The software architecture of the camera for the ASTRI SST-2M prototype for the Cherenkov Telescope Array

Sangiorgi, Pierluca, Capalbi, Milvia, Gimenes, Renato, La Rosa, Giovanni, Russo, Francesco, et al.

Pierluca Sangiorgi, Milvia Capalbi, Renato Gimenes, Giovanni La Rosa, Francesco Russo, Alberto Segreto, Giuseppe Sottile, Osvaldo Catalano, "The software architecture of the camera for the ASTRI SST-2M prototype for the Cherenkov Telescope Array," Proc. SPIE 9913, Software and Cyberinfrastructure for Astronomy IV, 99133T (26 July 2016); doi: 10.1117/12.2231647

SPIE.

Event: SPIE Astronomical Telescopes + Instrumentation, 2016, Edinburgh, United Kingdom

The software architecture of the Camera for the ASTRI SST-2M prototype for the Cherenkov Telescope Array

Pierluca Sangiorgi*[a], Milvia Capalbi[a], Renato Gimenes[a,b], Giovanni La Rosa[a], Francesco Russo[a], Alberto Segreto[a], Giuseppe Sottile[a], Osvaldo Catalano[a], for the ASTRI Collaboration[c], and CTA Consortium[d]

[a]INAF - Istituto di Astrofisica Spaziale e Fisica Cosmica di Palermo, Via U. La Malfa 153, 90146, Palermo, Italy
[b]Instituto de Astronomia, Geofísica e Ciências Atmosféricas, Universidade de São Paulo-IAG/USP, Rua do Matão 1226, 05508-090 São Paulo, SP, Brazil
[c]http://www.brera.inaf.it/astri/
[d]http://www.cta-observatory.org/

ABSTRACT

The purpose of this contribution is to present the current status of the software architecture of the ASTRI SST-2M Cherenkov Camera. The ASTRI SST-2M telescope is an end-to-end prototype for the Small Size Telescope of the Cherenkov Telescope Array. The ASTRI camera is an innovative instrument based on SiPM detectors and has several internal hardware components. In this contribution we will give a brief description of the hardware components of the camera of the ASTRI SST-2M prototype and of their interconnections. Then we will present the outcome of the software architectural design process that we carried out in order to identify the main structural components of the camera software system and the relationships among them. We will analyze the architectural model that describes how the camera software is organized as a set of communicating blocks. Finally, we will show where these blocks are deployed in the hardware components and how they interact. We will describe in some detail, the physical communication ports and external ancillary devices management, the high precision time-tag management, the fast data collection and the fast data exchange between different camera subsystems, and the interfacing with the external systems.

Keywords: Cherenkov Telescope Array, CTA, ASTRI, SiPM, Software Design

1. INTRODUCTION

The ground-based gamma-ray astronomy had recently a rapid development. In particular, the use of Imaging Atmospheric Cherenkov Telescopes (IACTs) allowed to obtain important results in the study of very high energy phenomena in the Universe. A further significant improvement will derive from the realization of the Cherenkov Telescope Array (CTA)[1] observatory, which is currently under development and will provide better performances (sensitivity, energy range, angular resolution) compared to any currently operating gamma-ray instruments.

In this context, the Italian National Institute of Astrophysics (INAF) has developed the ASTRI SST-2M telescope,[2] as a prototype designed to be compliant with the CTA requirements for the small size class of telescopes. It has been installed at the observing station of Serra La Nave, on Mt. Etna (Sicily, Italy). Thanks to the experience gained with the prototype, it is foreseen a subsequent production of at least nine ASTRI telescopes that will form the mini-array proposed, as pre-production units, to be installed at the CTA southern site. The ASTRI mini-array project, led by INAF, is carried on by Italy, Brazil and South-Africa.

Some innovative technological solutions for the detection of Atmospheric Cherenkov light have been chosen for the optical design and the detector. The optical system is characterized by a dual-mirror in the Schwarzschild-Couder configuration, which allows to have a good angular resolution over the whole field of view and to reduce the camera size. The camera on the focal surface is designed to detect the short Cherenkov light flashes produced

* sangiorgi@ifc.inaf.it, +39 091 68 09 566

Software and Cyberinfrastructure for Astronomy IV, edited by Gianluca Chiozzi, Juan C. Guzman, Proc. of SPIE Vol. 9913, 99133T · © 2016 SPIE
CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2231647

by secondary particles of the atmospheric showers generated by very high energy primary particles or gamma rays hitting the atmosphere. It is based on Silicon Photo-Multiplier (SiPM) sensors with a specifically designed front-end electronics. It is equipped with customized calibration system and thermal control system which have been studied to reach its scientific requirements.

