HIGH PERFORMANCE DATA PROCESSOR (HPDP) – IMAGE PROCESSING APPLICATIONS OF A NEW GENERATION SPACE PROCESSOR

Gerard Vives Vallduriola[(1)], Tim Helfers[(2)], Daniel Bretz[(3)], Mohsin Syed [(4)], Daniel Witsch[(5)], Constantin Papadas[(6)], Vincent Pérel [(7),] Stefan Bartels [(8)]

[(1)] Airbus Defence and Space GmbH. Taufkirchen, Germany. Email: gerard.vives@airbus.com
[(2)] Airbus Defence and Space GmbH. Taufkirchen, Germany. Email: tim.helfers@airbus.com
[(3)] Airbus Defence and Space GmbH. Taufkirchen, Germany. Email: daniel.bretz@airbus.com
[(4)] Airbus Defence and Space GmbH. Taufkirchen, Germany. Email: mohsin.syed@airbus.com
[(5)] Airbus Defence and Space GmbH. Taufkirchen, Germany. Email:witsch.da@hotmail.com
[(6)]ISD, S.A. Athens, Greece. Email: papadas@isd.gr
(7) Centrale Polytéchnique Lyon. Lyon, France. Email: vincent.perel@cpe.fr
(8) Airbus Defence and Space GmbH. Taufkirchen, Germany.

ABSTRACT

Modern society depends heavily on satellite infrastructure. Conventional communication satellite payloads have the drawback to have their architecture fixed for the lifetime of the satellite (15 years) and therefore cannot adapt to changing communication standards and market/application evolution (like multimedia applications). A way to alleviate the rigidity is the use of on-board reconfigurable technology in order to be able to modify the processing of the signals performed on-board.

Additionally, the increasing use of new earth-observation and communication technologies coupled with rapidly changing customer needs require a high performance and flexible processing technology for on-board data processing. The algorithmic and processing requirements for such processing are of a magnitude larger than those which could be successfully handled by classical processors. Therefore, there is a need for a powerful and flexible processor that can process large amounts of data at high speeds, and at the same time, is reconfigurable to adapt to changes.

This paper shows part of the work and results of simulations obtained up to present date with the High Performance Data Processor (HPDP), an array-based processor developed by Airbus Defence and Space GmbH in Munich and ISD, SA in Greece. Further results of tests performed on the hardware are presented.

1. INTRODUCTION

The HPDP was tested in several data processing environments for benchmarking purposes. In this context, basic image processing is the first field presented in this paper, followed by stereoscopic image processing.

For image compression simulations, the CCSDS 122.0-B-1 [2] algorithm was initially tested. For on-board object detection, the Boundary Tensor was tested extensively and for autonomous navigation the Sobel operator was implemented. For cryptographic applications (in the frame of a different study), a pseudo-random number generator, the AES-256 and a Diffie-Hellman key exchange algorithm were successfully ported to the HPDP's array.

2. HPDP ARCHITECTURE

The HPDP architecture integrates the XPP reconfigurable processing core IP, space suitable peripherals and memory interfaces. No specialized hardwired cores for specific functions are required since the reconfigurable core is fully programmable and provides full range processing capabilities. Figure 1 depicts the major building blocks of the HPDP architecture.

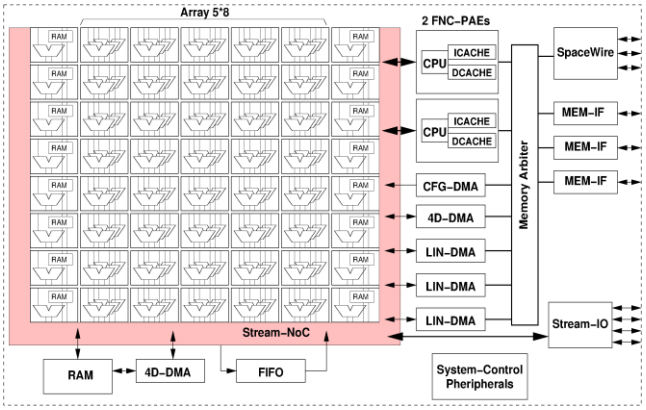

Figure 1. Overview of HPDP architecture.

