PROTOTYPING A LOW-COST EMBEDDED AIRSPEED SYSTEM

Razvan-Viorel MIHAI, Alexandru RADU, Ovidiu TABAN

Military Technical Academy, Bucharest, Romania (razvan.mihai@mta.ro)

DOI: 10.19062/2247-3173.2017.19.1.13

Abstract: *Velocity is one of the main parameters that must be known during the flight of an aircraft. In aeronautics a pitot-static system is used.. In this paper we present the implementation of a low cost embedded system design to measure in real-time the airspeed of the aircraft and a graphical user interface to provide the possibility to change density of the air and the speed of a DC motor used as a fan to produce a fluid flow to verify the performance of our system. In modern aircraft, an air data computer computes air speed based on the digital results obtained after extracting the static pressure from the total pressure. We use a two in one sensor that directly provides the difference of pressure, under the form of an analog signal.*

Keywords: *pitot tube, embedded system, low cost, Graphical User Interface*

1. INTRODUCTION

Airplane's speed is measured in knots or miles per hour and it is displayed inside the cockpit of the aircraft. The airspeed indicator is one of the basic aircraft instruments and is of importance to pilots because adherence to safe operating speeds is imperative.

Pilots need to know the speed during some critical phases of the flight, when taking off, landing, stalling.

The airspeed indicator works by subtracting the static pressure from the total pressure. This article presents the transition from conventional airspeed instruments to the computerized systems found on today's airplanes. Airspeed can also be obtained from a GPS unit, even though the data has significant errors that must be corrected. The best option is the differential GPS which requires a beacon fixed at some known coordinates, which provides corrections to the data that is obtained from the satellites. Unfortunately these systems are quite expensive for what we aimed to achieve in this research.

The airspeed indicator is part of the pitot-static system, used to provide the dynamic pressure. This parameter is used to measure the true airspeed obtained from an analogue device, using the following formula:

$$TAS = a_0 \sqrt{\frac{5T}{T_0}\left[\left(\frac{q_c}{P}+1\right)^{\frac{2}{7}}-1\right]} \tag{1}$$

q_c - impact pressure
P- static pressure
T- temperature
T_0 – temperature at the standard sea level
a_0 – speed of sound at the standard sea level

We chose to build an embedded system which is able to calculate and display the true airspeed at low speeds and altitudes, so we had to use the following formula for this case:

$$TAS = EAS\sqrt{\frac{\rho_0}{\rho}} \tag{2}$$

where:

ρ is the actual air density

ρ_0 is the standard sea level density (1.225 kg/m$^{3)}$

$$EAS = \sqrt{\frac{2q}{\rho_0}} \tag{3}$$

$$q = \frac{1}{2}\rho v^2 \tag{4}$$

where:

q is the dynamic pressure

v is the flow speed

Youngshin Kang et. all propose the Kalman filter setup to calibrate static pressure, or airspeed and barometric altitude using GPS ground velocity [1]. And also Kalman filter based AOS calibration method is proposed using flight data. Proposed methods are verified by simulation and real flight data of Smart UAV.

The next paper [2] claims that the coefficient α must be calibrated to correct the measurement result. The authors suggest to use a Doppler laser anemometer to calibrate Pitot-tube to acquire the coefficient α.

2. HARDWARE DESIGN

We tried to design a simple board on which we should be able to develop new features in a modular fashion. Figure 1 illustrates the resulted board:

