.5
3	2016–11–22	0.0	3.8	12.2	38.4
4	2016–11–22	12.1	3.7	12.8	32.0
5	2016–12–31	0.0	3.6	8.2	36.4
6	2016–12–31	0.0	3.6	8.2	36.6
7	2017–02–16	55.3	3.7	21.0	16.4
8	2017–03–07	45.6	3.9	28.8	20.8
9	2017–01–22	45.5	3.9	28.3	21.1
Average			**3.7**	**16.0**	**30.8**

including all processing and data transfers to the end user, was set to a maximum of 5 min. This requirement allowed up to 210 s for SAR processing and product generation. Table VI shows processing times achieved in the different scenes.

Image formation achieves an average processing time of 3.7 s with only minimal variation depending on the small differences in scene size. The image processing algorithms, on the other hand, show larger variations in runtime depending on scene properties. For ship detection, land filtering is only applied after the initial CFAR detections to allow filtering of azimuth ambiguities on water originating from land objects. This leads to a large number of detected and processed objects in scenes with high land content, as there are many bright reflectors on land. Consequently, the algorithm performs slowest on scenes with much land and fastest if only water with few ships is in the scene.

The opposite is true for wind and sea state detection. Land masking is performed before other processing steps, and grid cells with land are not calculated. This reduces the processing time for scenes with high land percentage, and the longest processing times are recorded for scenes where all grid cells are evaluated.

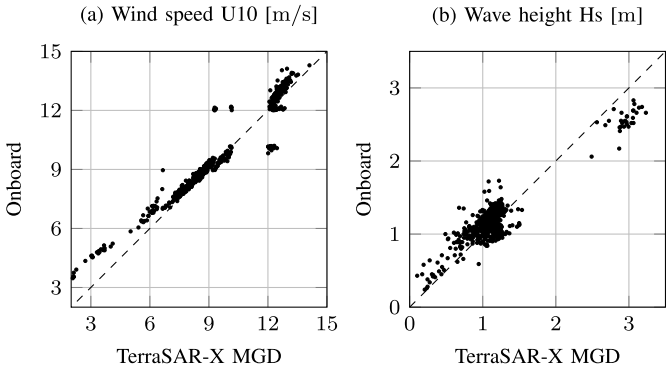

Fig. 11. Comparison of (a) wind speed U10 and (b) wave height Hs derived from original TerraSAR-X MGD and onboard scenes, respectively. The differences are a result of different radiometric accuracies, of different integration areas (depending on pixel size) and, in case of Hs, also of outstanding retraining of parameters.

Compared with the latency requirement, even the higher average for wind and sea state with 34.5 s (image formation

and processing) is way below the 210 s requirement. However, this also had to be achieved in the so-called max strip scenario of the project, which consists of three raw data blocks instead of one. For SAR, this is implemented by simply running the processing chain three times in a row on a single board. This choice was made, since even three scenes with a total latency of 103.5 s are processed in less than half of the latency requirement.

VI. Discussion

In this article, on-board SAR image formation and image processing were implemented on one MPSoC. The algorithm implementations of both image formation and image processing on a single hardware have not been attempted so far in any on-board SAR platform. The other major advancement of this work is the achieved speedup for L0 to L1 processing. L1 product generation took 3.7 s in the MSD case and 4.3 s in the SLC case, using a monochromatic ω-KA. Hence, the processing speed of the SLC product is 28.3 Mpx/s. In case of the MSD product, the multilooking means a reduction of image size by a factor of 4, and the increased speed is caused by the reduced data volume written to memory. Thus, for MSD data, the throughput is 8.1 Mpx/s, which, however, corresponds to 32.4 Mpx/s of SLC data at the input of multilooking. With respect to raw data, using an input size of 8192×32768 samples, the processing speed is 68.8 mega samples per second (MSPS).

In comparison with the previous work, the current implementation is significantly faster. In [7], the authors presented spaceborne on-board SAR processing with four Xilinx FPGAs and six 8-GB DDR3 DRAMs. The authors give the processing speed of their RDA-based SSC image formation as 18 Mpx/s. The CSA was used by [8] and [9] to generate an SSC product. The authors performed SAR image formation of raw data with 16384×16384 samples within 10.6 s on an FPGA with reconfigurable architecture and 12.1 s on an FPGA-ASIC PS, corresponding to 25.3 and 22.2 MSPS, respectively. Latency reduction in our implementation results from running four datapaths in parallel. This parallelization is possible, because the monochromatic ω-KA is computationally less complex and requires less resources per datapath in hardware.