The HPDP core i.e. XPP IP, consists of 40 ALU-PAEs and 16 RAM-PAEs and typically processes high bandwidth data streams. Two FNC-PAEs (Function PAEs) are coupled to the array's communication channels via versatile crossbars. The FNC-PAEs perform control flow tasks, sequential algorithms and system management. They communicate directly between each other through vertical communication channels.

Several DMA controllers transfer data in the background. DMA operation is controlled by a FNC-PAE. FIFOs uncouple the DMA channels from system RAM bursts, high speed interfaces and potentially stalled pipelines within the XPP. The specialized DMA controllers (4D-DMA) generate 4-dimensional access patterns for the on-chip buffers (X-RAM) or the off-chip memory (SRAM or SDRAM memory banks).

The HPDP architecture includes the following memory ports:

- Configuration Memory: The configuration memory is made up of PROM/EEPROM or SRAM devices and stores the code (boot code and/or application code).
- Data Memory: The memory port can be connected with either a SRAM or a SDRAM memory bank.

The communication with the external controller is based on the widely used SpaceWire standard.
In summary the following key features are provided by the HPDP architecture:

- Based on the XPP III Array Processor from PACT XPP Technologies providing more than 15 GigaOps/s. It has 40 ALU Processing Array Elements (16-bit), 2 Harvard type VLIW 16-bit processor cores (FNC-PAEs), and 256 Kbit high speed on-chip RAM with error protection.
- 2 external data memory interfaces for SRAM and/or SDRAM devices, error protected, with data bandwidth of up to 200 MByte/s
- 4 x 1.6 Gbit/s Streaming Ports
- Fully reprogrammable platform
- 3 SpaceWire interfaces operating at 100 Mbps on each channel with routing capability.

The architecture includes standard space relevant features like:

- ECC/EDAC and scrubbing function in the external memory interfaces
- Error protection in on-chip memory
- Triple mode redundancy reset and clock logic
- Clock synchronous design
- JTAG scan and BIST
- Space relevant control interface (SpaceWire).

3. COMPARISON WITH OTHER SPACE PROCESSORS

The currently available European space qualified data processors are not able to fulfill the algorithmic and processing requirements of future applications. The performance versus flexibility comparison between the XPP and currently available space processors is shown in Figure 2.

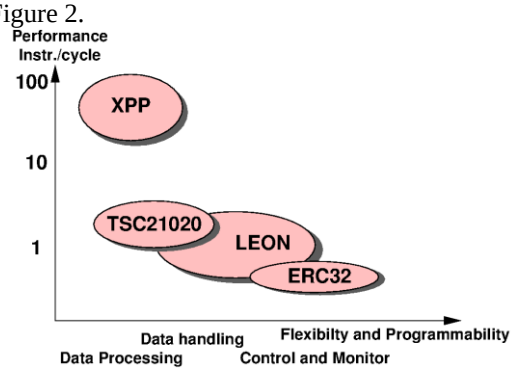

Figure 2. Performance comparison.

The SPARC processors including the ERC32 and the LEON provide processing performance in order to perform data handling as well as control and monitor tasks. Data processing is foreseen in the field of low rate applications such as attitude and orbit control.

The TSC21020 provides performance for medium rate signal processing and data handling applications. It is based on the commercially available ADSP21020. Its computation units support up to 3 floating points operations in parallel for special applications such as FFT. The clock speed of the TSC21020, however, is nailed down through the external memory interface, which is dependent on the memory access time.

As shown in Figure 2, the XPP reaches a much higher performance than traditional space processors for processing of high data volumes with less context switches requiring a reconfiguration of the array. Since the array architecture does not need program memory and contains internal data memory, the clock speed does not need to be tuned to external memory. For data streaming applications the speed is only limited by the maximum the chip technology and the I/O bandwidth can provide.

