DESIGN AND DEVELOPMENT OF AN ACTUATOR CONTROL AND MONITORING UNIT FOR SMALL AND MEDIUM SIZE RESEARCH UAVS

D. Teubl, T. W. Bitenc, M. Hornung
Institute of Aircraft Design, Technical University of Munich,
Boltzmannstraße 15, 85748 Garching, Germany

Abstract

Nowadays, a multitude of flight control computers and sensor systems are available to support research activities related to testing various unmanned aircraft systems. Either for initial flight testing of a newly designed aircraft or for retrofit purposes, most flight control computers offer a wide variety of functionalities, with a clear exception of servo actuators of flight control systems. Due to similarities in underlying operating principles, paired with the knowledge and expertise gained with monitoring and control, functionalities can be reused in Electro-Mechanical Actuators (EMA) dedicated for bigger UAV's and transport aircraft. Due to size restrictions, merely Commercially Off-The-Shelf (COTS) servos can be chosen for present research UAVs. These COTS servos lack desired functionalities, like power consumption or position monitoring. Having the ability to record such data would be beneficial for research purposes. Moreover, there is no commercially off-the-shelf servo actuator for small to medium-sized UAV's, which would allow high configurability on feedback and closed-loop control capability. The present work addresses this problem by introducing a novel actuator control and monitor unit. Detailed design considerations and goals will be introduced along with the state of the development. For validation of the functionalities and usability of the system, laboratory and field test results will be shown.

Keywords
UAV, Actuators, Servo, Monitoring, Control, Embedded Systems

1. MOTIVATION

Small and medium-sized research Unmanned Aerial Vehicles (UAVs) are well suited for testing and prototyping different algorithms and scalable technologies, like flight control computers or different control algorithms of electro-mechanical actuators. Actuators commonly used in UAVs with a maximum takeoff mass (MTOM) of $40 - 100kg$, are Commercial Off-The-Shelf (COTS) servo actuators used in Remotelly Controlled (RC) modelling. The benefits of such servos are the low weight and low cost, even at high torques. These servos are commonly controlled in an open-loop manner via a single PWM signal. There are variations on the market today [1, 2, 3], which rely on BUS-like connectivity (CAN, SBUS) and are capable of giving feedback, like shaft position to the user.

These servo actuators use a small Direct Current (DC) or Brussless Direct Current (BLDC) motor with an additional gear-train and potentiometer for position feedback. With this configuration, they utilize the same principles as electro-mechanical actuators used in manned transport aircraft [4]. From the COTS servos, the majority feautures simple PWM inputs and no data feedback. With digital interfaces like SBUS or CAN, it is already possible to set minimum or maximum allowed positions [1, 2], monitor actual position and life-cycle data [3]. However, there is currently no COTS servo under $0.1kg$ with maximum $8.4V_{DC}$ operation voltage which would offer position, current, and temperature measurement along with life-cycle data and configurability in terms of min/max positions and data feedback configuration.

In the past, few initiatives have been carried out to enhance the capabilities of simple PWM-driven servos by attaching custom external hardware for measurement and control [5, 6, 7]. These hardware, however, did only come with limited functionality, e.g. it allowed only measuring health-relevant data without logging capabilities or enough computational power to process the measured data in real-time, moreover, their installation required damaging the housing of the actuator to allow access to the internal electronics.

In this paper, we show the design and development process of an Actuator Control and Monitor Unit (ACMU) [8], which can enhance the functionality of any COTS PWM-driven servo-actuator. The control and monitoring functionalities are implemented on a small-size custom hardware running a Real-Time Operating System (RTOS). On the monitoring side, the system is capable of measuring current-consumption and input voltage. Via an I2C interface, the deflection of the control surface, as well as the adherent servo level arm deflection and the temperature of the servo can be measured. Along with the monitoring capabilities, the system is capable of collecting overall run-time information for the actuator and it is able to provide enough computing power to run prediction- and prediction-analysis algorithms. Furthermore, all the measured data, alongside system logs can be stored on an SD card, for later review or processing.

doi: 10.25967/550036

On the control side, it can provide configurable serial (e.g. USB, UART) or bus-like (e.g. CAN) input, and implementation for custom closed-loop control algorithms.

