CONTRIBUTION OF A POWERFUL IMAGE PROCESSING UNIT TO THE AUTONOMY OF ROBOTIC INSTRUMENTS

T. Wuergler [(1)], D. Kraehenbuehl [(1)], S. Beauvivre [(1)], P. Plancke [(2)]

[(1)] Micro-cameras & Space Exploration SA, Puits-Godet 10a, 2000 Neuchâtel, Switzerland, thomas.wurgler@microcameras.ch
[(2)] European Space Agency, Keplerlaan 1, 2200 AG Noordwijk, The Netherlands

ABSTRACT

Exploration missions have stringent constraints in terms of mass and power, while requiring more and more advanced processing to be performed locally. Indeed, local data processing provides autonomy to rovers and instruments and reduces the amount of data having to be downloaded to Earth, greatly increasing the science/technical return.

The Imaging Processing Unit (IPU) under development for ESA aims to be used to control imagers or other sensors and to perform data processing as well as processing for the control and setting up of the instrument in support of the autonomy of the system. Its significant processing capability allows reducing the main processor's activity for instrument related tasks. Its lightness and compactness is a valuable asset for tight payload mass & volume requirements. The IPU is designed to operate in harsh environment and especially at very low temperature without active thermal control, which is an advantage for various space missions, like Mars missions were operation at very low temperature is likely to occur.

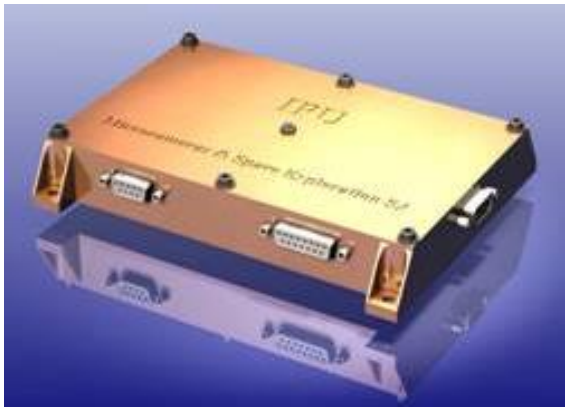

Figure 1. Example of an IPU external design

The new camera developments, based on large sensors, together with the need for more and more complex processing (compression, image processing, feature extraction, etc.) make the IPU a solution of choice for highly demanding systems. Its reconfigurable functionality is an additional benefit since it improves the versatility of running diverse applications and is well suited for onboard autonomous systems. The use of the IPU relieves workload from the onboard computer, improve the autonomy of the system and the overall efficiency of the mission.

Typical applications for the IPU include:

- Image compression;
- Distortion correction;
- On-board calibration;
- Integration time optimization;
- Image registration;
- Stereo reconstruction;
- Depth of field extension;
- High dynamic range imaging;
- Super-resolution;
- Image analysis;
- Support to navigation;
- Co-processor to the OBC;
- Monitoring.

These functions can be combined according to the mission's needs and to the IPU capabilities to propose a set of functionalities that can even be updated during the mission.

1. INTRODUCTION

The paper presents the architecture of the IPU and its potential applications in support to the autonomy of robotic instruments.

The instrument under development is aimed to be versatile enough to be able to work as an on-board computer (OBC) assistant as well as a specific image processing unit.

The selected most suited architecture for this purpose consists of the following components:

- A communication FPGA
- A data processing FPGA
- A "system on chip" (SOC) processor embedded in the data processing FPGA
- A mass data storage memory
- A processor code memory

Both FPGAs are built in anti-fuse non-reprogrammable technology. It allows a good immunity to radiation effects. The versatility of the system is guaranteed by the re-programmability of the SOC processor code.
Resources consuming functions are also programmed into FPGA, to parallelize operations and improve speed.

2. IMAGING PROCESSING UNIT DESCRIPTION

The IPU is able to communicate with a host component (e.g. on-board computer). It can receive, store and process images from two cameras.

The unit consists of a communication block and a data processing block. Two FPGA are used for both block realization. A mass memory allows the storage of telemetry data. The IPU software is executed by a SOC processor programmed into the data processing FPGA. The software code is stored into an external non-volatile memory.