4. CANDIDATE ALGORITHMS

For image compression purposes, the CCSDS 122.0-B-1 algorithm was tested while for on-board object detection, both the Boundary Tensor and the Difference Method [10] were tested extensively. The difference method was never ported to the HPDP as its execution needs so many decision loops that it slows down the actual performance of the HPDP.

4.1. CCSDS 122.0-B-1

The CCSDS 122.0-B-1 is an image compression standard published in 2005 by the Consultative Committee for Space Data Systems (CCSDS), which also released other compression standards for arbitrary and hyperspectral data. It is a recommendation for

compression of two-dimensional grayscale image data and was specifically designed for on-board processing of payload data on spacecrafts. The aim of the recommendation is to provide an image compression standard that can be implemented despite the limited computational power and memory [2][6].

Two different modes for lossless and lossy compression are supported. Lossless compression is achieved by quantization and entropy coding, for lossy compression in addition image information is removed, depending on compression factors and beginning with the least important information.

The compression is based on a Discrete Wavelet Transform (DWT). The resulting sub-bands of the original image signal are then compressed by a Bit Plane Encoder (BPE), as seen in Fig. 1.

Figure 3. Functional parts of the CCSDS 122.0-B-1 recommended standard (CCSDS, 2005).

4.1.1. Results

Two compression types are possible with this implementation of the CCSDS 122.0-B- 1, quality-limited and volume-limited compression.

factor	1	1.6	2.4	3.2	4.8	6.4	8	9.6	12.8
time (s)	26.27	21.47	17.29	14.33	12.06	9.94	9.81	9.37	7.27
speedup	1.0	1.2	1.5	1.8	2.2	2.6	2.7	2.8	3.6

factor	16	19.2	25.6	32	38.4	51.2	64	76.8	102.4	128
time (s)	7.20	7.16	6.76	6.64	4.61	4.47	4.46	4.45	4.41	4.31
speedup	3.6	3.7	3.9	4.0	5.7	5.9	5.9	5.9	6.0	6.1

Figure 4.Total runtime in seconds (100 million cycles) with different compression factors (11202 pixels, 16 bpp).

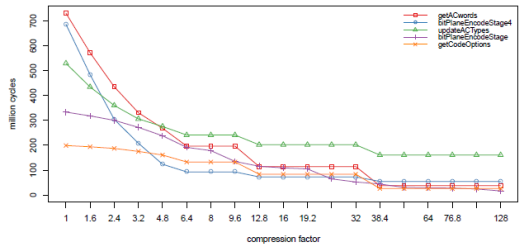

Figure 5. BPE part runtimes with increasing compression factor (11202 pixels at 16 bpp).

Although the implementation is not yet competitive, the performance gained from using the array instead of an FNC for the wavelet transform shows the potential of the HPDP for this kind of computations. The availability of in C programmable co-processors allows a developer to port programs incrementally to the dataflow array.

4.1.2. Current work

An implementation of the Discrete Wavelet Transform on the HPDP has been completed at Airbus Defence and Space. Currently, in the context of the EU H2020-funded HI-FLY project (776151), ISD focuses primarily on the efficient implementation of the bit-plane encoder. The goal is to boost performance by mapping as much as possible of the bit-plane encoder functionality on the dataflow array, a challenging task given the several levels of nested loops and branches required.

4.2. Boundary Tensor

The boundary tensor [3] is a symmetric and positive semi definite tensor with non-negative eigenvalues λ_1 and λ_2 and with the positive semi-definite symmetric tensor T of order 2.

These eigenvalues represent the variations in the pixel intensity in the direction of their orthogonal eigenvectors. In other words, the boundary tensor analyses the area around the processed pixel, and provides a local base showing the direction of the intensity variation, and the strength of the variation. As such, if λ_1 and λ_2 are both null it means that the area of the image has pixels of constant intensity. If λ_1 is strictly positive and λ_2 is null ($\lambda_1 \geq \lambda_2$) by definition) then the pixel intensity is only changing in the direction given by the eigenvector associated to λ_1: an edge is detected. If λ_1 and λ_2 are both not null, it means that pixel intensity changes in all the directions in the area of the image: a corner is detected.

