Reconfigurable Embedded Architectures for On-Board Synthetic-Aperture Radar Processing

Helena Cruz*, Mário Véstias†, José Monteiro‡, Horácio Neto§ and Rui Policarpo Duarte¶
*INESC-ID/Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal
Email: helena.cruz@tecnico.ulisboa.pt
†INESC-ID/Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa
Email: mvestias@deetc.isel.ipl.pt
‡INESC-ID/Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal
Email: jcm@inesc-id.pt
§INESC-ID/Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal
Email: hcn@inesc-id.pt
¶INESC-ID, Lisbon, Portugal
Email: rui.duarte@tecnico.ulisboa.pt

***Abstract*—SAR systems designed for on-board space environments present different challenges when compared to other systems. Constraints in performance, size, weight, power consumption and image quality are aspects that need to be taken into consideration when developing on-board systems. This paper presents an evaluation of two different multi-core embedded architectures for the Backprojection algorithm suitable for small satellites. Single and multi-core implementations are discussed, as well as the computation of approximations for the most time-consuming operations of the algorithm. Two different commercially available systems were tested: Pynq-Z2 and Ultra96, which are compared in terms of execution time, power consumption and efficiency, and image quality.**

I. Introduction

Synthetic-Aperture Radar (SAR) is a technology used to generate images from objects or landscapes using a moving radar system. The relevance of SAR has been increasing throughout the years due to its ability to operate regardless of weather conditions and without a light source, that is, day and night. SAR systems are installed onto moving platforms such as satellites, aircrafts or drones and can cover a large area due to the movement of the platforms [1]. For space applications, SAR introduces new concerns in terms of data availability since data acquired is not immediately visible and requires offline and off-site processing and an RF link with high bandwidth capacity and availability. The capacity to digest the SAR signals on-board and broadcast the processed images in real-time is of most importance and is yet to become widely available. This paper presents a comparison on the evaluation of performance, power, and quality of SAR images generated by the Backprojection algorithm, on two different small form-factor computing platforms. To compare the power efficiency of the different platforms it considers implementations with single and multi-core, and fixed-point and floating-point precisions.

II. SAR Image Processing

There are many SAR image generation algorithms, which can be divided into two large groups: Fast Fourier Transform (FFT)-based and non FFT-based. FFT-based algorithms, such as the Range-Doppler [2] or Polar Format [3] algorithms, are typically more efficient, however, the image has lower quality. For instance, in the Range-Doppler algorithm, the energy is not entirely concentrated on the range migration curve, introducing degradation in the range focus [4]. In the Polar Format algorithm, the two-dimensional FFT operation introduces geometrical warping and loss of focus as the distance increases from the scene center [3], [5], [4]. The Range-Doppler and the Polar Format algorithms have a computational complexity of $O\left(N^2 log_2\left(N\right)\right)$.

Non-FFT-based algorithms, such as the Backprojection algorithm, generate images with higher quality, however, have a higher computational complexity. The Backprojection algorithm has a complexity of $O\left(N^3\right)$ and was chosen for this architecture due to the quality of the generated images, even with the increased complexity.

A. Backprojection Algorithm

The Backprojection algorithm is based on the projection of the echoes received by the radar on a bitmap [6], [7], [8]. This projection is calculated for each of the pixels of the image and the resulting image is the accumulation of the projections. The steps of the Backprojection algorithm, for each pulse and each pixel, are as follows:

1) Compute the distance from the platform to the pixel.
2) Use the distance to calculate the position (range) in the dataset.
3) Using linear interpolation, sample the range calculated in the previous step.
4) Scale the sampled value using a matched filter to calculate the pulse contribution.
5) Add the contribution of each pulse to the final image.