The selected Imaging Processing Unit architecture can be represented by 5 main components, as shown in the following diagram:

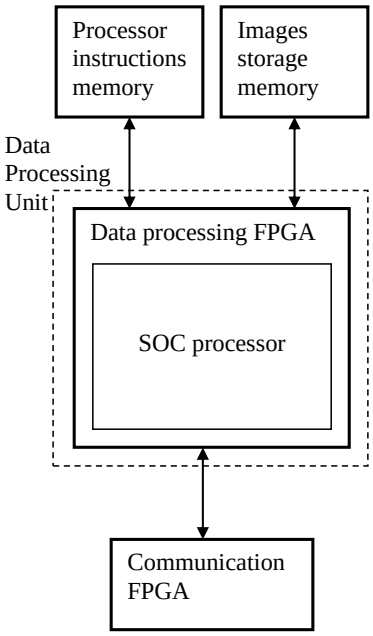

Figure 2. IPU global architecture

The data processing part of the IPU consists of a FPGA where a SOC processor is programmed. This part of the IPU is connected to a FPGA dedicated for the communication. It is also connected to external memories, for the SOC processor instructions and for science data storage.

The selected data processing unit includes a non-reprogrammable FPGA and a SOC processor programmed inside. The FPGA contains the processor core, with the usual blocks needed by it, like cache memory, MMU, etc.

The following diagram shows the data processing FPGA programmed functions, in the case of an image processing typical use (compression, distortion correction, integration time optimization, dark calibration).

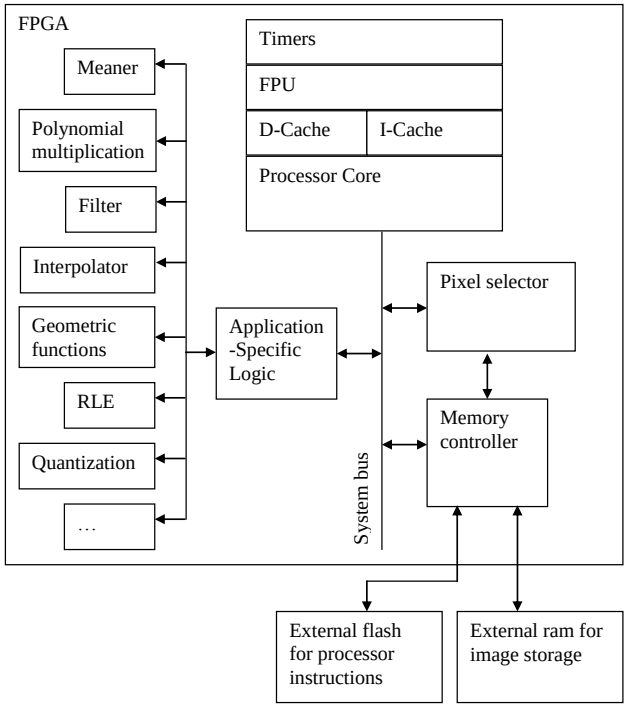

Figure 3. Data processing unit global scheme

The FPGA logical programmed blocks can be defined according to the mission purpose. The previous figure shows image processing typically using blocks. But if the IPU is to be used for positioning, other logical blocks should be programmed.

The advantages of this philosophy are:

- Speed improvement: basic common used signal processing functions (e.g. filtering) are slow when executed by a processor. Programming these functions into the FPGA allows the parallelization of the operations, and a pipelining processing of the data, improving considerably the speed. The processor is used to configure the logical blocks and to realize functions which are less time critical.
- Good updating capability: The functions blocks programmed into the FPGA are selected to be commonly used by standard algorithms. They are tuneable enough to be adapted to every

situation. For example, the filter block can be used for wavelet decomposition as well as for contrast detection purpose. Although the FPGA code cannot be changed, the processor code can be updated every time. It means that new signal processing algorithms can be added to the system and can use the FPGA logic blocks.

To truely benefit from this philosophy, the logical blocks programmed in the FPGA have to be carefully defined according to the mission. Only the functions that are commonly used by potential algorithms have to be programmed in the FPGA. To improve speed, the functions which have to be applied on large amount of data have to be programmed in a pipelined mode in the FPGA.