The presented work demonstrates that satellites with onboard SAR focusing and product generation for enhanced NRT product delivery are feasible with current technology. The resulting processing times from L0 raw data to L2 products vary between 11.8 and 42.2 s per scene of about 375 km^2 when distributing the workload between PL and PS with adapted processing algorithms. In the EO-ALERT project, the full chain starting from acquired raw data up to the end user receiving alerts was developed, including data processing as described in this article, data compression, and encryption, as well as transfer to the end user. In some scenarios, such as SAR ship detection, total latencies for the full chain below 1 min were achieved. The project goal of a maximum latency of 5 min was achieved in every realistic scenario. This low-latency availability of alerts will become even more important with increasing frequency of EO satellite overflights, offering new applications for EO technology for a large group of end users reliant on almost real-time data.

Acknowledgment

This work reflects only the author's view and the European Commission (EC) is not responsible for any use that may be made from the information contained in this work. The EO-ALERT project is coordinated by DEIMOS Space. More information on the project is available at eo-alert-h2020.eu. TerraSAR-X data were provided by DLR under Announcement of Opportunity (AO) proposal OCE3625.

References

[1] M. Kerr *et al.*, "EO-ALERT: A novel architecture for the next generation of remote sensing satellites supporting rapid civil alerts," in *Proc. IAC*, Oct. 2020, pp. 1–13.

[2] A. Bergeron *et al.*, "Satellite on-board real-time SAR processor prototype," in *Proc. Int. Conf. Space Opt. (ICSO)*, vol. 10565, E. Armandillo, B. Cugny, and N. Karafolas, Eds. Bellingham, WA, USA: SPIE, 2017, pp. 670–676.

[3] D. Romano, V. Mele, and M. Lapegna, "The challenge of onboard SAR processing: A GPU opportunity," in *Computational Science–(ICCS)*, V. V. Krzhizhanovskaya, G. Závodszky, M. H. Lees, J. J. Dongarra, P. M. A. Sloot, S. Brissos, and J. Teixeira, Eds. Cham, Switzerland: Springer, 2020, pp. 46–59.

[4] F. Zhang, G. Li, W. Li, W. Hu, and Y. Hu, "Accelerating spaceborne SAR imaging using multiple CPU/GPU deep collaborative computing," *Sensors*, vol. 16, no. 4, p. 494, Apr. 2016.

[5] N. Ding, Z. Zong, B. Liu, and S. Yuan, "An FPGA hardware implementation for omega-K SAR imaging algorithm," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2021, pp. 5155–5158.

[6] X. Zhou, Z. J. Yu, Y. Cao, and S. Jiang, "SAR imaging realization with FPGA based on VIVADO HLS," in *Proc. IEEE Int. Conf. Signal, Inf. Data Process. (ICSIDP)*, Dec. 2019, pp. 1–4.

[7] Y. Sugimoto, S. Ozawa, and N. Inaba, "Spaceborne synthetic aperture radar signal processing using field-programmable gate arrays," *J. Appl. Remote Sens.*, vol. 12, Jul. 2018, Art. no. 035007.

[8] B. Li *et al.*, "Real-time spaceborne synthetic aperture radar float-point imaging system using optimized mapping methodology and a multi-node parallel accelerating technique," *Sensors*, vol. 18, no. 3, p. 725, Feb. 2018.

[9] C. Yang *et al.*, "A spaceborne synthetic aperture radar partial fixed-point imaging system using a field-programmable gate array–application-specific integrated circuit hybrid heterogeneous parallel acceleration technique," *Sensors*, vol. 17, no. 7, p. 1493, 2017.

[10] Y. Lou, D. Clark, P. Marks, R. J. Muellerschoen, and C. C. Wang, "Onboard radar processor development for rapid response to natural hazards," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 6, pp. 2770–2776, Jun. 2016.

[11] M. Pfitzner, F. Cholewa, P. Pirsch, and H. Blume, "Development and potential of real-time FPGA frequency-based SAR image processing for short-range FMCW applications," in *Proc. Int. Conf. Radar*, Sep. 2013, pp. 101–105.

[12] S. K. Joshi and S. V. Baumgartner, "Range-Doppler tracking of ships using single-channel airborne radar data," in *Proc. 13th Eur. Conf. Synth. Aperture Radar (EUSAR)*, Apr. 2021, pp. 1–6.