2. DESIGN CONSIDERATIONS

This chapter explains the major design choices made during the development process of the first prototypes. First, the needs for monitoring and control capabilities will be described, followed by a description of the physical hardware requirements as well as the description of the desired input-output interfaces and considerations regarding software development.
Based on previous experiences [5, 7], monitoring of actuator parameters (e.g. temperature and position) during operation has positive benefits on the operation of a demonstrator. The measurement of the current consumption, for example, allows the user to monitor the load on the actuator and deploy software-based overload protection. The ability to monitor run-time, command and load information allows to develop and use early maintenance and failure prediction algorithms.
From a control point of view, reconfiguration of min/max and neutral position can be useful in demonstrators which need a high level of redundancy in given control surfaces. An ability to deploy high level control algorithms can slightly change the dynamic performance of the actuator. On top of that, the ability of turning on and off the actuator on demand - at startup operation, or in standby mode - can potentially reduce gear-wear and damage during extended ground operations, by eliminating the constant load caused by the weight of the control surface.
On the hardware side, the small form-factor is a key driving factor, due to the normally available space around the actuators. Separation of core functionalites into discrete hardware parts increases the net size of the system, but allows for flexibility and reconfiguration for different actuators.
The ability to apply the system to a high variety of actuators in a plug-and-play manner is mandatory. Causing damage or replacing critical electronics on a COTS system, to enchance its capabilities normally requires extensive manual work during the installation process and leads to possible unwanted permanent damage to the system.
Most available actuators of research interest provide a PWM interface as a control input, and all remote control and autopilot systems have a PWM output. Thus, the PWM-2-PWM interface is mandatory. On top of that, UART-2-PWM and CAN-2-PWM signal transformations are desirable as well to be able to give feedback for online monitoring purposes. In addition, a standardized USB connection to a PC is mandatory to provide a clean interface for system updates and parameter changes.
A flexible communication BUS interface for digital sensors like angular position or temperature measurement is necessary. That will allow a slim, and easy-to-extend interface for a high variety of sensors.
To be able to manage configurations for different systems, and safely store measurement data and system log in any configuration a local permanent data storage is desired.
Using a Real-Time Operating System (RTOS) and existing hardware libraries can drastically shorten the software development time for embedded systems, by reusing normally needed functionalities. A RTOS provides a task scheduler, which allows separation and prioritization of different functionalities. Hardware libraries provide an easy-to-use high-level interface for peripherals.
Previous personal experience and studies showed [9, 10, 11], that software development time can be highly reduced with proper tools and high-level languages. On top of that, external studies showed, that modern C++ can improve the design and effectiveness of an embedded software [12, 13].
The main design drivers are briefly summarized here.

- Monitoring functions - position, power consumption, temperature, usage informations etc.
- High and low level control functions
- No damage on actuator/easy actuator replacement
- Can be applied to a high variety of actuators
- Distributed hardware design
- Variable input for control signals - PWM, UART, CAN
- Flexible sensor interface
- Local data storage
- Usage of Real-Time Operation system
- Usage of Modern C++

During development, open-source and cross-platform software and technologies are prefered. This allow to use different text editors, IDE's and operating systems, and still maintain a highly cooperative environment.

3. HARDWARE DEVELOPMENT

In this section, possible configurations of the ACMU will be shown, followed by the introduction of the most important parts of the hardware. Then, the limitations of the system will be summarized. Finally, the sensor boards will be shown, which serve as an optional extension in the design.

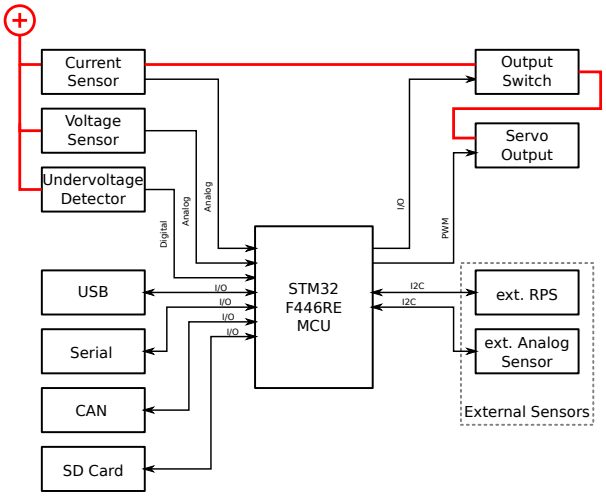


Figure 1. High level block diagram of the ACMU system.

The ACMU as a system features a distributed design, with functionalities decoupled as much as possible from each other. The full system shown in Fig. 1, consists of the ACMU main and interface boards, an analog extension board and at least one Rotary Position Sensor (RPS) board. Different configurations are possible based on the measurement requirements. The

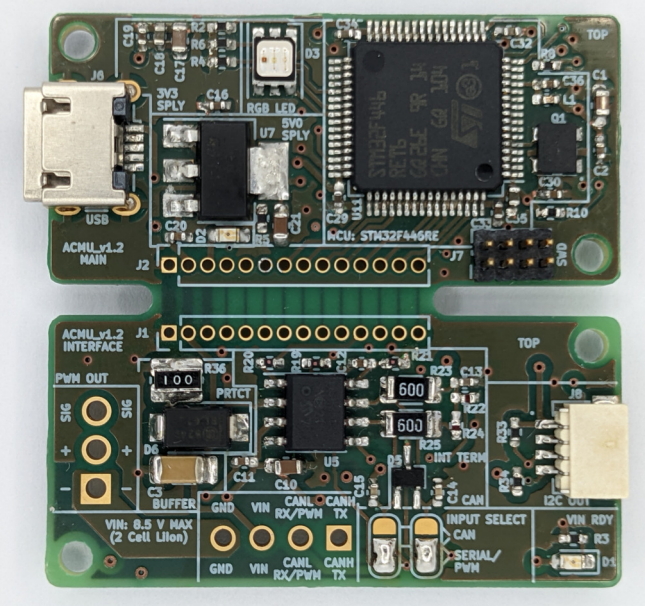


Figure 2. Top view of the ACMU v1.2 main board and interface board.