The software to manage all the telescope components certainly plays an important role. In this contribution we describe in particular the software architecture of the ASTRI SST-2M camera. This software implements the functional requirements of the camera allowing to perform the required operations of the instrument and to monitor the status of the different hardware elements. It is developed following the usual best practices of the software engineering like modularization and decoupling. This allows to optimize the development and maintenance phases and an easy integration in a more complex framework.

After a brief description of the camera hardware components and their functionalities in section 2, the design of the software is reported in section 3, where also some details are given about each module responsible for the different functions.

2. CAMERA HARDWARE DESCRIPTION

The camera of the ASTRI SST-2M prototype is composed of several components belonging to four hardware main block assemblies: Mechanical, Power Supply, Electronics and Thermal System.

Since the camera is controlled and operated by dedicated and customized software and firmware, we provide a brief description of some of the components that are involved in software management. For a more detailed description of the camera and its electronics, refer to [3] and [4].

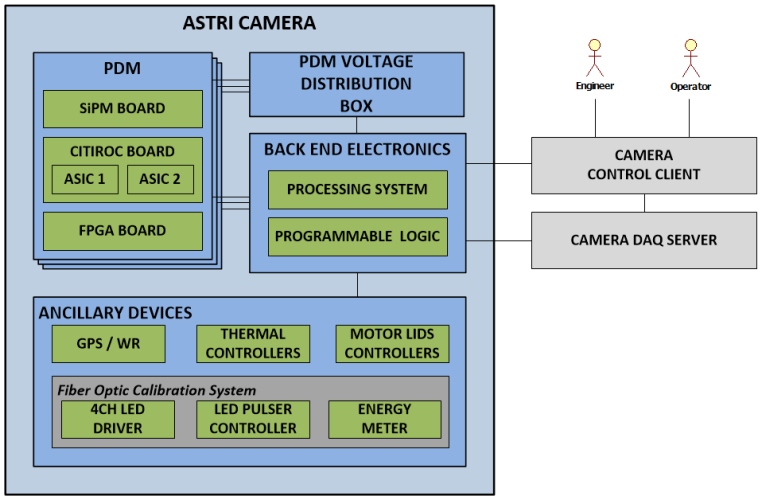

Figure 1. ASTRI SST-2M Camera logical view

2.1 Photo Detection Module

The Photo Detection Module (PDM), shown in Figure 2, is the electro-mechanical unit containing the SiPMs board, Front End Electronics (FEE) and ARTIX-7 FPGA[5] Printed Circuit Boards (PCBs).

SiPM PCB is formed of four SiPM units and 10 precision analog temperature sensors used for SiPMs temperature monitoring.

The FEE is designed to acquire electrical signals from the SiPMs and it is based on the CITIROC chip and two dual channel 12-bit ADCs that convert low-gain and high-gain sampled values to digital outputs.

In order to process the 64 PDM pixels two CITIROC ASICs[6,7] are connected in a daisy chain. Each ASIC is configured to undertake the desired functions by loading the configuration registers serially (slow control mode). This operation is managed by the ARTIX-7 FPGA under the control of the Back End Electronics (BEE).

The ARTIX-7 FPGA governs and controls all the operations of the PDM front-end. It is interfaced to the BEE through a bidirectional serial line and to the CITIROC PCB via a serial line and a few dedicated digital lines. It receives commands from the BEE and sends data and HK information to the BEE.