3. DATA PROCESSING UNIT SPECIFICATIONS

The data processing unit under development is expected to respect the following requirements:

- Immunity to radiations (level depends on the mission)
- Low power (max 2.5W)
- Miniaturisation (smallest packages, component number minimization)
- High data rate links to sensors and host spacecraft
- Embeds a mass storage memory
- Embeds high efficiency data processing algorithms
- Re-programmable capability.

To evaluate the IPU performances needs, a typical mission is considered where the IPU's tasks are the following:

- Store 8 pictures of 2048x2048 with 16 bits pixels
- Process images from two cameras (2048x2048 16bits) at 1Hz frequency, with the following algorithms:
    - High efficiency lossy or lossless compression (wavelet based)
    - Distortion correction
- Automatic optimization processes:
    - Integration time optimization
    - Dark calibration and removal.

According to the established needs, the most suited technologies have been selected to compose an IPU with the best processing capabilities.

The following table shows the calculated performances and specifications of the developed unit:

Table 1. Data processing unit specifications

Specification	**Value**
Radiation and fault tolerance. Two possible variants:	
- Variant 1	100 krads
- Variant 2	300 krads
Sensors data links:	
- Number of links	2 physical (numerous addressable peripherals)
- Links type	SpaceWire
- Links speed	100 Mbits/s
Interface with host spacecraft:	
- Link type	SpaceWire
- Link speed	100 Mbits/s
Storage memory:	
- Memory type	SRAM
- Memory size	512 Mbits
Processor code memory:	
- Memory type	EEPROM
- Memory size	1 Mbits
Global clock frequency	50MHz or 100MHz
Data processing FPGA:	
- technology	Anti-fuse
- capacity	~60k cells
Communication FPGA:	
- technology	Anti-fuse
- capacity	~20k cells
Power supply:	
- Voltage	+5V
- Power consumption	< 2.5W

This configuration allows the IPU to fulfil the requirements. There are indeed enough cells in the FPGA to program the needed processing functions, and the SOC processor. The SRAM of 512 Mbits allows the storage of 8 pictures of 2048x2048 with 16bits pixels.

The system is also capable of processing two pictures per second (compression and distortion correction) with a clock frequency of 100MHz. If a clock of 50MHz is preferred, half of this performance is reached.

The versatility of the IPU is reachable by the reprogramming capability of the processor code memory EEPROM. Indeed, despite the FPGA anti-fuse technology, the SOC processor software can be modified by updating the EEPROM.

4. POTENTIAL APPLICATIONS IN SUPPORT TO THE ROBOTIC INSTRUMENTS' AUTONOMY

Due to its good versatility, the use of the IPU is not

limited to the image processing functions described above. It can indeed be really useful in support to robotic applications with resources-demanding computing.

The identified well-suited robotic applications include:

- Path planning;
- Positioning, rendezvous;
- Image analysis (e.g. features detection);
- Sensors data processing.

4.1. Path planning

Path planning can be done from a three dimensional representation of the spatial environment. It needs complex and time consuming algorithms:

- Image registration;
- Stereo reconstruction;
- Features detection.

These algorithms are complex and necessitate resources and computing power. Implementing all these operations in an IPU is not compatible with the goal of a miniaturized and low-power unit.

However, the IPU can be really helpful to the onboard computer, acting as co-processor, by preparing the pictures from stereoscopic cameras (distortion correction, image registration, etc.) and let the OBC plan the path itself.

4.2. Positioning, rendezvous

Specific processing linked to positioning, rendezvous, etc., are also a potential application for the IPU, which could include algorithms needed by a positioning system based on the recognition of a target in an image produced by one or several camera(s).

The IPU is also able to manage data from several positioning sensors in order to convert raw signals to machine-readable position values.

4.3. Image analysis

Local image analysis is an interesting feature that can be applied for several missions. To save time and limit data exchange to the Earth, embedding image analysis capability on the spacecraft is increasingly needed.

The following identified algorithms could be interesting in many cases:

- Dynamic range enhancement (HDR): combine several differently exposed pictures together to improve the contrast in every region;
- Depth of field extension: combine several differently focused pictures together to obtain a fully focused picture;
- Features detection: detect sharp and recognizable forms in images in order to coincide them together;
- Exposition measurement: check if an image is well exposed before sending it to the Earth.

