GOVERNMENT OF DUBAI

MOHAMMED BIN RASHID SPACE CENTRE

Development of the Emirates Lunar Rover Camera Interface Board

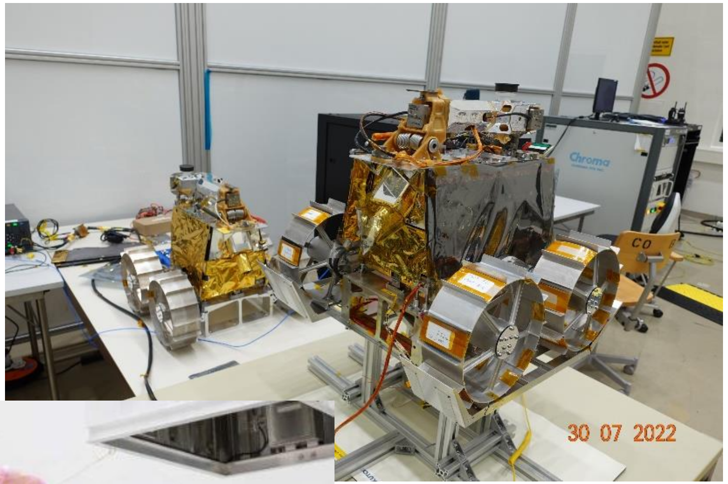
Mohammed Saeed Khoory – mohammed.khoory@mbrsc.ae

Santhosh Amilineni – santhosh.kumar@mbrsc.ae

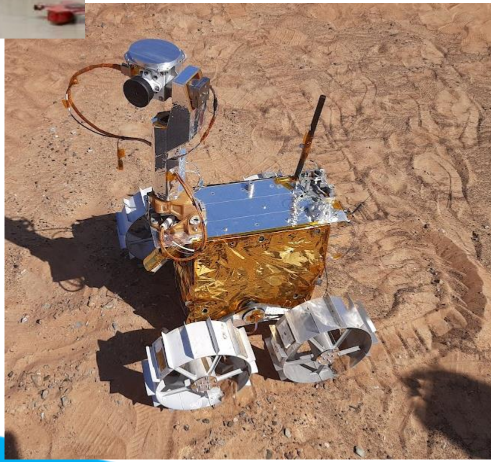
GR740 USER DAY 2022

Mission Overview

- Emirates Lunar Mission is the UAE’s first Lunar and Rover mission
- Small, low cost, manually driven 12kg rover, designed from scratch with COTS parts
- Carried with iSpace M1 Lander
- Launched on December 11 (just 2 days ago!)
- Technology Demonstration and Science mission
  - 3 Visible Cameras including 1 microscope
  - Thermal camera
  - Langmuire Probe
  - Material Adhesion (MAD) experiment
  - Deployable Mast and 2-axis gimbal for the primary camera
- Two communication systems
- Originally meant to launch in 2021
  - Tight schedule, tight budget, tiny team!