[13] H. Breit, S. Mandapati, and U. Balss, "An FPGA/MPSoC based low latency onboard SAR processor," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2021, pp. 5159–5162.

[14] S. Wiehle, D. Günzel, and B. Tings, "SAR satellite on-board ship, wind, and sea state detection," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2021, pp. 8289–8292.

[15] S. Wiehle, D. Günzel, B. Tings, H. Breit, U. Balss, and S. Mandapati, "A satellite on-board SAR processing chain for generation of rapid civil alerts," in *Proc. 13th Eur. Conf. Synth. Aperture Radar (EUSAR)*, Mar. 2021, pp. 1–5.

[16] A. Pawlitzki and F. Steinmetz, "multiMIND–high performance processing system for robust newspace payloads," in *Proc. Eur. Workshop-Board Data Process. (OBDP)*, Jun. 2021.

[17] J. Nalepa, P. Kuligowski, M. Gumiela, M. Drobik, and M. Nowak, "Leopard: A new chapter in on-board deep learning-powered analysis of hyperspectral imagery," in *Proc. IAC-CyberSpace Ed.*, Oct. 2020, pp. 1–4.

[18] R. Bamler and B. Schättler, *Geocoding: ERS-1 SAR Data and Systems*. Karlsruhe, Germany: Wichmann-Verlag, 1993, pp. 53–102.

[19] W.-C. Fang and M. Y. Jin, "On board processor development for NASA's spaceborne imaging radar with VLSI system-on-chip technology," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 2, May 2004, pp. II–901.
[20] L. Chen and T. Long, "The research and implementation of spaceborne SAR real-time quick look imaging system," in *Proc. Congr. Image Signal Process.*, vol. 2, May 2008, pp. 587–591.
[21] M. Pfitzner, S. Langemeyer, P. Pirsch, and H. Blume, "A flexible real-time SAR processing platform for high resolution airborne image generation," in *Proc. IEEE CIE Int. Conf. Radar*, vol. 1, Oct. 2011, pp. 26–29.
[22] C.-L. Yu and C. Chakrabarti, "Transpose-free SAR imaging on FPGA platform," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2012, pp. 762–765.
[23] M. Pfitzner, F. Cholewa, P. Pirsch, and H. Blume, "FPGA based architecture for real-time SAR processing with integrated motion compensation," in *Proc. Asia–Pacific Conf. Synth. Aperture Radar (APSAR)*, 2013, pp. 521–524.
[24] N. Hou, D. Zhang, G. Du, and Y. Song, "An FPGA-based multi-core system for synthetic aperture radar data processing," in *Proc. Int. Conf. Anti-Counterfeiting, Secur. Identificat. (ASID)*, Dec. 2014, pp. 1–4.
[25] M. Pfitzner, F. Cholewa, P. Pirsch, and H. Blume, "A flexible hardware architecture for real-time airborne Wavenumber Domain SAR processing," in *Proc. 9th Eur. Conf. Synth. Aperture Radar (EUSAR)*, Apr. 2012, pp. 28–31.
[26] R. Bamler, "A comparison of range-Doppler and wavenumber domain SAR focusing algorithms," *IEEE Trans. Geosci. Remote Sens.*, vol. 30, no. 4, pp. 706–713, Jul. 1992.
[27] I. Cumming and F. Wong, *Digital Processing of Synthetic Aperture Radar Data: Algorithms and Implementation*. Norwood, MA, USA: Artech House, 2005.
[28] S. Brusch, S. Lehner, T. Fritz, M. Soccorsi, A. Soloviev, and B. van Schie, "Ship surveillance with TerraSAR-X," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 3, pp. 1092–1103, Mar. 2011.
[29] B. Tings, C. A. Bentes da Silva, and S. Lehner, "Dynamically adapted ship parameter estimation using TerraSAR-X images," *Int. J. Remote Sens.*, vol. 37, no. 9, pp. 1990–2015, May 2016.
[30] A. Frost, R. Ressel, and S. Lehner, "Automated iceberg detection using high-resolution X-band SAR images," *Can. J. Remote Sens.*, vol. 42, no. 4, pp. 354–366, 2016.
[31] D. Gregorek *et al.*, "FPGA prototyping of a high-resolution TerraSAR-X image processor for iceberg detection," in *Proc. 53rd Asilomar Conf. Signals, Syst., Comput.*, Nov. 2019, pp. 1832–1835.
[32] X.-M. Li and S. Lehner, "Algorithm for sea surface wind retrieval from TerraSAR-X and TanDEM-X data," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 5, pp. 2928–2939, May 2014.
[33] A. L. Pleskachevsky, W. Rosenthal, and S. Lehner, "Meteo-marine parameters for highly variable environment in coastal regions from satellite radar images," *ISPRS J. Photogramm. Remote Sens.*, vol. 119, pp. 464–484, Sep. 2016.
[34] A. Pleskachevsky, S. Jacobsen, B. Tings, and E. Schwarz, "Estimation of sea state from Sentinel-1 synthetic aperture radar imagery for maritime situation awareness," *Int. J. Remote Sens.*, vol. 40, no. 11, pp. 4104–4142, Jun. 2019.
[35] A. Pleskachevsky, B. Tings, and S. Jacobsen, "Sea state from Sentinel-1 SAR wave mode imagery for maritime situation awareness," in *Proc. 13th Eur. Conf. Synth. Aperture Radar (EUSAR)*, Mar./Apr. 2021.
[36] A. Schubert, D. Small, M. Jehle, and E. Meier, "COSMO-skymed, TerraSAR-X, and RADARSAT-2 geolocation accuracy after compensation for Earth-system effects," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2012, pp. 3301–3304.
[37] U. Balss *et al.*, "High precision measurement on the absolute localization accuracy of TerraSAR-X," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2012, pp. 2991–2994.

