Multi-purpose simulator for Plato mission

SpaceWire mission and applications, Short Paper

Rafael Corsi Ferrão, Sergio Ribeiro Augusto, Cássio Berni, Franklin Ronald Ferreira dos Santos e Vanderlei Cunha Parro
Critical embedded systems group
IMT
São Caetano do Sul, SP, Brazil
corsiferrao@gmail.com

Saulo Finco
Citar project
CTI
Campinas, SP, Brazil
saulo.finco@cti.gov.br

Philippe Plasson and Loïc Gueguen
LESIA
Paris Observatory
Meudon, France
Philippe.plasson@obspm.br

Gisbert Peter
DLR
Berlin, Germany
Gisbert.Peter@dlr.de

Manfred Steller
IWF
Graz, Austria
Manfred.Steller@oeaw.ac.at

***Abstract*— This paper presents hardware and software solution for simulation of a group of cameras used by the PLATO satellite. The simulator can be configured either to work with the scientific processing unit or as the complementary loop attitude control processing unit. Configuration and monitoring can be performed remotely, which caters to cooperatively work and guarantees traceability for quality purposes. Because the system operates with database and configurable architecture, its performance can be modified to operate as standard generation element (CCSDS, e.g.) traveling via SpaceWire. Eight SpaceWire links for feeding processing system compose the simulator. The links may route data in real time rate configurable to 200Mbits/s, with representative images, using the RMAP protocol, it can run continuously up to two days of satellite operation. The information database is stored in two solid-state drives with 500 GBytes capacity each one. Access for configuration and monitoring are made using TCP-IP protocol. Each simulator has a unique ID and is automatically recognized when connected to the Ethernet network. The software layer has graphical interface compatible but offers component for integration with other EGSE systems. The system has own housekeeping, which enables diagnosis operation and viewing by the operator. The system will be used by European groups: LESIA (France), DLR (Germany) and IWF (Austria).**

***Index Terms*— SpaceWire, RMAP, Plato mission.**

I. Introduction

The main goal of this project is to have a realistic hardware simulation of the image acquisition system of the Plato satellite. Part of this architecture can be analyzed in Fig. 1. Each data processing unit (DPU) receives from the electronic front end (FEE) 4 SpaceWire (SpW) [3] links running at 100 Mbits/s. A similar system is used in the attitude control system, just changing the number of links to 8 and reducing the data amount to a half CCD. The simulator described here involves the CCDs, with dynamic images and the FEE. More details about Plato architecture is available in the ESA website [1].

The simulator can be subdivided in three subsystems:

- Electronic main board (EMB).
- Resident software (RSW).
- Supervisor software (SSW).

The EMB has the capability to work with eight SpW links and to generate all RMAP [2] commands necessary to transmit the image from the simulator to the DPU and to receive RMAP commands (Write and Read) from the DPU.

The RSW software controls EMB where the images are stored, running on a dedicated PC and communicates with the EMB by a S-ATA protocols. The set RSW+EMB forms the Simulator (SIMUCAM).

Supervisor software (SSW) is proposed to control a set of simulators and communicates with the RSW by a TCP/IP connection allowing debugging and configure several simulators. The simulator runs as stand-alone application to avoid timing glitches and the Human-Machine-Interface (HMI) will be use only to control the simulations.

The SIMUCAM is able to work with both: normal DPU (N-DPU) used for scientific processing and fast DPU (F-DPU), used for the attitude control loop. They work in a similar way besides the integration image time and the number of SpW interfaces.

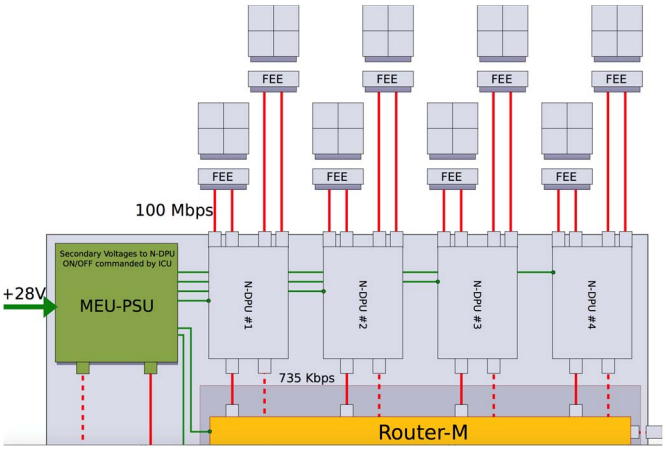

Fig. 1. Plato electrical architecture for science.