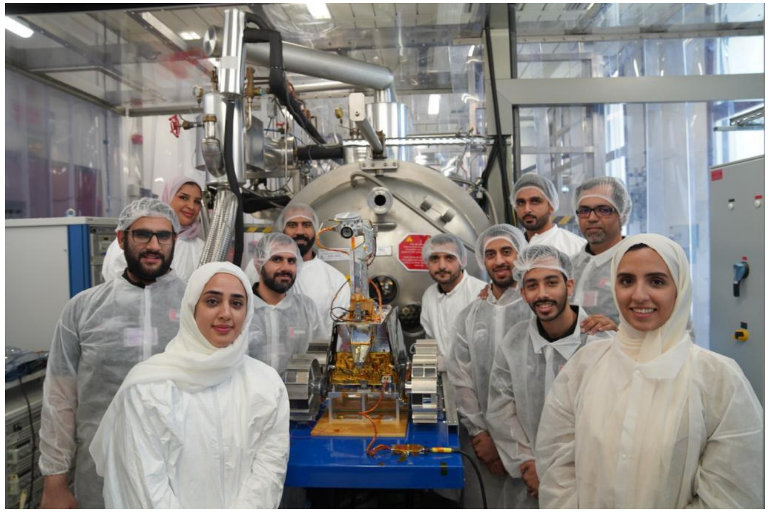


Credit: iSpace

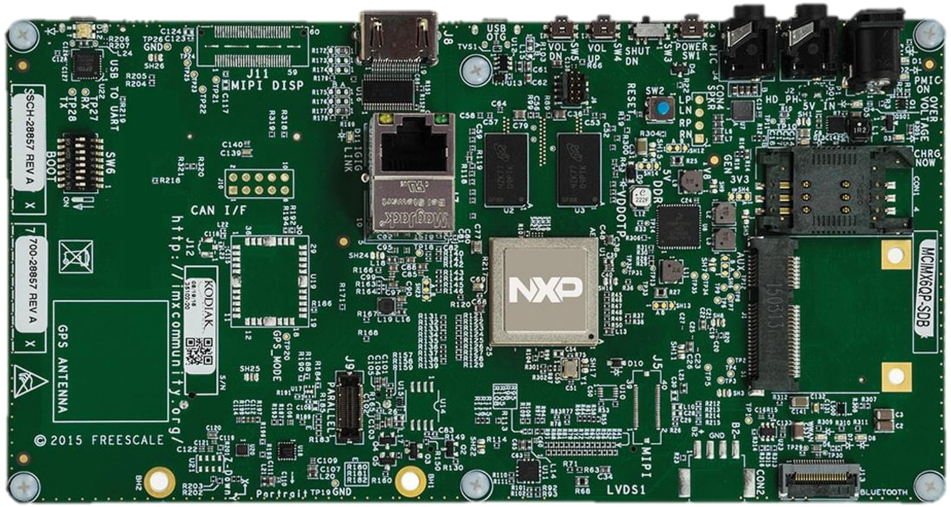


The ELM Team

- Space Robotics Lab (SRL) – started as an R&D team at MBRSC
- Small core team of around 10-14 people
  - 4 software engineers, two of which are dedicated to on-board software
  - Santhosh Amilineni dedicated to developing software for the CIB
  - Many were fresh graduates when they first started
  - Many had previous experience with satellites
  - Space robotics is a completely new field to all
- Small, agile team and development



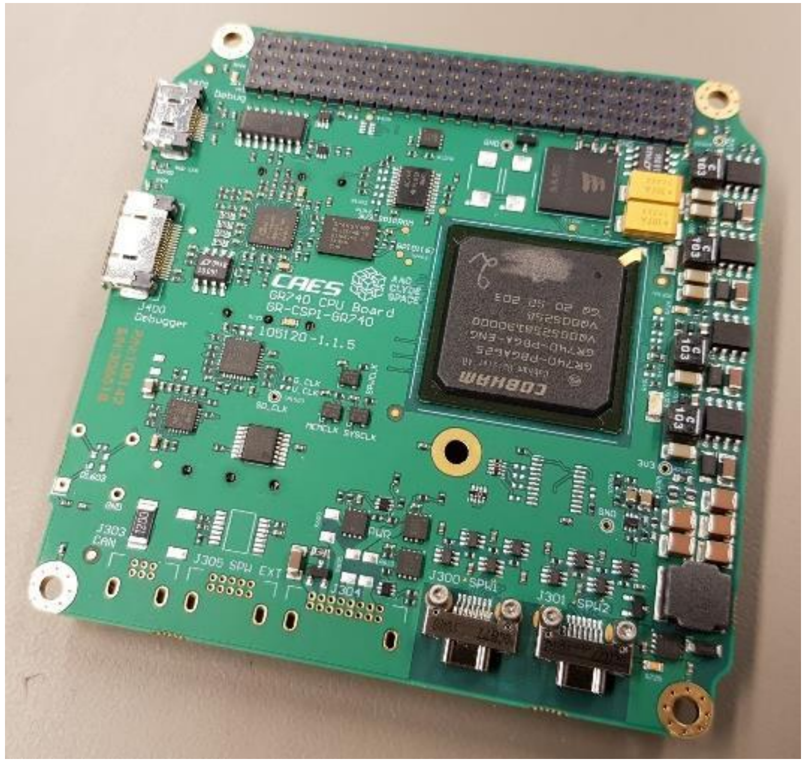
Initial Plans



- Initally planned to use a single ARM Cortex A8 iMX-6 board
- Off-the-shelf CSI/MIPI cameras originally planned
  - i.e. COTS Smartphone cameras
  - Easy to use and straightforward
  - Not great for science or space
  - Required a qualification process
- Space Qualified SD card for storage
- Yocto Linux OS
- Space-grade 3D-Plus cameras were later introduced into the mission
  - Caspex cameras provided through a collaboration with CNES
  - Spacewire based
- Space-grade cameras meant that we needed a more reliable solution to continue with iMX-6
  - Redundancy was necessary
  - Custom FPGA core to handle Spacewire
- First design: Custom board with dual redundant iMX-6 Cortex-A9 SoCs
  - Dual core SoC with video processing cores @ 800Mhz and 5000 DMIPS
  - Custom FPGA with a Spacewire core
  - Complex, densely packed board + a daughterboard