Stefan Wiehle received the Diploma degree in physics and the Dr. rer. nat. degree from Technische Universität Braunschweig, Braunschweig, Germany, in 2010 and 2014, respectively.

He is currently with the DLR Maritime Safety and Security Lab, Bremen, Germany, a group at DLR's Remote Sensing Technology Institute (IMF). His focus is on the implementation and automation of SAR maritime value-adding algorithms, such as land-water-line detection, bathymetry derivation, or sea ice classification. He coordinated the project BASE-platform combining satellite-derived bathymetry from multiple sources and led DLR's activities in the EO-ALERT project developing satellite onboard processing.

Srikanth Mandapati received the B.E. degree in electronics and communication from Jawaharlal Nehru Technological University, Hyderabad, India, in 2013, and the M.S. degree in embedded systems from the Chemnitz University of Technology, Chemnitz, Germany, in 2017.

Afterward, he started his career as a Verification Engineer with AKKA Technologies, Munich, Germany. Since 2019, he has been with the Remote Sensing Technology Institute (IMF) of DLR working on SAR processing. His main areas of interest are embedded platforms and data acceleration techniques with FPGAs.

Dominik Günzel received the M.Sc. degree in electrical engineering from the University of Bremen, Bremen, Germany, in 2019.

Afterward, he joined the Maritime Safety and Security Lab, Remote Sensing Technology Institute (IMF), German Aerospace Center (DLR), Bremen, as a Research Scientist. His research focuses on data processing with purpose-built integrated digital circuits, including on-board of a satellite or in the datacenter, to enable novel real-time or near real-time services.

Helko Breit received the Diploma degree in electrical and telecommunication science from the Technical University of Munich, Munich, Germany, in 1990.

Since 1990, he has been with the German Aerospace Center (DLR), Wessling, Germany, where he is the Leader of the team "SAR Processing" at Remote Sensing Technology Institute. He has worked on a variety of international missions, including SIR-C/X-SAR, SRTM/X-SAR, and the German Missions TerraSAR-X/TanDEM-X. He contributed to several European Space Agency (ESA) studies and supported the commissioning of the ESA's Sentinel 1 SAR satellites. He was responsible for the development of the TerraSAR-X multimode SAR processor TMSP and the bistatic processing of TanDEM-X mission data. His research activities comprise the development of SAR processing algorithms and systems for future SAR satellites and missions such as HRWS and Tandem-L.

Ulrich Balss received the Diploma degree in physics and the Dr. phil. nat. degree from Johann Wolfgang Goethe University, Frankfurt, Germany, in 1994 and 2004, respectively.

From 2003 to 2010, he was the Scientific Staff with the Technical University Munich, Munich, Germany. In this time, he worked within the framework of a cooperation of the Technical University Munich and German Aerospace Center (DLR), and he was permanently located at the Remote Sensing Technology Institute (IMF) of DLR. In 2010, he joined DLR's Remote Sensing Technology Institute. He is involved in the German TerraSAR-X and TanDEM-X Missions. In particular, he participated in the development of the TerraSAR-X Multimode SAR Processor (TMSP) and implemented the focusing kernel of the TMSP.