

FPGA Based Attitude Control System Architecture for Increased Performance

Georg Grillmayer, Marc Hirth
 Institute of Space Systems, Universität Stuttgart
 Pfaffenwaldring 31, 70569 Stuttgart, Germany
 grillmayer@irs.uni-stuttgart.de, hirth@irs.uni-stuttgart.de

Felix Huber, Viola Wolter
 Steinbeis Transferzentrum Raumfahrt
 Rötestraße 15, 71126 Gäufelden, Germany
 huber@tz-raumfahrt.de, wolter@tz-raumfahrt.de

ABSTRACT: For micro-satellites the required attitude accuracy, agility and autonomy along with desired parallel processing of payload images in real-time needs to be achieved within limited power and computational resources. Providing fast, parallel execution while still small in hardware size and modest in power consumption field programmable gate array (FPGA) based on-board computers (OBC) have the potential to highly increase system performance. The *Flying Laptop* is a micro-satellite currently under development at the Institute of Space Systems, Universität Stuttgart. The attitude control system together with the on-board navigation system is implemented as hardware in an FPGA. The architecture including the sensors, actuators and control algorithms are described in this paper.

NOMENCLATURE

ACS	Attitude Control System
ASIC	Application Specific Integrated Circuit
API	Application Programming Interface
FDIR	Fault Detection Isolation and Recovery
FOG	Fiber Optical Gyro
FPGA	Field Programmable Gate Array
GPS	Global Positioning System
I ² C	Inter-Integrated Circuit
IBIS	Integrated Bus for Intelligent Sensors
LTDN	Local Time of Decending Node
OBC	On Board Computer
PAL	Platform Abstraction Layer
PPS	Pulse Per Second
PSL	Platform Support Library
RAM	Random Access Memory
RW	Reaction Wheel
SHIP	Satellite Hardware Interface Protocol

A	Wheel alignment matrix	[-]
B	Magnetic field vector	[T]
C, c	Controller matrix\gain	[-]
E	Unity matrix	[-]
h	Angular momentum	[Nms]
I	Inertia matrix	[kgm ²]
K, k	Controller matrix\gain	[-]
m	Magnetic dipole moment	[Am ²]
P, p	Controller matrix\gain	[-]
P	Target position	[m]
q	Quaternion	[-]
R	Satellite position	[m]

T	Transformation matrix	[-]
T	Torque	[Nm]
u_{null}	Nullspace of A	[-]
V	Satellite velocity	[m/s]
ζ	Damping ratio	[-]
ϕ	Roll angle	[rad]
θ	Pitch angle	[rad]
ψ	Yaw angle	[rad]
ω	Angular velocity	[rad/s]
ω_n	natural frequency	[rad/s]
T_{ij}	describes the transformation from the j-frame to the i-frame	
q_{ij}	is the quaternion describing the attitude of the j-frame with respect to the i-frame	
X_j	describes the variable X in coordinates of the j-frame	

INTRODUCTION

The *Flying Laptop* (Figure 1) is a 100 kg, three-axis stabilized micro-satellite currently under development at the Institute of Space Systems, Universität Stuttgart and is planned to be launched into a sun-synchronous low earth orbit. The primary mission objective is technology demonstration.¹

Field programmable gate arrays (FPGAs) as a replacement of expensive application-specific integrated circuits (ASICs) are of increasing popularity for space and are currently being used in spacecraft payload and bus systems.² Though, few flight experiments with reconfig-

urable computers using FPGAs have been conducted due to the risk of non-radiation susceptibility. One of them being the Adaptive Instrument Module³ of the Australian micro-satellite Fedsat which was developed for in-orbit evaluation of RAM-based FPGAs.

A micro-satellite typically has limited power and computational resources. In order to achieve the required agility, accuracy of the navigation & attitude control system and autonomy along with the parallel processing of the payload images in real-time, for the *Flying Laptop* the decision was made to use an FPGA based on-board computer (OBC).

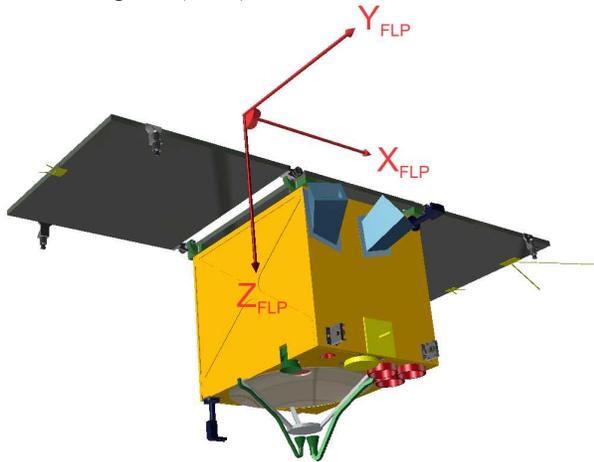


Figure 1: Flying Laptop

ACS HARDWARE

The satellite motion is monitored by five different types of sensors: two three-axis magnetometers, two coarse sun sensors, four fiber-optic rate sensors, one autonomous star tracker with two camera heads and three GPS receivers. The actuators that rotate the satellite to the desired attitude are four reaction wheels and three magnetic torquers.

All sensors and actuators are connected to the FPGA in a star like configuration, increasing the system reliability by using separate RS-422 ports, digital I/O lines, I²C buses and an IBIS bus.

All connections of the ACS hardware devices to the FPGA on-board computer are displayed in Figure 2.

Magnetometers

The ZARM-Technik AMR-magnetometer is a micro controller based 3-axis magnetometer with digital output. Two magnetometers will be installed on the *Flying Laptop*. Each of them is directly connected to the on-board computer via a RS-422 interface at 57600 baud. A measurement is returned approx. 160 ms after sending the read command.

