DEC/MEC Software Detailed Design Document

Doc. PACS-CL-SR-004, issue 1.0
11 April, 2006

Prepared by : A. Mazy

Verified by :

Authorised by : E. Renotte

Approved by :

filename : PACS-CL-SR-004v1.0 DDD.doc

Distribution List

Recipients	Affiliation	Copies
CSL PACS Team c/o E. Renotte	CSL - Liege	\\Titan_pacs\documents (sources)\e) Software\PACS-CL-SR-004v1.0.doc
Central File @ PACS Project Office c/o N. Gradmann	MPE - Garching	1 PDF copy
O. Bauer, H. Feuchtgruber	MPE - Garching	1 PDF copy

Document Change Record

Issue	Date	Comments
Issue 1.0	14/04/2006	

last saved by amazy on 11-Apr-06

List of Abbreviations

ASW	Application Software
AVM	Avionic Verification Model
BOLA	Bolometer Amplifier
BOLC	Bolometer Controller
CSL	Centre Spatial de Liège
DAC	Digital-to-Analog Converter
DEC/MEC	Detector & Mechanism Controller
DPU	Digital Processing Unit
EEPROM	Electrically Erasable PROM
EGSE	Electrical Ground Support Equipment
EM	Engineering/Electrical Model
ESA	European Space Agency
FM	Flight Model
FPGA	Field Programmable Gate Array
FPU	Focal Plane Unit
HK	HouseKeeping
HW	Hardware
ICD	Interface Control Document
NA	Not Applicable
OBS	On-Board Software
PACS	Photodetector Array Camera and Spectrometer
PROM	Programmable ROM
PSC	Processor Support Chip
QM	Qualification Model
RAM	Random Access Memory
ROM	Read-Only Memory
S/C	SpaceCraft
SPU	Signal Processing Unit
SSD	Software Specification Document
SW	Software
SUSW	Start-Up Software
TBC	To Be Confirmed
TBD	To Be Defined
URD	User Requirement Document

Table of Contents

1 Scope

1.1 Introduction

The Photodetector Array Camera and Spectrometer (PACS) is an imaging spectrometer-photometer which forms part of the science payload of the Herschel Space Observatory (formerly called FIRST), an ESA cornerstone mission (CS4) to be launched in 2007 on Ariane 5.

A presentation of the Herschel mission and status is available at URL: http://sci.esa.int/home/herschel/. Useful information on PACS instrument, mission and Consortium can be found at http://pacs.mpe-garching.mpg.de and http://pacs.ster.kuleuven.ac.be/.

1.2 Purpose

This document presents the detailed design of the software embedded in the DEC/MEC.

This document is addressed to :

- The programmer who will maintain the DEC/MEC application.

The software embedded in the DEC/MEC is split in two independent module:

- The Sart-up Software (SUSW) is used only at start-up to check the memory, perform patching and start the Application software. The SUSW is not described in this document.
- The Application Software (ASW) is the main application that is described in this document.

This document presents:

- The coding standards that have been used when coding the DMC OBS
- The DMC OBS code. Note that the code is provided as an external appendix

1.3 Organisational Responsibilities

The PACS project activities including project management and system engineering will be done at MPE-Garching under the direction of A. Poglitsch (PI). Design, fabrication, testing, and integration of the flight units will be done at Co-I and commercial facilities as appropriate.

In this programme the CSL is responsible for the design, production and unit-level verification of:

- the focal plane Grating Assembly;
- the Detector & Mechanism Controller (DEC/MEC);
- the Warm Interconnecting Harness.

2 Documents

2.1 Applicable Documents

[AD1]	PACS-CL-SP-001	DEC/MEC Software User Requirements Document
[AD2]	PACS-CL-ID-003	Interface Control Document DEC/MEC – DPU
[AD3]	PACS-CL-ID-004	Interface Control Document DEC/MEC - SPU
[AD4]	HPL-IC-1248-01-CRS	HW-SW Interface Control Document
[AD5]	HPL-ICD-1248-01-CRS	Low Level SW Drivers SW Interface Control Document
[AD6]	PACS-CL-SR-001	DEC/MEC SSD
[AD7]	PACS-CL-SR-002	DEC/MEC User Manual

2.2 Reference Documents

[RD1]	ESA PT-PACS-02126	Instrument Interface Document - Part B - Instrument "PACS"
[RD2]	PACS-ME-PL-002	PACS Design, Development and Verification Plan
[RD3]	1355-1995 (ISO/IEC 14575)	IEEE Standard for Heterogeneous InterConnect (HIC)
[RD4]	-	AxiomSys User's Manual
[RD5]	-	Virtuoso user guide for Version 4.1

3 Coding standards

3.1 File organisation

- Each file name clearly identifies the functionality of its content
- Each Virtuoso task is implemented in a separate file. The file name and the task name have a similar structure although file names are usually shorter.
- All structures and global variables are uniquely defined in header files.