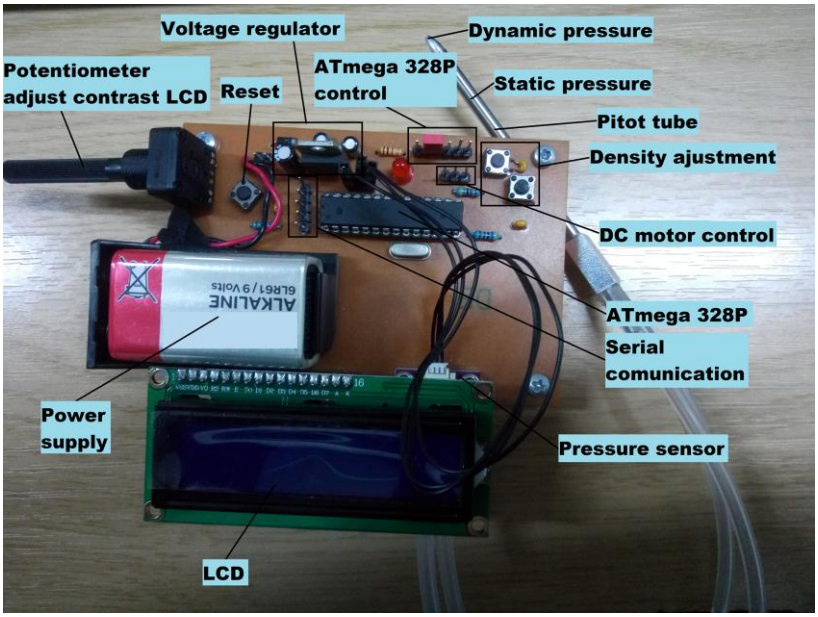

FIG. 1

In terms of data display, the aviation industry is moving from cathode ray tubes to the lighter and more reliable LCD (Liquid Crystal Displays). CRTs require high voltages–up to 50,000 volts–and generate a lot of heat, while LCDs, like large chips, can use power in the 5-volt range. The advantages of the LED (Light Emitting Diodes) over the incandescent lamps are longer life, lower voltage, faster on and off operations, and less heat. The LCD has an advantage over the LED in that it requires less power to operate. Where LEDs commonly operate in the milli-watt range, the LCD operates in the microwatt range. In order to keep a low power consumption we chose to use a 32 characters on two lines LCD, because we decided to display the density of the air and the true air speed.

The core component responsible with data conversion, processing and communication is a CMOS (Complementary Metal-Oxide Semiconductor) 8-bit AVR Atmel chip, respectively ATmega328p.This microcontroller is integrated in the well known Arduino Uno board which has a lot of features that we need to particularly configure them in our application. Thus we designed a prototype board aiming to minimize costs, comparing with Arduino Uno.

Because of the great variety of conditions, ranges, and materials for which pressure must be measured, there are many different types of pressure sensor designs. Often pressure can be converted to some intermediate form, such as displacement. The sensor then converts this displacement into an electrical output such as voltage or current. The three most universal types of pressure transducers of this form are the strain gage, variable capacitance, and piezoelectric.

Piezoelectric pressure transducer take advantage of the electrical properties of naturally occurring crystals such as quartz. These crystals generate an electrical charge when they are strained. We chose to use the differential pressure sensor – CJMCU-36, which is based on the component MPX5010 presented in Fig. 2. The MPxx5010 series piezo resistive transducers are state-of-the-art monolithic silicon pressure sensors designed for a wide range of applications, but particularly those employing a microcontroller or microprocessor with A/D inputs.

FIG. 2

The nominal transfer value is hardware calibrated under the following formula:

$$V_{out} = V_s * (0.09 * P + 0.04) \pm (Pressure\ Error * TempFactor * 0.09 * V_s) \quad (5)$$

V_s - Supply Voltage

V_{out} - Output Voltage

We used this sensor because it directly outputs the dynamic pressure analogue quantity, therefore helping us to avoid the usage of two analogue to digital channels and also decreasing the probable error.

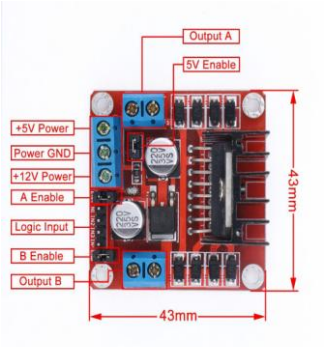

FIG. 3

One simple and easy way to control the speed of a motor is to regulate the amount of voltage across its terminals and this can be achieved using "Pulse Width Modulation" or PWM. As its name suggests, pulse width modulation speed control works by varying the duty cycle, of the pulses while keeping the frequency constant.

The power applied to the motor can be controlled by varying the width of these applied pulses and thereby varying the average DC voltage applied to the motors terminals. To be able to control the speed of the motor we used a motor driver – L298N, as depicted in "Fig.3".