Sun Sensors

The sun sensors system consists of two three-axis

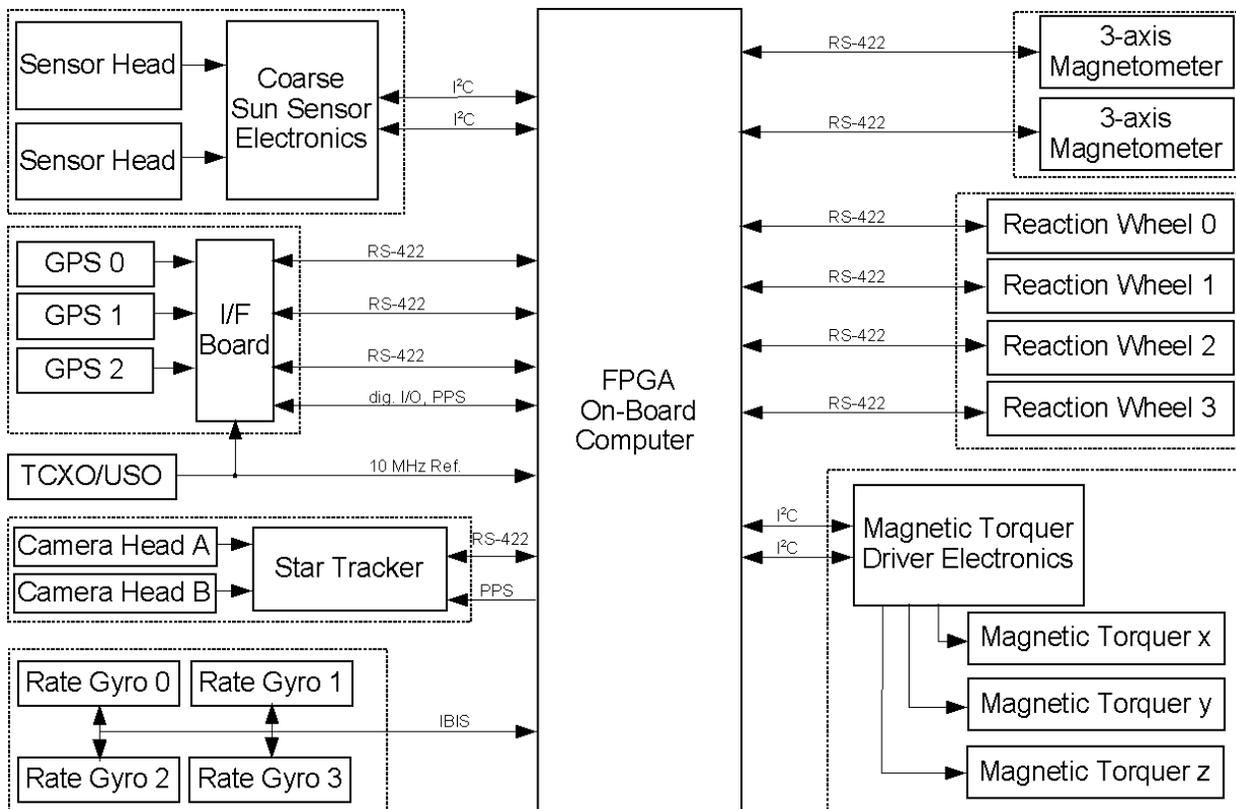


Figure 2: ACS Hardware Connections

Coarse Sun Sensors from Sun Space mounted at diagonal corners of the satellite. The sensors voltages are digitized and sent to the OBC via an I²C bus.

Rate Sensor

Four C-FORS fiber optic gyros from Litef measure the angular rate. They are arranged in a tetrahedron configuration, to allow for a single failure. The sensors have a drift rate of 3°/h. All four rate sensors are connected to the on-board computer by a common IBIS bus with a serial clock frequency of 2 MHz.

Star Tracker

The micro-Advanced Stellar Compass from the Technical University of Denmark is used. The system consists of two camera head units connected to a data processing unit. The star tracker provides an attitude knowledge of down to 3 arcseconds. It is connected to the on-board computer via a RS-422 interface at 115 kbaud and a PPS input for time synchronization.

GPS

The GENIUS (GPS Enhanced Navigation Instrument for the Universität Stuttgart micro-satellite) system is developed as an experiment for accurate determination of the spacecraft attitude using GPS. It consists of three COTS Phoenix GPS boards. Each of the receivers is

connected to separate GPS antenna via a low noise amplifier. The antennas are placed at three corners of the middle solar panel in an L-like arrangement. Furthermore the receivers are synchronized via an ultra-stable 10 MHz oscillator.

For normal non-attitude operation the envisaged accuracies are 10 m in position, 0.1 m/s in velocity and 1 μs in time.

All three Phoenix GPS receivers are connected to the on-board computer via separate RS-422 interfaces at 57600 baud and digital I/O lines. Each receiver sends a PPS pulse for time synchronization.

Reaction Wheels

Four reaction wheels RSI 01-5/28 from Teldix are running in a single hot redundant tetrahedron configuration. Each of the wheels has an angular momentum capacity of 0.12 Nms and a reaction torque of 5 mNm over the range of ±3000 rpm. The reaction wheels are connected to the on-board computer via a RS-422 interface at 9600 baud.

Magnetic Torquers

Three ZARM-Technik magnetic torquers (torque rods) with a linear dipole moment of 6 Am² are utilized. The torquers are connected to a power box that includes two I²C buses for connection to the OBC. The whole system is single redundant.