To build the boundary tensor, a set of separable polar filters is applied to the image which is to be analysed. These filters are defined as the product of an angular and a radial function in order to optimize its frequency behaviour. It will also help getting the invariance to rotations. The first step in applying the filters to the image is done by convoluting the image with each filter row-like, taking into account the intensity of each pixel and the coefficients of the filter.

In practice, the filter coefficients are taken equal to zero outside a radius r. r = 4 will be the used value, as it represents a good compromise between complexity and precision.

The result of the row-like convolution is then again convoluted with the filters, but this time column-like. For the algorithm, the kernels were chosen equal to those taken in [4].

After having simulated several algorithms on the HPDP (compression, boundary tensor and several communications algorithms) Airbus can point out which algorithms are most appropriate for the chosen architecture. This architecture, commonly used in space applications, is especially performant with loops, as these are processed in parallel. However, sequential programs are executed slowly. In contrast, a typical PC-architecture (i.e. programmed in C) is slow for loops, as they cannot be executed in parallel, but very fast for sequential execution.

There is a hardware limitation that for the HPDP data types should be preferably 16-bit fixed-point arithmetic, which can be interpreted as using "short integers" instead of "reals" in C. For the on-board image processing S/W module, this fact must be considered when choosing the corresponding algorithms for feature detection and filtering. Floating point could be emulated on on-board H/W, but with high degradation of performance.

The boundary tensor can be split in the odd energy which accumulates in the step edges and in the even energy which accumulates in the roof edges.

The final step of the algorithm is to determine if a pixel corresponds to a resident space object (RSO) or not. In order to do so, it is necessary to extract from the boundary tensor a measure of the probability of the pixel being an edge. The tensor trace is actually the energy contained in the edges: it is the sum of the eigenvalues of the tensor.

4.2.1. Properties of the Boundary Tensor

Concerning the time needed to process one single image using the Boundary Tensor, the HPDP implementation needs over 0.734 s, while the same implementation of the algorithm running on a Microsemi RTG4 FPGA only needs 0.06 s, which is much faster than the requirement of 1 s for processing 2 images for the on-board image processing project. The difference in timing between the two implementations can be explained by the fact the FPGA can process all the convolutions at the same time, as well as calculating the output, without needing to write any data in an external memory, unlike the HPDP. The resulting images show that the boundary tensor algorithm can also detect streaks of different intensity.

Some small differences were observed between the results of the implementation on both architectures (HPDP and RTG4). After an analysis of the dataflow of both architectures, it has been noted that the one on the HPDP had to scale up the kernel coefficients with a multiplying factor in order to make them bigger than 1. The RTG4 architecture uses fractional length to deal with this problem. The kernels used by the HPDP simulation must make the architecture a more sensitive one, as it seems to detect more debris, but also gets more noise. As a conclusion, both hardware architectures are comparable in terms of output results; the only difference is that the HPDP version changes the kernel coefficients and the threshold value while scaling them up, which takes processing time [9].

5. IMAGE-BASED ODOMETRY

In the frame of a demonstrator for image processing Airbus created a robotic platform to simulate a stereoscopic environment where images will be processed and autonomous navigation will be explored as in [11].

In order to prove that the HPDP is an ideal candidate for optical autonomous navigation applications, a test environment, STEVE (STEreoscopic Vision Environment), was developed.

Figure 6. Side image of STEVE (STEreoscopic Vision Environment).

Based on a commercial robotic platform with 6 motorised wheels and 2 cameras for stereoscopic vision, STEVE delivers images from two red-green-blue (RGB) cameras to the desktop HPDP simulator for image filtering and processing activities.

5.1. Autonomous navigation algorithm

At the beginning of the image processing chain (see Figure 7) of the autonomous navigation, the two red green blue (RGB) cameras act as the two eyes of humans and the rover evaluates the distances with different objects from stereo images.

