Prototype SpaceWire RMAP Boot Software for Secondary Processor

Ana Rugina, Technical Officer

Dr. Mattias Holm, Project Manager

Alberto Ferrazzi, Developer

TERMA

ALLIES IN INNOVATION

Study Overview

- Motivations
  - SpW increasingly used as command/control link between platform and instruments
  - Expected benefits: reduced complexity on instrument side, less non-volatile memory
- Objectives
  - System impact analysis
  - Prototype
- Small investment activity
  - Budget: 75k
  - Duration: 6 months (estimated) / 7.5 months (actual)
- Contractor: TERMA GmbH

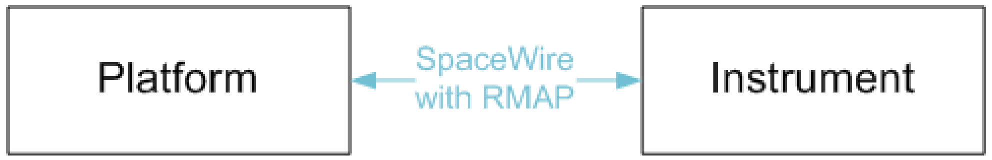
Outline

- Introduction
  - Objectives of the activity
  - Traditional Boot Software
  - Predecessor Project
  - What is SpaceWire and RMAP
- Project
  - Analysis
    - Objectives
    - Configurations
      - Application Software deployment via RMAP
      - Boot and Application Software deployment via RMAP
      - Full boot via RMAP
    - Recaps
  - Prototype
    - Overview
    - Architecture
    - Nominal Boot Sequence
    - To be noted
- Achievements

Introduction – Objectives of the activity

The objectives of this activity were:

- Analyze possible usages of RMAP over a SpaceWire link between the platform and an instrument in order to improve/simplify the boot of the instrument



- Implement a prototype that demonstrates the most advantageous configuration where RMAP is used to boot the instrument

Introduction – Traditional Boot Software

Boot software main objective: prepare the computer for the execution of the application software and launch the application software

Modes
- Nominal: nominal boot procedure
- Standby: interactive mode to perform maintenance
- Monitor: ground

Spacecraft Boot Software processes (nominal mode):
- Initialization: sets the hardware to a well know state
- Test: perform tests on hardware and enable fundamental devices
    - Produce a detailed boot report
- Launch: launches the Application Software Image (ASW) launch
    - Configuration (i.e. stack setup)
    - Check and launch of the selected ASW image

Other tasks:
- Manage ASW images
- Standby mode
    - ASW image patching
    - Debug

