

The most important thing we build is trust



ADVANCED ELECTRONIC SOLUTIONS

AVIATION SERVICES

COMMUNICATIONS AND CONNECTIVITY

MISSION SYSTEMS

## FDIR Validation Test-bed Development and Results

Alexander Karlsson, Anandhavel Sakthivel, Martin Aberg, Jan Andersson, Sandi Habinc  
Brice Dellandréa, Jean-Christian Nodet - TAS  
Farid Guettache, Gianluca Furano – ESA

Cobham Gaisler

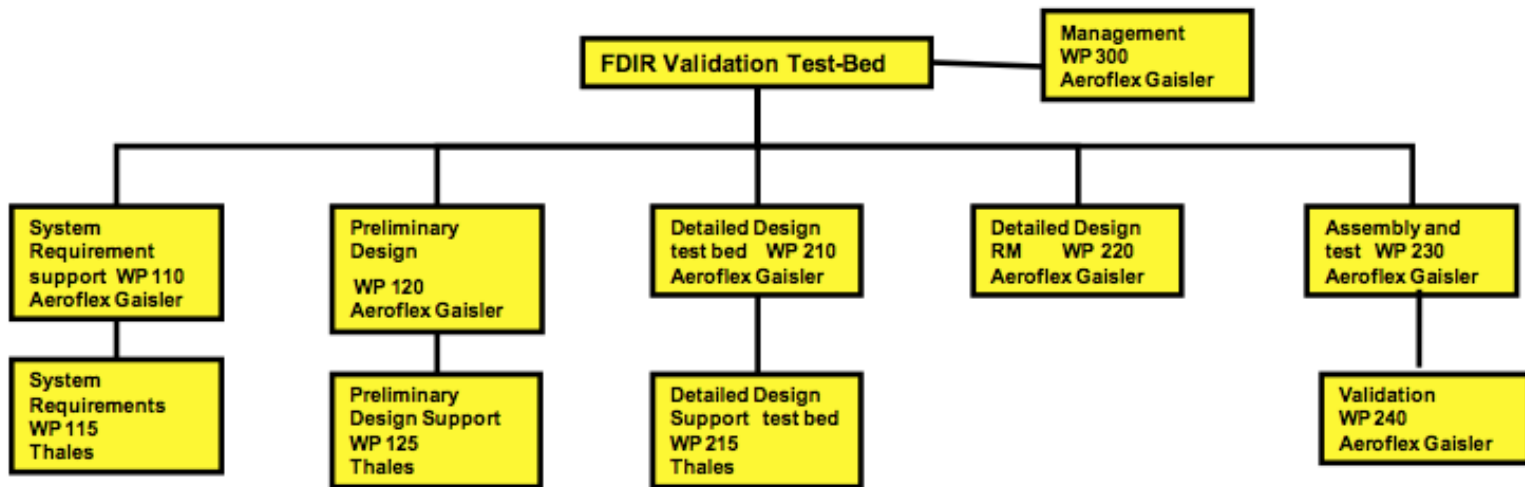
TEC-ED and TEC-SW Final Presentation Days - 01 June 2015

- Development of a FDIR Validation Test-bed is an activity initiated and funded by ESA/ESTEC under contract 4000109928/13/NL/AK.
- Cobham Gaisler and Thales Alenia Space France are to develop an extension of the existing avionics system testbed facility in ESTEC's Avionics Lab.
- The resulting FDIR testbed will allow to test concepts, strategy mechanisms and tools related to FDIR. Ultimately the purpose of the output of this activity is to provide a tool for assessment and validation at laboratory level.
- Team:
  - ESA: Farid Guettache, Gianluca Furano
  - TAS-F: Jean-Christian Nodet, Brice Dellandrea
  - Cobham Gaisler: Anandhavel Sakthivel, Alexander Karlsson, Martin Aberg, Sandi Habinc, Jan Andersson

- Activity flow
- Objectives
- Current RASTA system
- Work performed
- Test-bed
- Board design
- Reconfiguration Module
- Test-bed controller tool
- Summary

- Simple waterfall:

- Phase 1: System requirements, Preliminary Design
- Phase 2: Detailed Design Development, Assembling, Integration and Test, Validation and test case



- Extend the capability of the existing RASTA facility from a single string to a dual redundant string, supporting different redundancy schemes.
- Develop a test-bed controller software tool which enables the user to perform system-level emulation of full FDIR functionality using the RASTA facility.
- Engineering objectives:
  - Reuse the existing hardware as much as possible.
  - Design and develop a new RASTA module providing reconfiguration and safeguard memory.
  - Integrate the newly developed module in ESTEC's RASTA facility based on two identical chains, re-using the existing equipment.
  - test-bed controller software tool that enables the user to load, run and analyse applications and simulations, as well as inject error cases in the dual redundant system.

- RASTA system consisting of

- Processor board: GR-CPCI-AT697

- PCI, UART, ...

- I/O board: GR-CPCI-XC4V

- PCI
- MIL-STD-1553B, CAN (redundant),
- SpaceWire - three independent links with RMAP
- 10/100 Ethernet, Discrete I/O, UART and JTAG debug interfaces

- TMTC board: GR-CPCI-XC4V

- PCI
- ECCS/CCSDS TM and TC functions, including a CPDU which also has a PacketWire interface towards an external Reconfiguration Module. It also contains On-Board Time and support for automatic time transfer via SpaceWire.

- SW on processor board typically accesses peripheral units on I/O and TMTC boards through PCI



- Requirements established and mapped to RASTA building blocks
  - Review of FDIR architecture and characteristics in selected space missions TN 1-1
    - Covered SB4000, S1, S3, SAVOIR, L3G-AURUM
- Developed Hardware
  - GR-RASTA-FDIR: RM, SGM and CPDU inside FPGA. Mezzanine to support external interfaces
  - GR-CPCI-SPW4: New accessory board that consists of 4xSpW. Main purpose is to connect GR-RASTA-FDIR to GR-RASTA-TMTC with minimum changes to the latter system

continued

---

- Developed Reconfiguration Module IP core
- Updated GR-RASTA-TMTC
  - Connections to ensure that FDIR unit accessible from ground
- Interconnects between PM, TMTC and FDIR units
- Established Verification and Validation Plan
- GRMON2 support for LEON2
- Developed RTEMS drivers required for Reconfiguration Module and GR-RASTA-FDIR (PCI target driver).
- Controller tool development (Ongoing)
- Verification and Validation of the test-bed (Ongoing)
- Test application software development (Ongoing)



- The configuration of each RASTA single chain contains:
  - Processor Module (PM) based on AT697E/F
  - I/O Module
  - Telecommand (TC) and Telemetry Module (TM)
  - FDIR Module with RM and Safeguard Memory
  - Crate with power supply