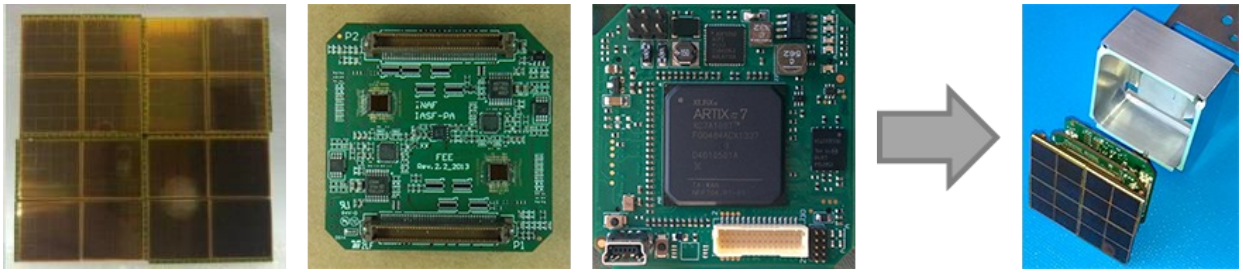

Figure 2. Components of the Photo Detection Module unit and its assembly

2.2 PDM Voltage Distribution Box

All the voltages needed for the PDMs and for some ancillary devices are provided by a Voltage Distribution Box (VDB) that consists of two main boards and 37 daughter boards. The VDB converts one low-voltage supply (24V) input to several regulated output voltages. A custom firmware provides control and housekeeping functions which can be accessed by an SPI communication interface.

Control and housekeeping functions as voltage or current are provided by a custom firmware.

2.3 Back End Electronics

The BEE is hosted on a custom built electronic board based on the Xilinx Zynq-7000 All Programmable SoCs[8] which is composed of two main parts: a Processing System (PS) formed around a dual-core ARM Cortex-A9 processor, and Programmable Logic (PL), which is equivalent to that of an FPGA. It also features integrated memory, a variety of peripherals, and high-speed communications interfaces.

The PL section is ideal for implementing high-speed logic, arithmetic and data flow subsystems, while the PS supports software routines and/or operating systems, meaning that the overall functionality of any designed system can be appropriately partitioned between hardware and software.

The BEE represents the elaboration unit of the camera software and is the heart of the electronics, since it is in charge of the complete data and command management of the instrument and it interfaces the detectors to the external world. It is the primary component of the real-time scientific data processing capability (the processing pipeline) of the ASTRI SST-2M Telescope.

The BEE is in charge of performing the following main functions:

- Receive commands from the Camera Control Client and perform the related procedures;
- Receive data from the front-end in real-time;
- Deliver results to the Camera Server System in the required ASTRI packet format;
- Perform control and monitoring operations of the ancillary devices;
- Perform a limited number of additional processes upon user request;

The Camera Control Client will provide all the external commands and controls to the BEE and will receive all errors, warning, performance and other reports from the BEE.

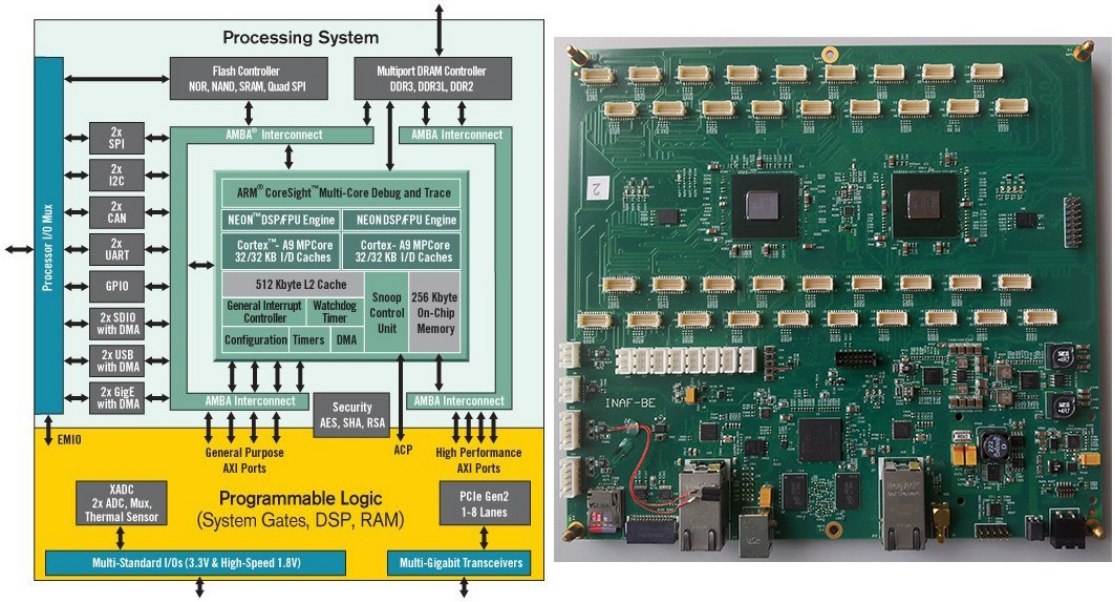

Figure 3. The Xilinx Zynq architecture on the left and the BEE on the right

2.4 Ancillary Devices

The camera is equipped with a set of additional devices which are described below.