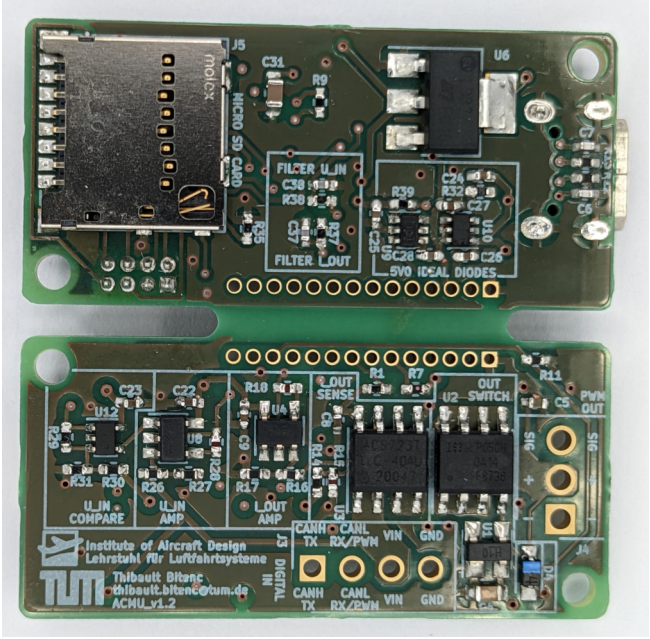


Figure 3. Bottom view of the ACMU v1.2 main board and interface board.

ACMU (Fig. 2, 3) itself can measure power consumption and the target position signal. One RPS board Fig. 6 can be connected via I2C to have the actual position measurement as well. For temperature and other analog measurements, the analog extension board (Fig. 5) is required. In case the deflections of the controlled flaps are required, the flap-RPS boards (Fig. 6b) can be utilized for multiple measurement point at the hinge-lines. On top of that, other I2C capable sensors can be connected to the existing bus.
The separation of main and interface boards allowed a flat and stacked layout. Depending on the available space and integration process, the one with better form-factor can be used. The ACMU has to be connected in series with the servo motor power and PWM line for monitoring but does not need to be placed on or close to the actuator. If external sensor boards are being used, the distance limitation is limited to the maximum length of the I2C bus.

3.1. Main and interface boards

The core of the system is the ACMU main board and interface board. The top and botton view of the ACMU board can be seen on Fig. 2 and Fig. 3 respectively.
The main board contains a STMF446RE Microcontroller Unit (MCU), an SD-card slot, a Micro-B type USB connection, an input power switch to enable powering the device from USB or the input voltage pinheader and Low-Pass filters for two analog measurements. The interface board contains the configurable control inputs, the PWM output for the actuator, the I2C interface connection for the external sensors, the current and voltage measurement circuits and a digital switch for the actuator power.
The main and interface board can be separated into two, and connected together via 1.27mm pinheaders to form a stacked layout. In future development, the two boards can be improved separately if the interface is kept the same.
To increase the accuracy of the analog measurements, several recommendations [14] were used. A voltage divider was used to match the maximum measurable voltage to $3.3V_{DC}$ reference voltage, an operational amplifier(OPAM) was placed in "following operation" mode, to drive the Analog to Digital Converter (ADC) and the filter. A low-pass filter was placed just before the ADC inputs, to suppress high frequency noise. A functional diagram can be seen in Fig. **??**. A similar circuit was used for the voltage and the current measurement as well.

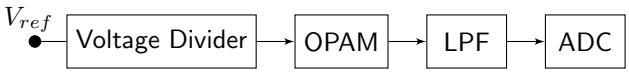


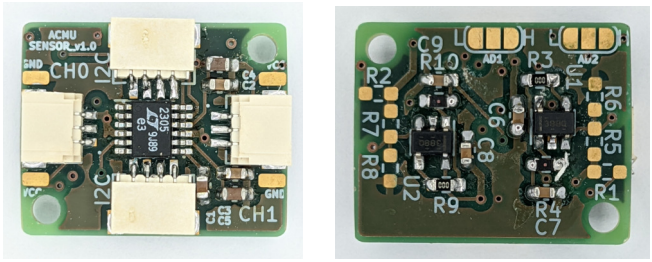
Figure 4. Functional diagram for analog voltage reading.