Introduction – Traditional Boot Software

Hardware requirements

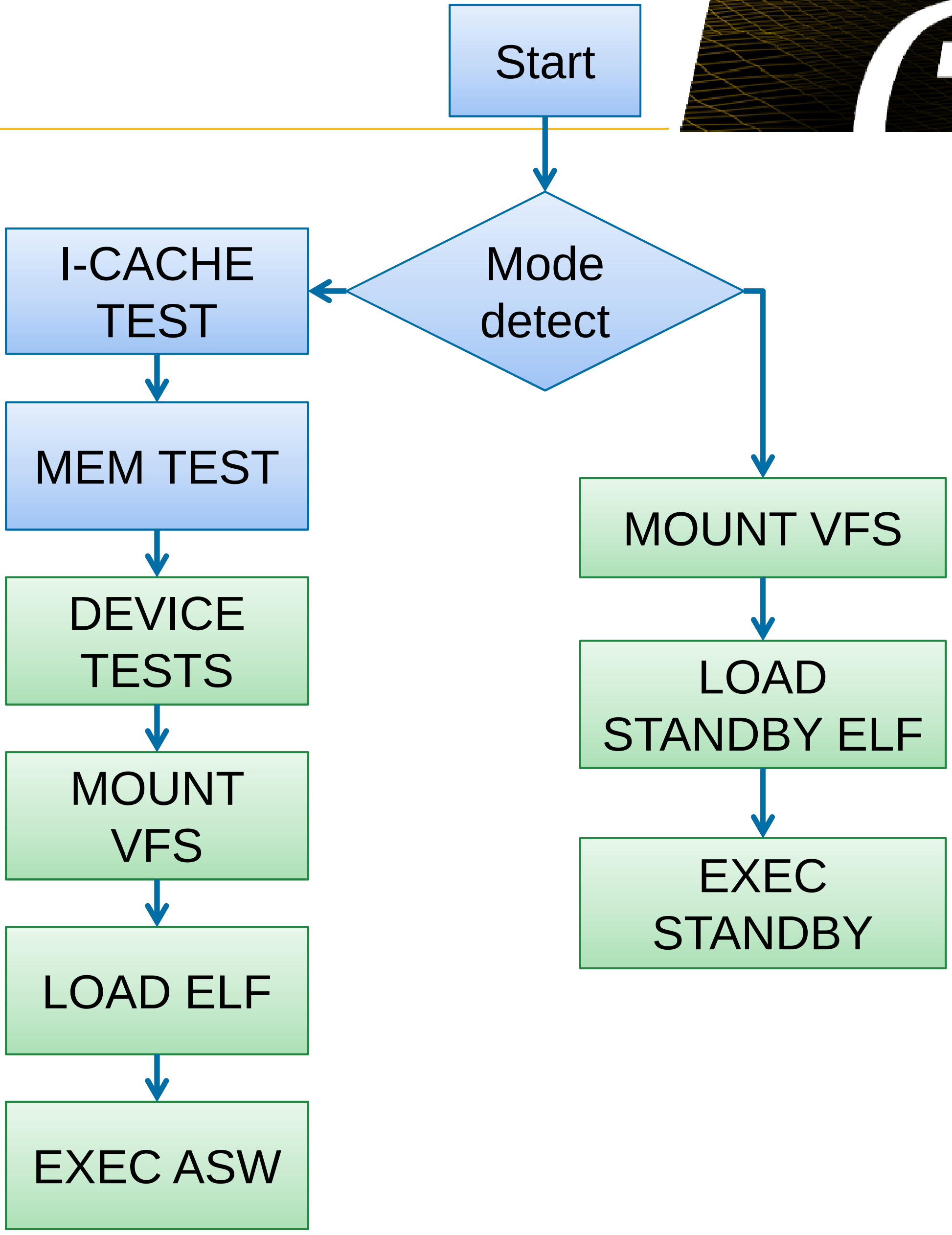
- Non-volatile memory to store
  - Boot software
  - ASW images
- RAM
- CPU

Instrument Boot

- The booting of the instrument is sort of independent from the platform (the platform has to turn on the instrument and the instrument will boot on its own)

Predecessor Project

- OBC Initialisation Sequence (ITT/AO/1-7323/12/NL/LvH)
- Purpose to verify the SAVOIR generic boot specification by developing a boot software from it.
- Booting LEON2 CPU including nominal and standby mode.
- Basic CPU init (register initialisation), memory tests, I-CACHE test and stack initialisation written in assembler
- Device tests, virtual file system and ELF loader written in C.
- Very modular system (BSP support)



Introduction – What is SpaceWire and RMAP

SpaceWire

- Standard for high-speed links and networks on Spacecrafts
- Up to 200 MBits/s
- Point to point
  - Wormhole routing
- Serial
- Full-duplex
- Basic unit is Transfer packets

Remote Memory Access Protocol (RMAP)

- Command / Response protocol to access memory mapped resources
  - Read
  - Write
  - ReadWrite
- Supports:
  - Verified writes
  - Packets up to ~16MB
- Can be implemented in hardware or software

Analysis - Objectives

The objective of the analysis was to:

- Find all possible configurations where the RMAP over SpaceWire can be used to boot an instrument
- Analyze the impact of each configuration
  - Platform
  - Instrument
  - Validation Infrastructure
  - Performance

- Output
  - Technical note

Analysis – Configurations

Three configurations were discovered:

- Instrument with PROM
  - Application software deployment via RMAP
- Instrument without PROM
  - Boot and application software deployment via RMAP
  - Full boot via RMAP

Application Software deployment via RMAP

- It is the closest configuration to the traditional boot software
  - The boot software image of the instrument is stored in PROM and executed automatically form there when the instrument is booted
  - The instrument boot software performs all the tasks of a traditional boot software up to the moment where it has to launch the application software