A compact and lightweight GPS[9] receiver with a precise one-pulse-per-second (1pps) output is used for time synchronization and time-tagging of the triggered events. In addition, the BEE provides the required 10 ns precision processing, with a proper circuit embedded in the FPGA part.

The camera is thermally controlled through a thermoelectric system based on 4 Peltier cells controlled by 4 thermal controllers[10] that are connected to the BEE through a RS485 port. The controllers can operate in both heating and cooling modes to maintain the temperature near to desired value and provide temperature monitoring and functions for the control of the thermoelectric system.

Opening and closing operations of the telescope lids are performed by means of two motor controllers[11] connected through RS485 port to the BEE.

The relative gain calibration of the camera is performed by means of a fiber optic calibration system which includes an I2C 4 channel led driver[12] to activate the led of the desired color, an RS232 led pulser[13] to perform full control of pulse width, pulse current and repetition rate. An USB power and energy meter[14] controls the stability of the led.

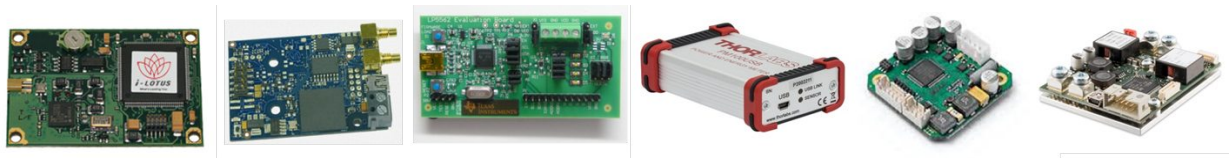

Figure 4. Ancillary devices. From the left GPS, Led Pulser, Led Driver, Energy Meter, Motor Lid Controller, Thermal Controller

2.5 Camera Control Client

The Control Client is represented by a personal computer where the high-level slow-control software is executed. This client represents the entry point for the final user (engineer or operator) who interacts with the camera using Graphical User Interfaces (GUI) in order to access monitoring and control functionality.

2.6 Camera DAQ Server

The Camera DAQ Server represents the Data Acquisition (DAQ) Workstation responsible for receiving and storing the different data packets (Housekeeping, Calibration and Scientific) produced by the camera.

3. CAMERA SOFTWARE DESCRIPTION

3.1 Software layers

The software realized to perform the control and monitor operations of the entire ASTRI SST-2M telescope is the ASTRI Mini-Array Software System (MASS).[15,16]

The MASS relies on the open-source ALMA Common Software (ACS)[17] for the high-level interfacing, and on the industrial-standard Process Control Unified Architecture (OPC-UA)[18] for low-level interaction with the hardware.

In order to ensure an high degree of decoupling with the specific hardware adopted, the MASS defines the interaction between user and subsystem in terms of high level commands which are described in a specific requirement document, the Interface Control Document (ICD).

In few words, this means that MASS is a high level framework that defines the interfaces starting from requirements and that specifies what the system must do without saying how. For this reason an additional software layer of lower level between MASS and hardware devices is necessary. This software, which is represented by the *Camera Application Software* in Figure 5, is responsible of the implementation of the interfaces defined at higher level. This is done by translating high level commands to low level directives that manage physical communication ports and external ancillary devices. The Camera Application Software is also in charge of managing data exchange between the camera hardware subsystem, in a way compliant with the hardware and software platform adopted for the camera.

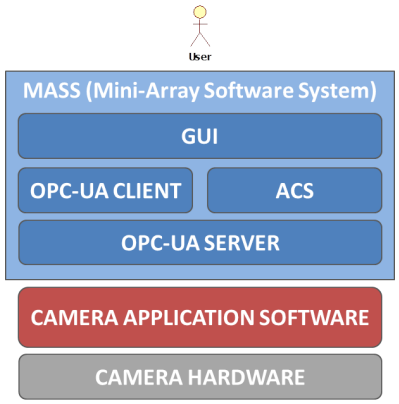

Figure 5. Software layers of the ASTRI SST-2M Camera

In the following the Camera Application Software will be described in terms of software architecture design and subsystem interactions.