Other algorithms can be foreseen for the IPU.

4.4. Sensors data processing

In robotic applications, many sensors are used for positioning or detection purposes. The signals from the sensor outputs have to be pre-proceeded before using them as reality representative values. This step can consume a lot of resources according to the sensor type.

5. APPLICATION EXAMPLE

To illustrate the ability of the IPU to embed several user defined algorithms according to the mission, a typical example is described hereafter.

In this example, the IPU is used for the following image processing functions:

- Image compression;
- Distortion correction;
- Dark calibration;
- Integration time optimization.

Each algorithm has to be programmed in the IPU data processing part. The algorithms can be generally decomposed into several basic or complex functions. The basic functions of each algorithm are programmed as logical blocks in the FPGA. The rest of the algorithms are programmed as software code executable by the SOC processor.

5.1. Compression

A wavelet based image compression algorithm has been selected, because of its high lossy and lossless compression efficiency. This algorithm uses the following methods:

- Wavelet transform;
- Vector and scalar quantization;
- Progressive image coding;
- Adaptive run-length/Rice (RLR) compression.

Four recurrent basic operations have been identified in the algorithm. These operations are realized several times per pixel. To process a several mega pixels picture, the following operations have to be done millions times:

- Filtering: the wavelet decomposition of an image consists of low-pass and high-pass filters applied both vertically and horizontally on pictures;

- Quantization: a lossy compression begins with the quantization of the data before encoding them;

- Run length encoding (RLE): a RLE based method is used for the compression step;
- Pixel selection: before being processed, the pixels have to be loaded from storage memory. To improve speed, this function reads the pixels in a pipelined mode.

Compressing a 2048x2048x16bits picture with this algorithm requires 397 MOP for a standard processor.

5.2. Distortion correction

The correction is done from an estimated distortion model. The latter is defined after the calibration of the camera's optical defaults. This model is an approximation by a polynomial equation.

To correct the distortion on a picture, the pixels have to be shifted according to the model equation. Interpolation is then needed because of the non-linear shift of the pixels.

Four recurrent basic operations have been identified in this algorithm. These operations are realized one or more times per pixels:

- Quadratic sum: to compute the distance from distortion centre to proceeded pixel;
- Polynomial multiplication: to calculate the pixel shifting value from the polynomial model;
- Interpolation: bilinear interpolation block;
- Pixel selection: before being processed, the pixels have to be loaded from storage memory. To improve speed, this function reads the pixels in a pipelined mode

Undistorting a 2048x2048x16bits picture with this algorithm requires 71 MOP for a standard processor.

5.3. Integration time optimization

The optimization is done by acquiring a picture with an arbitrary defined integration time. The mean value and the number of saturated pixels are extracted from the picture's histogram. These values are needed to compute a new optimized integration time before acquiring a second picture.

Two recurrent basic operations have been identified in this algorithm. These operations are realized one or more times per pixels:

- Mean calculation: to compute the mean value of the histogram;
- Pixel selection: before being processed, the pixels have to be loaded from storage memory. To improve speed, this function reads the pixels in a pipelined mode

Optimizing the integration time for a 2048x2048x16bits picture with this algorithm requires 28 MOP for a standard processor.

5.4. Dark calibration

The dark current can be characterised by acquiring pictures with two different integration times. The temporal noise has to be removed from pictures, in order to measure only the non-uniformity generated by the dark signal on the pictures.

For each pixel, a slope and an offset can be found, characterizing the dark level vs. the integration time. Knowing the properties of each pixel, the acquired pictures can be corrected according to the programmed integration time.

Two recurrent basic operations have been identified in this algorithm. These operations are realized one or more times per pixels:

- Mean calculation: to compute the temporal mean on the acquired pictures;
- Pixel selection: before being processed, the pixels have to be loaded from storage memory. To improve speed, this function reads the pixels in a pipelined mode.

Characterising the dark of a 2048x2048x16bits picture with this algorithm requires 69 MOP for a standard processor.