On top of that, a series of capacitors with values of $0.1\mu F$, $1\mu F$ and $10\mu F$ were placed after the $3.3V_{DC}$ converter to reduce the noise of the reference signal.
The layout of the board was designed in a way, to help during manual assembly by using the silk-screen layer drawings to show the different functionalities. All part numbers and references are visible and readable.

3.2. Analog extension board

An analog extension board was developed to measure additional analog voltages, and provide data via I2C to the ACMU board. The analog channels are digitized on the board and their values feed to the I2C bus. The low integrated design of the board allows custom redesign for special applications. For proof of

concept, the board shown on Fig. 5 was built to support two analog measurement and an I2C channel connection.

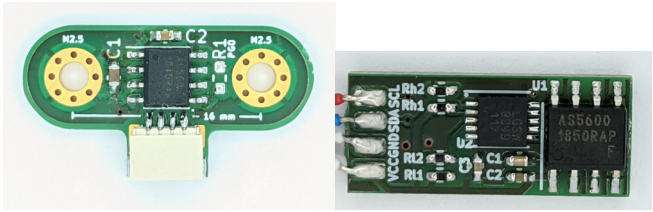


(a) Front side (b) Back side

Figure 5. Front (a) and back (b) side of the assembled Analog Extension Boad.

3.3. RPS sensors

For the measurement of the angle of the actuator shaft, the AS5600 hall type magnetic rotary position sensor is used [15]. This type of angle measurement has several advantages over other commonly applied techniques.
Namely, resistive potentiometers have limited lifetime due to mechanical wear and can show hysteresis during operation. The latter can increase over time due to wear or debris. The possibility of using an encoder based position measurement is not an option due to the size limitation for required precision.
The main advantages of magnetic rotary position measurement are the small size, linearity of the measured data and contactless measurement. On top of that, using external sensors on the actuator allows scalability over different types, and does not require opening the housing of the actuator e.g. to attach sensor wires.



(a) Servo version (b) Flap version

Figure 6. Different designs of the used rotary position sensors.

The same technology can be used to measure the position of the control surface as well. This can be used as more precise feedback for autopilots or to detect the failure of the linkage or unwanted flap movement.
One limitation of the AS5600 sensor is the I2C address, which is fixed. That allows easy usage and minimal additional hardware but requires more parts if multiple sensors are desirable.
Currently, two PCB layouts exist for the RPS sensor. The first, the servo version shown in Fig. 6a, is meant to measure the actuator shaft position, by mounting it over the servo rotational shaft. The second, the flap version shown in Fig. 6b, is meant to mount close to the hinge-line of the actuator to measure the actual position of the actuator surface.
The flap version was designed with an I2C address converter on board, to allow the usage of multiple AS5600 on the same I2C bus.
A servo mounting assembly is shown in Fig. 7. Although, it requires extra parts like the magnet holder, a ring magnet, the RPS bracket and the RPS sensor itself, it gives a flexible assembly, which can be easily used on most of the available servos. The custom parts may be easily manufactured using 3D printing techniques.

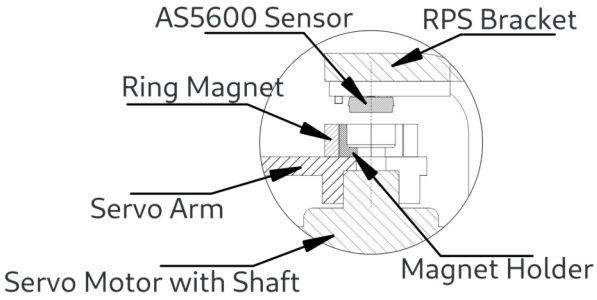


Figure 7. Servo mounting assembly for the RPS sensor.

3.4. Limitations

The electrical, physical and input-output configurations are presented here.
The ACMU board can operate either from USB or from external voltage up to 8.5V_{DC}. The analog extension and RPS boards are operating at 5V_{DC}, and the ACMU board can supply them via the power lines of the I2C connection. Table 1 shows the physical dimensions of the two possible configurations of the ACMU board with housing, and the dimensions of the external parts. Table 2 shows the possible working configurations in terms of control inputs and provided output data.

	x	y	z.
ACMU flat	40	40	11
ACMU stacked	40	20	19
Analog extension	18	22	8
RPS actuator	25	13	3.3
RPS flap	17.5	7.5	3.3

Table 1. Physical dimensions of different parts, configurations. The values are in [mm].

4. SOFTWARE DEVELOPMENT

This part focuses on the main architecture of the developed software and shows some key decisions, like programming language, development environment and the used development practice.
The aim of the initial software development phase was, to have a working prototype with sufficient functionality, and leave open

Out \ In	PWM_{in}	UART	CAN	USB
PWM_{out}	X	X	X	X
USB	X	X	X	X
CAN			X	X
UART		X		X

Table 2. Possible configuration of control inputs and outputs.

the design for further changes. With that, the system was flight-ready within 6 months after the initial requirements were set. Since then, new functionalities and improvements were gradually introduced to the system without limiting the original capabilities.

