

## Design and implementation of a compact avionics instrument for light aviation

Mustafa Emre AYDEMİR\*

Department of Electronics Engineering, Turkish Air Force Academy, Yeşilyurt, İstanbul, Turkey

Received: 21.07.2014

Accepted/Published Online: 30.04.2015

Final Version: 20.06.2016

**Abstract:** In this study, a low-cost compact avionics instrument has been designed and implemented for light aviation vehicles such as ultralight airplanes, hang glider-trikes, and paramotors. At the moment there are a few commercial products that provide flight data and vehicle statuses such as altitude, temperature, fuel level, and engine hours meter. However, due to low production rates, customization for different air vehicles, and complex electronics and signal processing hardware, these instruments are generally far from being affordable; their cost may even be more than that of the air vehicle. This study presents a novel approach such that recent state-of-the-art hardware and sensors are utilized to provide critical flight data to the user accurately. The commercial off-the-self hardware used in the instrument may easily be acquired from the electronics market. Devices like these may be utilized not only for aviation but also sea and land vehicles, providing the user with important data at an affordable cost. In this study, design of the device is explained in detail, which may be constructed with basic electronic circuit construction tools and be utilized as an educational tool as well. The instrument that leverages flight security may be tailored for any air vehicle by using the methods in this study.

**Key words:** Instrumentation, embedded systems, sensors

### 1. Introduction

Ultralight aviation is becoming popular fast. As the vehicles are becoming more affordable and capable, they are being utilized by not only amateurs but the police, the film industry, and fire control and prevention departments, as well. Mostly being powered by simple 2-stroke engines, these vehicles are easy to fly and maintain. In most cases, these air vehicles are flown without use of any avionics. Most users avoid complexity and just want to use their vehicles casually without expensive avionics. So far, some avionics have been developed for ultralight aircraft. However, these systems are designed equivalent to professional avionics, some even involving embedded Linux operating systems [1]. In some systems, the pilot-vehicle interface is almost the same as in a commercial transportation aircraft [2]. As poor user interface design has been a contributing cause of many aerospace accidents and causalities, the avionics software development process is required by law and it has been optimized for safety [3]. Ensuring the safety criteria is a long and costly process and this price is reflected on the avionics price tag.

Meanwhile, as MEMS (microelectromechanical sensors) and GPS (Global Positioning System) sensors have become commercially available at very low prices, some companies have started building avionic instruments for limited-budget users by using these sensors. A group of these instruments (Alti-Variometer, GPS, and radio) mounted on a powered paraglider is shown in Figure 1. Although these instruments lack the functionality

\*Correspondence: [m.aydemir@hho.edu.tr](mailto:m.aydemir@hho.edu.tr)

of professional avionic systems, they provide some basic information for flying and are very popular among the ultralight aircraft community. However, discrete instruments are separately dedicated to flight information, navigation, and engine monitoring. These devices are powered with their own batteries and each has different operating instructions, which require a considerable time to learn.



**Figure 1.** Some flight instruments during flight.

This configuration is not good for pilots who are not familiar with these types of instruments. Therefore, bringing different instruments together is not a very efficient method because the pilot-vehicle interface is very important for flying the vehicle safely and effectively [4]. Today, even though smartphones with GPS are abundant, still a considerable number of pilots are lost during flight or experience controlled flight into terrain because they may not evaluate GPS information during flight, because it is almost impossible to use these devices in windy and vibrational environments.

The purpose of the study is to demonstrate that, with today's technologies, it is possible to construct an affordable and reliable flight avionic system that provides the fundamental information needed by the ultralight aircraft pilot.

In this study, a novel flight instrument for ultralight aircraft has been designed and implemented. In addition to the altitude, temperature, and time, this device also displays and stores rpm, engine hours, fuel capacity, and air/oxygen ratio information. This information has critical value to the healthy operation of the engine because monitoring the engine during the flight is vital; if the engine stops over a nonlandable area the results may be catastrophic. The instrument draws around 95 mA current, which is very low. When any malfunction is detected the user is warned immediately. The system also includes a datalogger, which has similar functionality of a CSFDR (crash survivable flight data recorder) that logs all this information on an SD memory card in order to provide postflight analysis data for training or accident investigation. With the help of the GPS sensor, the user always monitors the ground speed, which is very important for landing planning. Distance and bearing to the takeoff location is also always displayed as most accidents are due to being lost and not being able to find somewhere to land.

