Przemysław GĄSIOR, Adam BONDYRA, Stanisław GARDECKI
POZNAŃ AERIAL ROBOTICS TEAM, INSTITUTE OF CONTROL AND INFORMATION ENGINEERING
POZNAŃ UNIVERSITY OF TECHNOLOGY
3A Piotrowo St., 60-965 Poznań, Poland

Comparison of different methods of flight data logging used in multirotor UAVs

Abstract

This paper describes the comparison of different methods of flight data logging in multirotor unmanned aerial vehicles. Solutions used in commercial avionics systems are presented and compared with the current design developed by authors. Based on the literature and analyzed examples, requirements for two separate logging systems are formulated. This approach allowed to modify current solutions and develop new methods for the high frequency data logging. Three different memory storages were considered for this task - SD card, RAM and external NAND Flash. Based on the theoretical calculations, developed prototypes and performed experiments, the best solution was selected.

Keywords: microavionics, flash, RAM, memory card.

1. Introduction

The information on the status of the aerial platform is vital to perform a safe flight. In most cases, only control commands from the operator are sent to the flying robot and processed by control algorithms. Additionally, a second radio channel can be added to transmit basic telemetry data. This method is characterized with relatively low frequency and is used mostly in on-line status monitoring along with the ability to transmit high-level commands (like following a path on waypoints). Therefore, one important problem arises - range and quality of those wireless connections. During a flight, the platform can exceed the area of the radio signal coverage which leads to the corruption or loss of received data. Thus, the system ought to be equipped with an additional on-board logging capability of crucial status variables in case of an accident. This feature is not dependent on issues with radio connection quality and allows to investigate causes of system malfunction. It is also known as a so-called *black box* in classical aviation.

More sophisticated scientific research, like the development of estimation and control algorithms, requires significantly higher data refresh rate along with access to inner variables of the system. Mentioned conditions are caused by the verification process of algorithms under their standard working conditions. Solution to this problem is also to log this information on the platform during the flight and process it offline. This subsystem should not interfere with previously described *black box* approach in order to maintain the high reliability.

Finally, scientific platforms require two systems of data logging - with low and high frequency rates. Both stated solutions can be achieved in many ways, which all come down to the one main ingredient - additional memory storage connected to main avionics elements. Block diagram with mentioned division including approaches described in this paper is presented in Fig. 1.

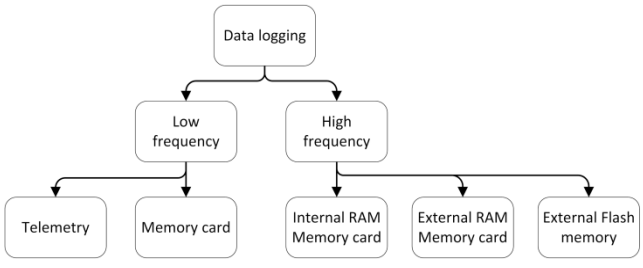


Fig. 1. Block diagram of described data logging techniques

2. Related work

Our research group is developing customised avionics systems for multirotor aerial vehicles adapted to different areas of scientific research. One of the latest versions of control modules was described in [1] and it is currently used in the *Falcon V5* platform [2]. Mentioned avionics system consists of three main modules - *AHRS* (Attitude and Heading Reference System), *Mainboard* and *Powerboard*. The most important component from the perspective of this paper is the *Mainboard*. In this module, control algorithms are executed and it merges all the data from the whole system. Therefore, it should be the place to implement additional memory storages for logging tasks. Fortunately, the *Mainboard* already has a capability of the so-called *black box*. Its implementation is based on a SD card connected via SPI (Serial Peripheral Interface) with usage of DMA (Direct Memory Access). The filesystem has been employed and utilized to simplify data downloading. During the initialization phase, flight parameters are read from the configuration file. Then, the second file with a unique name is created, which will be used to store status variables. Write process is sequential and represented as a separate line (around 500 Bytes) for each time stamp. The maximum write speed of this approach is only around 126 kB/s because of the non-optimal size of written data and converting floating point variables to its text representation. Because data logging is the task with one of the lowest priorities in the flight control system and the 10 Hz write frequency was selected, mentioned data rate is lowered to 5 kB/s with overall processing time equal to 36.7 ms per 1 s. The performance of this subsystem was not satisfactory for current development stage and can be improved by the modification of the software procedure. Bigger buffers can be utilized and data could be sent in sequential multiblock write procedure. Additionally, SD mode communication based on the SDIO (Secure Digital Input Output) standard can be implemented. As stated in [3], it has better results than SPI in the data transfer with memory card. Authors decided to focus on new communication approach (SD mode) based on planned upgrades in the hardware. Therefore, this paper will focus on modification of existing *black box* subsystem and mostly on developing the solution for a second logging method - the high frequency logger.