A. N-FEE description operating in normal mode

Each N-FEE is in charge of four CCDs. Each CCD is acquired sequentially, the period of activity for normal camera is 25 seconds, The full process (4 CCD) takes 25 seconds. During these 25s, a full image from one CCD is delivered to N-DPU each 6.25s. The full-image transfer time from N-FEE to N-DPU should take considering the maximum optimization value of the time transfer less than 3.3 seconds. Two SpaceWire link is implemented as communication protocol between N-DPU and N-FEE.

The CCDs are divided vertically in two parts (Left and Right), and each part is transmitted by one of the two links. The CCDs transfer are swapped each 6.25 seconds.

For safety reason the SpaceWire channel utilization shall not exceed 80 %, that means the data rate averaged over the transfer duration, including the SpaceWire overhead, shall not exceed 80 Mbps. N-FEE is connected to a N-DPU by two SpaceWire links, each link is responsible for transfer half of CCD each CCD at time. The transfers occurs by encapsulating half line of one CCD (Left and Right) on a RMAP write command (FEE to DPU) and send it by one of the links, as shown on Fig.2 The address of each write commands is incremental and should be restarted at a new image transfer (25 s).

Each write command (IMAGE) have at the begining of the data a top sync counter that is incremented at a new received synchronism, and at the end of the data some *prescan* pixels. The *prescan* pixels can be part of the image (loaded into the simulator) but the top sync counter must be added by the MEB on real time. The top sync (6.25s) is transmitted to de DPU by a *timecode* command; this is the only way that DPU can access this signal. It will be sent by both SpW links to guarantee robustness. Housekeeping is sent by the N-FEE to the N-DPU at the end of a full image transmission by a RMAP write command (N-FEE to N-DPU). The HK can also be accessed asynchronous at any operation mode by a read command from the DPU (N-DPU to N-FEE).

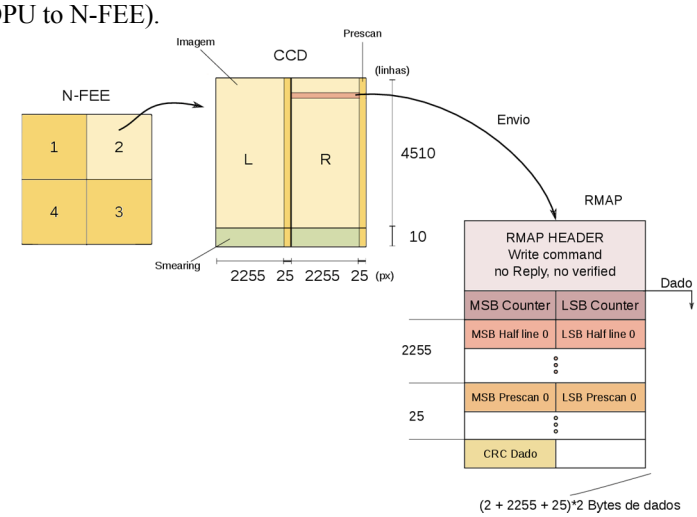

Fig. 2. RMAP data diagram.

The N-FEE has the following functional modes:

- Operational mode: the CCDs are read with the synchronization signals. Data packets including image and Housekeeping are sent to N-DPU.
- Stand-By mode: no sequencing signals and no data packet sent. Bias to CCD at nominal value, only Housekeeping data are sent on request from the MEU.
- Integration mode: during AIT specific read must be done to compensate the huge dark current generated by the CCD, the use of the Dump-Drain or a different exposure time management must be implemented. In that mode, the N-FEE may function without synchronization signals from the *SylBox*.
- Test Mode: the N-FEE sends a pattern to the MEU. This mode is mainly used during AIT step with the MEU without the FPA. This pattern will be defined latter.

B. F-FEE description operating in fast mode

As the N-FEE each F-FEE is in charge of four CCD. Each CCD is acquired sequentially, the period of activity for the fast cameras is 2.5 seconds, The full process (4 CCD) takes 2.5 seconds. During these 2.5, all full images are delivered to the F-DPU each 1.5s. Eight SpaceWire link is implemented to transfer the images to de DPU. The CCDs are divided vertically in two parts (Left and Right), and each part is transmitted by one of the eight links. The CCDs transfer concurrently (all at the same time). F-FEE is connected to a F-DPU by eight SpaceWire links, each link is responsible for transfer half of CCD, this process occurs concurrently. The transfers occurs by encapsulating half line of one CCD on a RMAP write command (FEE to DPU) and

send it by one link, as shown on Fig. 2 The address of each write commands is incremental and should be restarted at a new image transfer (2.5 s) this means that the images are saved on the same memory address at the DPU side.

C. ICU

There are 2 ICU channels, which work, in cold redundancy. The ICU is responsible for the management of the payload, the communication with the Service Module (SVM), the compression of scientific data before transmitting them as telemetry to the SVM. Two SpaceWire routers (RU) a Data Compression Unit (RDCU), a memory unity (MU) and a processor unit (PU) compose the ICU.