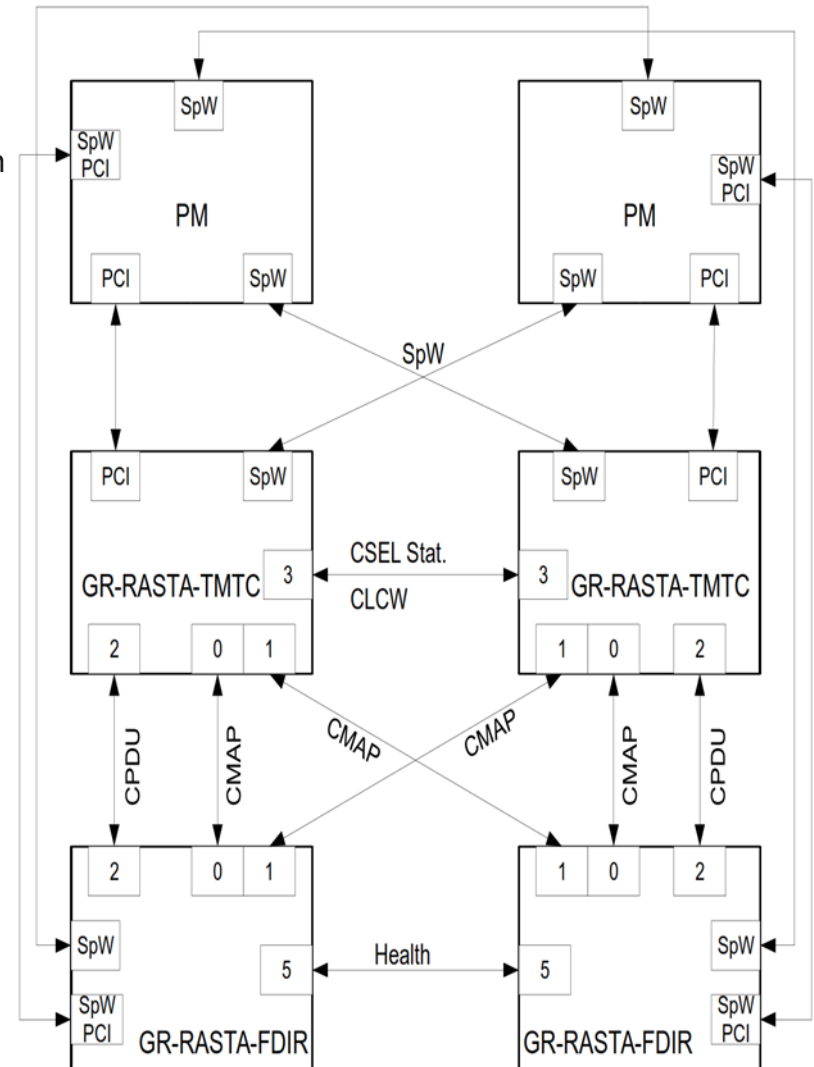
# Test-bed

## Single chain



## Dual chain with interconnects

- PM here is LEON and GR-RASTA-IO
- PM - TMTC - FDIR
  - PCI bus is the main communication interface within each chain
  - While SpaceWire is used for cross-strap links
- FDIR 2 – TMTC 2
  - CPDU links encapsulated in a PacketAsynchronous interface
- FDIR 0 – TMTC 0
  - TC and TM communication FDIR and TMTC (direct link) that is encapsulated in a PA interface
- FDIR 1 – TMTC 1
  - TC and TM communication FDIR and TMTC (cross-strap) that is encapsulated in a PA interface
- TMTC 3 N – TMTC 3 R
  - Link Between the TMTC modules carrying CLCW and CSEL Status
- FDIR 3 N – FDIR 3 R
  - Health communication between RM's



# Test-bed

Dual chain



# Test-bed

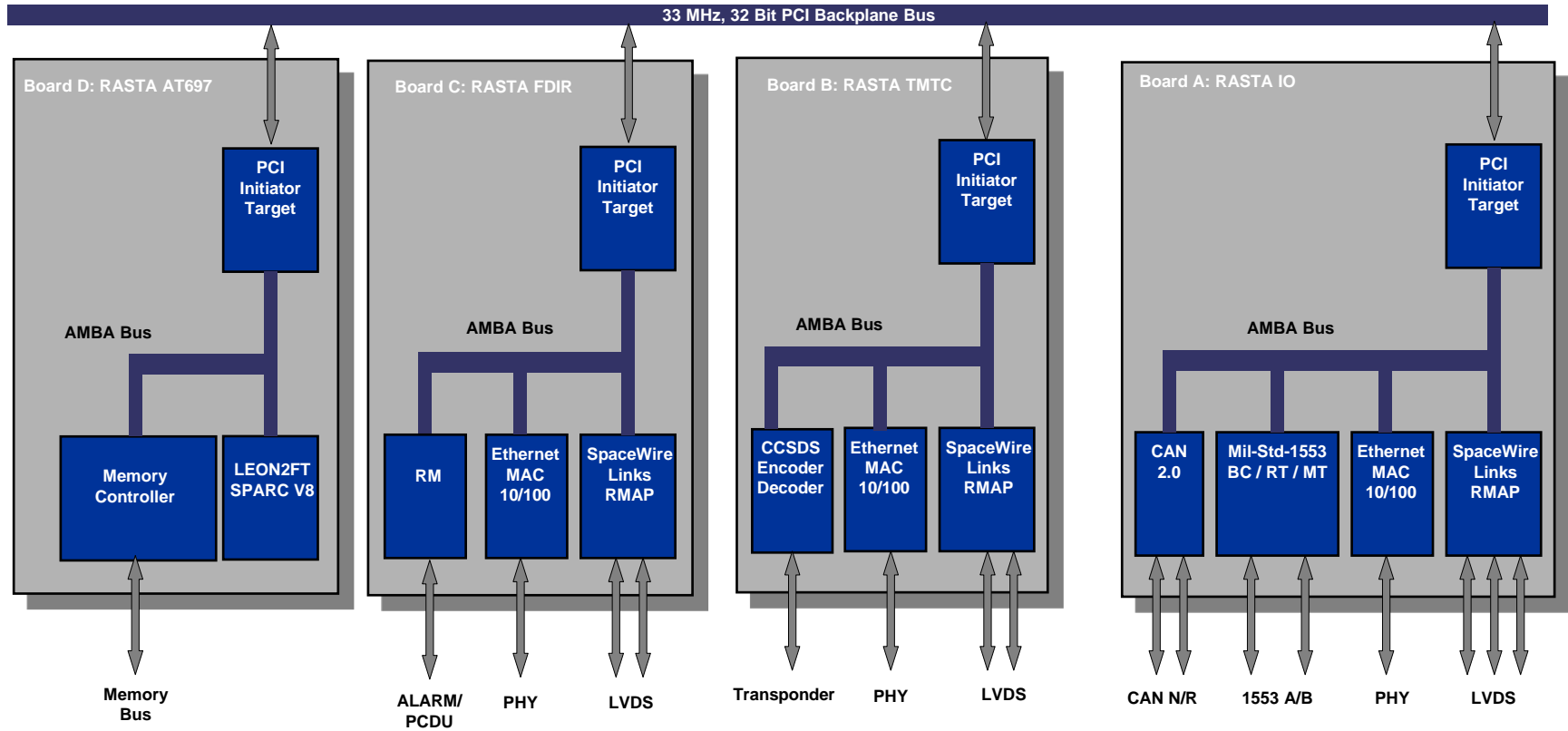
## Dual chain with interconnects



# cPCI Crate Arrangement

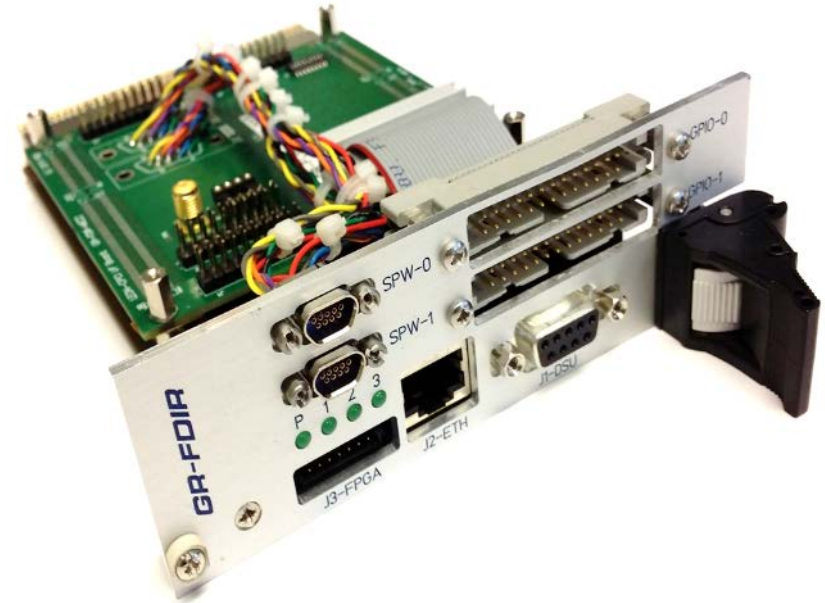
	PCI 1 (HOST)	PCI 2	PCI 3	PCI 4	PCI 5	PCI 6	PCI 7	PCI 8				
GR-CPCI-SER	GR-CPCI-AT697	GR-ACC-SPW4	GR-RASTA-FDIR		GR-RASTA-TMTC		GR-TMTC-PW	GR-RASTA-IO				GR-ACC-SPW4
1 (ACC → PCI 1)	2	3 (ACC → PCI3)	4	5	6	7	8 (ACC → PCI 5)	9	10	11	12	13 (ACC → PCI 5)