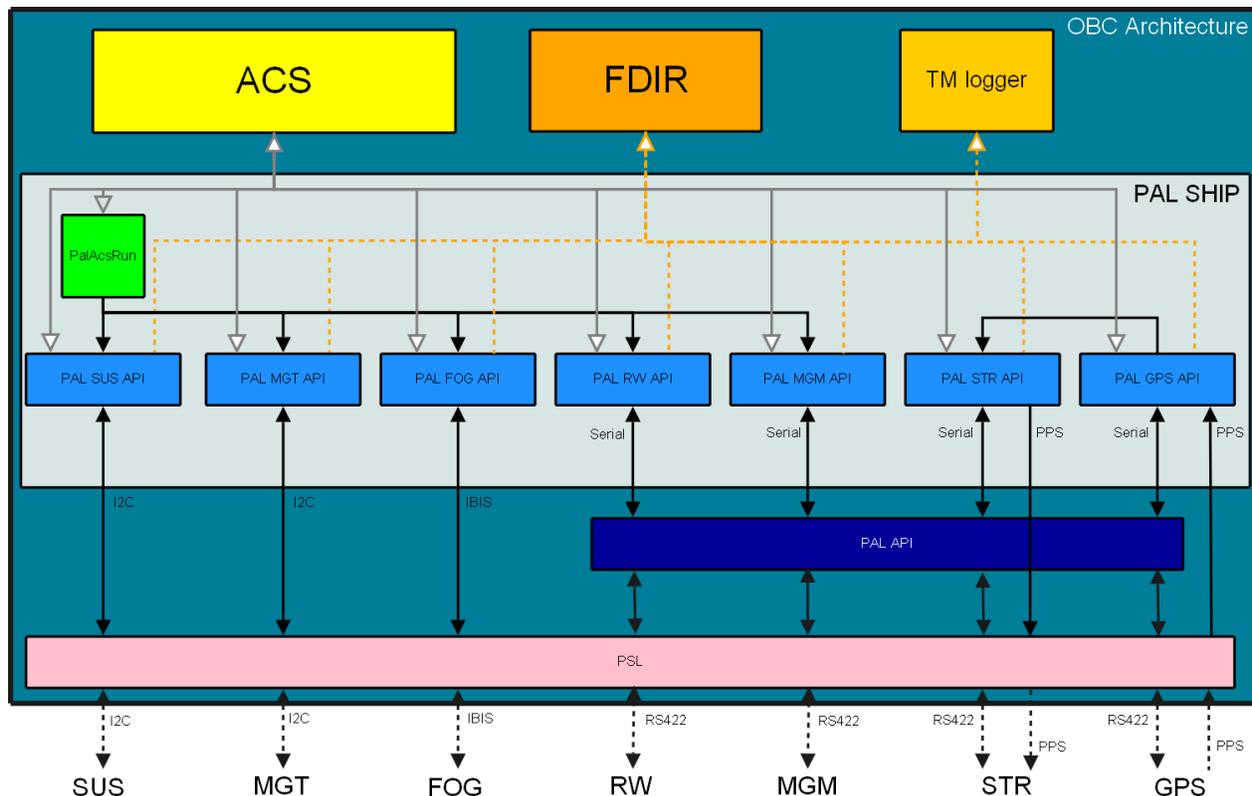


Figure 3: FPGA ACS Interface Structure

ACS ARCHITECTURE

FPGA Characteristics

As the on-board computer of the *Flying Laptop* is a Field-Programmable Gate Array, it requires programming that differs significantly from previous micro-processor software design. An FPGA is a semiconductor device that consists of programmable logic components and programmable interconnects. The components can duplicate the functionality of basic logic gates or more complex functions like look-up tables, RAM, shift registers, multiplexers, carry logic or arithmetic logic. The *Flying Laptop OBC* is based on the Virtex-II Pro FPGA from Xilinx.⁴ Two configurations, with 4 and 7 million system gates and up to 200 MHz clock frequency, are being evaluated. The hardware set-up is currently under development by the Steinbeis Transferzentrum Raumfahrt in cooperation with the Fraunhofer Institute for Computer Architecture and Software Technology. Hardware designs for the OBC are generated with the Handel-C compiler⁵ from Celoxia and a place and route tool from the FPGA manufacturer Xilinx. Handel-C is a high level language for hardware description, similar to ANSI C, but is also capable of providing low level functionality. Bit manipulation, digital input and output and even the use of basic logic elements like flip-flops can be implemented directly in the code. The Handel-C compiler is part of the DK Design Suite⁶ from Celoxica. Within the DK environment, the netlist for the programming of the FPGA is synthesized automatically. Using a FPGA for the ACS implies that the control and navigation algorithms will not run as software but are implemented in hardware. Thus, algorithms run with the speed of dedicated hardware circuits, but can be created with the programmability of software. This is a different approach for designing a system. Extensive parallel execution and cycle precise synchronization between parallel routines is possible, which in turn allows very accurate timing. Every code line takes exactly one clock cycle every time it is executed. Signals like interrupts are inherent to the design and are handled instantaneously. All resources are available in parallel, thus, no underlying operation system for scheduling is needed.

Satellite Hardware Interface Protocol

The implementation of an FPGA design in Handel-C usually consists of three layers. The uppermost layer is the user application layer, where the high level functionality and algorithms are implemented. Hence, these are the ACS algorithms, the FDIR and the TM logger. This layer is independent of the FPGA hardware board it runs on. The FPGA and board specific functionality is included in the remaining two layers, the Platform Abstraction Layer (PAL) and the Platform Support Library (PSL). The Platform Abstraction Layer can consist of

multiple PAL APIs, where each contains the functionality of a specific resource. The input and output functionality of a board is encapsulated in a Standard-PAL API. It provides functions for the control of interfaces like serial, PS2 or parallel interfaces, and simplifies the use and control of different types of RAM. For the *Flying Laptop* ACS interfaces the serial interface functionality of the Standard-PAL API is used. Further input and output functionality that is needed for the ACS is contained in the Satellite Hardware Interface Protocol (SHIP) API. In the SHIP API the I²C bus and the IBIS bus are implemented. The I²C bus is used for the magnetic torquers and for the sun sensors. The IBIS bus controls the communication with the fiber optic rate sensors. Also included in the SHIP API are the PAL APIs for the ACS devices. They provide functions that run, enable and disable the ACS devices and offer access to sensor data and control values of actuators. PAL APIs can also build on other PAL APIs. For example the RW API (Reaction wheel API) uses the Standard-PAL API for the serial interface functionality.

The PSL contains the settings for the specific FPGA board. There, the number of interfaces of a specific type, like serial interfaces, is defined and input and output pins are assigned to the corresponding hardware sensors and actuators.

The structure of the specific Handel-C design for the *Flying Laptop* ACS with the necessary PAL APIs and PSL is displayed in Figure 3. In the drawing only a single instance of each hardware component is shown. Multiple instances are generated by initializing another resource of the existing PAL items.

In Handel-C it is possible to implement parallel execution of code. Thus the capabilities of the FPGA can be utilized to create algorithms with performance factors of 40-100 better than on a conventional processor. The exact execution time of each statement, depending on the system clock frequency, can be controlled during the implementation and supervised during debugging. The PAL SHIP contains a function (*PalAcsRun*) that controls the fixed timing of all ACS devices where values must be requested or set within a distinct time frame (see Figure 3). The control function sends a signal to a specific resource when a certain process in the resource must be started. All resource APIs of one type, for example all reaction wheels, get the same signal at exactly the same time and command the hardware device simultaneously. As the timing is as precise as the system clock, the time interval between data acquisition processes does not vary and the ACS algorithms can be employed accurately. For the ACS cycle only the delay between a read command and the time instant when the requested value is sent must be known and taken into account for the timing. The delivered variables are written to the list of ACS accessible variables.