Each ICU Router Unit (RU-A and RU-B) is connected to

- 4 MEU (Router Unit A is connected to MEU routers A and Router Unit B to MEU routers B).
- 2 F-DPU
- 4 N-AEU
- 2 F-AEU

Other functional units of ICU throughout 4 additional SpW links:

- ICU internal Memory Unit (MU)
- Processing Unit A (PU-A)
- Processing Unit B (PU-B)
- The other Router Unit to connect together the two SpW network.

The RDCU will collect the data from the twelve front end DPUs and compress the data. Finally, the ICU generates the telemetry packets to be sent to ground.

II. Simulator description

The global vision of the Simulator is synthesized in the Fig. 3 where we can see the components that compose the project. In order to understand the MEB role in overall process it is important to establish the main interaction during a nominal operation. The MEB is responsible to transmit data (ack. Images) from its internal SSD and DDR memory to the SpaceWire links, encapsulating it on a RMAP protocol. All transfers are started by a synchronism pulse (sync) that can be external our internal. With the sync detected, the processor core starts the tasks responsible to load the images stored on the DDR2 memory to the FIFOs on each SpaceWire/RMAP peripheral. Eight different DMA controllers, each one associated to a specific link, perform this transfer.

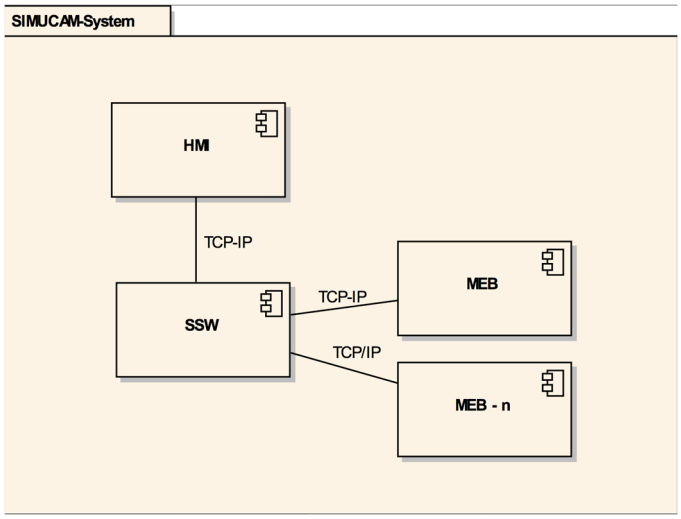

Fig. 3. SIMUCAM system and its components.

In order to understand the MEB role in overall process it is important to establish the main interaction during a nominal operation. The MEB is responsible to transmit data (ack. Images) from its internal SSD and DDR memory to the SpaceWire links, encapsulating it on a RMAP protocol. All transfers are started by a synchronism pulse (sync) that can be external our internal. With the sync detected, the processor core starts the tasks responsible to load the images stored on the DDR2 memory to the FIFOs on each SpaceWire/RMAP peripheral. Eight different DMA controllers, each one associated to a specific link, perform this transfer.

The DMA core supports a buffer of 512 transfers each transfers cares 9020 bytes. At the beginning of transfer the transfer command buffer is full filled by the task that controls the specific link. During the transmission of the image, the DMA core generates a interruption at every DMA transfer executed, this causes the link task to fill again the command buffer.

The 8 DMA cores share the DDR2 memory, the round robin scheduling is used to give access to DDR2, this scheduling is performed, not on software level, but at hardware level (bus controller).

The SpaceWire/RMAP FIFO is of 32 bits wide (to be compatible with the Avalon bus) and has a depth of 512. This peripheral is responsible to fragment the data into RMAP packets and transmit it to the SpaceWire link. No software innervation is necessary once the peripheral is configured, its perform the auto increment of the destination address on the RMAP command, inserts the data CRC, inserts (if configured) extra data on the data (ID info, top counter).

All this is done to free the uC from critical tasks the MEB avoiding glitch on the data transfer. Figure 4 is a resume from the proposed architecture. The SSD are used to stored images, this images must be loaded by the Ethernet TCP/IP connection

that communicates with the SSW component. These images are than cached on the DDR2 memories, and transfer by each SpaceWire link respecting all specification imposed for the N-FEE and F-FEE.