# Block Diagram



## GR-RASTA-FDIR (1)

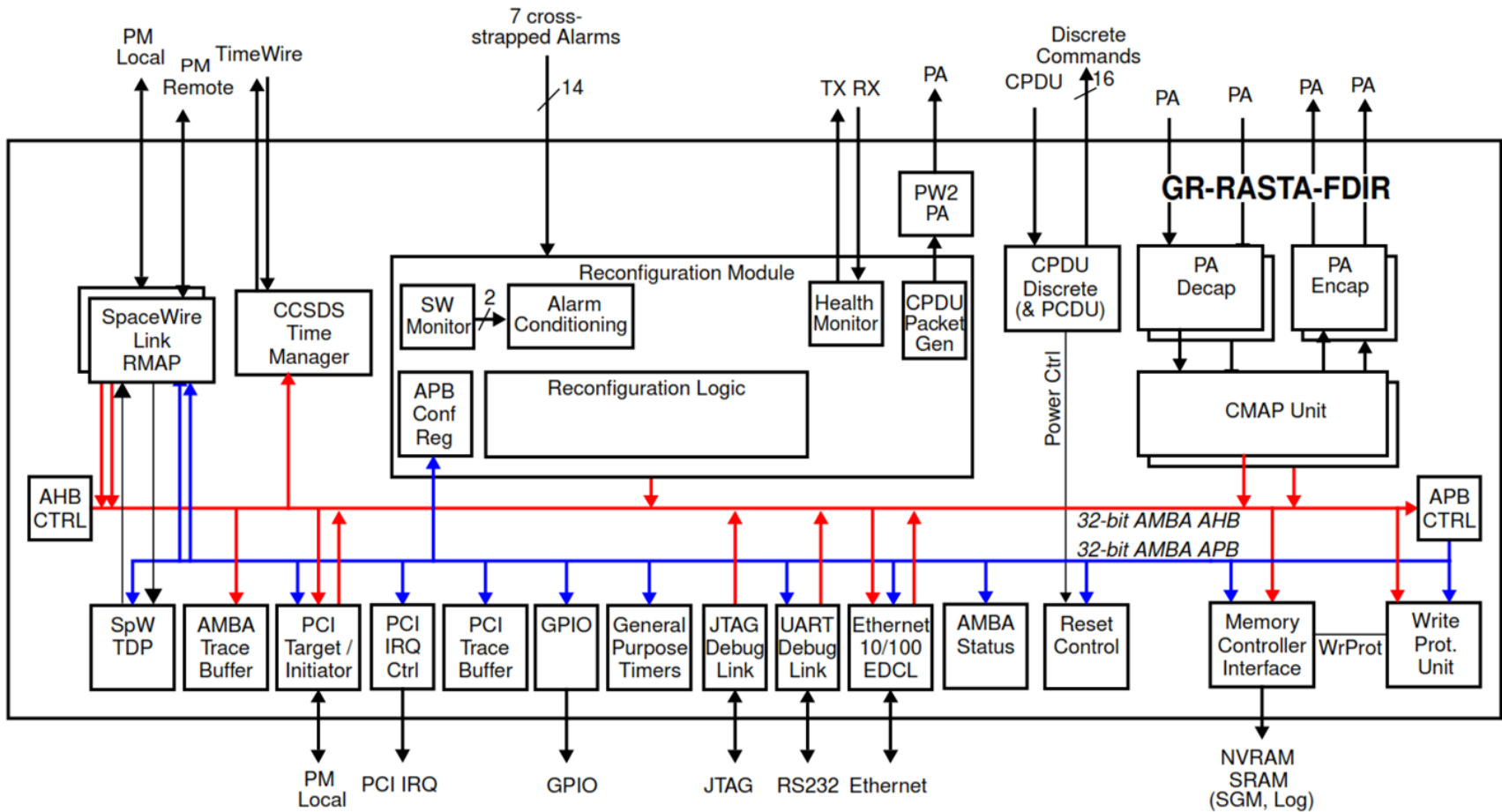
- The GR-RASTA-FDIR board is based on the GR-CPCI-XC4V:
  - Two SpaceWire interface
  - Ethernet core
  - JTAG Interface
  - Debug serial link UART
  - Compact CPCI
  - Memory
    - 128 MB SDRAM
    - 8 MB Flash PROM
  - Two GPIO interfaces:
    - 18 LVTTTL input/output digital I/O
    - 2 LVDS input/output digital I/O
- Reprogrammable FPGA with programming interface
- FPGA programmed from on-board configuration memory at reset
- On-board configuration memory is reprogrammable via JTAG I/F
- In addition one GR-ACC-SPW4 accessory board provides 4 MDM9 connectors with LVDS electrical levels