There are many open-source or commercial avionics modules available on the market. The most popular are described in [4]. Despite that their purpose is mainly aimed at users not performing scientific research, some are equipped with specific routines for data logging. One of them is Ardupilot [8]. Analyzing mentioned avionics system, in terms of this paper, reveals that it has two options for data logging - *tlogs* and *dataflash logs*. The first one is the telemetry log sent to the GCS (Ground Control Station) and saved there by specialized software. The maximum frequency of the data acquisition executed this way is less than 10 Hz. The second technique is based on an additional on-board flash memory with the size of 4 MB. This method allows gathering information with a frequency equal to 50 Hz. Above parameters are extracted from the online documentation.

Less popular designs, like *Paparazzi* [9], have no good logging capabilities but they are still under development of the community. Various commercial manufacturers, such as Mikrokopter [10], deliver not only flight controllers but also complete aerial systems. Despite many possible application areas, this particular avionics system is not very effective in high

frequency data logging. Its capabilities are limited to 1 Hz telemetry link and 5 Hz data recording thanks to the on-board SD card.

Probably the most advanced open-source system available is the *Pixhawk* [11]. It consists not only of the standard flight controller but also in some versions of an additional embedded computer. In addition, this avionics has the best logging abilities. Similar to previous examples, data can be sent via telemetry link and it is also equipped with an additional SD card. High frequency logging can be executed with arbitrary rate limited to CPU load and speed of the memory card. The typical refresh rate is around (100-200) Hz. Unfortunately, if the controller cannot meet required frequency, some packets may be skipped, which is unacceptable in many cases of scientific research. It is also worth to mention, that this avionics uses 32 kB buffers in the transfer to the SD card.

All presented examples are open-source. Thus, a developer can port such firmware to arbitrary hardware and develop customized logging routines. Professional drones with full support for the scientific research, which includes many desired functions, are also available on the market. Examples of this kind of UAVs (Unmanned Aerial Vehicles) are robots developed by the *Ascending Technologies* company [12]. Unfortunately, such platforms are expensive and do not have an open source code.

3. Requirements and proposed solutions

All examples presented in previous sections have advantages and disadvantages. However, for our research purposes, there are strict requirements to meet. Those requirements are divided into two groups - for low and high frequency logging. They specify a minimum data frequency rate, a minimum number of saved variables and the duration of data acquisition.

Black box logging task is required to maintain a minimum 10 Hz refresh rate. This frequency allows diagnosing what happened to the system during unexpected behavior. Currently this routine records around 50 variables in which at least 90% of them are floating point types. Raw data size is around 200 bytes, but after conversion to the text representation, the size reaches to almost 500 bytes because of high precision in floating numbers. This excess can be reduced to 300 bytes by rounding them. As mentioned in the introduction, a *black box* data logging task has one of the lowest priorities in the system, therefore it is preempted by control and communication functions. Thus, in this approach is better to divide written data into separate vectors assigned to specific time stamp than cumulating them and save as big buffers.

This is a demanding task, but additionally, the created file is periodically closed and opened in order to keep data saved in case of an emergency power cutoff. Mentioned parameters should be maintained or even improved along with the reduction of the processing time by the modification in software routine or usage of the SD mode in communication with the memory card.

High frequency logging is a much more demanding task. The goal is to achieve a minimum of the 800 Hz sampling rate. It will allow diagnosing estimation algorithms in the AHRS module and record raw data from the IMU (Inertial Measurement Unit) sensor without skipping samples. These actions are vital for proper development of those algorithms, for example during determining variances of measurements. Additionally, control routines which are executed at the frequency equal to 400 Hz could also be debugged. Based on [5] and examples from the previous section, we selected three main solutions to the stated problem, which are listed and described below.

3.1. First approach – internal RAM + SD

The first solution was based on already implemented mechanism utilizing the memory card and internal RAM (Random Access Memory) of the *Mainboard*'s microcontroller. This approach was used until now, but it required a modernization because of diagnosed limitations.

In the idle RAM area, the table of measurements was declared and it was being filled with data in the software loop. The start of the sequence is initiated by the operator and duration is strictly depended on a number of variables and sampling frequency. For example, in the current software version, there is a possibility to declare a table with 27,800 floating point elements. During recording of four variables (the time increment and 3 other floating point variables), a number of maximum elements should be divided by four, which gives the result of 6950 measurement cycles. Recording time at 400 Hz sampling frequency is equal only to 17.375 s. If more variables have to be saved within the same period of time, the frequency has to be reduced.