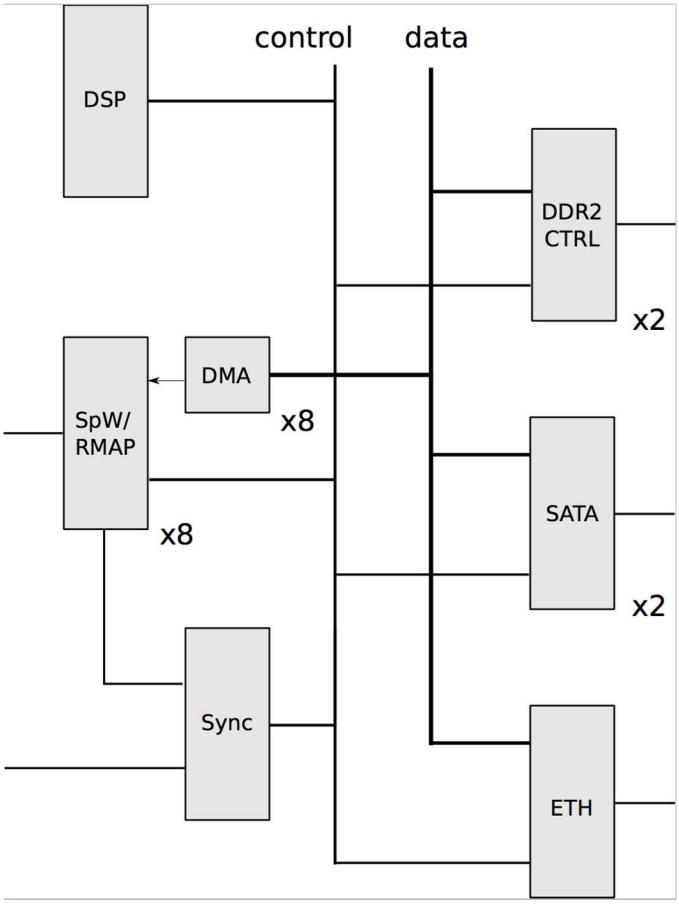

Fig. 4. The simulator architecture.

The SpW/RMAP is peripheral developed for optimize the data transfer from de memory to the SpaceWire, which can be analyzed in Fig. 5, its six main parts can be listed:

- RMAP: generates the RMAP head and data CRC; its operation is controlled by the REG.
- FIFO: used to cache de data of the RMAP (image), is a dual port with a reading clock of 200 MHz (8b) and writing port of 100 MHz (32b).
- DELAY: block that generates the sample time (4MHz) to emulate the A/D .
- REG: register with all RMAP configuration.
- TC: Time code.
- SpW: SpaceWire core.

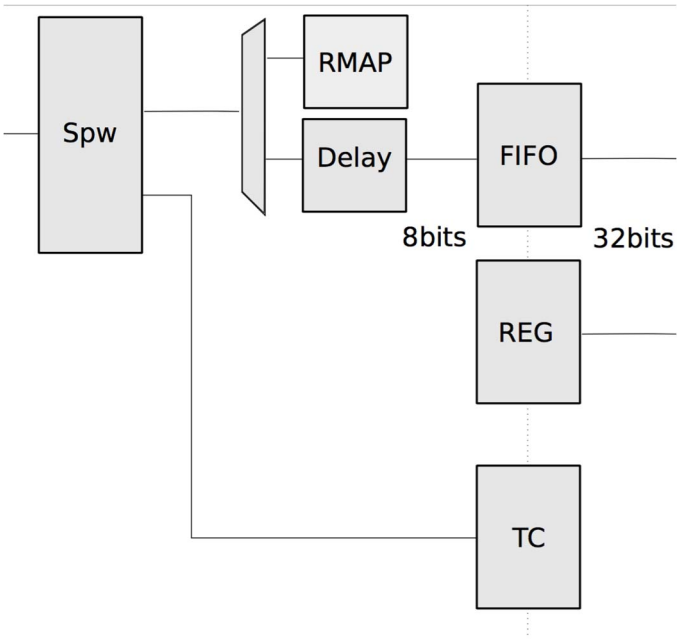

Fig. 5. The SpW/RMPA peripheral architecture.

ACKNOWLEDGMENT

The authors thanks to: Fapesp grant 2008/57866-1; CNPq grant 574004/2008-4; FINEP and Mauá Institute of Technology.

REFERENCES

[1] ESTEC-ESA. Plato payload definition document. ”http://sci.esa.int/ science e/www/object/index.cfm?fobjectid=42793#, 2008. •

[2] ESTEC-ESA. Remote memory access protocol remote acess memory protocol. http://www.ecss.nl/forums/ecss/dispatch.cgi/standards/showFile/100770/d2010020912165 E-ST-50-52C(5February2010).pdf, 2010. •

[3] ESTEC-ESA. Spacewire protocol identification. http://www.ecss.nl/forums/ecss/dispatch.cgi/standards/showFile/100769/d2010020912161 E-ST-50-51C(5February2010).pdf, 2010. •