The value of each pixel is the accumulation of the pulse contributions calculated using steps 1-5, therefore, the computation of each pixel is independent and can be easily parallelized. The pseudocode of the Backprojection algorithm is in Figure 1.

for all pixels k **do**
  $fk \leftarrow 0$
  for all pulses p **do**
    $R \leftarrow ||ak - vp||$ {calculate distance from platform to pixel (step 1)}
    $b \leftarrow \lfloor (R - R0)/\Delta R \rfloor$ {range bin (integer) (step 2)}
    if $b \in [0, Np - 2]$ **then**
      $w \leftarrow \lfloor (R - R0)/\Delta R \rfloor - b$ {interpolation weight}
      $s \leftarrow (1 - w) \cdot g(p, b) + w \cdot g(p, b + 1)$ {data sample (step 3)}
      $fk \leftarrow fk + \exp^{i \cdot ku \cdot R}$ {add pulse's contribution (steps 4 and 5)}
    end if
  end for
end for

Fig. 1: Pseudocode of the Backprojection algorithm.

The Backprojection algorithm can be applied to different SAR modes: stripmap, spotlight and circular SAR. In stripmap mode, the radar moves along the azimuth with a fixed antenna, illuminating a different region with the movement. In spotlight SAR, a moving antenna is mounted on a moving platform, illuminating the same region as the platform moves along the azimuth region. When compared to stripmap, the covered region is smaller, but the range resolution is superior. Lastly, circular SAR consists of a platform moving around the illuminated area in a circular motion, gathering data from all 360 angles. Circular SAR, similarly to spotlight, has superior resolution, however, the covered area is inferior to stripmap [2], [9]. In this paper, the Backprojection implementation used is for spotlight SAR.

B. Image Quality

The quality of the computed SAR images can be measured using quality metrics such as Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE) and Structural Similarity (SSIM) [10]. PSNR or MSE calculate the absolute error between the pixel values of the resulting image and a reference, whereas the SSIM is a perception-based metric based on the degradation of structural information. Equation 1 calculates the norm of the resulting pixel value (rk) divided by the norm of the difference between the values of the resulting pixel and the pixel from the reference image (tk).

$$SNR_{dB} = 10 \cdot \log_{10} \left(\frac{\sum_{k=1}^{N} |rk|^2}{\sum_{k=1}^{N} |rk - tk|^2} \right) \tag{1}$$

SSIM considers that pixels that are spatially close will have stronger dependencies when compared to other pixels and takes into consideration three different components: luminance, contrast, and structure. The SSIM of two images x and y is given by Equation 2, where μ_x is the mean intensity of x, μ_y is the mean intensity of y, σ_x is the standard deviation of x, σ_y is the standard deviation of y, C_1 and C_2 are constants included to avoid instability when $\mu_x^2 + \mu_y^2$ and $\sigma_x^2 + \sigma_y^2$, respectively, is close to zero, $C_1 = (K_1 L)^2$ and $C_2 = (K_2 L)^2$ where L is the dynamic range of the pixel values, $K_1 = 0.01$ and $K_2 = 0.03$, by default.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{2}$$

III. Algorithm Acceleration

To accelerate the algorithm, two approaches were followed: parallelization and utilization of approximations to speed the execution time. The following sections detail each of these approaches.

A. Wordlength Optimization

The implementations of the Backprojection algorithm considers all variables to be double precision floating-point by default. However, double-precision floating-point arithmetic is too time consuming and leads to long execution times. Because of this, the most time-consuming operations were implemented using approximations using fixed-point format from the LIBFIXMATH library. Alternative implementation such as CORDIC [11] were evaluated but provided worst results, thus they were not considered for further study.

B. Algorithm Parallelization

From the pseudocode presented in Figure 1 is it easily identifiable that it is possible to parallelize the Backprojection algorithm since the pixels can be computed independently. The parallelization of the algorithm was performed using a static division of the workload for the available processors.

IV. Target Reconfigurable Embedded Systems

This section presents the details of the processing systems selected for this research work. Two different Reconfigurable Systems on-a-Chip with a multi-core processor and reconfigurable logic (SoC-FPGA) were considered. Even though the reconfigurable fabric of the device was not used in this work, these devices were considered because they have the possibility to add a custom hardware accelerator in the future. In this work the following Xilinx SoC-FPGAs boards were used: Pynq-Z2 from TUL and Ultra96 from Avnet.

A. Pynq-Z2

The Pynq-Z2 board has a Zynq XC7Z7020 SoC-FPGA from Xilinx. This device is populated with reconfigurable logic and has a hard processor with a Dual-core ARM A9 running at 650MHz. This board measures 87x140 mm and can be supplied with 5V from an USB port.