# Board Design

## GR-RASTA-FDIR (2)

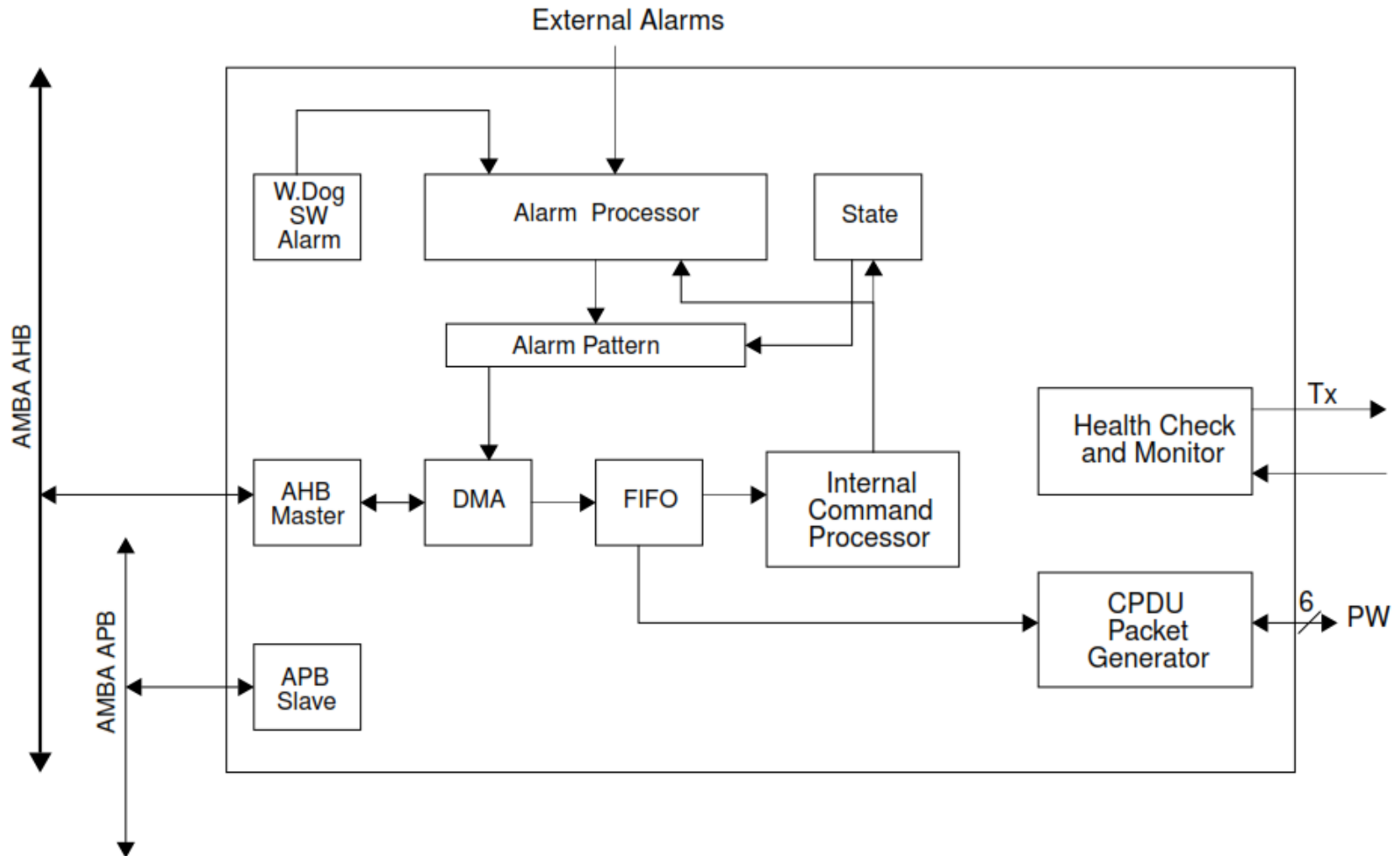


- Consist of four SpaceWire interfaces
- Connects to the extension connectors available on the GR-TMTC-MEZZ and GR-FDIR-MEZZ
- LVDS drivers available on the accessory board



# Reconfiguration Module

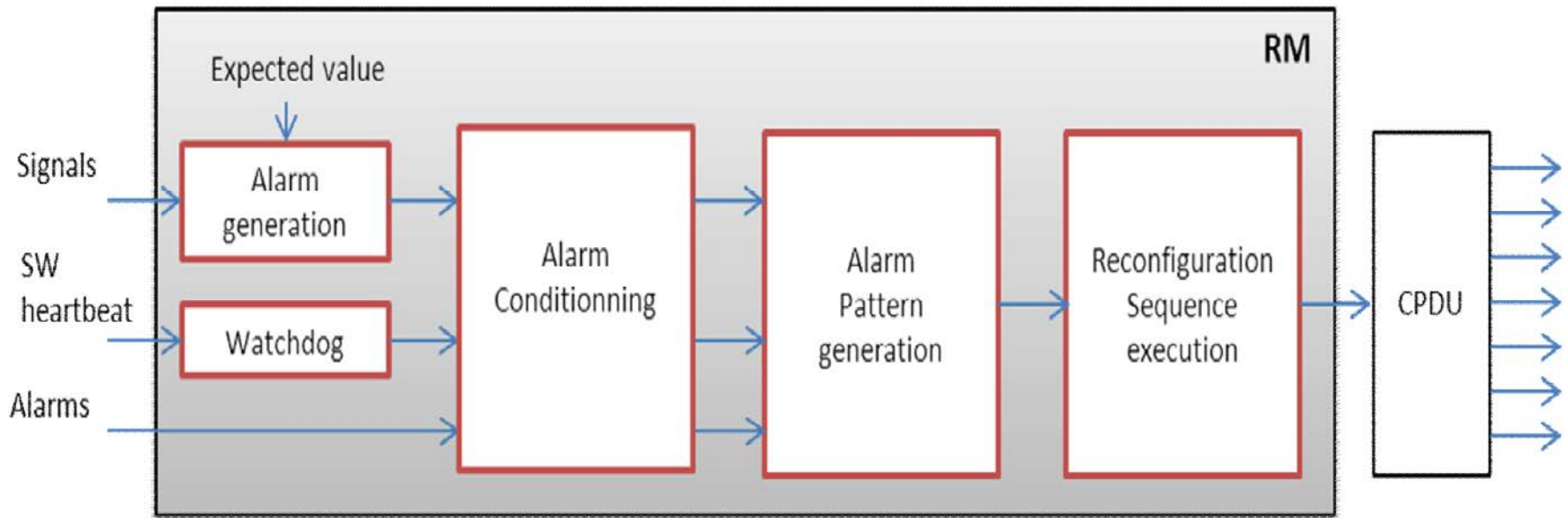
## Block Diagram



# Reconfiguration Module

## Function

- Various options are provided for configuration of the alarm inputs such as masking, shaping and activation delay
- Based on the alarm configuration and input signals (alarm event) the RM IP core will execute user defined reconfiguration sequences (internal and external commands)



# Reconfiguration Module

## Reconfiguration sequences

---

- The internal 64 commands are split into 2 blocks each containing 32 commands
  - Clear all pending
  - Clear current pending
  - Mask all, Mask current
  - Unmask all, Unmask current
  - Rearm Watchdog
  - Delay
  - Set state
  - Check alarm
  - Do nothing
- The external commands are formed into packets and sent to the command pulse distribution unit using PacketWire interface

## Function

---

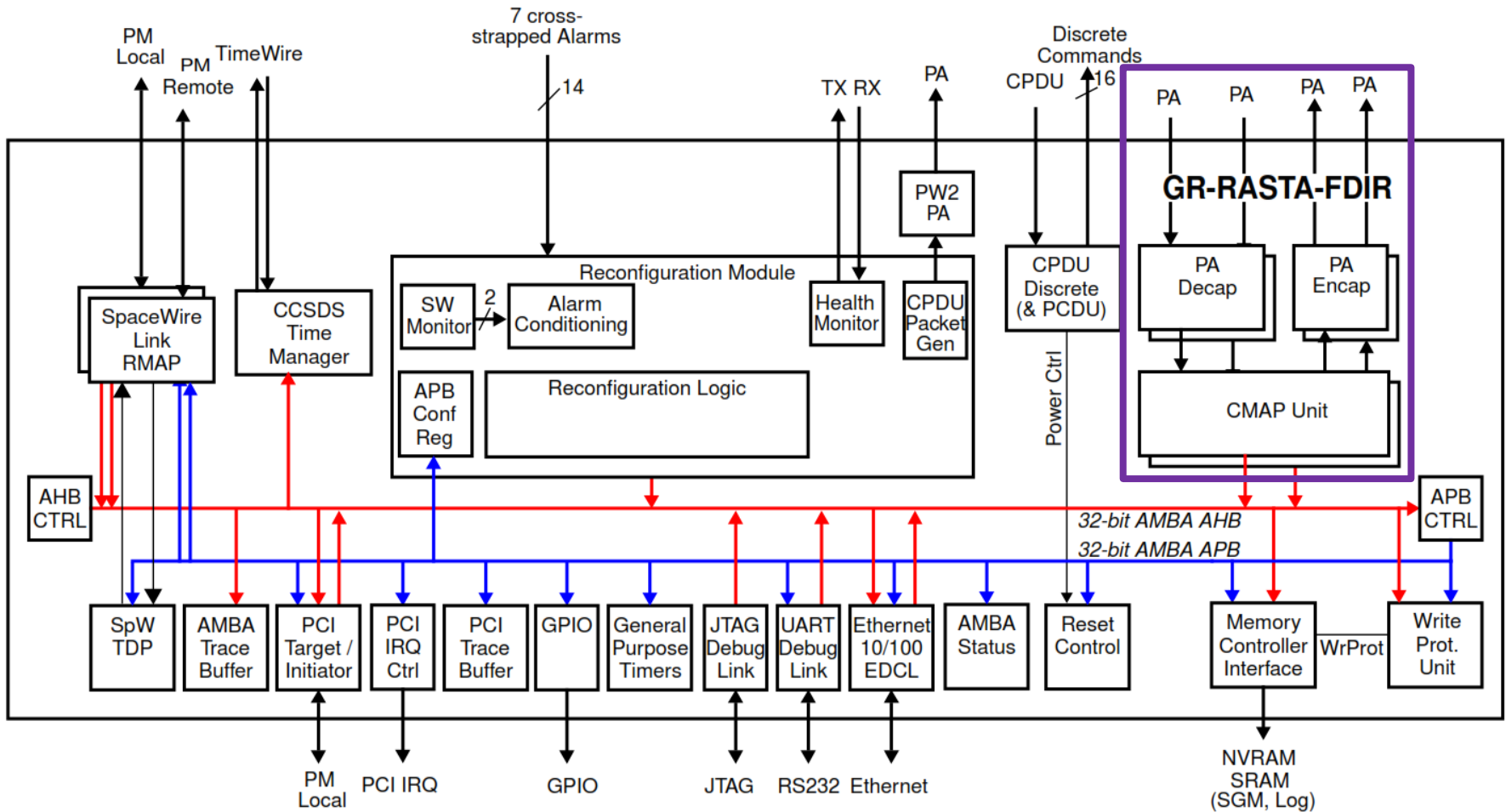
- The alarm pattern provides the address from which the reconfiguration sequences should be fetched.
- The AMBA AHB bus is used for retrieving the reconfiguration sequences in memory external to the core.
- When an alarm persists after performed reconfiguration the internal state-machine can be utilised to execute alternative reconfiguration sequences that depend on previously executed reconfigurations
- The alarms are logged with the time instance at which the alarm is triggered.