## 2. System overview

Avionic systems are composed of one or more of the communication, navigation, engine monitoring, CSFDR, and collision-avoidance systems. Light aviation aircraft mostly do not have autopilot or meteorological information (radar) systems as these systems are bulky and rarely needed in this form of aviation. Therefore, these functions are excluded from the flight instrument. The components of the system are shown in Figure 2. The heart of the system is two 8-bit microcontrollers [5]. The reason for using a low-end microcontroller is that no digital/analog signal conditioning is needed as the selected sensors provide filtered and calibrated data.

Therefore, the microcontrollers' role is to act as a timing and I/O unit by sending all the information to display, record data on the SD card, and inform the user about malfunctions or make warnings for low fuel, etc. The microcontrollers' tasks are shown in Table 1. The main reason for using two separate microcontrollers is to decrease the number of tasks per microcontroller. With this approach, even if one microcontroller fails, the second will keep functioning and provide at least half the information the user needs. Putting all the information on a single graphical LCD is possible, but it may be confusing for the user. This configuration allows the user to reach the information at a glance. This is a critical issue for flight safety. The system block diagram is given in Figure 2.

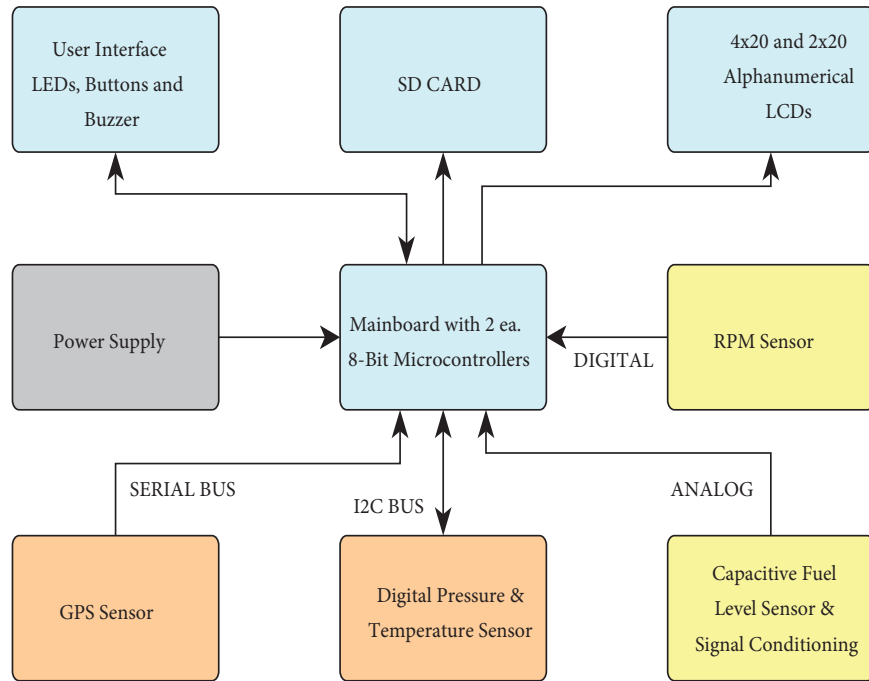


Figure 2. System block diagram.

Table 1. Tasks of microcontrollers.

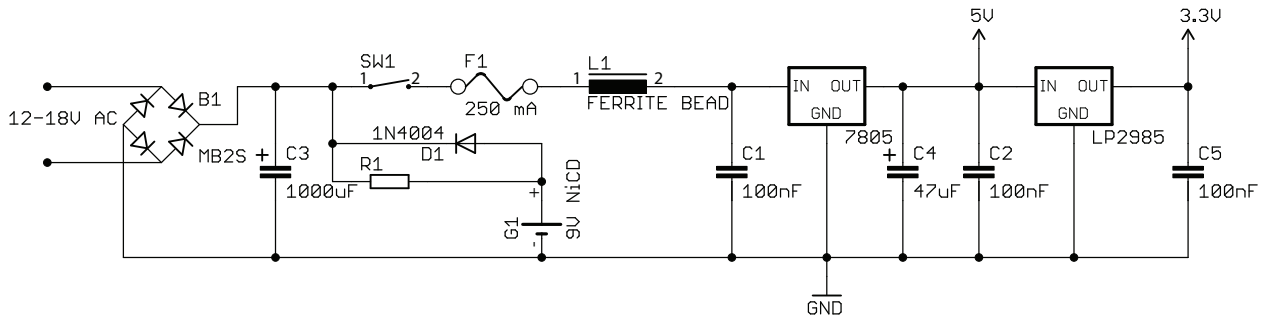
Microcontroller #1	Microcontroller #2
<ul style="list-style-type: none"> <li>• Altimeter</li> <li>• Variometer</li> <li>• Temperature</li> <li>• Engine rpm</li> <li>• Chronometer</li> <li>• Engine hours</li> <li>• Air to oxygen ratio</li> </ul>	<ul style="list-style-type: none"> <li>• Ground speed</li> <li>• Continuous heading</li> <li>• Distance information to takeoff</li> <li>• Real time clock and date</li> <li>• Latitude/longitude information</li> <li>• SD card read/write</li> </ul>

