

ESA's First-Ever Lunar Mission Satellite Orbits Moon with Automatically Generated Flight Code

On February 27, 2005, Europe's first-ever moon satellite began its lunar science observation phase, having already achieved the ESA's primary objective of using electric propulsion for deep-space projects.

The project's prime contractor, Swedish Space Corporation (SSC) developed the attitude and orbit control system (AOCS) of the Small Missions for Advanced Research and Technology (SMART-1) using automatically generated flight code. The AOCS orients the spacecraft for thrust vectoring, scientific instrument pointing, and ensuring that the solar arrays are illuminated by the sun. It also controls the electric propulsion thrust vector alignment within the spacecraft body during lunar transfer and descent phases while providing an advanced failure detection, isolation, and recovery (FDIR) system.

"MathWorks tools for Model-Based Design helped us to design and automatically generate code for the SMART-1 flight application software for attitude control, power control, thermal control, and FDIR," explains Per Bodin, SMART-1 AOCS Manager at SSC. "Based on this success, we plan to develop all onboard application software, nearly 95% of the full flight software, using Simulink® and Real-Time Workshop® Embedded Coder."

THE CHALLENGE

SSC needed to develop an AOCS within a low-cost mission profile, strict software development standards, and a short software development cycle of fewer than two years. Moreover, because the AOCS needed to perform in a harsh space environment of intense radiation, minimal gravity, and other effects not testable in a lab or on earth, SSC required rigorous proof-chain test capabilities to ensure the system performed correctly during flight.



Artist rendition of SMART-1 traveling to the moon. Photo courtesy of the European Space Agency.

"SMART-1 was ESA's first lunar mission, and we started with only a few fundamental system requirements," says Bodin. "We needed an efficient development process because traditional approaches based on paper designs and hand code would not work given the time constraints and the small development team."

THE SOLUTION

SSC implemented a new development process based on MathWorks tools for Model-Based Design to model, simulate, automatically generate code, and to test the onboard AOCS software. Engineers developed accurate simulation models to predict system behavior and to create exhaustive system and software test cases, which met the ESA PSS-05 software development standard.

Using Simulink, SSC first established the system's architectural design model, which contained only a few subsystem layers and blocks but was sufficient to support initial simulation scenarios. The architecture models were then elaborated into the detailed design by using only allowed blocks, following certain parameter-naming conventions, and adding robustness checks such as divide-by-zero protection.

Engineers used Stateflow® to develop a state machine that configures the subsystems of

THE CHALLENGE

To develop ESA's first lunar mission in a short time frame and at minimal cost

THE SOLUTION

Use MathWorks tools for Model-Based Design to model, simulate, generate, and test the flight code

THE RESULTS

- Reduced system development time by 50%
- Improved process efficiency
- Produced efficient code



We successfully developed the SMART-1 AOCS in a very short time frame and with a very low budget. MathWorks tools for simulation and flight-code generation played a key role in this success and will serve as the foundation for future satellite programs, such as Prisma.



Per Bodin, Swedish Space Corporation

the AOCS model based on the operating mode, including detumble, safe, electric propulsion, and science. In addition, they used Stateflow to control the autonomous momentum management function to heat the thrusters.

SSC used MathWorks tools to automatically generate the application C code, which was then compiled, integrated, and linked into the overall onboard software. Low-level device drivers and operating system software were developed using traditional, hand-written methods. SSC then deployed the onboard software onto a radiation-hardened ERC32 embedded processor.

“Simulation models helped us to create the detailed software design and to accurately predict the system behavior, including spacecraft dynamics,” Bodin says. “We then developed detailed test scenarios and generated code with a high degree of confidence that the implementation would match the model behavior.”

The Simulink subsystems of the AOCS were unit-tested and integration-tested on a simulated ERC32 target by reusing test cases that the engineers developed and executed within Simulink. In addition to meeting the low-level software requirements, the unit and integration tests also included structural code coverage analysis, input range testing, and max-path testing.

SSC performed software system testing on a hard real-time simulation environment and analyzed the results with MATLAB®. They verified the AOCS at the system level using the integrated spacecraft. These tests included open and closed-loop tests at the European Space Research and Technology Centre (ESTEC) in the Netherlands.

“We are quite pleased with the results so far,” says Bodin. “We plan to use more MathWorks tools for future project activities, including xPC Target for hardware-in-the-loop testing.”

THE RESULTS

- **Reduced system development time by 50%.** “I estimate that it would have taken approximately 50-100% longer to develop our system without MathWorks tools,” says Bodin.
- **Improved process efficiency.** “With MathWorks tools, we greatly reduced the number of design iterations and versions of our software,” notes Bodin. “The automatically generated code contained no coding errors—only design issues that were mostly eliminated during simulation studies.”
- **Produced efficient code.** “The generated code was roughly equal to hand code for RAM, ROM, and execution speed,” Bodin explains. “We certainly found no major inefficiencies or other concerns with the quality of the generated code.”

To learn more about ESA and SSC, visit www.esa.int and www.ssc.se

www.mathworks.com

APPLICATION AREAS

- Control Design
- Hardware-in-loop testing
- Model-Based Design
- Production code generation
- Simulation

PRODUCTS USED

- MATLAB
- Simulink
- Real-Time Workshop
- Real-Time Workshop Embedded Coder
- Stateflow
- Stateflow Coder