4.1. Use of embedded operating system

The Mbed OS [16] is used as a RTOS. It has good integration with the STM32 microcontroller family, good support for modern C++ , reliable documentation, and many available libraries. Other free and open-source embedded operating systems were considered as well, like freeRTOS [17] or the Zephyr Project [18]. The Mbed OS was chosen due to the reasons listed above.
The extensive external libraries and build in functionalities allowed to use of pre-build and tested functionalities like PWM reading and generation or creation of a local file system on an SD-card.

4.2. Main architecture

In the current state, the software is divided around its two main functions of control and monitoring. The control part is responsible for driving the servo itself, based on the input PWM signal. The highest priority thread is responsible for copying the captured input signal to the output PWM generator. On top of that, the architecture allows injecting a custom digital control algorithm, to reshape the PWM signal.
The monitoring part is responsible for the collection and storage of the measurements. The data collection and storage parts are separated into different threads, and an asynchronous workflow is set up for smooth operation. Figure 8 shows a detailed sequence diagram of the data acquisition design.
Data is requested from each sensor in a real-time thread with 5ms intervals. When all the data is available, a comma-separated string is created, which contains the new data set and a time-stamp. The string is stored in a double-buffer configuration. When one buffer is full, the buffers are swapped, and the content of the full buffer is written to a file on the SD-Card. With this setup, the PWM to PWM translation is done independently from other parts of the system, which allows having a reliable control of the servo itself.

4.3. Use of C++

During software development, C++ was used to write the software for the MCU. That allowed the use of high-level language elements like classes or polymorphism.

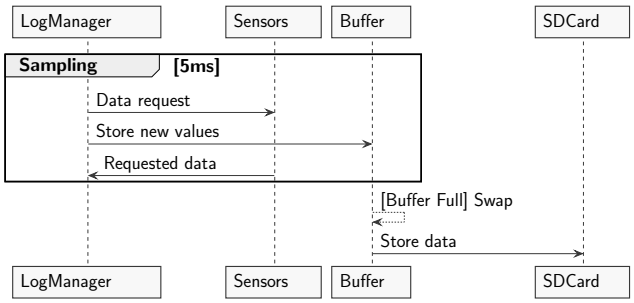


Figure 8. Sequence diagram of the data acquisition.

For example, the use of polymorphism allows the usage of the vector class of the C++ standard library for collecting all the available sensor classes and provides an uniform interface for requesting data from the different sensors. With this approach, the overall source code only has to be changed at a single point to allow the use of new sensors.
Although it is possible to use polymorphism in C as well with function pointers, C++ does it in a safer and more understandable way [10].
Modern language elements like constant expressions, auto, enum classes, literals and initialization list are used extensively to improve readablitly. At the current state, the high-efficient hardware resources like hardware buffers and DMA unit of the MCU are not fully exploited.

4.4. Use of Test-Driven Development

Studies showed, that the usage of Test-Driven Development (TDD) in embedded systems can fasten up the development progress, by eliminating most of the compile-download-test cycles [19]. Among other positive effects, software written in a test-driven way will have a good module separation via proper interfaces [20], which allowed to change modules or even change the underlying operating system, if necessary.
For example, a simple control class was developed to allow the control of the output PWM signal via messages over USB. Altough the initial effort was high due to the need for developing a test-harness which mimics some specific hardware dependent libraries, adding and testing new messages and functionaliteis proved to be minimal effort. All parts of the class were developed without actually donwloading program to the MCU, thus allowing immediate feedback after any changes of the implementation. The developed class was tested in a real environment as well, when it had sufficient functionality, and it worked on first try as expected.

5. LABORATORY AND FIELD TEST RESULTS

To make sure that all the designed hardware elements are working as expected, laboratory tests were conducted. The overall goal of the development process was to have a working hardware, with minimal software which is reusable in early flight-test. Hardware functionalities like CAN or the USB interface were tested with individual software.
Each hardware interface was tested and internal and external

measurement data were collected for comparasion. Here, only relevant results will be shown. All the test and their configuration can be read in Thibault W. Bitenc Master Thesis [8].

5.1. Laboratory tests

The input voltage measurements were validated against the measurements of a multimeter. The results can be seen in Fig. 9. It is clearly visible, that the measured voltage by the ACMU is different from the target voltage. However, the fitted regression line shows good linearity along the individual points.