5.5. Performances predictions

The algorithms described in the example use commonly the following function blocks:

- Pixel selection;
- Meaning;
- Polynomial multiplication;
- Filtering;
- Interpolation;
- Run length encoding;
- Quantization;
- Geometric functions.

All these blocks are programmed in the FPGA. The "Polynomial sum" function is included in a more global block containing several geometric functions. A supplementary logical block is also needed to configure and control all the listed blocks. The latter, called the main control block, realizes the interface between the SOC processor bus and the logical specific blocks.

The resources needed in the FPGA by the different blocks (number of logical registers and combinatorial cells) are listed in the following table:

Table 2. FPGA resources needed for blocks

Function	Registers	Combinatorial
Pixel selector	366	732
Meaner	1144	2288
Polynomial multiplication	1000	2000
Filter	1132	2264
Interpolator	1050	2100
Geometric functions	324	648
Run length encoding	100	200
Quantization	100	200
Main control	4800	9600
Total	**10'016**	**20'032**

Around 10k registers and 20k combinatorial cells are needed in the data processing FPGA for the logical blocks. A total of 30k cells are required in the FPGA.
The SOC processor needs around 8k cells of the signal processing FPGA.

The following diagram shows the space repartition of the data processing FPGA:

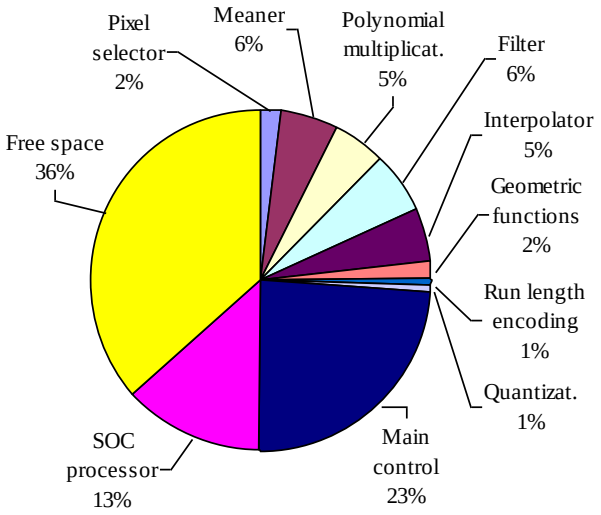

Figure 4. Data processing FPGA space repartition

The diagram shows that with this configuration, only 64% of the signal processing FPGA is used. Other logical blocks can be added to the FPGA firmware, to allow the use of other algorithms.
The SOC processor and main control block use 36% of the full FPGA space. These parts stay the same for every mission. It means that twice of the actual programmed logical functions can be included in the signal processing FPGA.

The organization of the data processing FPGA allows a significant efficiency improvement compared to a simple processor. The operations can indeed be executed for every algorithm in pipelined mode.

The following table shows for every described algorithm the number of operations needed, the number of clock cycles to compute a 2048x2048x16 bits picture and the efficiency in MOPS (Mega Operation Per Second):

Table 3. Selected algorithms performances

Algorithm	Operations number	Clk cycles number	Efficiency @ 50MHz
Compression	397 MOP	28 M	700 MOPS
Distortion correction	71 MOP	21 M	170 MOPS
Integr. Time optimization	14 MOP	4 M	175 MOPS
Dark calibration	69 MOP	69 M	50 MOPS

The improvement of the efficiency by pipelining the operation into specific logical blocks is clearly visible. For compression, where the number of operation is especially important, the gain brought by this method is high.

The efficiency of the "dark calibration" algorithm is limited by the read of the data from the storage memory.

According to these performances specifications, the IPU is capable of processing distortion correction and compression at a rate of 1 frame per second (2048x2048x16bits picture), with a 50MHz global clock.

6. USE OF THE IPU FOR SPACE MISSIONS

The philosophy of the IPU is to be useable for a large range of space missions without modifying the hardware. Two parts of the IPU can be adapted with respect to the mission's requirements:

- The firmware: the FPGA code is programmed once on Earth and cannot be updated. The specific logical blocks to program the FPGA are selected according to the needed algorithms
- The software: the SOC processor code can be updated at any time, even during the mission. It contains the functions that use the logical blocks programmed in the FPGA

When developing an application for the IPU, the algorithms are programmed in C language for the processor. When needed by a function, a logical VHDL block can be added in the FPGA firmware code. This block is used as a specific function by the software. The idea is to regroup all the primary VHDL blocks in a dedicated library, to allow their reuse for other projects.