The microcontrollers operate on a real-time operating system, which is based on the internal timer interrupts. Instead of 32-bit microcontrollers, an 8-bit microcontroller has been utilized in the system due to

easier soldering and simpler instruction set. Building the electronic circuits for 28 pin DIP (dual inline package) microcontrollers is easy and simple circuits are generally more reliable than complex ones. It is also easier to construct or repair these types of integrated circuits w.r.t. the ones with land grid array or ball grid array pins.

**2.1. The power source**

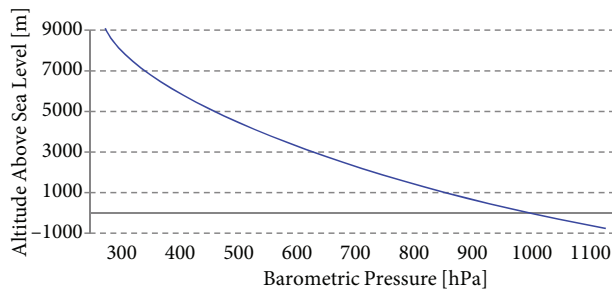
As most light aircraft engines have an alternator for powering the electronic ignition systems, this source may be utilized for powering the instrument as well. However, in order to save weight and reduce complexity, most units lack a battery, and therefore the engine is started by pulling a rope. A backup battery is included to provide power when the engine is stopped. The power system specific to the flight instrument is shown in Figure 3. Here, the L1 choke inductor with metal core is a transient suppressor and has big importance in noisy circuits. The value of R1 should be chosen such that the 9 V NiCd battery will always be trickle-charged. The D1 diode provides isolation of the 9 V battery while the alternator is working. The power source provides 5 and 3.3 V DC for the instrument.



**Figure 3.** The power source schematic.

**2.2. Altitude, descent/ascent, and temperature information**

As MEMS sensors have been merged with electronic self-calibration and compensation circuitry, low-cost altitude sensors having an accuracy of less than 1 m have been widely available. Therefore, the complex analog and digital calibration circuitry has been eliminated. The selected sensor is the recently announced BOSCH BMP180 pressure sensor [6]. It measures the atmospheric pressure with very high accuracy and sends through the I2C (inter-integrated circuit) bus, which is very popular in industry. The precision of the sensor output depends on the data sampling rate and an accuracy of 40 cm is achieved for 1 Hz measurements, which is fine enough for a flight monitoring system but not for autopilot. The plot for altitude vs. pressure values obtained is shown in Figure 4. The altitude may be calculated as [7]:



**Figure 4.** Altitude vs. pressure.

$$h = 44330 \left(1 - \left(\frac{p}{p_0}\right)^{5.255}\right) \tag{1}$$

where  $h$  is the altitude in meters,  $p$  is the pressure in hPa, and  $p_0$  is the pressure at sea level (101.325 hPa). Calculating Eq. (1) requires floating math. Today's 8-bit microcontrollers may handle this task easily due to their increased RAM, which is 2 KB for the ATMEL AVR 328 used in the device; however, taking into account many other duties, the floating-point math process spares considerable machine cycles. Instead, it is a better idea to calculate the altitude for some intervals and calculate the altitude for any pressure value by interpolation [7] (Figure 5). Here, the  $J_n$  and  $I_n$  coefficients are the first-order Taylor series coefficients for calculating Eq. (1) [8]. The interface between the pressure/temperature sensor and microcontroller #1 is shown in Figure 6. The 24-bit pressure information is sent through the I2C bus. The variometer data provide the rate of descent/ascent and this information may easily be acquired by differentiating 2 consequent altitude data at 1 Hz. The pressure sensor also has an internal temperature sensor for pressure compensation. The temperature information is also acquired from the pressure sensor.

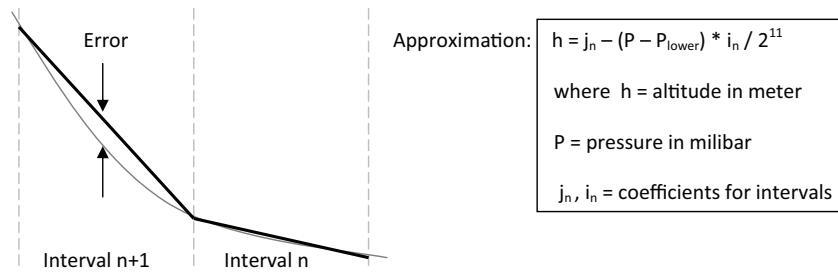


Figure 5. Altitude vs. pressure.