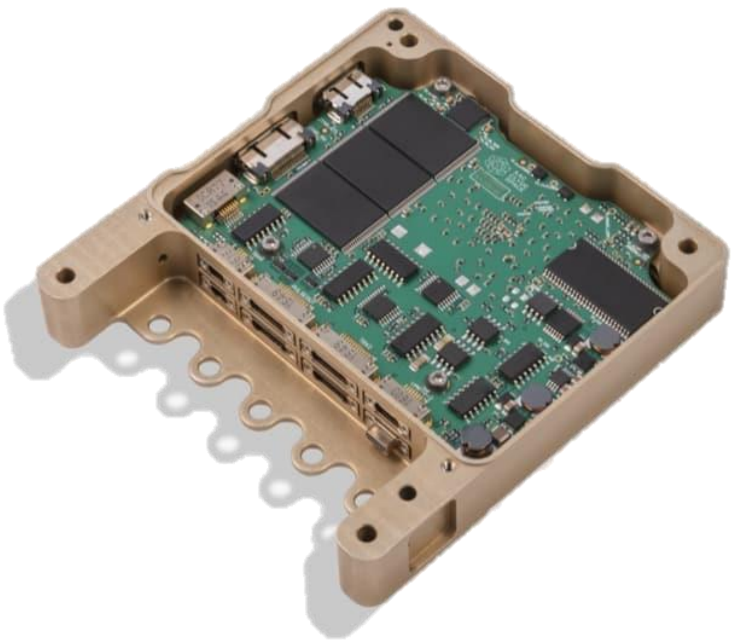
New Camera Interface Board with GR740

- Based on heritage AAC ClydeSpace OBC design
  - Customized to fit the Rover bus
- Quadcore LEON4-FT PBGA at 250MHz
  - 1840 DMIPS
- 512MB SDRAM
- 16Mbit MRAM
- 256Mbit SPI Flash
- 4 SpaceWire interfaces, 3 used for cameras
- GR-CAN cores



Mass Memory

- The new CIB does not have mass memory
  - Only MRAM (used for bootloader) and SPI flash (used for Linux image)
- AAC Clydespace TCM is used for Mass Memory storage
  - Connected via SpaceWire
  - Used as a black box, no customization necessary
- Provides 32 gigabytes
  - Used for images, video, previews, etc.
  - Continuous partitions
- Partition 0 used for file lists
  - Random access partition, but smaller size
  - New versions of the ISW also stored in partition 0



Imaging Software Design

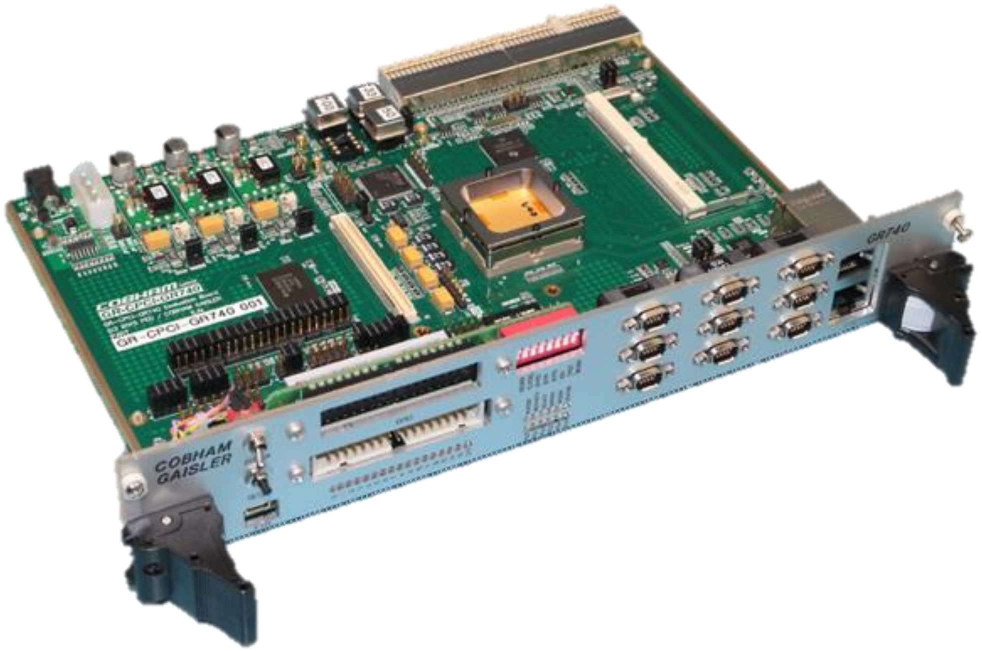
- The ISW is a C-based Linux user-space executable
  - Controls cameras and TCM directly via GR-Linux SpaceWire driver
  - Controls the thermal camera via SPI
  - Spawns additional linux processes, e.g. gstreamer, xz, shell commands
  - Communicates with ground via CAN and CSP
- Accepts telecommands from either the ground system, or from the rover OBC
- Tempfs used for acquiring images
  - Later stored into the TCM by command
- libcsp used for communication via CAN
- Downloads handled via separate ELR FTP executable
  - Same C code is used for the OBC running FreeRTOS on Cortex-M3