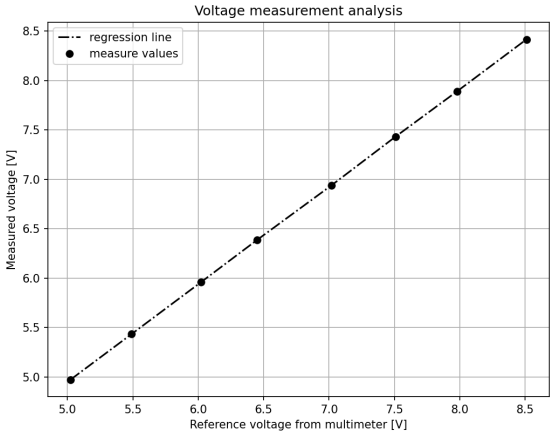


Figure 9. Input voltage measurement validation.

The position feedback from the RPS sensor shows acceptable results. On the raw data, filtering is required to reduce the noise of the internal ADC of the AS5600 sensor. In overall, with digital actuators, the $0.5-0.1$ degree accuracy is achievable. Figure 10 shows the reference and measured position.

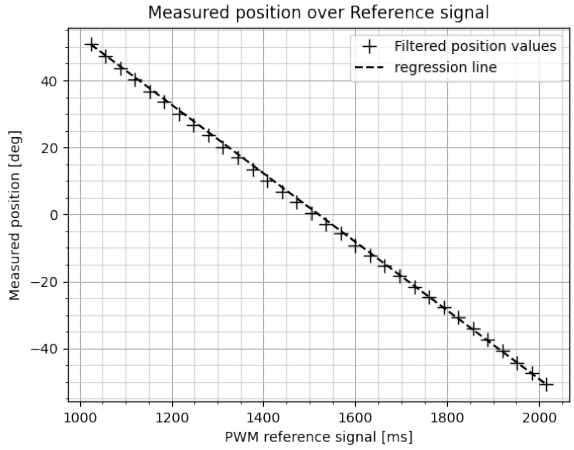


Figure 10. Position feedback results against input reference signal of a Hi-Tech D980-TW actuator.

In general, there was no post-processing used on the measurements of the board. It is clear from the laboratory tests, that better calibration on the software side (gain and offset compensation), and minimal filtering is needed to have an accurate measurement, despite the effort made on the hardware side.

5.2. Field tests

After the laboratory tests, a series of flight-tests were conducted with a stable software version, which had all logging features enabled.

For the flight, only the ACMU board was used. The board was connected in series to the PWM and power line of the elevator of a Bixler hobby glider [21]. With that, reference signal and power consumption measurements were made in flight. During the test, three consecutive flights have been conducted. The system was powered on and off between flights.

During the flights, no additional delay was experienced by the pilot. A new log file was created for each flight, and all the expected data were properly logged to the SD-card.

The measured values can be seen in Fig. 11 from the third flight in above mentioned configuration. From the consumed current and the target position measurements, unexpected load cases can easily be identified. From system startup, it took about 3 minutes to check the correct operation of the system and to do the required manual adjustments. The actual flight lasted approximately 8 minutes.

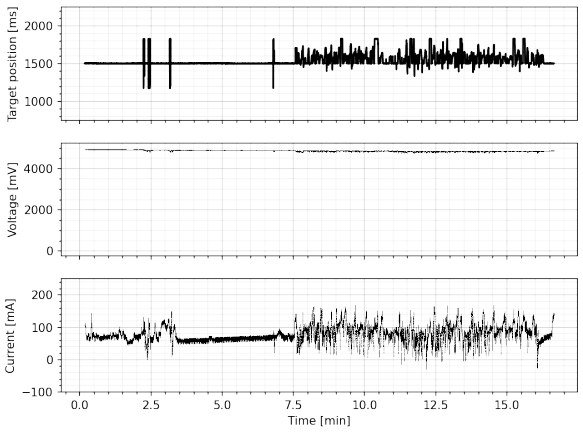


Figure 11. Data recorded during test flight-test. The actual flight starting at 7.5 minutes.

6. FUTURE PLANS

The original work in which the ACMU was developed [8], focused mainly on hardware development and preparing a reliably working prototype. Due to that, the development of several high level or complex functionalities was not addressed yet. For example, the utilization of DMA, standard USB connection, permanent memory, software configuration changes via configuration files or the system log have not been implemented or only been developed partially.

The built-in DMA of the MCU could be used for automatic data sampling into an n long buffer, which can serve as a moving average filter for n samples with low computation cycle. Similar buffering could be used for all the measurements with individual buffers.

At current state, there is one sampling frequency used for every measurement. Individual sampling frequencies would allow to use the right sampling for each measurement based on the applied technology and sensors. Along with the individual filtering, dedicated sampling frequencies would allow to implement an over-sampling and filtering measurement approach, which will yield well filtered data already on the MCU.

Due to the standard USB interface, easy connection to any PC is possible, which can allow functionalities like firmware update, parameter reconfiguration or accessing the data logs. An easy-to-use cross platform application can be developed, to allow semi-technical users to operate an ACMU, without the need of any development equipment.