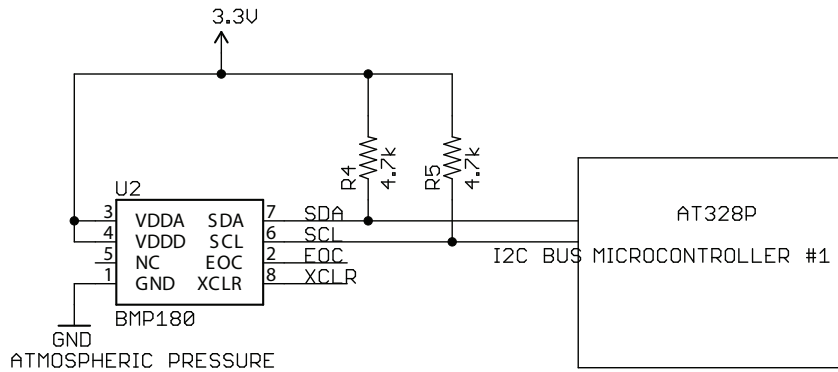


Figure 6. Microcontroller and pressure/temperature sensor interface.

### 2.3. The engine information system

The engine operation is controlled by monitoring the fuel tank gas level, engine rpm, and exhaust gas/oxygen ratio. In order to monitor the gas level, a parallel plate capacitor is submerged in the gas tank. The capacitance is defined as:

$$C = \epsilon_0 \epsilon_r \frac{A}{d} \tag{2}$$

where  $\epsilon_0$  is the dielectric permittivity of free space ( $8.85 \text{ times } 10^{12} \text{ F/m}$ ),  $\epsilon_r$  is the dielectric permittivity of media (2.2 for gas),  $A$  is the plate area ( $\text{m}^2$ ), and  $d$  is the distance (m) between plates. As the level of gas varies, so does the capacitance. The capacitor  $C$  is in pF levels, which is part of an oscillator circuit built with a 555 timer (Figure 7). Here the oscillation frequency is:

$$f = \frac{1.44}{(R_1 + 2.R_2).C} \tag{3}$$

This frequency is measured by an internal counter of the microcontroller. The counter value is fetched at 1 Hz, the value is multiplied by a constant specific to tank size, and gas level data are acquired.

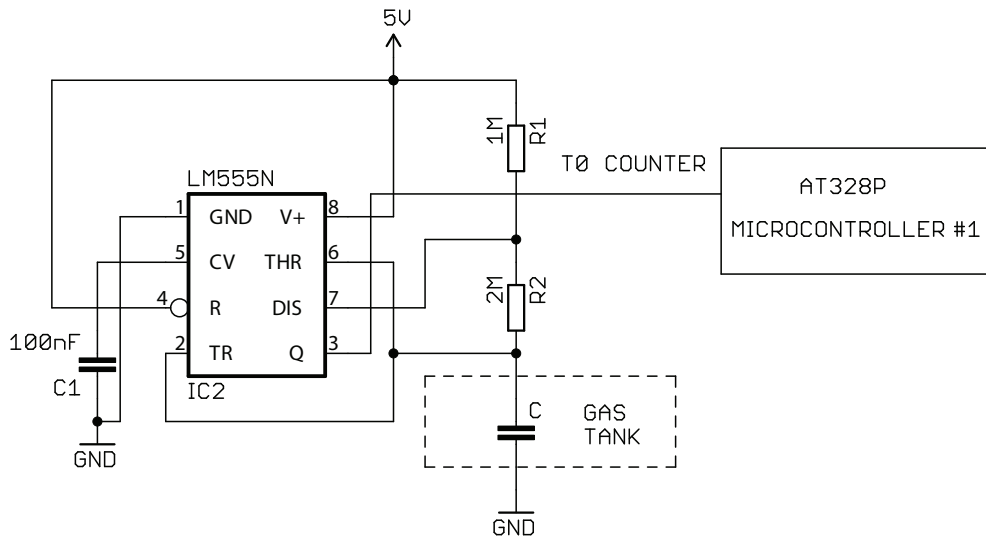


Figure 7. Microcontroller and gas level sensor interface.

On the other hand, the rpm information is directly received by inducing voltage from the L1 coil, which is wrapped a few turns around the spark plug high voltage cable (Figure 8). This signal is clipped, conditioned, and amplified by a simple BJT amplifier. Similar to the frequency measurement, this signal is fed to another counter in the microcontroller.