B. ULTRA96

Similarly to the Pynq-Z2, the Ultra96 is also based on an SoC-FPGA. This board holds a Xilinx Zynq UltraScale+ MPSoC ZU3EG. Inside the device there is a Quad-core ARM A53 CPU running at 1.2GHz. This board also features 2 GB of LPDDR4 memory, measures 85x54 mm, and its power supply provides 12V/3A. It has active cooling.

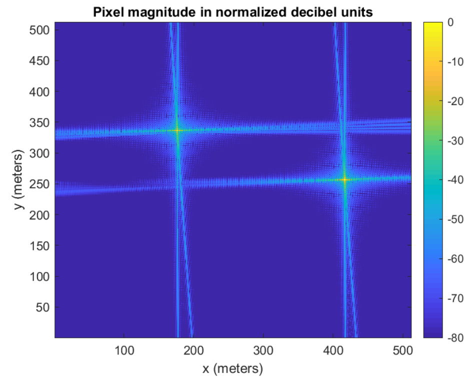
V. Evaluation

Each system was evaluated in terms of execution time, power, and approximation errors. The implementation of the Backprojection algorithm used in this paper is part of the PERFECT Suite [12], and was written in C. The Backprojection was tested using a set of synthetic pulses to generate a 512 × 512px image. To determine which functions could benefit from the fixed-point approximations, the profiling was done using gprof[1], compiled with GCC[2] with the optimization level `-o3`[3]. The trigonometric functions (sine/cosine) consume almost 85% of the execution time, while the rest of the algorithm required less than 16%. From this analysis, it is possible to conclude that the trigonometric functions should be the target of the wordlength optimization and be replaced with fixed-point approximations.

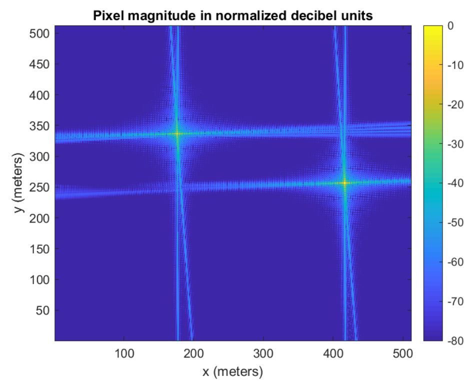
To reduce the processing time, the trigonometric functions were implemented using fixed-point arithmetic from the Libfixmath library. However, such approximation impacted the image quality due to loss of precision. Floating-point computations were performed using double-precision floating-point format, and hence are considered as reference. When comparing with the image compute with approximations the differences are not visible, Figure 2. The image above is the original version, calculated using the Backprojection algorithm implementation available in the PERFECT Suite [12], while the image below was generated using the LIBFIXMATH approximation to calculate the sine and cosine functions. To quantify the the precision loss, the SNR was 100.66dB and the SSIM was 0.9999973523, which means they are very similar to each other.

A. System Performance and Power

The system's performance was assessed by measuring the time which implementation required to complete the processing of the same image. The power consumption was measured using a programmable Thurlby Thandar (TTi) PL303QMD-P power supply. The energy (e) consumed by each implementation is defined as consumed power(p) × execution time(t). Table I summarizes the execution times, power and energy consumed by each implementation on all devices. For each device, the most energy efficient implementation is highlighted. As expected, the fixed-point multi-core implementations had a lower execution time when compared to either the single-core or the floating-point implementations. Both devices have potential to further optimize these implementations, since the FPGA fabric is not being used in this work. Even though the power consumption of the Ultra96 is superior to the Pynq-Z2, when the execution time is considered the Ultra96 is the most efficient device and implementation.



(a) Image generated using the original implementation of the Backprojection algorithm. Has a SSIM of 0.9999997399 when compared to the provided with the dataset and a SNR of 142.75dB.



(b) Image generated using the LIBFIXMATH library to calculate the cosine and sine functions instead of the standard version used in the Backprojection algorithm. Has a SSIM of 0.9999973523 when compared to the reference provided with the dataset and a SNR of 100.66dB.

Fig. 2: Images generated using the Backprojection algorithm and the PERFECT Suite dataset. The first image is the original version and the second was generated using an approximation for the sine and cosine functions.