3.2 Software architecture

The first step of any design project is to define the desired behaviors of the system, i.e. to create an appropriate specification from a set of requirements. As mentioned in the previous chapter, the Zynq architecture combines an ARM processor and an FPGA which provide, respectively, the software elements and the hardware elements of the designed system.

Thus, a key element of the system design stage, is to identify and partition the requested functionalities of the system between hardware and software, and to define the interfaces between the two sections.

In terms of hardware development, the task is to identify the functional blocks necessary to achieve the design, and to thereafter assemble them through some combination of design reuse and new Intellectual Property (IP) Core development, and make appropriate connections among the blocks. Similarly, the software aspect of the project can be realized through developing custom code or by reusing pre-existing software.

In the next we will use generically the term software also to refer to the hardware description language (VHDL) used to generate the code for the FPGA that implements the desired hardware blocks.

Starting from these assumptions, we identify two main classes of software components: Slow and Fast. Slow software components are implemented in the Processing System (PS) of the BEE and run over ARM processors under Linux and Java Runtime environment installed on it. Fast software components do not implement particularly complex logic but they are responsible for simple tasks that require high speed in their execution and for this reason they are realized in the Programmable Logic (PL) side of the BEE.

Figure 6 shows the software architecture in terms of logical functional blocks implementing the functional requirements of the camera identifying where they are deployed in the hardware subsystems.

In the following a detailed description of these blocks is provided.

The *Serialport Handler* module is responsible for the management of the serial communication ports of the BEE, like TTL, RS232, RS485, SPI and I2C and its main basic functionalities are:

- port reservation and initialization, by specifying the right parameters for communication like baud rate, databit, stopbit, parity bit and flow control.
- transmission over the port, by sending strings or byte array of data.
- receiving strings or byte array of data, by using a port event listener which notify upcoming port activities

In this way it is able to generically send and receive data, but also manage communication protocol of request-response type implementing, on received data, exit conditions like prefixed size or token identification and timeout in order to provide replies from interconnected devices as soon as possible.

Since the interactions with every external device equipped with serial interface pass through this module, we can say that it represents the most used external interface and its implementation is the most reused code of the camera application software. In a similar way, the *USBTMC Handler* performs over the USB port same functionality of the Serialport Handler for communication with device compliant to the USBTMC standard, like for the adopted Power and Energy Meter.

Ancillary Managers are a set of components for each ancillary devices which export to the high level software the desired functionalities. They basically represent a wrapper between the desired high level commands and proper data (strings or byte array) to be sent to the devices managing the specific communication protocol, in order to perform configuration of the device. By managing the specific communication protocols, they configure the device, monitor a set of attributes, perform actions.

Many, but not all, vendors provide their device with a proper software to manage it. However we had to rewrite a custom built software for each of them for several reasons. First, our particular hardware architecture with ARM processor and Linux environment is not supported by the software provided from vendors. Second, a uniform device management is required in order to facilitate the integration into an unique software which communicates and exchanges data with every subsystems. Furthermore, we are not interested to all functionalities provided by vendors for their devices, but only that ones useful for the users, in order to obtain a more simple software to be managed by the elaboration unit of the BEE and easy to use.

In this way, integrating a device inside the camera application software implies the following phases:

- Management of the physical interconnection port;
- Identification of the subset of commands of our interest to be compliant with the ICD;
- Implementation of the communication protocol;