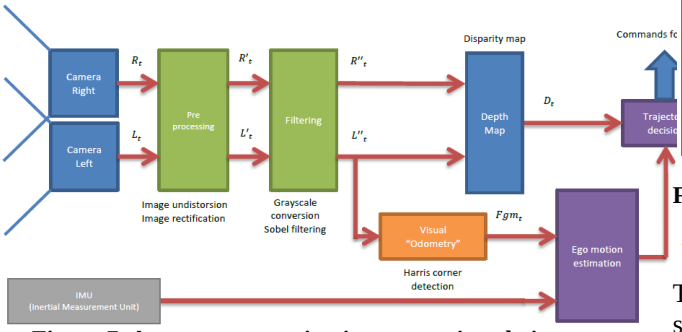

Figure 7. Autonomous navigation processing chain.

The pre-processing module is responsible of the 3D rectification and the distortion that a camera could have. The utility of the 3D reconstruction is that each image is taken regarding to a common referential that needs to be known.

Each image is processed by converting the RGB intensity from the camera into a grayscale value. At the output, pixels will carry information of the scene which varies between 0 and 255 (8 bits) instead of 24 bits (8 bits per colour). The Sobel operator is used in image processing particularly for edge detection [7]. It is based on the convolution with two kernels (see tables below) that compute the vertical and horizontal change in intensity over the image (gradient).

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Figure 8. Sobel operator masks. At the top, the x component and at the bottom the y component.

One or two images are passed either to the visual odometry or the depth map computation. To determine the motion performed of the robot, some key points of the frame are extracted and tracked over time. The Harris Corner/edge detection reuses the two directional gradients of the Sobel operation to compute the "cornerness" of a region. By shifting a window all around the frame and computing the change of intensity over eight directions, it determines the nature of the area (see Figure 9).

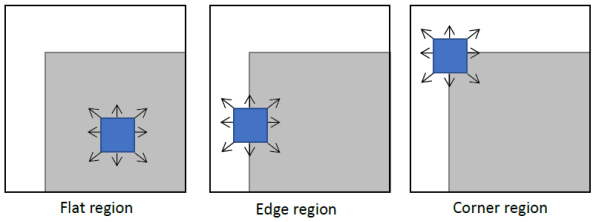

Figure 9. Representation of the Harris window on different regions.

The analysis of the stereo images is based on basic stereoscopic geometry and matching algorithm. One key element is the disparity which expresses the shift of pixel of the exact same region between the two images. This correlation criterion is calculated by comparing a fixed window from the first image with a shifting window along the same row (see Figure 10). The sum of absolute distances of intensities of each respective pixel of the windows characterizes the metric evaluation criteria. Then, it is sufficient to get the smallest criteria to get the most likely match between the two windows to evaluate the real distance.

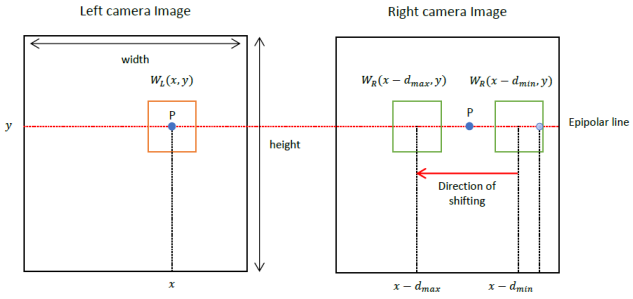

Figure 10. Schematic of the Stereo correlation process [8].

5.2. Results and performance

To see the performance of the algorithm, a reproduction of an "equivalent" Martian soil was done. For that experiment, boxes which represent obstacles were spread all around the scene as rocks of different sizes (see Figure 11). And a new pair of images (1280x720 pixels) were recorded and passed into the simulator of the HPDP.