3.2 Naming conventions

3.2.1 General

- All names shall be self explanatory. Therefore, there are no limitation on name length
- When a name is made of more than one word, the beginning of each word is marked by a capital letter (ex: SizeInBytes)

3.2.2 Global variables

- All global variable names start with 'g' (ex: gParams)
- If the global variable is a pointer, it starts with 'gp' (ex: gpParams)
- If the global variable is an array, it starts with 'ga' (ex: gaRedSpectroPacketBuffer)

3.2.3 Local variables

- All local variable names start with a tiny letter (ex: nbMeasureBySample)
- If the local variable is a pointer, it starts with 'p_' (ex: p_nbMeasureBySample)

3.2.4 Structure

- Structure names always start with a capital letter (ex: MyStructure).
- Structure member names always start with a capital letter except for pointers that starts with a 'p' (ex: pMyMemberPointer).

3.2.5 Enumeration and defines

- Enumeration type names always start with a capital letter.
- Enumeration member are always written in capital letter, the words are separated by underscore (ex: SIMULATION_DATA)
- When an enumeration member or a define defines a mask, its name starts with K_BMASK_ (this is to be compatible with some code provided by CRISA and used in DMC OBS)
- When an enumeration member or a define defines a bit position, its name starts with K_BPOS_ (this is to be compatible with some code provided by CRISA and used in DMC OBS)

3.2.6 Function

- All function names start with a capital letter; except for function written in assembly that start with a tiny letter. (ex: PerformHkAcquisitions() and pmpsc_isr())
- All function implementing a trigger command start with 'Command' (ex: CommandStartSequence)
- Function arguments names have the same naming conventions as local variables

3.2.7 Assembly code

- Assembly is never self explanatory. Therefore, each subset of instruction shall be documented
- label names are in tiny letters. Words are separated by '_' (ex: read_chopper_pos). Self explanatory label names is a good way to comment the code that follows.

- the compiler only considers the first 32 letters of label names. Therefore, we make sure that the first 32 characters of labels are unique. This does not prevent to use longer names.

4 Source Code

4.1 List of files

allnodes.h	File generated by Virtuoso, containing the definition of all Kernel Services that can be shared among nodes (in our case, we have only one node)
BaseMosaic	Part of the makefile
BDecCtrl.c	Contains the code of the Blue DEC Controller and the code shared between Blue and Red DEC Controller.
BDecRec.c	Contains the code of the Blue DEC Receiver and the code shared between Blue and Red DEC Receiver
bol.h	Contains the definition of variables related to the BOLC.
BolCtrl.c	Contains the code of the BOLC Controller.
BolRec.c	Contains the code of the BOLC Receiver.
BPackEnc.c	Contains the code of the Blue Packet Encoder and the code shared between Blue and Red Packet Encoders
ChopCtrl.c	Contains the code of the Chopper Controller except for some code which is common to all PID controllers that can be found in pid_ctrl.c
ChopCtrl.h	Contains the definition specific to Chopper Controller
ChopGrat.s	Contains the 8KHz interrupt routine : PID Controllers and analog housekeeping.
Com_list.h	Contains the list of all trigger commands and their description.
Commands.h	Contains the definition of Command structures.
Constant.h	Contains the definition of some constants shared in the whole OBS
crc.c	Contains the algorithm used to compute the CRC used in the communication with DPU.
crc.h	Contains the definitions related to the CRC algorithm.
Crisa.ach	Architecture file defining the HW of the CRISA board
CsCtrl1.c	Contains the code of the first calibration source controller and the code that is shared between the 2 controllers
CsCtrl2.c	Contains the code of the second calibration source controller.
csl_type.h	Contains the definition of basic data types.
Dec.h	Contains the definition of all variables related to the DEC.
Det.h	Contains the definition of all variables related to the detectors in general (DEC + BOLC)
DetSim.c	Contains the code of the Detector Simulator task.
DetSim.h	Contains the definition related to the Detector Simulator task.
DpuCom.h	Contains the definitions related to the communication with DPU.
DpuRec.c	Contains the code of the DPU Receiver task.
DpuSend.c	Contains the code of the DPU Sender task.
error.c	Contains some function used to signal errors.