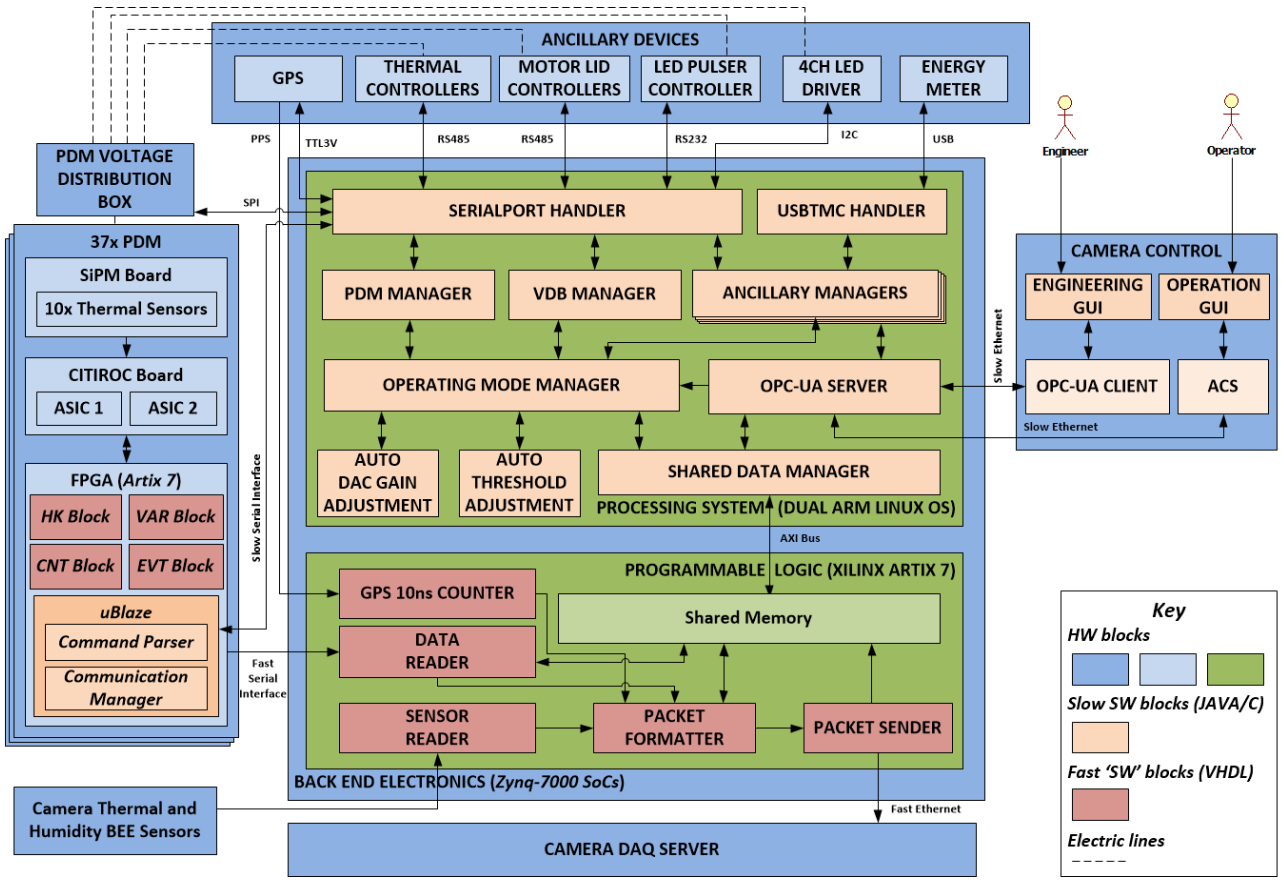

Figure 6. Astri SST-2M Camera Software Architecture

Note that the first phase is often the same because is valid for each device with the same interconnection and the port handler can be reused.

The *VDB Manager* is very similar as concept and internal architecture to the ancillary managers, but it is responsible of the monitoring and control of the power supply management of the PDM. It implements our custom build communication protocol of the VDB in order to switch on and off low voltages and high voltage of the PDMs, and keep track of some parameters to ensure that the PDMs work always in the proper range of levels of voltages.

It also provides functionalities to allow to turn on and off some devices connected to the VDB for their long lifetime management. For example, consider the case of motor lids controllers that must work for a very limited time just to open and close the lids. Using the VDB and its manager, we are able to automatically keep on the device only when it is needed.

The *PDM Manager* is responsible of the control of the PDMs. It implements our custom build communication protocol of the PDM in order to initialize the PDM, send, write and execute the configuration tables for the ASICs and the FPGA, reset the FPGA modules and request data relative to calibration, housekeeping and variance.

Commands and data sent through the slow serial interfaces are received by the FPGA of the PDM which implements, in a firmware deployed in a soft processor, the communication protocol manager and the command parser. Once a command is correctly received and recognized, the firmware sends data or activate the proper lines of the hardware modules realized in VHDL to perform the respectively functionalities of housekeeping, variance, calibration and scientific acquisition.