- RMAP is used to
  - Upload the instrument application software
  - Transfer the boot-report
  - Communicate between platform and instrument

- Nominal procedure
  1. Platform processor starts and runs a nominal boot sequence
  2. The instrument processor starts and runs the instrument boot program automatically
  3. The boot program runs tests
  4. The boot program notifies the platform it is ready to
  5. The flight software on the platform uses the RMAP to upload the application software to the instrument RAM and signals the instrument boot software to resume
  6. The instrument boot program launches the uploaded application software

Application Software deployment via RMAP

- Requirements
  - No additional requirement with respect to traditional case
- Advantages
  - Less memory required in the instrument to store the application software (but more in platform)
  - Simpler instrument boot software (does not have to manage ASW image)
- Disadvantages
  - PROM and boot software are still required

Boot & Application Software deployment via RMAP

- The boot software is not stored on the instrument, it is injected in RAM via RMAP and the instrument executes it from there

- RMAP is used to
  - Perform memory test on a sub area of memory
  - Upload the instrument boot software and command execution of it
  - Upload the instrument application software
  - Transfer the boot-report
  - Communicate between platform and instrument

Boot & Application Software deployment via RMAP

Nominal procedure:

1. Any processor of ICUs starts and halts / waits immediately.
2. The platform processor starts and runs a normal boot sequence that leads to flight software start up
3. The flight software boots the instruments. For each instrument the platform performs the following steps via RMAP:
   - Turn on EDAC on the ICU RAM
   - Upload the ICU boot image
   - Turn off EDAC on the ICU RAM
   - Run a checksum test on the uploaded image (code and data segment) to prevent execution of bad code
   - Resumes ICU processor execution and wait a notification (like receiving the boot-report) that means that the ICU is ready to receive the ASW
4. The ICU boot software executes the RAM test and EDAC diagnostics on every location except the text segment
5. Basing on the EDAC validity flag set by the platform during ICU RAM, either enable or disable EDAC (basically this allows the platform to force the ICU to leave the EDAC off in case the unit or cells have failed)
6. Perform the rest of self-tests in C code
7. The ICU provides the boot-report to the platform via RMAP and wait for the ASW to be provided

Boot & Application Software deployment via RMAP

- Requirements
  - The SpaceWire link between platform and instrument must go in running state without intervention of instrument CPU
  - Possibility to control CPU execution via RMAP
  - Full access to RAM from RMAP

- Advantages
  - Instrument does not require a PROM or any other memory beside RAM
  - Simpler instrument boot software
  - Can boot multiple instruments in parallel

- Disadvantages
  - A boot software is still required

Full boot via RMAP

- There is no boot program to be run on the instrument
- The platform performs the initialization, the tests, and uploads the ASW directly in RAM of the instrument via RMAP

- RMAP is used to
    - Initialize the instrument
    - Perform tests
    - Upload the application software

Full boot via RMAP

Nominal procedure:

1. ICU processor starts and halts / waits immediately
2. Platform processor starts and runs a normal boot sequence that leads to the start-up of the flight software
3. The flight software boots the instruments. For each instrument the platform performs the following steps via RMAP:
   - Turn on EDAC on the ICU RAM
   - Perform a ICU RAM test
   - Depending on RAM test results and configuration turn off EDAC on the ICU RAM
   - Perform other applicable tests (i.e. L2C)
   - Write the Boot-Report in ICU RAM
   - Copy the ICU-ASW directly into ICU-RAM and resume processor execution
4. ICU-ASW setup the processor registers and does necessary self-tests for functionality not tested remotely, updating the boot-report:
   - Setup integer unit, floating point unit, on-chip processor registers, MMU registers
   - Setup a default handler for traps
   - Test I- and D-caches
   - Test Interrupts
5. ICU-ASW notifies flight software when it has finished booting preliminary activities

Full boot via RMAP

- Requirements
  - The SpaceWire link between platform and instrument must go in running state without intervention of instrument CPU
  - Possibility to control CPU execution via RMAP
  - Full access to RAM from RMAP