Figure 4 shows the ACS control cycle. For all devices a certain offset is set in order to receive all measurements

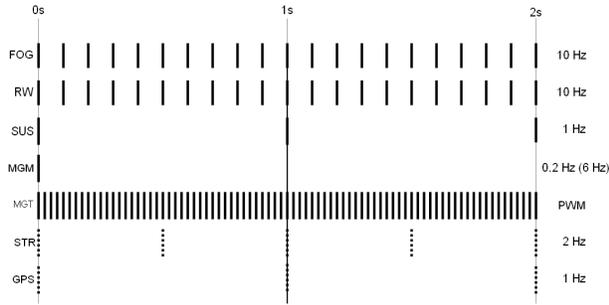


Figure 4: ACS Control Cycle Loops

synchronized to each other and compensate for the delay caused by the communication time. For example the fiber-optic gyros (FOG) and the reaction wheels (RW) work with the same sampling frequency of 10 Hz. The communication with the reaction wheels however, runs through a serial interface with 9600 baud and uses ASCII characters for the data transmission. This takes longer than the communication with the fiber-optic gyros, where the data is transmitted in binary and with a clock cycle of 2 MHz. So the communication with the reaction wheels must be started earlier than the communication with the gyros to be on time with the sync impulse.

Time Synchronization

The *Flying Laptop* is equipped with a 10 MHz ultra stable oscillator (accuracy 10^{-13} s for a 1000 s interval) that is used as a reference for the FPGA system clock as well as the timebase of the GPS receiver oscillators. The GPS time is used for absolute time referencing of the OBC clock.

The star tracker internal time, which runs independently, is synchronized to the OBC time every 10 s using a built in time telecommand. Furthermore the GPS position is fed to the star tracker in order to compensate orbit aberration effects and provide the best available accuracy.

Figure 5 shows the sequence of the PPS time synchronization from the GPS receivers via the OBC to the star tracker. The PPS signal from the GPS receivers to the on-board computer is 1 ms long. The navigation data are sent approximately 75 ms after the PPS signal. The OBC processes up to three received GPS PPS signals and uses an internal phase locked loop to generate the PPS signal for the star tracker with an extended impulse high time of 20 ms.

The star tracker and the GPS receivers deliver the attitude (2 Hz update) and the position (1 Hz update) unsynchronized to the ACS control cycle and with an update frequency less than the 10 Hz of the internal ACS control loop. In order to get higher rates from the GPS receiver an orbit propagator with a Kalman filter implemented in the internal firmware will be used. For the star tracker the current attitude is extrapolated from the two last valid samples using quaternion algebra.

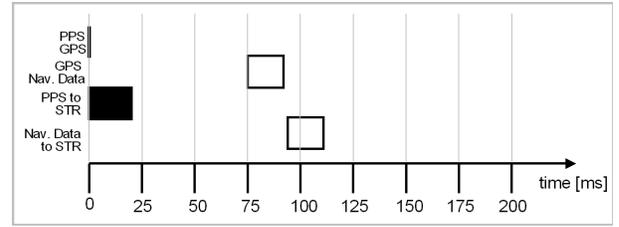


Figure 5: PPS Timing

Attitude Control

The attitude control block in Figure 3 uses the functions that are provided by the SHIP API. Through variable access functions, control values for the actuators are written and measurement values from the sensors are read. The following sections show the control loops which will be implemented.

ATTITUDE DYNAMICS

The attitude dynamics of a satellite, acted upon by internal and external torques is given by

$$T_{dist} + T_{mag} = \mathbf{I} \dot{\omega} + \omega \times (\mathbf{I} \omega + \mathbf{h}_w) + \dot{\mathbf{h}}_w \quad (1)$$

where T_{dist} are external disturbance torques, T_{mag} the torque generated by the magnetic torquers, ω the inertial referenced body rate, \mathbf{h}_w the angular momentum of the reaction wheels and \mathbf{I} the satellite's inertia matrix.

As the latter will be known for the *Flying Laptop* after design freeze, at the present time it is approximated with BIRD's inertia matrix, another micro-satellite with comparable dimensions

$$\mathbf{I} = \begin{bmatrix} 3.90 & -0.03 & 0.02 \\ -0.03 & 4.09 & -0.25 \\ 0.02 & -0.25 & 4.26 \end{bmatrix} \quad (2)$$

For the kinematic equation using quaternion notation, it is given

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & \omega_3 & -\omega_2 & \omega_1 \\ -\omega_3 & 0 & \omega_1 & \omega_2 \\ \omega_2 & -\omega_1 & 0 & \omega_3 \\ -\omega_1 & -\omega_2 & -\omega_3 & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \quad (3)$$

Finally, the error between actual (a) and commanded (c) attitude is described by the error quaternion \mathbf{q}_e :

$$\mathbf{q}_e = \begin{bmatrix} q_4 & q_3 & -q_2 & -q_1 \\ -q_3 & q_4 & q_1 & -q_2 \\ q_2 & -q_1 & q_4 & -q_3 \\ q_1 & q_2 & q_3 & q_4 \end{bmatrix}_c \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix}_a = \begin{bmatrix} q_{e1} \\ q_{e2} \\ q_{e3} \\ q_{e4} \end{bmatrix} \quad (4)$$

As the implemented controllers should work under any occurring conditions, no linearization of the equations of motion is made. The stability of the control laws is proved with Lyapunov theory.

MODES AND CONTROLLER DESIGN

Detumbling Mode

The objective of the detumbling mode is to reduce the angular velocity ω after launcher separation. An additional use is rate damping in case of accidentally exceeding the angular velocity limit.

Due to high angular rates, the measured magnetic field vector changes mainly as a result of the satellite's rotation and not because of the orbit variation of the Earth's magnetic field. Thus, the derivative of the magnetic field vector is a good approximation for the magnitude of the satellite's angular velocity and detumbling can be performed with magnetometer measurements alone.

Because a rotation about the Earth's magnetic field vector does not change its derivative, only vector components perpendicular to the magnetic field can be measured at every given point of time. Nevertheless, the magnetic field variation over a whole orbit allows a full 3-axis rate damping.