This solution does not have any delay or strict timing requirements to meet during recording of given measurements but the another problem arises - how to save those data to a non-volatile memory. Due to the characteristic of RAM, all data will vanish after the power-off. Thus, right after end of the measuring phase, a new file is created in the SD card and the data is being copied. Unfortunately, the standard frequency of memory card logging, which is equal to 10 Hz, will cause a great increase of time. In the case of 6950 measurement cycles, this process will last almost 12 minutes. This is a big disadvantage of described approach because there is no possibility to increase write frequency during the flight to ensure proper operation of flight control tasks. Therefore, we proposed one simple improvement – raise a mentioned frequency only after the touchdown and with disarmed motors. Such modification resulted with five times higher rate of data acquisition. Unfortunately, a problem of a limited number of variables still persists.

3.2. Second approach – external RAM + SD

The second approach is an expansion of the first one. We decided to increase the size of RAM by using an external circuit connected to the microcontroller with FSMC (Flexible Static Memory Controller) interface. Described solution was tested with IS42S16400J 64-Mbit SDRAM (Synchronous Dynamic Random Access Memory) with 12-bit address and 16-bit data buses. This approach allowed to gather more data, over two million floating point variables. Unfortunately, the bigger is the number of variables, the longer is the time of saving the data to the SD card. The usage of this method is almost identical to the previous approach, with the only difference that table was declared in an arbitrary address in the external memory.

3.3. Third approach – external Flash

The third, significantly different solution, is based on the external, non-volatile flash memory integrated circuit. There are two main types of those memories, NAND and NOR. They differ in some functional aspects because of a structure of the memory cell. First one is better suited for storing sequential data. However, the second type preferable in random access and executing programs [6]. NAND type was selected because of a better performance in write and erase operations, which are vital for logging applications. In the prototype, the 1Gb memory connected by 8-bit FSMC interface was used. This storage circuit is divided into pages, blocks and planes. Described version has only one plane, which is composed of 1024 block which consists of 64 pages and finally, each page has 2048 bytes. This type of memory eliminates the need for moving data to the SD card. Variables are written directly to a non-volatile area. But there are two major problems in this approach, accessibility of those data and write speed.

The capacity of a implemented memory encourages to store multiple measurement intervals, not only the current one. This approach will be even healthier for the circuit, because it will use a whole plane, not only first sectors. The manufacturer ensures reliability for 100,000 Program/Erase cycles which is a standard level for those devices. The recording process has to be managed properly, therefore, a simplified system for identifying each

sequence with the specification of address range is essential. This is implemented in the first block of the memory. Each measurement has a separate page which stores this information along with the date and names of variables. There is also a possibility to implement a file system on the mentioned external memory, like in [7]. But we decided to keep our custom routine because of a simplicity and customization to our needs. Recorded high frequency measurements can be downloaded by dedicated PC software communicating with *Mainboard* by UART (Universal Asynchronous Receiver and Transmitter), CAN (Controller Area Network) or USB.

The second problem, connected with the programming time, can be shown by an example. Assuming that a number of variables to write is equal to 8 and the sampling frequency is set to 800 Hz, there is a possibility to calculate needed time to write this data. As stated earlier, the NAND flash has better performance during a sequential write process, therefore we decided to buffer measurements and write the whole page afterwards, instead of variables from the single cycle. Based on this assumption, buffer overflow period will be equal to 80 ms (2048 bytes = 512 floats, 512 floats / 8 = 64 cycles, 64 cycles @ 800 Hz = 0.08 s). The manufacturer stated in the datasheet that page programming time is not greater than 25 μs which fulfils calculated limit and all comes down to the processing time in the microcontroller. Based on experiments, a time needed to write one page is equal to 332 μs, therefore the speed of this interface is sufficient.

4. Conclusions

Logging methods are commonly used in many aerial avionics designs. They are implemented as a telemetry link or an on-board memory storage (*black box*). As stated earlier, telemetry may be limited by range, throughput and faulty data, therefore the solution based on the SD card is a better suited for this task. Current SPI implementation of data transfer to the memory card has been replaced by the SD mode communication with DMA and resulted in over ten times reduction of processing time. This reservoir can be used for implementation of new algorithms or increase the sampling frequency of data recording. The comparison of developed solutions is presented in Tab. 1.