- Features

- External, Watchdog and Software alarms
- Alarm monitoring unit
- Alarm log (with time instance)
- Dedicated health link
- AMBA APB bus for configuration, control and status handling
- Reconfiguration sequences (internal and external commands)
- CPDU Packet Generator
- Configurable reconfiguration sequences

# Communication between FDIR and TMTc unit

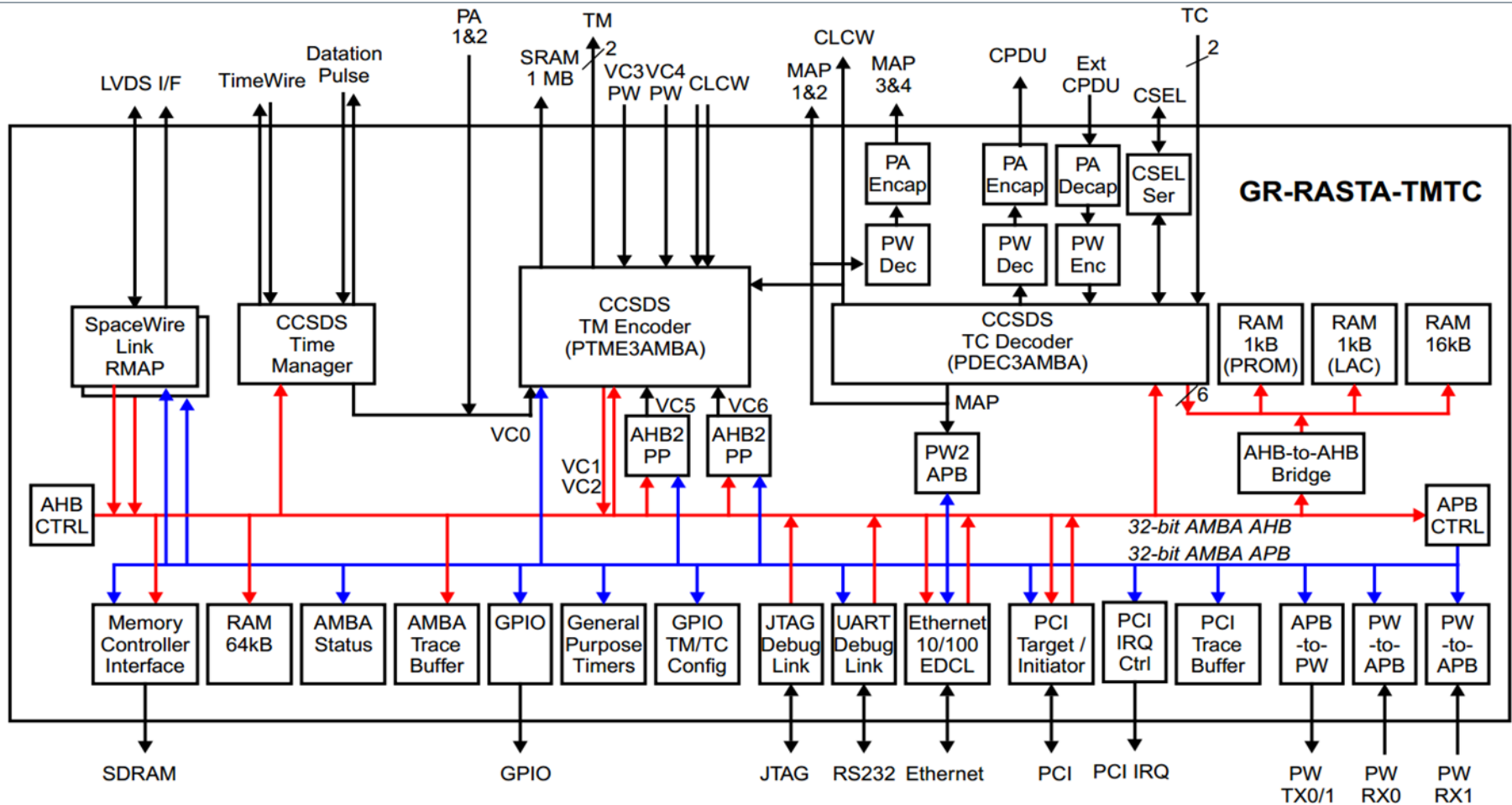
## FDIR unit





# Communication between FDIR and TMTC unit

## TMTC unit



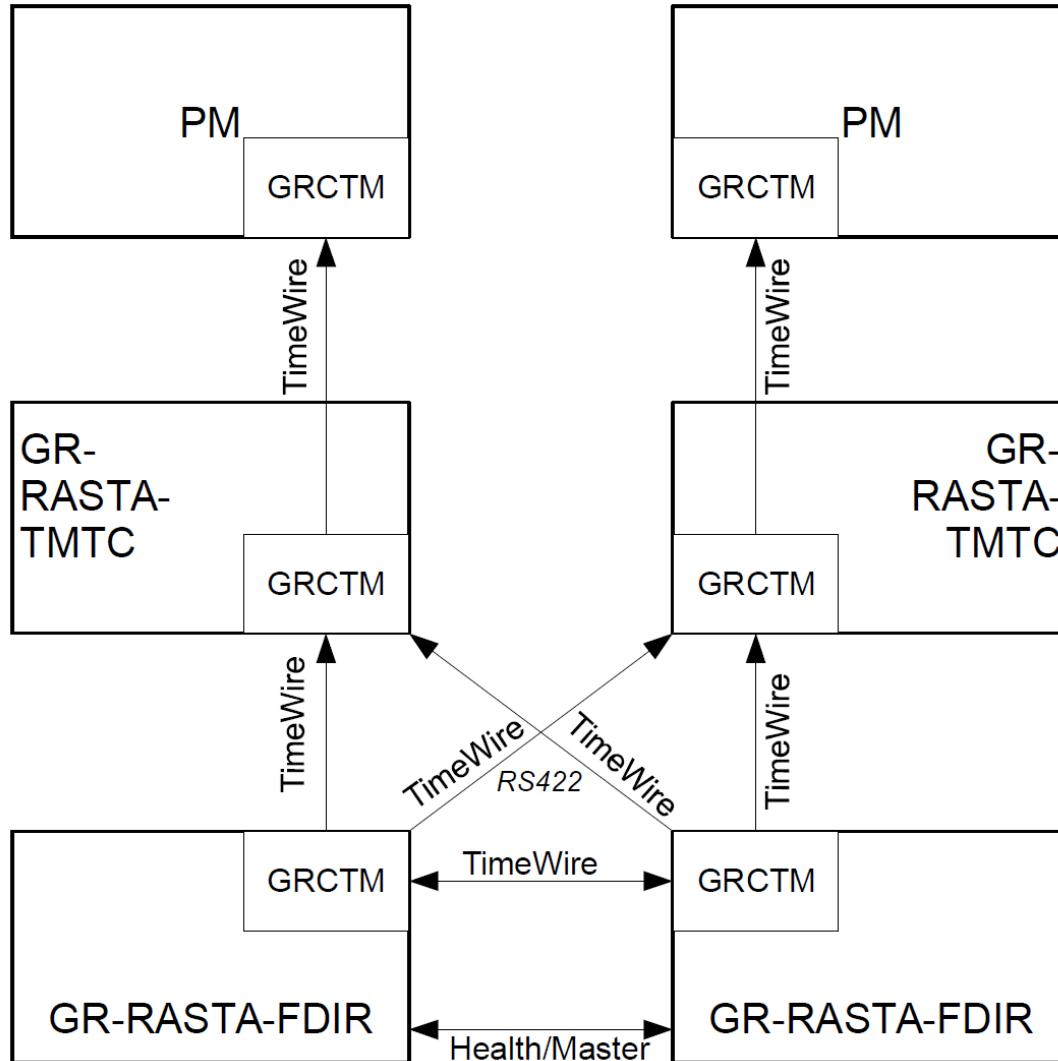
### CCSDS Memory Access Protocol Unit (CMAP)