TABLE I: Performance and power consumption according to device and implementation.

SoC-FPGA	sin/cos	core	t [s]	p [mW]	e [mWh]
Pynq-Z2	float	single-core	473	1715	225
	fixed-point	single-core	125	1720	60
	float	multi-core	238	1840	122
	fixed-point	multi-core	63	1810	32
Ultra96	float	single-core	273	2035	154
	fixed-point	single-core	124	2020	69
	float	multi-core	69	2205	42
	fixed-point	multi-core	31	2130	18

VI. Conclusions

This work compared the performance of two computing systems candidates for on-board SAR image processors. Usually, the performance of a system is the only observed factor in choosing a computing system. However, on-board systems are powered via batteries and therefore a high-performance system

[1] https://ftp.gnu.org/old-gnu/Manuals/gprof-2.9.1/html_mono/gprof.html
[2] https://gcc.gnu.org/
[3] https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html

which demands a lot of current requires heavier batteries than a more power efficient system. In this case, the Zynq Ultrascale+ SoC-FPGA is the fastest system.

SoC-FPGAs are best suited for fixed-point precision, and there they have good power efficiencies. Moreover, considering that the reconfigurable fabric of the FPGA can be configured to have an accelerator, it opens the possibility to have systems in the future real-time performance on-board.

ACKNOWLEDGMENT

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT), under grants with references UIDB/50021/2020 (INESC-ID multi-annual funding) and project SARRROCA (PTDC/EEI-HAC/31819/2017). Helena Cruz would like to acknowledge FCT for the support through grant with reference SFRH/BD/144133/2019.

REFERENCES

[1] M. Soumekh, *Synthetic Aperture Radar Signal Processing with MATLAB Algorithms*, J. W. t. Sons and Inc., Eds. New York: John Wiley & Sons, 1999. [Online]. Available: https://www.wiley.com/en-pt/Synthetic+Aperture+Radar+Signal+Processing+with+MATLAB+Algorithms-p-9780471297062

[2] I. G. Cumming and F. Wong, "Digital Processing of Synthetic Aperture Radar Data: Algorithms and Implementation." Artech House, 2005.

[3] R. Deming, M. Best, and S. Farrell, "Polar format algorithm for SAR imaging with Matlab," vol. 9093, 2014, pp. 47 —- 66. [Online]. Available: https://doi.org/10.1117/12.2050681

[4] W. Hughes, K. Gault, and G. Princz, "A comparison of the Range-Doppler and Chirp Scaling algorithms with reference to RADARSAT," *IGARSS '96. 1996 International Geoscience and Remote Sensing Symposium*, vol. 2, pp. 1221—1223 vol.2, 1996.

[5] B. D. Rigling and R. L. Moses, "Taylor expansion of the differential range for monostatic SAR," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 1, pp. 60—64, 2005.

[6] D. Pritsker, "Efficient Global Back-Projection on an FPGA," *2015 IEEE Radar Conference (RadarCon)*, pp. 0204—0209, 2015.

[7] M. Desai and W. Jenkins, "Convolution backprojection image reconstruction for spotlight mode synthetic aperture radar," *IEEE Transactions on Image Processing*, vol. 1, no. 4, pp. 505–517, 1992.

[8] L. A. Gorham and L. J. Moore, "SAR image formation toolbox for MATLAB," vol. 7699, 2010, pp. 46 —- 58. [Online]. Available: https://doi.org/10.1117/12.855375

[9] J. Lu, "Design Technology of Synthetic Aperture Radar," 2019, pp. 75—111. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119564621.ch3

[10] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600—612, 2004.

[11] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Transactions on Electronic Computers*, vol. EC-8, no. 3, pp. 330–334, 1959.

[12] K. Barker, T. Benson, D. Campbell, D. Ediger, R. Gioiosa, A. Hoisie, D. Kerbyson, J. Manzano, A. Marquez, L. Song, N. Tallent, and A. Tumeo, *PERFECT (Power Efficiency Revolution For Embedded Computing Technologies) Benchmark Suite Manual*, 2013. [Online]. Available: http://hpc.pnnl.gov/projects/PERFECT/