Any modern system which keeps usability in mind, is capable of reconfiguration with minimal user effort. The basics of such functionality are available on the current system, but not developed yet. Potential usage could cover the settings of minimum, maximum and neutral positions of the actuator, maximum operation temperature, data sampling frequencies, usable sensors types etc.

If the internal permanent memory is used, the ACMU would be able to start and run without an SD-card and still keep a dedicated configuration.

An operation or system log is a proved method for long term monitoring of the functionality of an embedded device, which is not reachable via some terminal application. Such a log will be usable not just during the further development process, but to prove the reliability of the system as well.

Future plans for these sytems include functionality extensions like the implementation of model-based observers for increased measurement accuracy [22, 23], and usage in the DG-800 S [24] UAV available at our Institute.

6.1. DG-800 S retrofitting

A framework is developed for the purpose of helping the retrofitting process of the DG-800 S research UAV, with ACMU's for the primary control functions. The base of the framework is a wooden platform, where all the equipment can be placed and connected, in an easily accessible manner. It consists of four actuators with one ACMU each, an SBUS capable receiver and a Raspberry Pi as shown on Fig. 12. A planned future configuration is shown in Fig. 13, where a CAN-converter is intended to be used to communicate with the ACMU and provides interfaces to a Raspberry Pi and an RC receiver.

The framework is intended to be used as an integration and test platform, first for prooving safety and steadiness of the basic functionalities, secondly before introducing any new functionality to the flyable demonstrator itself.

Developing such a platform can help to gain knowledge and experience with the system, and keep the actual aircraft unoperational period as short as possible.

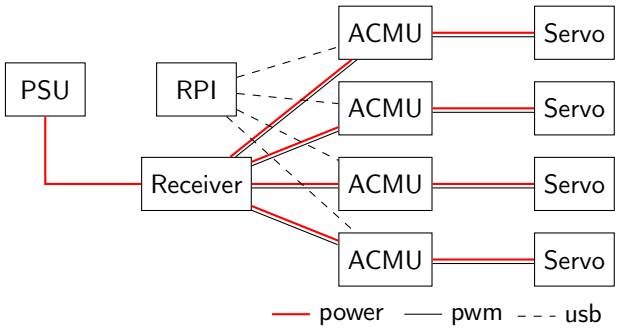


Figure 12. Current test configuration for the DG-800 S in a woodbird setup.

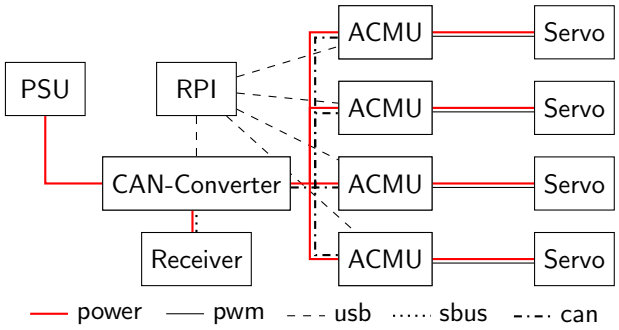


Figure 13. Planned test configuration for the DG-800 S in a woodbird setup.

7. CONCLUSION

A novel hardware platform was developed for in-flight operations to monitor and control a simple PWM-driven UAV actuator. The developed hardware and software were successfully tested in a laboratory environment. On top of that, a series of successful flight tests were conducted with the ACMU board connected to the elevator servo of a hobby glider.

Based on the early results, the ACMU as a system can be a beneficial extension in the overall life-cycle of a research aircraft. Its extensible sensor interface can be helpful to install additional measurements with minimal effort onto an already working system.

On top of that, it proves to be a versatile platform for control and monitoring functions dedicated for electro-mechanical actuators in UAV applications.

8. Glossary

ACMU Actuator Control and Monitor Unit

ADC Analog to Digital Converter

BLDC BrushLess Direct Current

CAN Controller Area Network

COTS Commercially off-the-shelf

DC Direct Current

DMA Direct Memory Access

EMA Electro-Mechanical Actuator

I2C Inter-Integrated Circuit protocol

IDE Integrated Development Environment

MCU Microcontroller Unit

MIMO Multiple Input Multiple Output

MTOM Maximum Take-Off Mass

OPAM Operational Amplifier

OS Operating System

PC Personal Computer

PCB Printed Circuit Board

PWM Pulse Width Modulation

RC Radio Controlled

RPS Rotary Position Sensor

RTOS Real-Time Operating System

SBUS Serial BUS, in Remote Controlled applications

TDD Test-Driven Development

UART Universal Asynchronous Receiver-Transmitter

UAV Unmanned Aerial Vehicle

USB Universal Serial Bus

9. REFERENCES

[1] Futaba s.bus high voltage system. https://www.rc.futaba.co.jp/english/hv/index.html. Accessed: 2021-03-15.