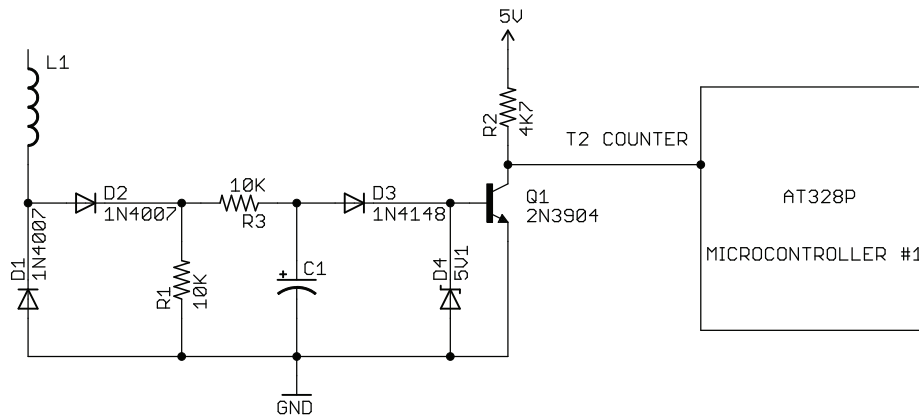
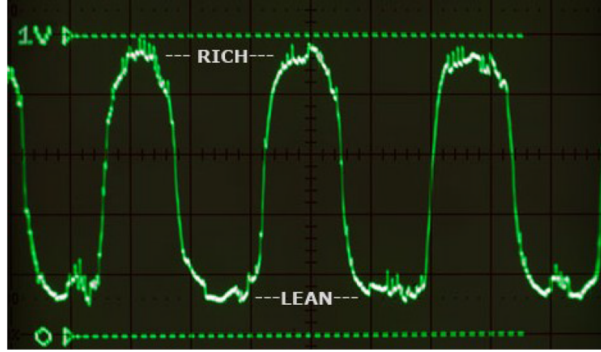


Figure 8. Microcontroller and rpm sensor interface.

In order to measure the exhaust gas oxygen level, a lambda sensor is used [9]. The lambda sensor is installed at the cylinder exhaust outlet. It provides information about the oxygen level in the exhaust gas. This information is important for engine tuning and efficiency. If the mixture is too rich in fuel, it will clog the spark and leave deposits on the cylinder; on the other hand, if the mixture is lean, the engine may seize. As the mixture varies depending on atmospheric pressure, the carburetor or injector has to be tuned before every flight manually. The lambda sensor helps the pilot tune the carburetor or injector for perfect mixture setting. The variation of the voltage at the output of the lambda sensor is given in Figure 9.



**Figure 9.** Voltage at the output of the lambda sensor.

The exhaust gas oxygen level is not constant. The reason is varying levels of oxygen during the inlet and exhaust cycles of the 2-stroke internal combustion engine. In order to measure the oxygen level, the varying voltage is converted to digital and 1000 samples are averaged at 1 Hz frequency while 0.5 V corresponds to an ideal air to fuel ratio of 14.7:1. For any other value, the carburetor mixture setting must be readjusted to reach the ideal mixture setting while the engine is running.

During flight, if high altitudes of more than 3000 m are gained, the partial atmospheric pressure of the oxygen drops, and therefore the mixture becomes rich again. The lambda sensor again may help the pilot adjust the carburetor setting for the best performance.

#### 2.4. GPS and navigation

The latitude, longitude, and real-time date and time data are received from the GPS sensor. This sensor is directly connected to a free digital pin of the second microcontroller. As the information is sent via serial port, a virtual serial port is established by software. The information coming from the GPS sensor is parsed into text [10]. At the end of the flight, most ultralight aircraft pilots land where they took off. To help them do that, the initial latitude and longitude is recorded at power on sequence and distance and bearing to takeoff coordinates is monitored continuously by the help of the cosine haversine formulation for the takeoff ( $la_1, lo_1$ ) and the existing ( $la_2, lo_2$ ) coordinates [11]:

$$\Delta lat = la_2 - la_1 \quad (4)$$

$$\Delta lon = lo_2 - lo_1 \quad (5)$$

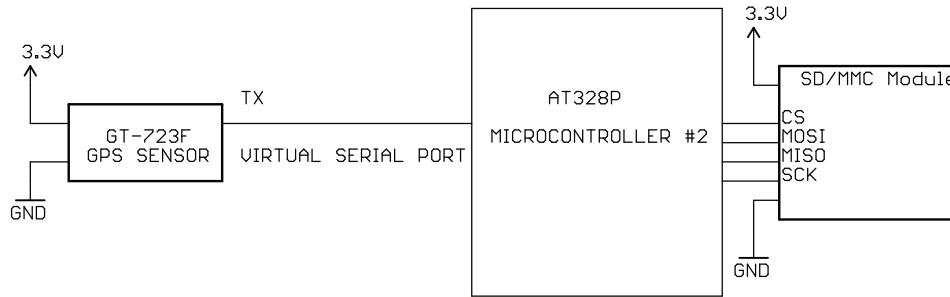
$$a = (\sin(\Delta lat/2))^2 + \cos(la_1) \cdot \cos(la_2) \cdot (\sin(\Delta lo/2))^2 \quad (6)$$

$$c = R \cdot 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a}) \quad (7)$$

$$\theta = \text{atan2}(\sin(\Delta lon) \cdot \cos(la_2), \cos(la_1) \cdot \sin(la_2) - \sin(la_1) \cdot \cos(la_2) \cdot \cos(\Delta lon)) \quad (8)$$

where  $\text{atan2}$  is the arctangent function with two arguments,  $R = 6371$  km is the radius of the earth,  $c$  is the distance to takeoff position, and  $\theta$  is the heading vector to takeoff position.

The navigation parameters are also recorded on an SD card. The connections between microcontroller #2 and the GPS sensor and SD card module are shown in Figure 10. As the hardware serial port of the microcontroller is reserved for testing, a virtual serial port is established on one of the free digital pins by the help of a special software library.



**Figure 10.** Microcontroller #2 and GPS sensor and SD card module interface.

### 3. The firmware

The firmware is written in the Arduino IDE (Integrated Development Environment) cross-compiler for Atmel AVR microcontrollers, which is open-source. This makes the programming much more easier compared to using the assembly language. Two lines of sample code are given in Listing 1. Here, the first 3 lines calculate heading vector to takeoff given by Eq. (8), while the fourth and fifth lines convert the result back to degrees and send it to the LCD. The cross-compiler vastly simplifies complex 32-bit floating math and LCD interface routines.

Listing 1:

```

1 temp1 = sin(d_lo)*cos(la2);
2 temp2 = cos(la1)*sin(la2)-sin(la1)*cos(la2)*cos(d_lo);
3 theta = atan2(temp1,temp2);
4 lcd_s.setCursor(7,1);
5 lcd_s.print(theta*180/3.14159,0);

```

During the power on, the battery level is checked and displayed. After a few seconds, microcontroller #1 starts measuring and displaying two altitudes (ASL: altitude above sea level, ATO: altitude above takeoff), vario (rate of ascent/descent), temperature, engine rpm, engine hours, and chronometer information at 1 Hz. The (+) and (-) buttons are used to correct the altitude offset error. Meanwhile, microcontroller #2 monitors the information from the GPS sensor and as soon as the number of satellites is greater than 3, the initial latitude and longitude ( $la_1$ ,  $lo_1$ ) values are stored. After that, groundspeed, heading and vector to takeoff coordinate, and real time and date information are monitored continuously (Figure 11).

The maintenance mode may be reached by the menu button. In this mode, the lambda sensor voltage and exact latitude and longitude information are displayed (Figure 12).

The firmware flowchart is given in Figure 13. The microcontroller resources (RAM, ROM, CPU speed, etc.) are listed in Table 2.





Figure 11. Engine (a) and GPS (b) LCDs during flight mode.