- Advantages
  - Very flexible: all boot operations are handled by platform software
  - No specific boot software has to be implemented for the instrument
- Disadvantages
  - Slow: the memory test is the most time-consuming step in the boot process. Running it over the SpaceWire link would have big impact on performances
  - May not configure/test internal processor registers, D- and I- caches. This task has to be shifted to ASW
  - Cannot boot multiple instrument in parallel

Analysis – Recap

- Memory Test has major impact on boot time

- Common advantage(s) of introducing RMAP:
  - Instrument boot software simplification
    - No need of a StandBy mode: post mortem investigations performed via RMAP
    - No need to store an application software image

- The verification infrastructure has to provide SpaceWire support (in the “instrument with non-volatile memory configuration” it may be avoided)

- Impact on platform
  - More memory required (host instruments ASW and/or boot software)
  - Increase of software complexity

Analysis – Recap

- The comparison table provided as part of the technical analysis can be used to identify the best configuration

Boot time estimations for:

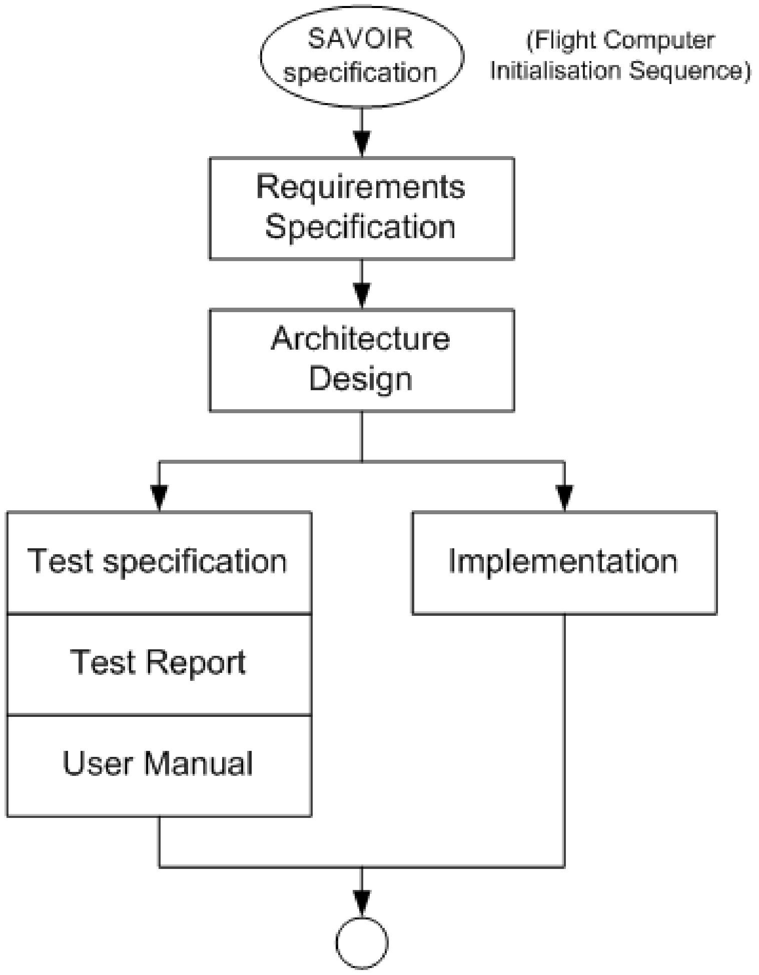
- ICU RAM: 1024 MB
- SpaceWire: 100 MB/s
- ICU ASW size: 1 MByte
- ICU Boot SW size: 25 KByte
- ICU Clock: 100 MHz
- ICU bus: 32 bit

The prototype implemented boots in 10-12 seconds (16MB RAM, 50MHz processor)

	Standard	RMAP deploys ASW	RMAP deploys ICU boot-sw and ASW	Full boot through RMAP
ROM / PROM Memory on ICU	Required ~32 KB for boot software	Required < 32KB (simpler boot SW)	Not required	Not required
SpaceWire link with RMAP active at reset without processor intervention	Not applicable	Not required	Required	Required
Control of the CPU execution through RMAP (i.e. using signals or DSU)	Not applicable	Not required	Required	Required
ICU Boot Time		~1.300933424 min	~1.300987957 min	~53.24353549 min
Multiple ICU boot	Parallel ICU booting	Mostly parallel	Mostly parallel	Sequential