The measured rate perpendicular to the magnetic field B can be written as

$$\omega_m = \left[\mathbf{E} - \frac{BB^T}{B^T B} \right] \omega \quad (5)$$

The control law for the commanded torque T_{mag} is a simple P-controller with a positive definite diagonal gain matrix $\mathbf{K} = \text{diag}(k)$:

$$T_{mag} = -\mathbf{K} \omega_m \quad (6)$$

This torque is generated by three magnetic torquer rods, one aligned along each body axis. Its value is the cross product of the torquer dipole moments m_{MGT} and the Earth's magnetic field vector:

$$T_{mag} = m_{MGT} \times B \quad (7)$$

Combining Eq. (5) and (6) yield to the corresponding dipole moment

$$m_{MGT} = -\mathbf{K} \omega_m \times B \quad (8)$$

Because $\omega_m \times B$ points in the direction of \dot{B} , Eq. (7) can be rewritten as

$$m_{MGT} = -\mathbf{K}_b \frac{\dot{B}}{\|B\|} \quad (9)$$

$\mathbf{K}_b = \text{diag}(k_b)$ is again a positive definite diagonal gain matrix. The absolute value of the magnetic field vector is used for scaling purposes.

To allow a designated bang-bang actuation of the magnetic torquers as well as to ensure that evolving torquer B-fields to not falsify magnetometer measurements the dipole moment command is applied in a 5 s periodic cycle containing a downtime of 0.25 s for magnetometer measurements. The commanded dipole moment is pulse-width modulated.

Inertial Pointing

The inertial pointing mode will mainly be used for calibration and for special purposes of automated asteroid detection using the star tracker. A pointing stability of 150 arcseconds is required. Attitude and rate information is provided by the star tracker and the fiber optical rate sensors. Four reaction wheels are used for actuation.

The controller for this mode must allow large eigenaxis slews from an arbitrary position to the commanded attitude. At the same time the maximum slew rate must not exceed $\omega_{max} = 1^\circ/s$. Therefore the following control law for the commanded torque T_w on the satellite body is used.⁷

$$T_w = -\dot{h}_w = -\mathbf{K} \text{sat}(Pq_e) - \mathbf{C} \omega \quad (10)$$

It contains controller matrices of the form

$$\mathbf{P} = \text{diag}(p_1, p_2, p_3), \mathbf{K} = \text{diag}(k_1, k_2, k_3), \mathbf{C} = c \mathbf{I} \quad (11)$$

where p_i, k_i, c are positive constants to be determined later on. q_e is the error quaternion vector consisting of the first three elements of \underline{q}_e in Eq. (3).

With $q_e(0)$, the error quaternion vector at the beginning of the maneuver, k_i and p_i can be determined as follows

$$k_i = c \frac{|q_{e,i}(0)|}{\|q_e(0)\|} \omega_{max} \quad (12)$$

$$\mathbf{P} = \mathbf{K}^{-1} k \mathbf{I} \quad (13)$$

The choice of k and c finally results from the desired damping ratio ζ and the natural frequency ω_n of the closed loop:

$$k = 2\omega_n^2 \quad \text{and} \quad c = 2\zeta\omega_n \quad (14)$$

The sat-function in Eq. (10) limits the output value to ± 1 and helps together with ω_{max} in the controller matrices to keep the maximum slew rate bounded.

Now, the torque command is transformed from body axis to wheel axis using the pseudo-inverse \mathbf{A}^+ of the wheel alignment matrix

$$T_{0-3} = -\mathbf{A}^+ T_w \quad (15)$$

Once a torque command exceeds the reaction wheel limit of $T_{max} = 5 \text{ mNm}$, the realized torque direction would no longer match the commanded direction. Thus, the whole torque command is scaled with respect to its largest element in case it exceeds T_{max}

$$T_c = \text{sat}_\sigma(T_{0-3}) = \begin{cases} T_{0-3} & \text{for } \sigma \leq 1 \\ \frac{T_{0-3}}{\sigma} & \text{for } \sigma > 1 \end{cases} \quad (16)$$

with

$$\sigma = \max_i \left| \frac{T_{0-3,i}}{T_{max}} \right| \quad (17)$$

This leads finally to

$$T_c = -\underset{\sigma}{\text{sat}}(\mathbf{K} \text{ sat}(\mathbf{P}q_e) - \mathbf{C}\omega) \quad (18)$$

Compared to the original control law described in ⁷, one modification was made. In Eq. (11) it is required that all $q_{e,i}(0) \neq 0$ because the matrix \mathbf{P} can not be computed otherwise. To avoid this problem and to allow certain maneuvers (i.e. rotation about a single body axis) a small bias $q_{i,\min} \ll 1$ is added if the corresponding element is zero. Simulations showed no negative effect due to this modification.

Another task for the inertial pointing mode is to turn the solar arrays (negative z-axis, see Figure 1) towards the sun. This is achieved with sun sensor data as attitude reference. As it provides only 2-axis information, the attitude of the x-z-or y-z-plane respectively must be artificially generated in order to use the same 3-axis controller.

Nadir Pointing

During earth observation in nadir pointing mode the satellite's body axes are aligned with the nadir coordinate system. The same type of non-linear controller with quaternion and rate feedback as for inertial pointing is used (the error quaternion now describes the error between body and nadir frame)

$$T_w = -\mathbf{K} \text{ sat}(\mathbf{P}q_e) - \mathbf{C}\bar{\omega} \quad (19)$$

The controller allows large angle slew maneuvers to a nadir-pointed orientation from an arbitrary attitude and then stabilizes the satellite's nadir orientation within 150 arcseconds.

No earth referenced attitude is directly available from any sensor, but with the help of GPS data it is possible to transform the initial quaternions from the star tracker to the nadir frame. First the GPS-time is converted to JD2000 to calculate the Greenwich Mean Sidereal Time (GMST). With its help, GPS position and velocity can be transformed from the earth-fixed to the earth-inertial frame (I).