Tab. 1. Performance comparison of developed solutions

	Maximum data rate, kB/s	Single cycle processing time, ms	Data packet size, B
SD (SPI)	126.66	3.670	476
SD (SD mode)	1541.77	0.302	476
Ext. Flash	6024.1	0.332	2048

All mentioned approaches for high frequency logging have advantages and disadvantages, therefore, it is hard to compare them in an easy way. The first solution can be used in any avionics system equipped with the memory card and a microcontroller with sufficient amount of RAM. This idea is relatively easy to implement and can help other researchers to verify their algorithms and outputs of sensors. The second method requires not only software modifications but also hardware modernisation of existing avionics. Even if external SDRAM memory multiples available storage and measurement time, it extends transfer time to the SD card. Unfortunately, sometimes there are no conditions for keeping the platform running for this long.

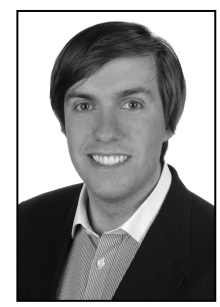
The third solution based on external flash memory also requires many modifications in firmware and hardware. Additional PC software is likewise expected, but thanks to that, there is a possibility to develop a fully reconfigurable logging system. It has also the best performance compared to the both implementations of memory cards (Tab. 1.). This approach is best suited for fully enclosed avionics modules with output data buses and eliminates mechanical interaction with avionics (inserting memory card). Based on recent development plans, our system will pursue this concept. In future work, a research on different algorithms for buffering data before writing to the memory card will be conducted. Implementation of bigger buffers should improve achieved data rate and optimise processing time. Additionally, memory cards with higher speed class will be tested.

5. References

[1] Bondyra A., Gardecki S., Gąsior P.: Distributed control system for multirotor aerial platforms. Measurement Automation Monitoring, vol. 61, no. 07, pp. 343-346, 2015.
[2] Bondyra A., Gardecki S., Gąsior P., Kasiński A.: Falcon: A compact multirotor flying platform with high load capability. Advances in Intelligent Systems and Computing, vol. 351, pp. 35-44, 2015.
[3] TOSHIBA: TOSHIBA SD Card Specification. 2006.
[4] Lim H., Park J., Lee D., Kim H.J.: Build Your Own Quadrotor: Open-Source Projects on Unmanned Aerial Vehicles. Robotics & Automation Magazine, IEEE, vol. 19, pp. 33-45, 2012.
[5] Suzdalenko A.: Guidelines for autonomous data logger design. Industrial Electronics (ISIE), 2011 IEEE International Symposium on, pp. 1426-1429, 2011.
[6] Micron Technology Inc.: NAND Flash 101: An Introduction to NAND Flash and How to Design It In to Your Next Product. Micron Technol-ogy Inc., Tech. Note, 2006.
[7] Penson W., Fazackerley S., Lawrence, R.: TEFS: A flash file system for use on memory constrained devices. Electrical and Computer Engineering, 2016 IEEE Canadian Conference on, pp. 1-5, 2016.
[8] http://www.ardupilot.co.uk/
[9] http://wiki.paparazziuav.org/
[10] http://www.mikrokopter.de/en/products/feateng
[11] https://pixhawk.org/
[12] http://www.asctec.de/en/

Received: 06.02.2017 *Paper reviewed* *Accepted: 04.04.2017*

Przemysław GĄSIOR, MSc



Research assistant in Institute of Control and Information Engineering at the Poznan University of Technology. Graduated in 2014 and received a MSc diploma in Automatic Control and Robotics with major in Robotics. Now, as a member of PART research group, his areas of research are aimed to control algorithms, modeling, estimation and development of multirotor aerial platforms.

e-mail: przemyslaw.gasior@put.poznan.pl

Adam BONDYRA, MSc

Research assistant in Institute of Control and Information Engineering, Poznan University of Technology. Graduated in 2014 and received a MSc diploma in Automatic Control and Robotics. Now, as a member of PART research team his areas of research are control algorithms, telemetry techniques, fault detection and fault-tolerant systems in micro UAVs.

e-mail: adam.bondyra@put.poznan.pl

Stanisław GARDECKI, MSc

Received the MSc degree in control engineering and robotics from Poznań University of Technology (Poland) in 2009. He was honored several times for achievements in the field of robotics. Currently he is an assistant at the Institute of Control and Information Engineering of Poznan University of Technology and a leader of PART (Polish Aerial Robotics Team). He is developing flying robots and researches their stabilization under varying load and configuration.

e-mail: stanislaw.gardecki@put.poznan.pl