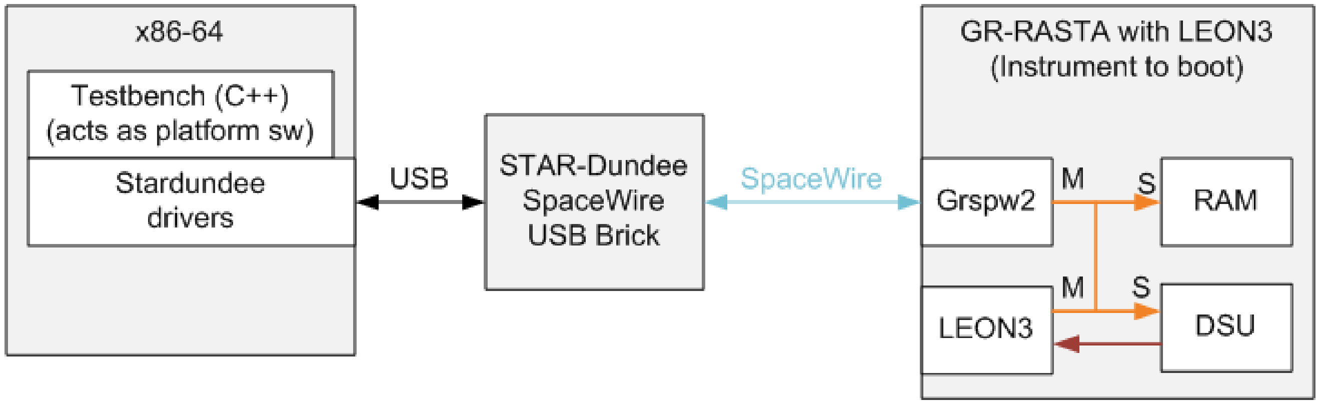
Prototype - Overview

- Implements the most advantageous configuration, where a boot software is uploaded to instrument RAM