[2] Hitech Multiplex industrial series servos. http://www.hitecrcd.com/products/servos/industrial_series/md950tw-20mm-coreless-titanium-gear/product. Accessed: 2021-03-15.

[3] VOLZ-servos standard actuators. https://volz-servos.com/produkte/standard/. Accessed: 2021-03-15.

[4] J.C. Maré. *Aerospace Actuators 2: Signal-by-Wire and Power-by-Wire*. Number v. 2 in Robotics series. Wiley, 2017. ISBN 9781848219427. URL https://books.google.de/books?id=fiFIDgAAQBAJ.

[5] István Réti, Márk Lukátsi, Bálint Vanek, István Gőzse, Ádám Bakos, and József Bokor. Smart mini actuators for safety critical unmanned aerial vehicles. In *2013 Conference on Control and Fault-Tolerant Systems (SysTol)*, pages 474–479. IEEE, 2013.

[6] Dr. István Harmati Tamás A. Fehér. Mechanical construction and control of biped robot master thesis.

[7] Jurij Sodja, Roeland De Breuker, Yasser Meddaikar, Johannes Dillinger, Keith Soal, Yves Govers, Wolf Krüger, Panagiotis Georgopoulos, Christos Koimtzoglou, Christian Roessler, Sebastian Köberle, Julius Bartasevicius, Daniel Teubl, Gyulai László, Szabolcs Toth, Mihaly Nagy, Daniel Balogh, Miklos Jasdi, Péter Bauer, and Bálint Vanek. Ground testing of the flexop demonstrator aircraft. 01 2020. doi: 10.2514/6.2020-1968.

[8] Thibaul W. Bitenc. Development of an electro-mechanical actuator control and measurement unit master thesis. URL https://www.lrg.tum.de/lls/startseite/. Accessed: 2021-07-10.

[9] Robert C. Martin. *Clean Code: A Handbook of Agile Software Craftsmanship*. Pearson, 2008. ISBN 9780136083238.

[10] Robert C. Martin. *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Pearson, 2017. ISBN 9780134494272.

[11] Scott Meyers. *Effective Modern C++: 42 specific ways to improve your use of C++11 and C++14*. O'Reilly Media, 2014. ISBN 9781491903995. URL https://www.oreilly.com/library/view/effective-modern-c/9781491908419/.

[12] Modern C++ embedded systems – part 2: Evaluating c++, . URL https://www.embedded.com/modern-c-embedded-systems-part-2-evaluating-c/. Accessed: 2021-07-18.

[13] Modern C++ embedded systems – part 1: Mythn and reality, . URL https://www.embedded.com/modern-c-in-embedded-systems-part-1-myth-and-reality/. Accessed: 2021-07-18.

[14] AN2834 application note - how to get the best adc accuraccy in stm32 microcontrollers. https://www.st.com/resource/en/application_note/cd00211314-how-to-get-the-best-adc-accuracy-in-stm32-microcontrollers-stmicroelectronics.pdf. Accessed: 2021-07-12.

[15] AS5600 - datasheet. https://ams.com/documents/20143/36005/AS5600_DS000365_5-00.pdf/649ee61c-8f9a-20df-9e10-43173a3eb323. Accessed: 2021-07-21.

[16] mbed-os. URL https://os.mbed.com/mbed-os/. Accessed: 2021-07-18.

[17] FreeRTOS: Real-time operating system for microcontrollers. URL freertos.org. Accessed: 2021-07-18.

[18] Zephy project. URL https://www.zephyrproject.org/. Accessed: 2021-07-18.

[19] James W. Grenning. *Test-Driven Development for Embedded C*. Number v. 1. The Pragmatic Programmer, 2011. ISBN 9781934356623. URL https://pragprog.com/titles/jgade/test-driven-development-for-embedded-c/.

[20] Kent Beck. *Test-Driven Development by Example*. Pearson, 2003. ISBN 9780321146533.

[21] H-king bixler 1.1. URL https://hobbyking.com/en_us/h-king-bixler-1-1-epo-1400mm-glider-pnf.html. Accessed: 2021-07-18.

[22] R. Pantonial, A. Kilantang, and B. Buenaobra. Real time thermal estimation of a brushed dc motor by a steady-state

kalman filter algorithm in multi-rate sampling scheme. In *TENCON 2012 IEEE Region 10 Conference*, pages 1–6, 2012. doi: 10.1109/TENCON.2012.6412194.

[23] W. Zhang, S. A. Gadsden, and S. R. Habibi. Nonlinear estimation of stator winding resistance in a brushless dc motor. In *2013 American Control Conference*, pages 4699–4704, June 2013. doi: 10.1109/ACC.2013.6580564.

[24] Sebastian J. Koeberle, Adrian E. Albert, Lars H. Nagel, and Mirko Hornung. *Flight Testing for Flight Dynamics Estimation of Medium-Sized UAVs*. doi: 10.2514/6.2021-1526. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2021-1526`.