---

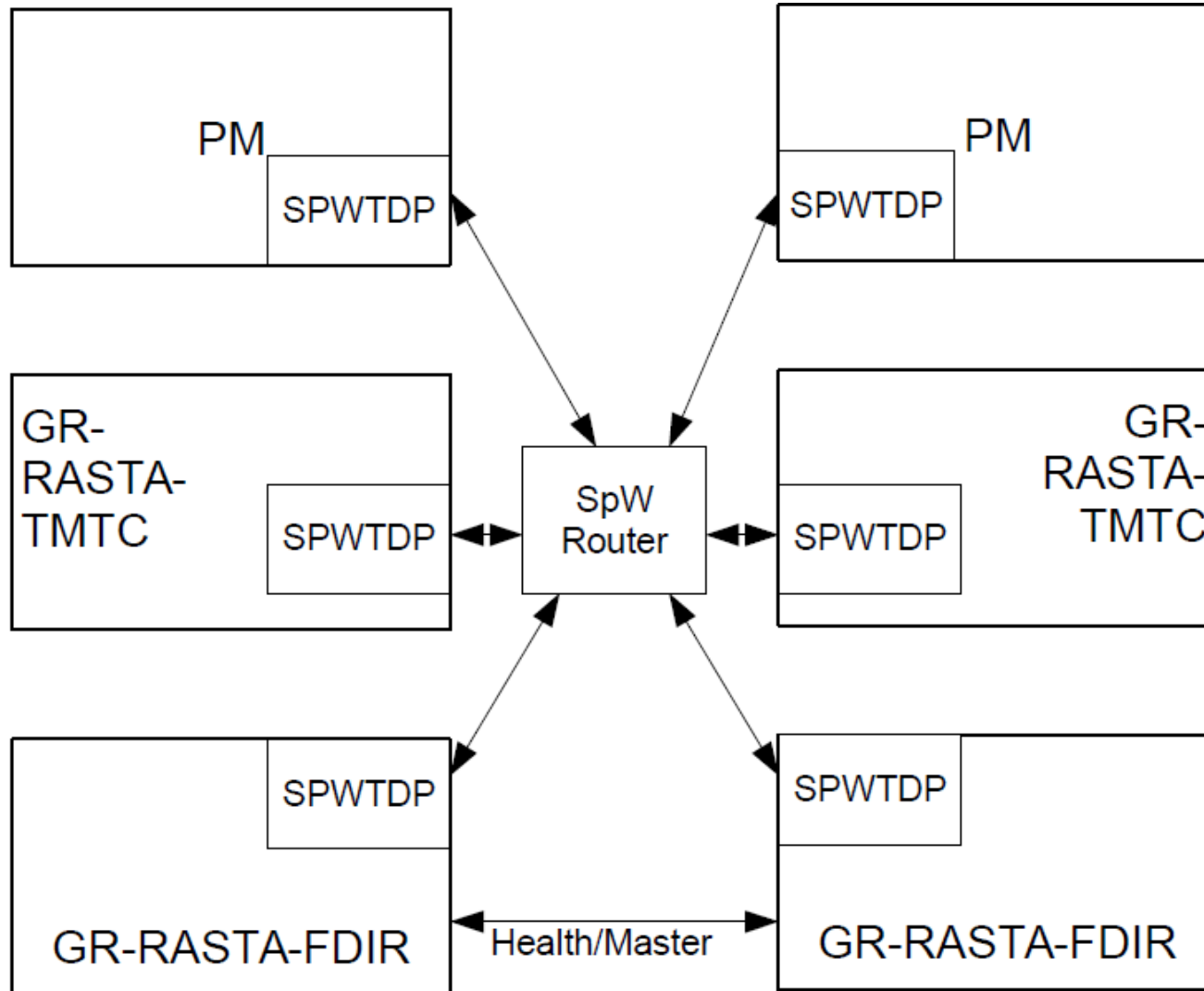
- CCSDS packet interface for performing AMBA bus accesses.
  - Packets conform to the Packet Utilization Standard (PUS)
  - Register load commands
  - Load and Dump Memory commands
- Hardware IP-core
  - Requires no software involvement
  - RM and spacecraft recovery
- Interfaces directly with a TC MAP and a TM VC
  - Asynchronous bit serial link (PacketAsynchronous) used for data transfers

- Automated TM packet generation
- Generation may be triggered by external events:
  - RM Error
  - RM Reconfiguration
  - Bus Error Interrupt (EDAC Errors)
  - Timer Units (for periodic status)
- Report Data for each trigger is fetched through AMBA Bus
- Available trigger Configuration options:
  - Severity Level
  - Transfer start address
  - Data length