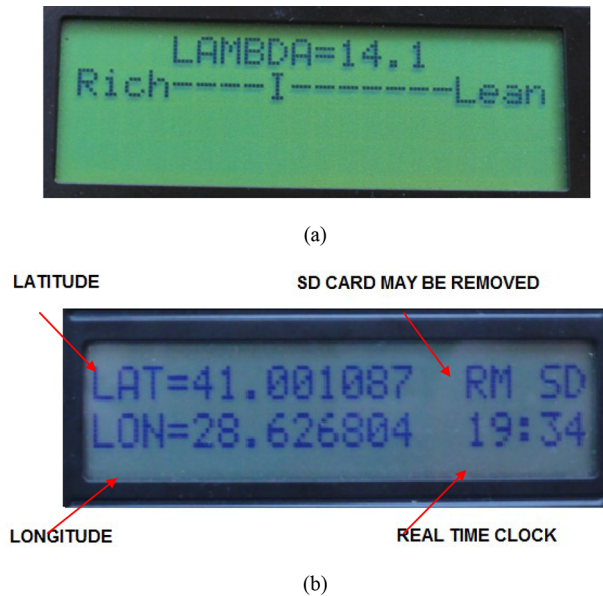
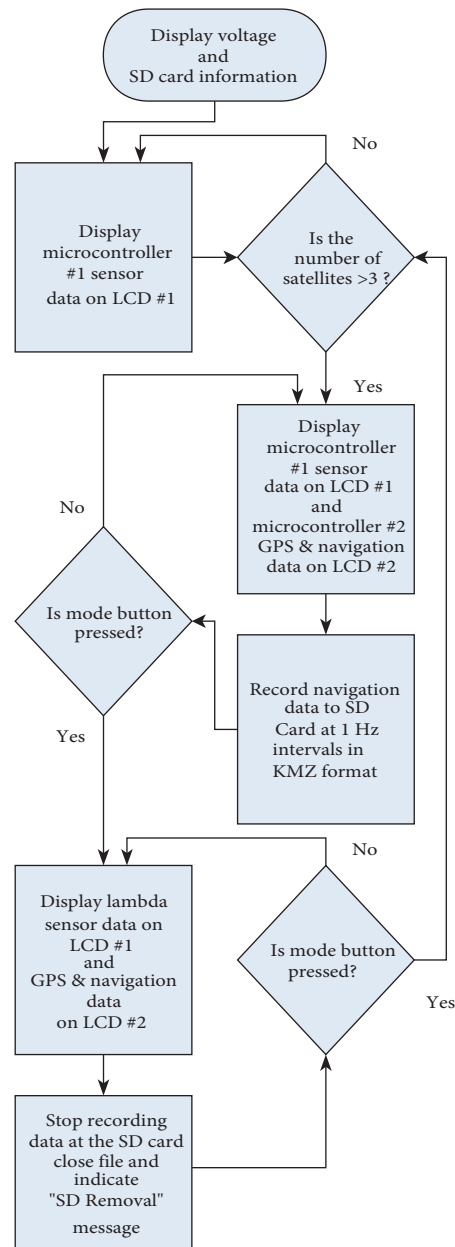


Figure 12. Engine (a) and GPS (b) LCDs during maintenance mode.

The system is assembled and put in a box (Figure 14). During the tests it was seen that the current drawn from the battery was 95 mA; therefore, the system may run for 21 h on a 2000 mAh NiCd battery. The whole system weighs only 250 g, which is quite light weight among other avionics systems.

**Table 2.** Microcontroller resource usage.

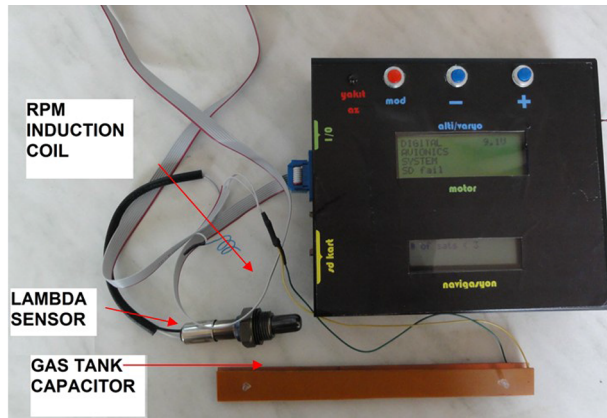
Resources	Microcontroller #1	Microcontroller #2
C source file size	3 KB	14 KB
ROM	10 KB	22 KB
RAM	2 KB	
CPU speed	16 MHz (single cycle RISC architecture)	
CPU usage duty cycle	50%–90% depending on the gas and rpm sensor interrupts	90%
Size of KML file on SD cart	-	130 KB/h at 1 Hz sampling rate



**Figure 13.** Firmware flowchart.

#### 4. System performance and cost efficiency

The system was installed on a paramotor (Figure 15) and some test flights were made. During the test period, the unit worked accurately. After the flight, the SD card was removed and flight data were visualized by manually inserting latitude, longitude, and altitude information to a KML (Keyhole Markup Language) file [12] for Google Earth (Figure 16). The 3-D navigation information is helpful for postflight evaluation.