The following scheme shows the principle of selecting the FPGA logical blocks according to the needs of the required algorithms:

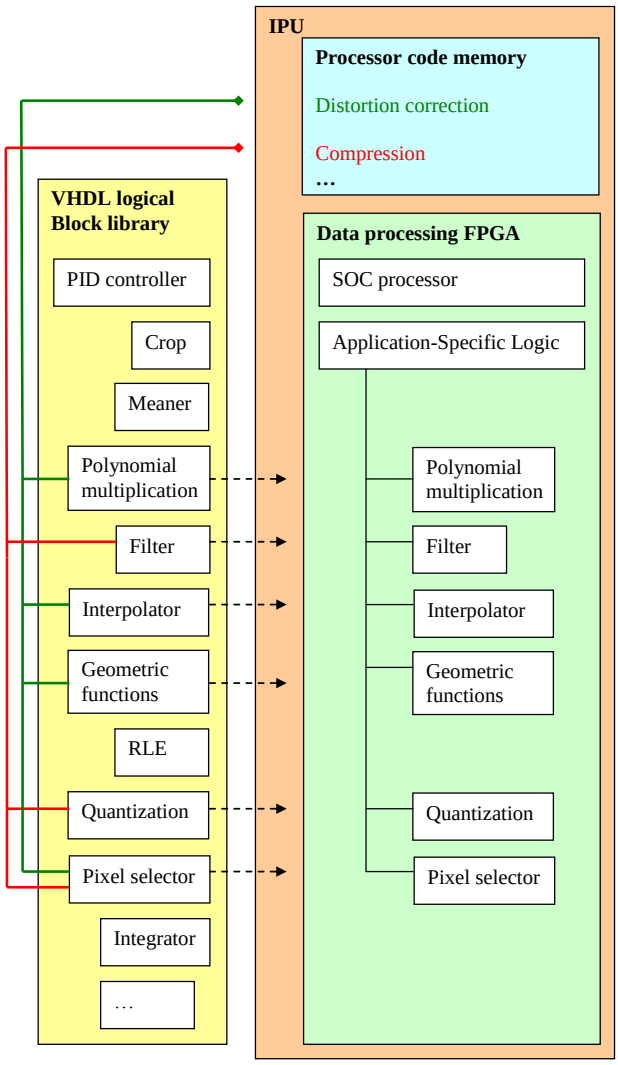

Figure 5: Selection of the FPGA logical blocks

A VHDL library of primary logical functions is progressively established. When defining the algorithms to be programmed in the IPU, the logical VHDL blocks to use are selected and added to the FPGA firmware

7. CONCLUSION

The IPU has not the ambition to replace any on-board computer. The goal of this element is to offer a powerful support to the spacecraft electronics for specific low level applications.
The processing unit is indeed designed to be compliant with low mass and low power requirements. This instrument is however capable of processing a substantial amount of data in a restricted time.

The IPU is ideally suited when connected between sensors and OBC. The acquired raw data from sensors can indeed be pre-processed by the IPU. The latter can extract the relevant part of the read data and convert it to values which are usable by the OBC. The IPU can be used for example to convert raw pictures from stereoscopic cameras to a 3D representation plan. Another typical application could be the use of the IPU to transform position data from several sensors (accelerometer, gyroscope, magnetometer, GPS, etc.) into absolute position values. In addition, high efficiency compression algorithms can be implemented in the IPU to decrease considerably the amount of data transferred to Earth.

The described IPU is able to bring a valuable contribution to the autonomy of robotic space instruments. Adding a powerful data processing unit to a space instrument relieves the workload of the OBC. The latter can then devote its power to high level applications, increasing considerably the computation speed. The amount of data having to be downloaded to Earth can be moreover considerably reduced.

The low size and power consumption of the IPU make it easily embeddable. Its processor reprogrammability and FPGA tuneable logical blocks guaranty a good versatility of the imaging processing unit.