The *Operating Mode Manager* manages all the operations needed for the specific operational mode (Idle, Scientific, Calibration) of the camera. It receives the desired operation requested by the user and the related parameters and prepare the proper CITIROC and FPGA configurations for the FEE to be dispatched by the PDM Manager. Simultaneously, it schedules all the others needed operations requested for that particular mode using functions provided by the other modules with which it is interconnected. As an example, in relative calibration mode it provides the high voltage to the PDM through the VDB Manager, closes the lids through the motor lid manager, starts the pulsing of the fiber optic and measures the emitted light through the ancillary managers of the fiber optic calibration system, and so on.

Once all the requested procedures are accomplished, it applies the configurations to all the PDMs and the operating mode starts.

During scientific operation mode, it also performs continuous monitoring of temperature and rate meter of the SiPMs and passes this data to the *Auto DAC Gain Adjustment* and *Auto Threshold Adjustment* modules. These two modules make the camera a dynamic control system able, through specific algorithms, to dynamically evaluate the ideal PDM configurations for the runtime context.

In the PL side of the BEE we have the modules that requires major priorities and not so suitable to run under an operating system.

The *Sensor Reader* is used for continuous monitoring of temperature and humidity sensors of the BEE.

The *Gps 10ns Counter* provides high precision time information. This is done by using the Pulse Per Second line of the GPS device to triggers a counter clocked at 100 Mhz. Every second this line resets the counter and in the meanwhile the operating system reads the time of the GPS through the GPS ancillary manager with a precision of 1 second. When a scientific event occurs, the information of GPS and the value of the counter are combined together and the event can be tagged with the time information of 10 nanoseconds of precision.

During scientific mode the PDM works as master device and the data related to events are automatically sent to the BEE. These data are collected from all the PDMs by the *Data Reader* module which passes them to the *Packet Formatter* module. It is in charge to organize these data, plus additional information, according to the current operating mode in a packet data format compliant with the ASTRI telemetry specification.

Once the packet is correctly formatted, it is sent to the DAQ Camera Server[19] through the Fast Ethernet Channel by the *Packet Sender* module which implements an UDP transmission.

This implies that the entire process of scientific data acquisition and sending, which is the crucial activity for the camera, is totally managed in parallel with the operating system. In this way it is managed with higher priority and without affecting or being affected by computational complexity.

Since the PL and PS are two independent subsystems of the BEE, we use IP core implementations of the AXI bus, a shared memory accessible from the user space and the Shared Data Manager in order to read and write data that the systems must exchange. As an example, PL can provide to the PS information about the status of the sending process of scientific data. As well as, PS can provide to the PL all that information not coming from the PDMs needed to format the telemetry packets.

All described modules implement the functionalities requested for the camera. But as the BEE is installed in the camera itself on board the telescope, we use an *OPC-UA Server* and the other software layers of the MASS in order to provide this functionalities to the users through graphical user interface.[20]

ACKNOWLEDGMENTS

This work is supported by the Italian Ministry of Education, University, and Research (MIUR) with funds specifically assigned to the Italian National Institute of Astrophysics (INAF) for the Cherenkov Telescope Array (CTA), and by the Italian Ministry of Economic Development (MISE) within the "Astronomia Industriale" program. We acknowledge support from the Brazilian Funding Agency FAPESP (Grant 2013/10559-5) and from the South African Department of Science and Technology through Funding Agreement 0227/2014 for the South African Gamma-Ray Astronomy Programme. We gratefully acknowledge support from the agencies and organizations listed under Funding Agencies at this website: http://www.cta-observatory.org/. This paper has gone through internal review by the CTA Consortium.

REFERENCES

[1] Acharya, B.S., et al., "Introducing the CTA concept," *Astroparticle Physics* **43**, 3 – 18 (2013).

[2] Pareschi, G., et al., "The dual-mirror Small Size Telescope for the Cherenkov Telescope Array," *Proc. 33rd ICRC* (2013).

[3] Catalano, O., Maccarone, M. C., Gargano, C., La Rosa, G., Segreto, A., Sottile, G., De Caprio, V., Russo, F., Capalbi, M., Sangiorgi, P., Bonanno, G., Grillo, A., Garozzo, S., Marano, D., Billotta, S., Romeo, G., Stringhetti, L., Fiorini, M., La Palombara, N., Incorvaia, S., Toso, G., Impiombato, D., and Giarrusso, S., "The camera of the ASTRI SST-2M prototype for the Cherenkov Telescope Array," *Proc. SPIE* **9147**, 91470D–91470D–15 (2014).