The axes of the nadir-frame (N) can now be described with the (normalized) position vector R_I and velocity vector V_I . The z-axis (see Figure 1) of the nadir frame points towards the Earth's center (opposite direction of the position vector)

$$z_{N,I} = -R_I \quad (20)$$

The y-axis is perpendicular to the orbital plane, therefore it is perpendicular to the plane spanned by R_I and V_I

$$y_{N,I} = -R_I \times V_I \quad (21)$$

The x-axis completes to a right-hand system

$$x_{N,I} = y_{N,I} \times z_{N,I} = (-R_I \times V_I) \times (-R_I) \quad (22)$$

Now, all axes of the nadir frame are represented in inertial coordinates. These are exactly the columns of the di-

rection cosine matrix T_{IN} describing the attitude of the nadir frame with respect to the inertial frame

$$T_{IN} = (x_{N,I} \ y_{N,I} \ z_{N,I}) \quad (23)$$

After transforming the matrix T_{IN} to an equivalent quaternion notation q_{IN} the desired attitude between nadir and body frame can finally be computed with the help of the inertially referenced star tracker quaternions q_{BI}

$$q_{BN} = q_{BI} \cdot q_{IN} \quad (24)$$

Contrary to inertial pointing, the reference body rate for the controller is non-zero. It must be set according to the orbital rate ω_O :

$$\bar{\omega} = \omega - \begin{bmatrix} 0 \\ -\omega_O \\ 0 \end{bmatrix} \quad (25)$$

In case of a non-circular orbit the nadir and orbit frame are not identical and the angular velocity varies and needs to be continually re-computed with the help of GPS data. First, the normalized position vector R_I is transformed into the nadir frame

$$-R_N = T_{IN}^T (-R_I) \quad (26)$$

The x-component of its derivative equals the desired body reference rate and can then be adjusted at every GPS (or orbit propagator) update:

$$-\dot{R}_{Nx} = -\omega_O \quad (27)$$

Target Pointing

In the target pointing mode the satellite remains aligned towards a fixed point on the Earth's surface. This mode is mainly used for scientific investigations of the bi-directional distribution function (BRDF), off-nadir inspection of ground areas and during contact with the ground station using the high gain antennas. Suitable images for a BRDF target need to be taken along $\pm 60^\circ$ pitch off nadir and within a maximum roll angle of $\pm 5^\circ$. An image with the payload cameras needs to be taken at every degree between the $\pm 60^\circ$ during the overflight.

The final controller for this mode still has to be determined, but it will likely be a quaternion and rate feedback controller as well. Star tracker, GPS and rate sensors will be used to precisely follow the calculated attitude profile within a pointing accuracy of 150 arcseconds.

An exemplary rate profile for a target pointing maneuver is shown in Figure 6.

The reference system for target pointing is again obtained with the help of the normalized GPS position and velocity vector. Together with the known earth-fixed target position vector P (normalized) it is transformed to the earth-inertial frame. Now the axes of the target frame (T) can be described with the inertial vectors to build the transformation matrix T_{IT} .

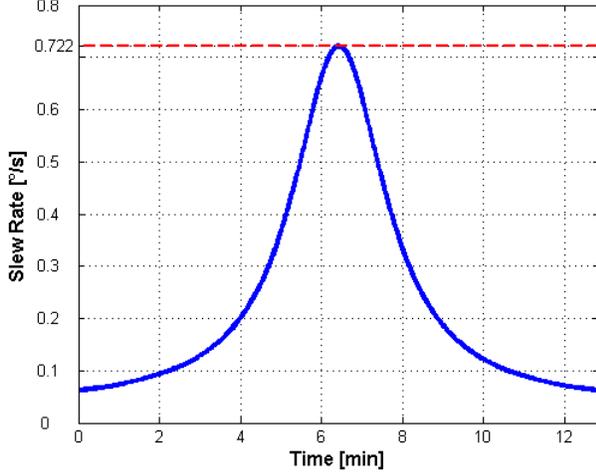


Figure 6: Slew Rate Profile for Target Pointing (600 km Orbit, Direct Overflight)

The z-axis should point towards the target center. This direction is expressed by the normalized difference of satellite and target position

$$z_{T,I} = \frac{P_I - R_I}{|P_I - R_I|} \quad (28)$$

As no yaw is allowed during target pointing, the x-axis remains in the orbital plane. It is perpendicular to the orbit normal n_{orb} as well as to the target z-axis

$$x_{T,I} = n_{orb} \times z_{T,I} = (V_I \times R_I) \times z_{T,I} \quad (29)$$

The y-axis completes to a right-hand system

$$y_{T,I} = z_{T,I} \times x_{T,I} \quad (30)$$

The matrix $T_{IT} = (x_{T,I} \ y_{T,I} \ z_{T,I})$ is converted to quaternion notation q_{IT} and the desired attitude between target and body frame can again be computed by multiplication with the star tracker quaternions q_{BI} :

$$q_{BT} = q_{BI} \cdot q_{IT} \quad (31)$$

Desaturation and Nullspace Control

Both, desaturation and nullspace control, are not independent attitude modes. They are needed to keep the reaction wheel speeds within their limits. As the wheels are active during all pointing modes, the respective controllers will be briefly described.

For wheel desaturation, total angular momentum control is used. The measured angular momentum

$$h_B = (I \omega + h_w) \quad (32)$$

of the system satellite + wheels is measured and compared the desired reference value

$$h_{B,ref} = (0 \ 0 \ 0)^T \quad (33)$$

If a difference is detected, it is counteracted using the following control law with the scalar gain $k_{offload}$ for the commanded torque

$$T_c = -k_{offload} (h_B - h_{B,ref}) \quad (34)$$

The torque is generated by the magnetic torquers. Because only torques perpendicular to the current magnetic field vector can be realized, only these components are commanded to the torquers. This finally yields to the magnetic dipole command

$$m_c = \frac{B \times T_c}{\|B\|} = -k_{offload} \frac{B \times (h_B - h_{ref})}{\|B\|} \quad (35)$$

The given formula shows that it is guaranteed that the total angular momentum is bounded.

However – as all four wheels are in use at the same time – a momentum component can be built up in the appropriate nullspace u_{null} . That component can not be detected by total angular momentum control and an additional nullspace controller is needed. The implemented control law is

$$T_c = k_{null} u_{null} u_{null}^T h_{0-3} \quad (36)$$

where k_{null} is a small scalar gain and h_{0-3} is the angular momentum of the four wheels in wheel coordinates. Eq. (36) shows that the magnitude of nullspace component $u_{null}^T h_{0-3}$ is measured and fed back along the nullspace u_{null} . Therefore, attitude control is not affected by the control signal as it is within the nullspace itself.