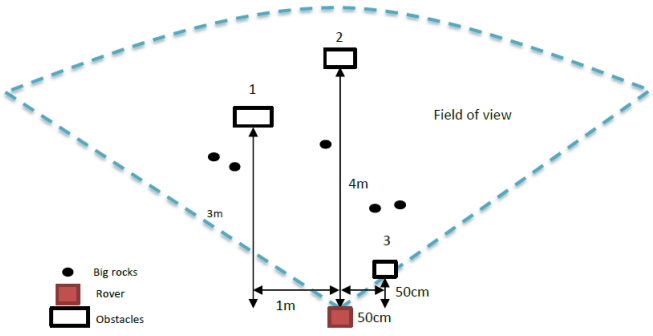

Figure 11. Top view of the scene.

The XPP simulator (XPPSIM) is simulating the behaviour of the HPDP and continuously displaying the cycle currently simulates. The different performances of each module were seized and gathered as well as the quality of the images.

		Clock cycle	Real time (in ms)	Bite rate (Mbit/s)	Throughput (Mpx/s)	SNR
HPDP boot		8129	0,03251	NA	NA	NA
RGB2grayscale	*Value*	1356528	5,426	1290,285	161,285	18,3006
Sobel Operator right	DRAM => SRAM	158989	0,6359	5504,494	688,061	18,3006
	2D convolution 3x3	1196875	4,787	4387,195	274,199	25,1414
Sobel Operator left	DRAM => SRAM	161694	0,646	5504,494	676,551	18,5048
	2D convolution 3x3	1205021	4,820	4357,537	272,346	23,9468
Harris Corner detection (on Left picture)	DRAM => SRAM	266832	1,067	6559,588	409,974	NA
	DRAM => SRAM	269173	1,076	6502,539	406,408	NA
	« Cornerness » computation	1808943	7,23	60,474	60,474	NA
Depth Map computation	DRAM => SRAM	267142	1,068	6551,976	409,498	NA
	DRAM => SRAM	269573	1,078	6492,890	405,805	NA
	Depth computation	33261394	133,045	2,405	0,1503	NA
Halt FNC+XPP Array		2506	0,010024	NA	NA	NA

Figure 12. Performance of each module inside the HPDP.

However, due to the very low speed of the HPDP simulator, the depth map computation was achieved on a degraded image (200x100 pixels) only. This means that resolution was voluntarily reduced just to increase the speed of the simulation. Figure 13 shows the different resulting images (933x469 pixels) at each stage of the image processing chain.

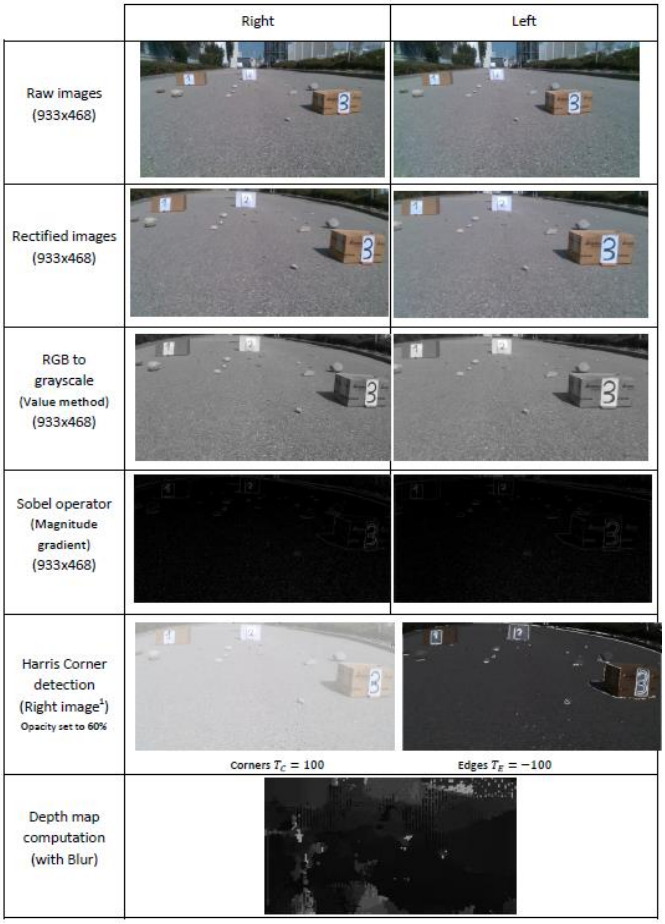

Figure 13. Result of the Right and Left images (before motion) at each different step of the algorithm. Note that images produced by the Harris algorithm were superposed with the original input to visualize the features.