[4] Sottile, G., Catalano, O., et al., for the ASTRI Collaboration and the CTA Consortium, "ASTRI SST-2M Camera Electronics," *these proceedings* (2016).

[5] XILINX, "Artix-7 FPGA Family." http://www.xilinx.com/products/silicon-devices/fpga/artix-7/.

[6] Fleury, J., Callier, S., de La Taille, C., Seguin, N., Thienpont, D., Dulucq, F., Ahmad, S., and Martin, G., "Petiroc and Citiroc: front-end ASICs for SiPM read-out and ToF applications," *Journal of Instrumentation* **9**, C01049 (Jan. 2014).

[7] Impiombato, D., Giarrusso, S., Mineo, T., Catalano, O., Gargano, C., Rosa, G. L., Russo, F., Sottile, G., Billotta, S., Bonanno, G., Garozzo, S., Grillo, A., Marano, D., and Romeo, G., "Characterization and performance of the ASIC (CITIROC) front-end of the ASTRI camera," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **794**, 185 – 192 (2015).

[8] XILINX, "Zynq-7000 Silicon Devices." http://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html.

[9] I-Lotus Pte. Ltd., "GPS Products M12 Timing Receiver datasheet." http://www.ilotus.com.sg/~ilotus/sites/all/themes/zeropoint/pdf/m12m/M12M%20-%20User%20Guide%20(Ver%201.0.0).pdf.

[10] Meerstetter Engineering, "TEC-1089 TEC Controller Datasheet." http://www.meerstetter.ch/category/29-latest-datasheets?download=236.

[11] Nanotec, "SMCI12 Stepper Controller technical manual." http://en.nanotec.com/fileadmin/files/Handbuecher/Motorcontrols/SMCI12_Technical-Manual_V1.2.pdf.

[12] Texas Instrument, "LP5562 Four-Channel LED Driver with Programmable Lighting Sequences Datasheet." http://www.ti.com/lit/ds/snvs820a/snvs820a.pdf.

[13] PicoLAS Schulz Electronic GmbH, "PLCS-21 user guide." http://www.schulz-electronic.de/out/media/PLCS-21_B_E.pdf.

[14] Thorlabs, "PM100USB Manual." http://www.thorlabs.de/thorcat/19500/PM100USB-Manual.pdf.

[15] Tosti, G., Schwarz, J., Antonelli, L. A., Trifoglio, M., Catalano, O., Maccarone, M. C., Leto, G., Gianotti, F., Canestrari, R., Giro, E., Fiorini, M., La Palombara, N., Pareschi, G., Stringhetti, L., Vercellone, S., Conforti, V., Tanci, C., Bruno, P., Grillo, A., Testa, V., di Paola, A., and Gallozzi, S., "The ASTRI/CTA mini-array software system," *Proc. SPIE* **9152**, 915204–915204–9 (2014).

[16] Tanci, C., Tosti, G., et al., on behalf of the ASTRI Collaboration and the CTA Consortium, "The ASTRI mini array software system (MASS) implementation: a proposal for the Cherenkov Telescope Array," *these proceedings* (2016).

[17] Schwarz, J., Farris, A., and Sommer, H., "The ALMA software architecture," *Proc. SPIE* **5496**, 190–204 (2004).

[18] OPC Unified Architecture, "Specification, Part 1: Overview and Concepts, Release 1.02, July 10, 2012." https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-1-overview-and-concepts.

[19] Conforti, V., Trifoglio, M., Bulgarelli, A., Gianotti, F., Fioretti, V., Tacchini, A., Zoli, A., Malaguti, G., Capalbi, M., and Catalano, O., "The ASTRI SST-2M telescope prototype for the Cherenkov Telescope Array: camera DAQ software architecture," *Proc. SPIE* **9152**, 91522D–91522D–14 (2014).

[20] Tanci, C., Tosti, G., et al., on behalf of the ASTRI Collaboration and the CTA Consortium, "Software design and code generation for the engineering graphical user interface of the ASTRI SST-2M prototype for the Cherenkov Telescope Array," *these proceedings* (2016).