SIMULATION RESULTS

Calculations were made to determine the orbital disturbance torques acting on the *Flying Laptop* and resulted in the magnetic torque having the major influence. It is mainly caused by the magnetic dipole (1 Am²) of the traveling-wave tube aligned with the satellite's body z-axis. For the following simulation results a higher dipole moment of $m_{dist} = [0 \ 0 \ 1.5]^T$ Am² is assumed resulting in a disturbance torque by interaction with the Earth's magnetic field

$$T_{dist} = m_{dist} \times B \quad (37)$$

Detumbling Mode

Initial conditions:

- Orbit: 600 km, sun-synchronous, 10:30 LTDN
- Initial attitude: $[\phi \ \theta \ \psi]_{BI} = [0^\circ \ 0^\circ \ 0^\circ]$

- Initial rates: $\omega_0 = \sqrt{\frac{\omega_{max}^2}{3}} [1 \ -1 \ 1]^T$,

$$\omega_{max} = 10^\circ/s$$

- Control gain: $k_b = 1000$
- Duration: 12 000 s (\approx two orbits)

To show the performance of the detumbling controller, a worst-case scenario with an overall initial rate of 10 °/s is simulated. The magnitude of the rate is evenly distributed on all three body axes with randomly chosen signs. The results are shown in Figure 7.

The absolute value of the body rate decreases monotonically and rate reduction is performed in about one orbit. The varying slope of the decrease is caused by the varying attitude of the satellite with respect to the magnetic field and therefore changing possible control torques. As no inertial rates can be measured by the magnetometers a non-zero rate of < 0.2 °/s remains due to the variation of the Earth's magnetic field.

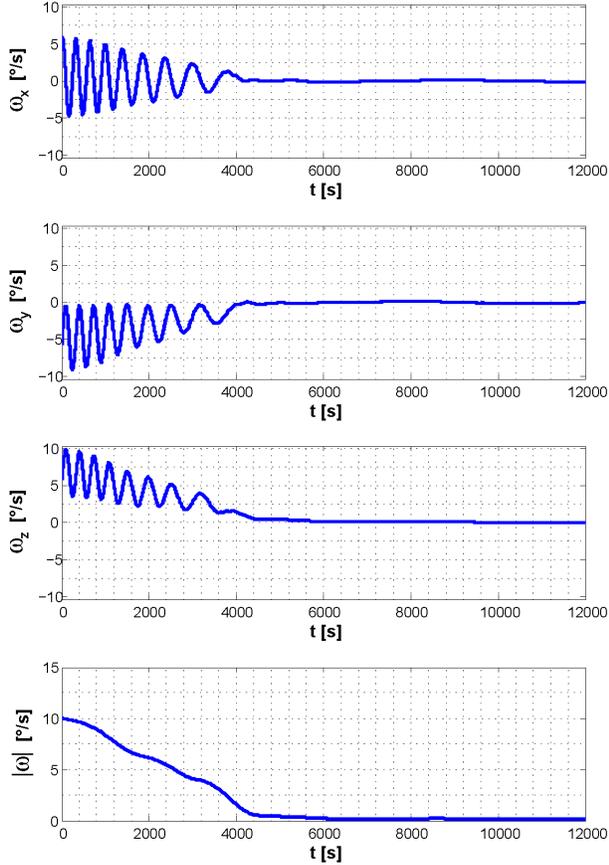


Figure 7: Body Rates and Magnitude of Angular Velocity

Inertial Pointing

Initial conditions:

- Orbit: 600 km, sun-synchronous, 10:30 LTDN
- Initial attitude: $[\phi \ \theta \ \psi]_{BI} = [0^\circ \ 0^\circ \ 0^\circ]$
- Attitude command: $[\phi \ \theta \ \psi]_{BI} = [45^\circ \ 25^\circ \ 5^\circ]$
- Initial rates: $\omega_0 = [0 \ 0 \ 0]^T$
- Controller parameters: $\omega_n = 0.1$, $\zeta = 1$
- Duration: 12 000 s (\approx two orbits)

For the simulation of the inertial pointing controller a randomly chosen large angle slew is commanded. The attitude error and the magnitude of the angular velocity at the beginning of the maneuver is shown in Figure 8. First the satellite accelerates to an angular body rate close to 1 °/s, without exceeding it. After a coast phase it starts to decelerate towards the desired attitude.

Within about 120 s the error between commanded and actual attitude has reached approximately zero for all three axes. The pointing accuracy remains less than

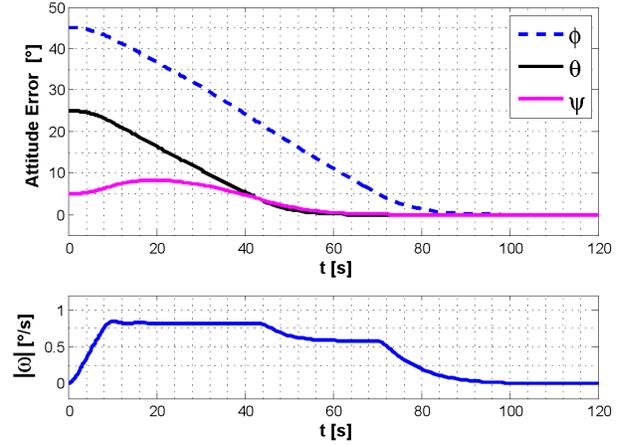


Figure 8: Attitude Error and Magnitude of Angular Velocity at the Beginning of the Slew Maneuver

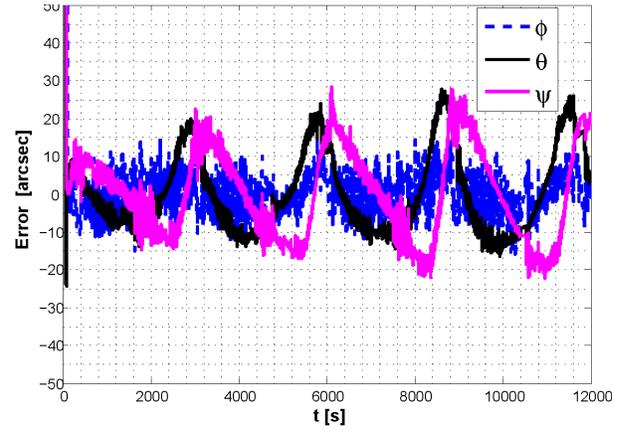


Figure 9: Deviation from Commanded Attitude over Two Orbits

30 arcseconds for the simulated time span of 2 orbits (Figure 9) and therefore clearly meets the requirement of 150 arcseconds. The long-periodic oscillations are caused by the varying amplitude of the magnetic disturbance torque, the short-periodic by the pulse-width modulated torquer commands.