" Figure 4" and "Figure 5" show the implementation of a Graphical User Interface in python to be able to check all USB ports from PC and recognize our board if is connected, on which port and what baud rate, in order to provide the possibility to change air density and to control the speed of the motor through serial communication with the microcontroller ATmega328p which will give a modifiable PWM signal to the motor driver.

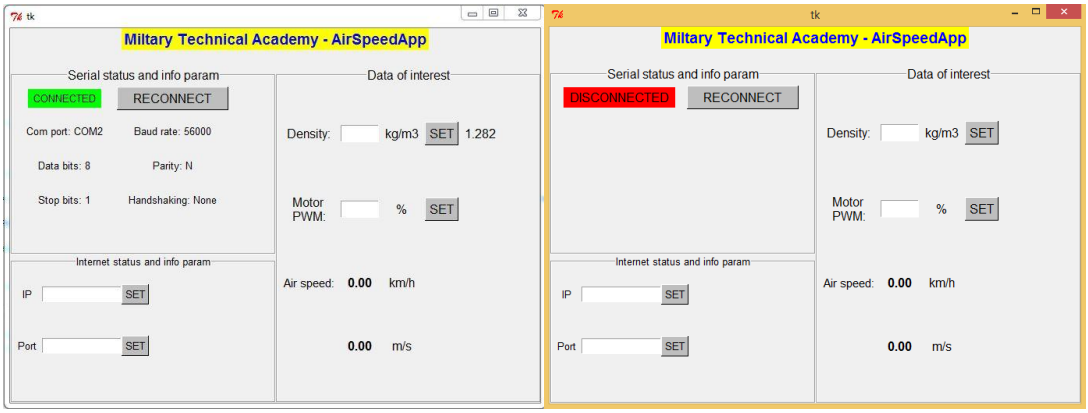

FIG. 4 **FIG. 5**

Schematic capture or schematic entry is a step in the design cycle of electronic design automation (EDA) at which the electronic diagram, or electronic schematic of the designed electronic circuit is created by a designer.

The most known software programs for PCB design are OrCAD, EAGLE, Altium. Taking into account that we don't have a big budget or a large project, we chose to use EAGLE, for his friendly environment and low cost price, to design the printed circuit board which is presented in "Fig. 6".