error.h	Contains the definition of all errors that can be generated in the OBS.
FWCtrl.c	Contains the code of the Filter Wheel Controllers except for some code which is common to all PID controllers that can be found in pid_ctrl.c
GratCtrl.c	Contains the code of the Grating Controller except for some code which is common to all PID controllers that can be found in pid_ctrl.c
GratCtrl.h	Contains the definition specific to Grating Controller
hk.c	Contains the code of the nominal HK task.
Hk.h	Contains the definition specific to the HK process
HkDiag.c	Contains the code of the diagnostic HK task.
HkList.h	Contains the list and definition of all the HK entries.
l_errcod.h	Definition of error codes for the low level software provided by CRISA
l_gendef.h	General definition for the low level software provided by CRISA
l_hwmap.h	Definition of addresses and bit fields specific to the CRISA DSP board
l_memasm.s	Low level functions to handle memory.
l_memory.c	Low level functions to handle memory.
l_memory.h	Low level functions to handle memory.
l_pscgen.c	Processor support chip functions.
l_pscgen.h	Processor support chip functions.
link1355.c	High level functions to handle the 1355 interface
link1355.h	High level functions to handle the 1355 interface
links.h	Affectation of the 1355 links
Makefile	The makefile
m_smcsco.c	Low level function to handle the SMCS chip provided by CRISA
m_smcsco.h	Low level function to handle the SMCS chip provided by CRISA
m_smcsge.c	Low level function to handle the SMCS chip provided by CRISA
m_smcsge.h	Low level function to handle the SMCS chip provided by CRISA
m_smcsin.c	Low level function to handle the SMCS chip provided by CRISA
m_smcsin.h	Low level function to handle the SMCS chip provided by CRISA
m_smcsre.c	Low level function to handle the SMCS chip provided by CRISA
m_smcsre.h	Low level function to handle the SMCS chip provided by CRISA
m_smcstr.c	Low level function to handle the SMCS chip provided by CRISA
m_smcstr.h	Low level function to handle the SMCS chip provided by CRISA
mim.c	Functions to handle the MIM boards
mim.h	Functions to handle the MIM boards
NODE1.c	File generated by Virtuoso, containing the definition of all Kernel Services specific to the node.
NODE1.h	File generated by Virtuoso, containing the definition of all Kernel Services specific to the node.
OnBoardMosaic.nli	Hardware definition file used by Virtuoso
OnBoardMosaic.vpf	Virtuoso Project File
PackEnc.h	Definitions related to Packet Encoder Tasks.
par_list.h	List and definition of all the Write commands.
Params.c	Functions to initialize the parameters area (shared memory)

Params.h	Definition of the parameters area (shared memory)
pid_ctrl.c	Functions global to all PID controllers
pid_ctrl.h	Functions global to all PID controllers
RDecCtrl.c	Contains the code of the Red DEC Controller
RDecRec.c	Contains the code of the Red DEC Receiver
RPackEnc.c	Contains the code of the Red Packet Encoder
Seq.c	Contains the code of the Sequencer Task
Seq.h	Contains the definitions related to the Sequencer Task
seq_msg.h	Contains some definitions related to the Sequencer Task
smcs.s	Contains the code of the interrupt routine for the SMCS chips and PSC.
SmcsDrv.c	Contains the high level functions of the 1355 drivers
SpuCom.h	Definitions related to the communication with SPU.
TempSens.c	Contains the code of the Temperature Sensor Task
ToolsMosaic	Part of the makefile
t_status.h	Bit fields definition used in Task Status housekeeping
u_bits.h	Some helper functions to manipulate bit fields
u_res.h	Some helper functions to lock and unlock shared resources
V_Macro.h	Some functions definition to guarantee the same behavior when compiling the code on a PC to build a Simulator

The listing of these files are provided in appendix