Nadir Pointing

Initial conditions:

- Orbit: 600 km, sun-synchronous, 10:30 LTDN
- Initial attitude: $\mathbf{q}_{BN} = [-0.1855 \ 0.4873 \ -0.659 \ 0.542]^T$ (equivalent to $[\phi \ \theta \ \psi]_{BI} = [0^\circ \ 0^\circ \ 0^\circ]$)
- Attitude command: $\mathbf{q}_{BN} = [0 \ 0 \ 0 \ 1]^T$ (fixed for nadir pointing)
- Initial rates: $\omega_0 = [0 \ 0 \ 0]^T$
- Parameters: $\omega_n = 0.1$, $\zeta = 1$
- Duration: 12 000 s (\approx two orbits)

The initial attitude for the simulation is again chosen randomly. After a slew maneuver from an inertially fixed attitude, the satellite aligns its axes with the nadir frame. The deviation from ideal nadir-pointing over a period of about two orbits is shown in Figure 10. It does not exceed the limit of 150 arcseconds and even stays within a maximum error of ± 20 arcseconds. Again long and short periodic oscillations are visible, caused by disturbance torque variations and torquer pulsing respectively.

Unlike for inertial pointing, the magnitude of the body rate does not decrease to zero after the initial slew maneuver, but settles at the orbital slew rate of approximately $0.062^\circ/\text{s}$.

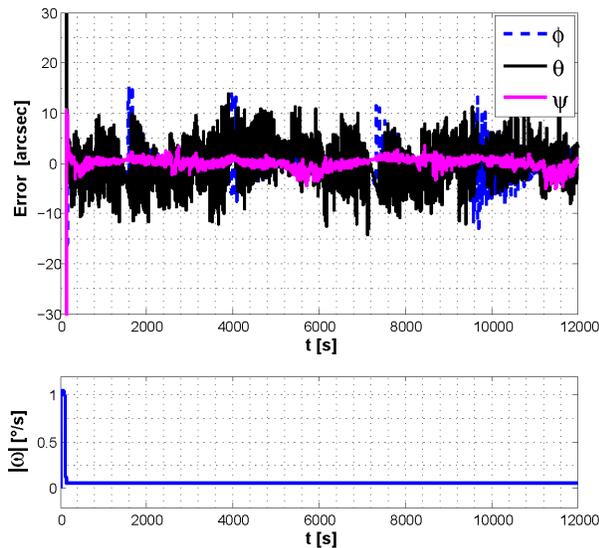


Figure 10: Nadir Pointing Error and Magnitude of Angular Velocity

IMPLEMENTATION

Figure 11 shows the hardware used for developing the software interfaces. As a development board a Spartan-III with 400 k-gates with 50 MHz clock frequency is used. The Spartan III FPGA is a downsized version of the Vertex II. Still the entire SHIP PAL could be compiled for the smaller chip leaving enough free system resources at the larger Vertex II FPGA.

Tests with the engineering models of the sensors and actuators in the laboratory show very good timing behavior and the time-parallel communication with the devices could be verified.

CONCLUSIONS

The conducted research shows that building an ACS entirely in hardware is possible. The described ACS architecture sets the basis for a high performance navigation and attitude control system where timing issues are already addressed at a low level.

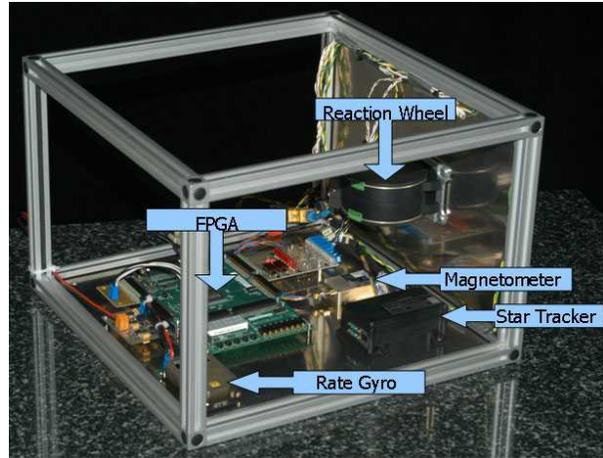


Figure 11: "SpaceCube" for ACS development

The advantage of the FPGA over a micro-processor is already noticeable during performance tests with the engineering models. The parallel structure of a FPGA allows to exploit the limits of the sensors and actuators in order to achieve a high precision navigation system so far not possible for university built micro-satellites.

REFERENCES

1. Grillmayer, G., Falke, A. and Roeser, H.P., "Technology Demonstration with the Micro-Satellite Flying Laptop", Selected Proceedings of the 5th IAA Symposium on Small Satellites for Earth Observation, Berlin, Germany, 4-8 Apr. 2005, pp. 419-427.
2. Kevin M., "FPGAs in Space - Programmable Logic in Orbit", FPGA and Programmable Logic Journal [online journal], 3 Aug. 2004, URL: http://www.fpgajournal.com/articles/20040803_space.htm [cited 25 April 2006].
3. Conde, R.F., Darrin, A.G., Dumont, F.C., Luers, P., Jurczyk, S., Bergmann, N. and Dawood, A., "Adaptive Instrument Module - A Reconfigurable Processor for Spacecraft Applications", The 2nd Annual Military and Aerospace Applications of Programmable Logic Devices (MAPLD'99) Conference, Laurel, Maryland, USA, Sep. 1999.
4. Xilinx, "Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet", URL: http://www.xilinx.com/products/silicon_solutions/fpgas/virtex/virtex_ii_platform_fpgas/resources/ [cited 1 June 2006].
5. Celoxica, "Handel-C Language Reference Manual - For DK version 4", 2005.
6. Celoxica, "DK Design Suite", URL: <http://www.celoxica.com/products/dk/> [cited 1 June 2006].
7. Wie, B., Space Vehicle Dynamics and Control, AIAA Education Series, 1998.