GR-Linux

- The original imaging software for iMX-6 uses Yocto Linux
- Use GR-Linux instead of the usual RTEMS
  - Based on Buildroot
  - Prevents a complete rewrite of ISW
- Port to GR-Linux was quick
- SocketCAN code made porting easy
- GStreamer-based code was also ported successfully
  - Initially used for video encoding, now only used for previewing images

Development Environment

- Development started on the GR740 development kit
- Transfer code to CIB after it has become available
- Several differences:
  - Clock speeds, performance
  - Memory layout and sizes
- Use different build configurations for each target



Difficulties (1/2)

- Storage complexity
  - No NVRAM, TCM was required, increasing complexity
  - TCM cannot be “mounted” like a normal filesystem
  - Custom software to handle storage required
  - TCM continuous partitions require management
- Booting
  - Linux Image is rather large, doesn’t fit into MRAM
  - Use GR-Boot to boot image from SPI flash
  - Boot time is 2 minutes (acceptable, but a little long)
- Software updates
  - Takes 6 hours (!!!) to update the Linux image in the SPI flash via GRMON
    - Workaround: avoid updating image, update executable only in TCM
  - Another possible workaround: don’t use GRMON directly, have an executable in RAM flash the SPI.
- Spacewire driver limitations
  - GR-Linux driver does not support hardware RMAP
    - Workaround: implement RMAP in software
  - Longer length Spacewire packets not supported
    - Workaround: modify memory map drivers
- CAN drivers in Linux
  - GR-Linux drivers did not initialize GPIOs properly by default
  - Some modification was necessary to CAN drivers in the kernel, but Linux kernel development is not easy

Difficulties (2/2)

- Other GR-Linux limitations
  - Device tree
- Endianness
  - The GR740/SPARC is big endian
  - Not a big issue if you write code correctly
  - Can catch you off-guard if you're initially working on little endian systems
- Power consumption
  - Twice as much power consumption despite much slower performance (vs iMX6)
    - 3 Watts on idle, 5 watts under load. Not much for a satellite, but a lot for rover
  - Power saving modes are lacking (compared to ARM)
    - E.g. S3 sleep, variable clock speeds
- Performance
  - Generating previews takes around 5-10 seconds
  - Lossless compression of a raw image takes nearly as long as actually downloading the raw image (and also consumes power)
  - No hardware compression of video
  - Live encoding of video (for streaming) considered impractical

Lessons Learned

- “Live” video streaming
  - Not a good requirement for a manually-controlled Lunar rover mission
  - Was considered required for Ops, and feasible with iMX6 and MIPI cameras
  - Considered impractical with new cameras, SpaceWire and GR740
  - As development progressed, we realized it’s not really a requirement
  - However, hardware selection was driven by this
- Executing shell commands for contingencies
  - The Linux shell is very powerful
- Linux makes user code easy, driver code hard
  - Driver abstractions allow us to reuse code between different platforms
    - Great for testing, don’t need to use an embedded board for basic tests
  - Can use a variety of open-source libraries/programs (e.g. xz, gstreamer) that “just work”
  - Kernel drivers are not as straightforward as on an RTOS, lots of complexity

Conclusion

- The rover uses the GR740-based CIB for camera control
  - Quadcore processor used for compression and image previews
  - Live video also considered but removed due to technical difficulties
- Transitioning to GR740 from an ARM board was successful, but not without difficulties
  - Linux helped with this transition due to ready abstraction layers
  - Several drivers in GR-Linux were not mature enough, requiring custom development
- Use of Linux made writing user code easy, driver code difficult

Thank You