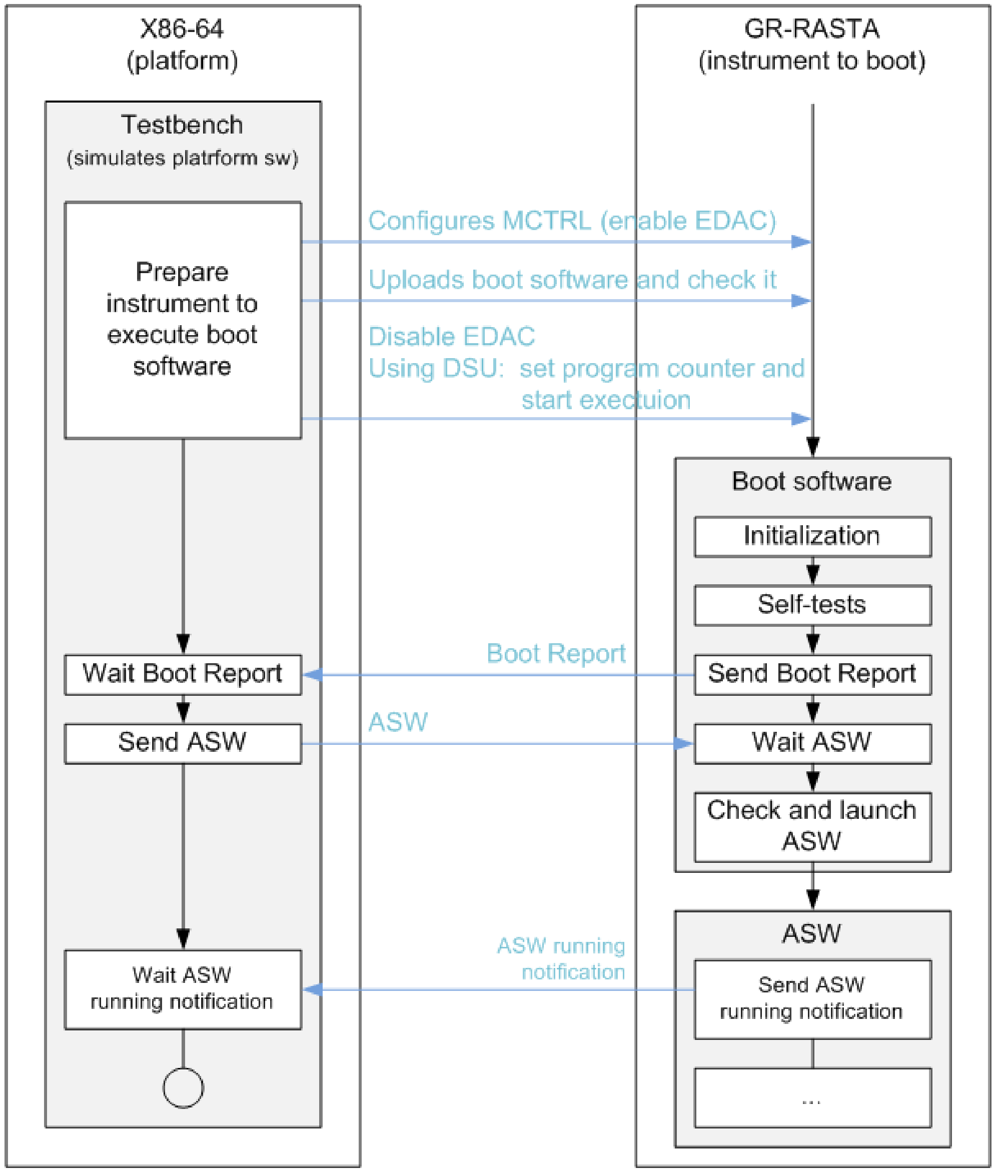
Prototype – Architecture

- Hardware used:
  - Platform: x86-64
  - Instrument: GR-RASTA
  - SpaceWire link: STAR-Dundee SpaceWire USB Brick
- Software implemented:
  - Boot software developed extending OBCINIT boot software
    - New linker script, new BSP for LEON3 with self tests
    - Build system extended to supports compilation of the traditional boot software, or any of the RMAP versions that support compile).
  - Testbench acts as platform flight software
    - Base class can be derived and methods reused to implement custom platform logic (template pattern)
  - Instrument test ASW


x86-64
Testbench (C++) (acts as platform sw)
Stardundee drivers
USB
STAR-Dundee SpaceWire USB Brick
SpaceWire
GR-RASTA with LEON3 (Instrument to boot)
Grspw2
M
S
RAM
LEON3
M
S
DSU

Prototype – Boot sequence

- Testbench reads ELF images of boot software and application software and uploads them as segments
- CRC used to check integrity
- Platform configures the boot software by uploading the configuration in the reserved memory page prior starting the boot software
  - Example: fast boot path (skip memory and cache tests)



Prototype – To be noted

- Requires Debug Support Unit accessible via RMAP
  - Set the program counter to the boot software entry point
  - Resume execution of the boot software
- Requires platform intervention for warm boot
- Do not uses LEON3 plug & play

Achievements

- A comparison table presenting the advantages/disadvantages of the three configurations has been produced
    - Shows the impact from different perspectives
    - Confirms that the introduction of RMAP in the boot process can simplify the instrument hardware and boot software
    - Shows that the impact on the platform is reasonably low
        - The platform shall already contain code to address the tasks shifted from the ICU boot software
        - Hardware: platform requires only more memory
- The most advantageous configuration has been identified
- A working prototype that demonstrates the most advantageous configuration has been implemented
    - Ported the boot software from ERC32 to LEON3

- Other items produced
    - Requirements specification derived from SAVOIR Flight Computer Initialisation Sequence Generic Specification (http://savoir.estec.esa.int/)
    - Design Architecture
    - Test specification
    - Test report
    - Operations manual

Meet us at

www.terma.com

www.terma.com/press/newsletter

www.linkedin.com/company/terma-a-s

www.twitter.com/terma_global

www.youtube.com/user/TermaTV