## Using GRCTM

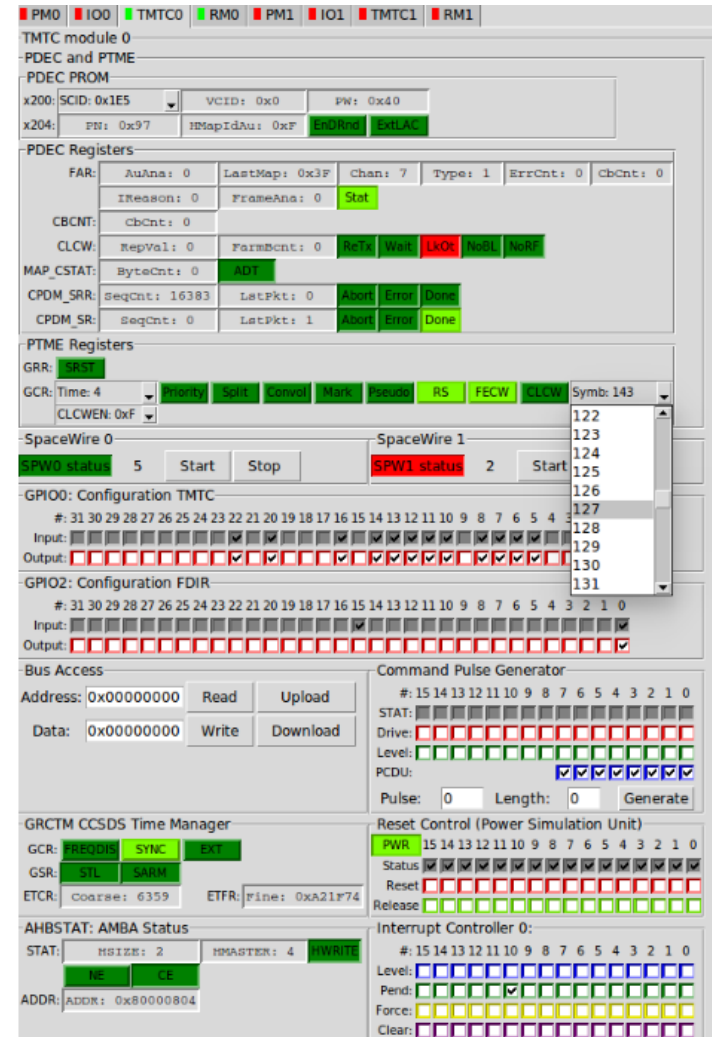


## Using SpaceWire



## Overview

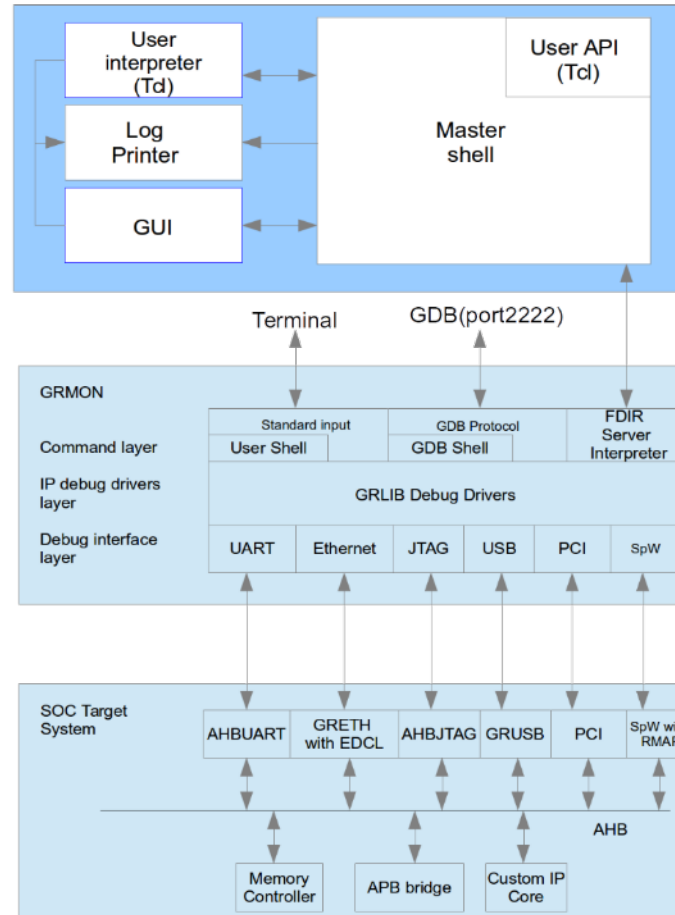
- 8 GRMON instances to connect all the RASTA units
- Continuous monitoring and presentation of register values on GUI
- Up to 20 hz pollrate
- Essential control of key IP-cores
- Load Programs to PM memory and execute/break
- Reset of individual IP cores
- Power management of RASTA units
- Inject error cases in the dual redundant system
- Work ongoing



# Test-bed Controller Tool

## Interaction with GRMON

Test Bed Controller Tool



# Test-bed Controller Tool

## Example 1 : Control and Monitor of RASTA-FDIR

The screenshot displays the FDIR Controller Tool interface, which is divided into several functional areas:

- Control Panel:** Includes buttons for 'Quit', 'Launch GRMON', 'Update period (ms): 400', 'Pause', and 'Script...'. Below this is a 'Debug link status' section with a color-coded bar for PM0, IO0, TMTCO, RM0, PM1, IO1, TMTC1, and RM1.
- Link cross-strapping Connection Status:** A table showing the status of connections between modules:

PM0 - TMTC1	Down	Enable	Disable
TMTCO - PM1	Down	Enable	Disable
RM0 - PM1	Down	Enable	Disable
PM0 - RM1	Down	Enable	Disable
- Log:** A scrollable text area showing system logs with columns for time (e.g., 67834 tt1), source (e.g., INTO), and message content (e.g., cpgen::logdiff pulse=0 len=0 coarse=0 fine=0x00000000 logent).
- Reconfiguration module 0:** A top status bar with color-coded indicators for PM0, IO0, TMTCO, RM0, PM1, IO1, TMTC1, and RM1. Below it are sections for 'Alarms' (a grid of status indicators for 31 channels), 'RM Status' (a grid of status indicators for various modules like CTRL, HCTRL, RME, STAT, WDRP, PSC, GSC), 'SpaceWire 0' (SPW0 status), and 'SpaceWire 1' (SPW1 status).
- Command Pulse Generator:** A section for generating pulses, including a grid for 15 channels, 'Drive' and 'Level' indicators, and 'PCDU' checkboxes.
- GPIO: Configuration:** A section for configuring GPIO pins, including a grid for 31 channels and 'Input'/'Output' checkboxes.
- Bus Access:** A section for reading and writing to a bus, with fields for 'Address' (0x00000000) and 'Data' (0x00000000), and buttons for 'Read', 'Write', 'Upload', and 'Download'.
- GRCTM CCSDS Time Manager:** A section for managing time, with fields for 'GCR', 'GSR', 'ETCR', and 'ETFR'.
- AHBSTAT: AMBA Status:** A section for monitoring AMBA status, with fields for 'STAT', 'HMASTER', 'HWRITE', 'NE', 'CE', and 'ADDR' (0x80000804).
- Reset Control (Power Simulation Unit):** A section for controlling power simulation, with a grid for 15 channels and 'PWR', 'Status', 'Reset', and 'Release' indicators.
- Interrupt Controller 0:** A section for monitoring interrupt controller status, with a grid for 15 channels and 'Level', 'Pend', 'Force', and 'Clear' indicators.



## Other features

- TCL Scripting Language
    - Used for developing tool
    - Powerful user scripts
  - Logs
    - User generated commands
    - Register changes
- Store value from 0x40000000 into set
    - set val [::fdirm::send pm0 {mem 0x40000000 4}]
  - Write value of set to 0x40000008
    - ::fdirm::send pm0 "wmem 0x40000008 \$val"

The screenshot displays a terminal window with a log of system events and command executions. On the left, a vertical list of timestamps and device identifiers (e.g., 1910740, 1916874, 1916951, 1916952, 1918322, 1918553, 1918554, 1932955, 1935826, 1936154, 1940581, 1940956, 1940964, 1940965, 1940969, 1940972, 1940984, 1940993) is shown. On the right, a larger window displays the corresponding log entries and command outputs. Annotations with colored boxes and arrows point to specific elements:

- PC Time in ms** (blue box) points to the timestamp column.
- RASTA device** (red box) points to the device identifier column.
- User shell input** (green box) points to the command execution area at the bottom.
- Trigger RM Alarm** (yellow box) points to the command `grrm0::set cmd regname=Mask field={Msk} flipbit=1 regwr=0x5`.
- Reconfiguration seq. unmasks alarm input** (purple box) points to the command `grrm0::diff regname=Mask fields="{Msk1 1}" regval=0x00000000`.
- RM Reconfiguration Log** (blue box) points to the command `grrm0::logdiff pattern=1 coarse=90 fine=0x64407e logentry=6`.
- Simulated Discrete Pulse Generator Logs** (red box) points to the command `cpgen::logdiff pulse=0 len=0 coarse=90 fine=0xcb logentry=6`.

```
1910740 rmo info
1916874 rmo info
1916951 rmo info
1916952 rmo info
1918322 rmo info
1918553 rmo info
1918554 rmo info
1932955 rmo info
1935826 rmo info
1936154 rmo info
1940581 rmo info
1940956 tt0 info
1940964 rmo info
1940965 rmo info
1940969 rmo info
1940972 rmo info
1940984 tt0 info
1940993 tt0 info

resetctrl::release dev=rm0 flipbit=2 regwr=0x55000004
resetctrl::release dev=rm0 flipbit=2 regwr=0x55000000
grrm0::diff regname=RME fields="{RMER 0} {ITER 0}" regval
grrm0::diff regname=STAT fields="{IDD 0}" regval=0x00000002
grrm0::set cmd regname=CTRL field={CONFDONE} flipbit=1 regw
grrm0::diff regname=CTRL fields="{CONFDONE 1}" regval=0x000
grrm0::diff regname=STAT fields="{CD 1}" regval=0x00000003
grrm0::diff regname=STAT fields="{IDD 1}" regval=0x00000007
grrm0::set cmd regname=Mask field={Msk} flipbit=1 regwr=0x5
grrm0::diff regname=Mask fields="{Msk1 1}" regval=0x00000000
grrm0::set cmd regname=CTRL field={CxEAlarm N} flipbit=12 r
pdec::diff regname=CPDM_SR fields="{SeqCnt 0} {LstPkt 1} {C
grrm0::diff regname=CTRL fields="{CxEAlarm N 1}" regval=0x00
grrm0::diff regname=Mask fields="{Msk1 0}" regval=0x00000000
grrm0::logdiff pattern=1 coarse=90 fine=0x64407e logentry=6
cpgen::logdiff pulse=0 len=0 coarse=90 fine=0xcb logentry=6
cpgen::logdiff pulse=0 len=0 coarse=90 fine=0xcb logentry=6
irqmp0::diff regname=Pend fields="{Pnd10 1}" regval=0x00000000
```

- Controller tool log the events, the time originates from the timer on the PC running the tool
- Additionally time units (CUC Time Manager) in all modules
- Dedicated time tag units for alarm events and command generation
  - To accurately measure time difference between alarm events and command generation

The screenshot shows a terminal window with a log of events. The log entries are as follows:

```
1910740 rm0 info resetctrl::release dev=rm0 flipbit=2 regwr=0x55000004
1916874 rm0 info resetctrl::release dev=rm0 flipbit=2 regwr=0x55000000
1916951 rm0 info grrm0::diff regname=RME fields="{RMER 0} {INITER 0}" regval=0x00000000
1916952 rm0 info grrm0::diff regname=STAT fields="{IDD 0}" regval=0x00000002
1918322 rm0 info grrm0::set cmd regname=CTRL field={CONFDONE} flipbit=1 regwr=0x00000000
1918553 rm0 info grrm0::diff regname=CTRL fields="{CONFDONE 1}" regval=0x00000000
1918554 rm0 info grrm0::diff regname=STAT fields="{CD 1}" regval=0x00000003
1932955 rm0 info grrm0::diff regname=STAT fields="{IDD 1}" regval=0x00000007
1935826 rm0 info grrm0::set cmd regname=Mask field={Msk} flipbit=1 regwr=0x55000000
1936154 rm0 info grrm0::diff regname=Mask fields="{Msk1 1}" regval=0x00000000
1940581 rm0 info grrm0::set cmd regname=CTRL field={CxEAlarm_N} flipbit=12 regwr=0x00000000
1940956 tt0 info pdec::diff regname=CPDM_SR fields="{SeqCnt 0} {LstPkt 1} {Dlts 0}" regval=0x00000000
1940964 rm0 info grrm0::diff regname=CTRL fields="{CxEAlarm N 1}" regval=0x00000000
1940965 rm0 info grrm0::diff regname=Mask fields="{Msk1 0}" regval=0x00000000
1940969 rm0 info grrm0::logdiff pattern=1 coarse=90 fine=0x64407e logentry=0
1940972 rm0 info cpgen::logdiff pulse=0 len=0 coarse=90 fine=0xcb logentry=0
1940984 tt0 info cpgen::logdiff pulse=0 len=0 coarse=90 fine=0xcb logentry=0
1940993 tt0 info irqmp0::diff regname=Pend fields="{Pnd10 1}" regval=0x00000000
```

Annotations on the left side:

- PC Time in ms (blue box) points to the time tags on the left.
- RASTA device (red box) points to the device names (rm0, tt0) on the left.
- User shell input (green box) points to the prompt 'fdirm::send' at the bottom.

Annotations on the right side:

- Trigger RM Alarm (yellow box) points to the 'grrm0::diff regname=STAT' entries.
- Reconfiguration seq. un.masks alarm input (purple box) points to the 'grrm0::set cmd regname=Mask' entry.
- RM Reconfiguration Log (blue box) points to the 'grrm0::diff regname=Mask' entry.
- Simulated Discrete Pulse Generator Logs (red box) points to the 'cpgen::logdiff' entries.

# Test-bed Controller Tool

## Example 2 : Control and Monitor of RASTA-TMTC

The screenshot displays the FDIR Controller Tool interface, which is used for controlling and monitoring the RASTA-TMTC system. The interface is divided into several main sections:

- Control Panel:** Located at the top left, it includes buttons for 'Quit', 'Launch GRMON', 'Update period (ms): 800', 'Pause', and 'Script...'. Below these are 'Debug link status' and 'Link cross-strapping' options.
- Link cross-strapping:** A table showing connection status for various modules:
 

Connection	Status	Enable	Disable
PM0 - TMTC1	Down	Enable	Disable
TMTC0 - PM1	Down	Enable	Disable
RM0 - PM1	Up	Enable	Disable
PM0 - RM1	Down	Enable	Disable
- Log:** A large text area at the bottom left showing a stream of log messages. The messages include timestamps and module identifiers (e.g., '1431436837507 rm0 info').
- System Status:** A top right panel with a color-coded legend (PM0, IO0, TMTC0, RM0, PM1, IO1, TMTC1, RM1) and a 'TMTC module 0' status indicator.
- PDEC and PTME:** Configuration panels for PDEC PROM (x200, x204) and PDEC Registers (FAR, CBCNT, CLCW, MAP\_CSTAT, CPDM\_SRR, CPDM\_SR).
- PTME Registers:** Configuration for GRR, GCR, and CLCW.
- SpaceWire:** Configuration for SpaceWire 0 and SpaceWire 1, including SPW0 status and SPW1 status.
- GPIO:** Configuration for GPIO0 (TMTC) and GPIO2 (FDIR), showing input and output pin states.
- Bus Access:** A panel for reading and writing data to a specific address (0x00000000).
- Command Pulse Generator:** Configuration for generating pulses, including pulse width and length.
- GRCTM CCSDS Time Manager:** Configuration for GCR, GSR, and ETCR.
- AHBSTAT: AMBA Status:** Configuration for STAT, ADDR, HSIZE, HMASTER, and HWRITE.
- Reset Control (Power Simulation Unit):** Configuration for PWR, Status, Reset, and Release.
- Interrupt Controller 0:** Configuration for Level, Pend, Force, and Clear.

- RTEMS 4.10
- Several examples available (SpW, CAN, 1553, ...)
- New FDIR test applications are developed
- RTEMS drivers required for Reconfiguration Module and GR-RASTA-FDIR (PCI target driver) available
- Validation of the Test-bed

# Features

## Overall activity features

---

- Evaluation platform for a dual redundant system
- Allow upgrade and evolution of architecture
- A user interface able to access the system resources and easy to use
- Ability to perform recording of simulation events and results
- Ability to measure the switching time between system re-configurations
- Ability to inject errors and to simulate failure mode
- Ability to time stamp events
- Configurable cross strap
- Generate on board time and provide the time synchronization and distribution function
- Example application software

- In this activity, a dual redundant architecture based on the current single chain RASTA architecture is developed.
- An RM IP core is developed together with a new RASTA-FDIR hardware module that will complement the existing RASTA hardware by providing the necessary functionality to support dual redundant architecture.
- Testbed controller tool will be developed which will enable the users to load, run and analyse applications and simulations, as well as inject error cases in the dual redundant system.

Thank you for listening!

<http://www.Cobham.com/Gaisler>  
[info@gaisler.com](mailto:info@gaisler.com)