Each step fulfils the requirements of each algorithm. A precision regarding the Harris images, the corner are blacks and white for the edges.

If we consider the maximum speed of the rover with a peak at around 5 cm/s, the Harris Corner detection could run, subject that the rest of the visual odometry algorithm (feature descriptor, tracking and motion estimation) are taking less than 975ms. The 3D scene detected by both cameras reveals a correct map, since each black and white pixel encodes distance information. Also, degradation from black (bottom of the picture) to white (top) shows the linear increase of the distance over the frame. However, the obstacles are hard to be spotted.

If the images were larger, such as 1024x1024 pixels, the images would have been treated in the exact same way but with more details due to a better resolution. Indeed, with more information of the scene perceived, the Harris Corner detection could spot more corners and in a more accurate way. This effect can also affect the depth map computation by giving more details of the studied region.

5.3. Conclusion

In conclusion, the objectives of the project were achieved by implementing different algorithms inside of the HPDP to assess its qualities to insure autonomous navigation processes. This architecture is already showing good performance in execution time and overall image quality.

6. CONCLUSION

The High Performance Data Processor for Space Applications developed at Airbus Defence and Space GmbH features a configurable fine grained dataflow array. The telecommunication algorithms implemented in the past meet the bandwidth performance requirements. Image processing algorithms are usually very demanding and have a high data throughput, which bring the HPDP to its limits. Even though the HPDP is in simulations not as performant as other platforms, it still offers enough processing power to fulfil many on-board functions.

7. REFERENCES

[1] PACT XPP Technologies. 2017. “Processor Licensing”. URL: http://www.pactxpp.com

[2] The Consultative Committee for Space Data Systems CCSDS. Image Data Compression, Recommended Standard CCSDS 122.0-B-1, Blue Book, 2005.

[3] Köthe, U. 2003. “Integrated Edge and Junction Detection with the Boundary Tensor”. Proceedings of the 9th IEEE International Conference on Computer Vision.

[4] Suárez Trujillo, D.A. 2015. “Design and Implementation of a feature detection algorithm for space debris detection on a High Performance Data Processor (HPDP)”. Master Thesis. Airbus DS GmbH, unpublished.

[5] XPP-III reference Manual – XPP Dataflow Array. PACT XPP Technologies AG, 2009.

[6] The Consultative Committee for Space Data Systems CCSDS. Image Data Compression, Informational Report CCSDS 120.1-G-1, Green Book, 2007.

[7] C. Harris and M. Stephens, "A combined corner and edge detector," in Proceedings of the 4th Alvey Vision Conference, Manchester, 1988.

[8] M. Winter, C. Barclay, V. Pereira, R. Lancaster, M. Caceres, K. McManamon, B. Nye, N. Silva, D. Lachat and M. Campana, "ExoMars rover vehicle : detailed description of the GNC system," in ASTRA Conference, Noordwijk, 2015.

[9] Vives, G., Helfers, T., Biersack, F., Linssen, S., Utzmann, Dr. J., Vananti, A. 2018 “The use of different architectures and streak observations algorithms to detect space debris”. ESA On-Board Processing and Data Compression Conference (OBDPC 2018). Matera, Italy, 2018.

[10] Métrailler, L., Vananti, A., Schildknecht, T., Pittet, J-N., Utzmann, J., Flohrer, T. 2017. „The Difference Method: A simple and effective on-board algorithm for space debris detection “. 68th International Astronautical Congress. Adelaide, Australia.

[11] M. Winter, C. Barclay, V. Pereira, R. Lancaster, M. Caceres, K. McManamon, B. Nye, N. Silva, D. Lachat and M. Campana, "ExoMars rover vehicle : detailed description of the GNC system," in ASTRA Conference, Noordwijk, 2015.