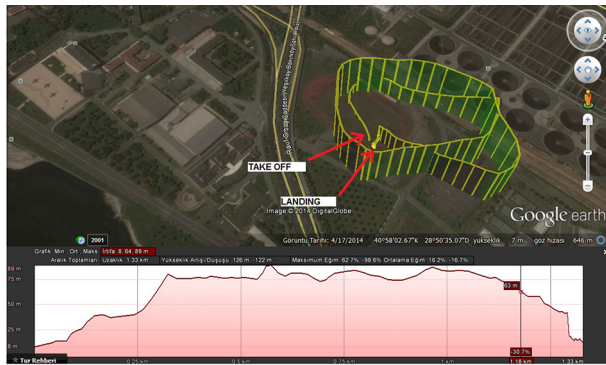


**Figure 14.** Flight information system hardware assembled.

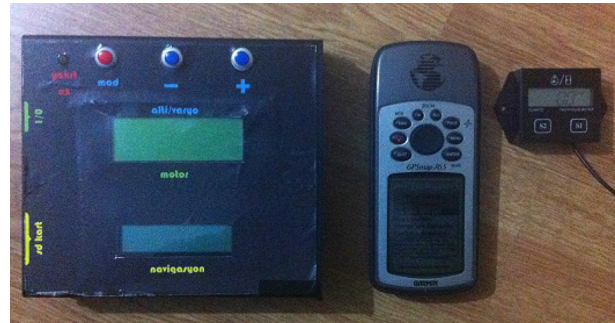


**Figure 15.** Unit installed on a paramotor.

The unit's accuracy was also compared with a rugged commercial GPS and engine monitor device (Figure 17). It was seen that the accuracy of the unit was very high.



**Figure 16.** Visualization of flight details from the SD card KML file.



**Figure 17.** Size comparison with commercial units.

#### 5. Conclusion

There is a trade-off between the cost and functionality of existing avionics in the market. The avionics system developed in this study has a very simple user interface; in fact, it has only 3 push buttons, and therefore anyone without prior operating instruction knowledge may use it with ease. The microcontroller-based, lightweight, low-energy flight instrument in this study presents a considerable alternative to the heavy, energy-consuming but certified existing avionics systems. The cost of the system is 1/9 of the lowest-cost alternative with similar functionality in the market. The device was tested on a paramotor and excellent results were achieved. The usage

of freeware open-source design software (Arduino IDE for code generation, Eagle CAD for PCB and schematics, and Google Earth for flight path visualization) makes the system very affordable, easy to customize, and reliable. The system is a good candidate for ultralight avionics and may also be used as a teaching tool for aerospace engineering.

### Acknowledgment

The author would like to thank the Marmara Paramotor Club for providing the paramotor in this study.

### References

- [1] Spinka O, Krakora J, Sojka M, Hanzalek Z. Low-cost avionics system for ultra-light aircrafts. In: IEEE 2006 Emerging Technologies and Factory Automation Conference; 20–22 September 2006; Prague, Czech Republic. pp. 102-109.
- [2] Paces P, Levora T, Bruna O, Popelka J, Mlejnek J. Integrated modular avionics onboard of small airplanes - Fiction or reality? In: IEEE/AIAA 30th Digital Avionics Systems Conference; 16–20 October 2011; Seattle, WA, USA. pp. 7A1-1-7A1-12.
- [3] Leslie A. Software Considerations in Airborne Systems and Equipment Certification. Malakoff, France: European Organization for Civil Aviation Equipment, 1992.
- [4] Harrison L, Castronuovo M, Janowitz J, Saraceni P. Pilot-vehicle interface. In: IEEE/AIAA 11th Digital Avionics Systems Conference; 5–8 October 1992; Seattle, WA, USA. pp. 296-300.
- [5] AT 328P Datasheet. San Jose, CA, USA: ATMEL Inc., 2014.
- [6] BOSCH BMP180 Sensor Datasheet. Stuttgart, Germany: BOSCH GmbH, 2014.
- [7] Application Note 501: Using MS5534 for altimeters and barometers. Hampton, VA, USA: Intersema Sensors Inc., 2012.
- [8] Aydemir M, Okatan A. An applied practical approach for determining altitude from an analog pressure sensor. In: IKS 2005 2nd International Conference on Intelligent Knowledge Systems; 5–7 May 2005; İstanbul, Turkey. pp. 173-175.
- [9] Bozon A, Koberstein E, Pletka H, Voelker H. Sensor for measuring the oxygen content in the exhaust gas of combustion engines and method thereof. US Patent US4362605-A, 1982.
- [10] Kennedy M. The Global Positioning System and GIS: An Introduction. London, UK: T&F Books, 2003.
- [11] Ismaeel A. Effective technique for allocating servers to support cloud using GPS and GIS. In: SAI 2013 Science and Information Conference; October 7–9 2013; London, UK. pp. 934-939.
- [12] Ballagh LM, Raup BH, Duerr RE, Khalsa SJS, Helm C, Fowler D, Gupte A. Representing scientific data sets in KML: methods and challenges. *Comput Geosci* 2011; 37: 57-64.