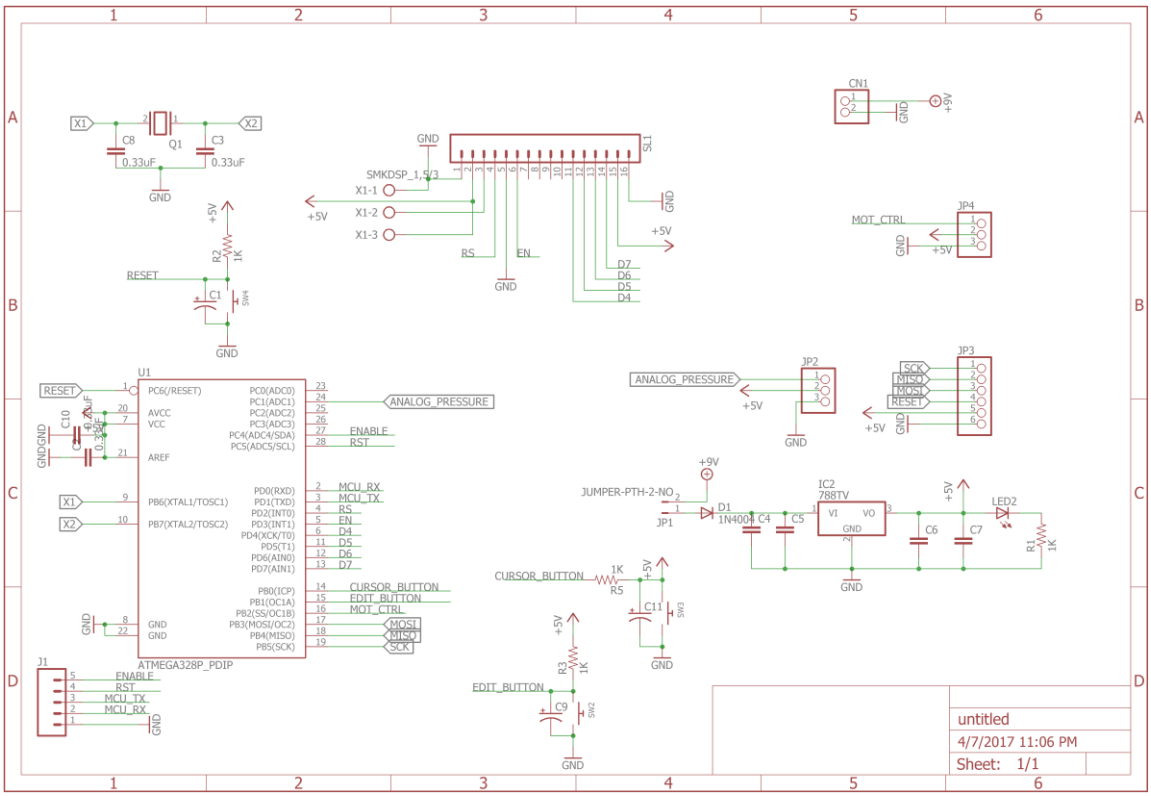

FIG. 6

3. SOFTWARE IMPLEMENTATION

The software we chose to work is Atmel Studio 7. Atmel Studio is an integrated development platform for developing and debugging all AVR microcontroller applications. The Atmel Studio integrated development platform gives you an easy-to-use environment to write, build and debug your applications written in C/C++ or assembly code. Studio 7 can also seamlessly import your Arduino sketches as C++ projects.

Another software we could chose was Arduino in which a program may be written in any programming language for a compiler that produces binary machine code for the target processor. The biggest lack of this software is that you cannot debug your program and it is really hard to determinate your flaws. In addition to that it offers a slower response time to register access.

The main loop in the software implementation:

```
while(1)
    {
            switch(StateMachine)
            {
                    case CALIBRATION:
                            //dis_data('C');
                            start_calibration();
                            //dis_data('a');
                    break;
                    case PRIMARY_SCREEN:
                            //dis_data('B');
                            primary_screen();
                            ADC_START_CONV;
                            StateMachine = FREE_RUN;
                    break;
```

```
                case FREE_RUN:
                        get_adc_chunck();
                        get_msg();
                break;
                case EDIT_DENS:
                        incrVal = '0';
                        dens = 0.0;
                        start_to_edit_screen();
                        //dis_cmd();
                        StateMachine = INSERT_VAL;
                break;
                case INSERT_VAL:
                        if (flag_digit_incr)
                        {
                                if (incrVal > '9')          incrVal = '0';
                                user_input_screen(incrVal++);
                                flag_digit_incr = FALSE;
                        }
                break;
        }
}
```

The program that we developed consists in a main loop in which it is executed one of the 5 states of this software. Those 5 states consist in: calibration, primary screen, free run, edit density and insert value.

The first state is calibration in which the message that will be displayed on LCD is made intelligible by calibrating the data received from ADC connected to the pitot tube using a mediation of 1000 values. This calibration method is not the best because inserts critical errors when the velocity of the flow is quick changed.

This method of calibration can be improved by reducing the values that are mediated, but reducing these values can change the way in which the data are displayed, making them appear quicker on the LCD resulting in an hazy display.

The second state primary screen represent the communication of the microcontroller with the LCD. With the help of this state the data received from ADC and compiled by the microcontroller is displayed on LCD, but first is displayed an introduction message. Aside from displaying velocity it also display the density value which can be changed.

The free run state refers to the nominal state of our application, during which we simply get data from ADC and display it on the LCD, but also in this state we expect a message from GUI to change the PWM of the DC motor control resulting in the change of the velocity flow through the Pitot tube.

The edit density state is called when the user press the edit density button and change the message displayed on the LCD to resemble a new density to be introduced. Because density varies with altitude it is essential to be able to insert updated values. The edit state can also be accessed through GUI.

Lastly the insert value refers to the moment when the edit density is pressed and is expected to be introduced the new value. This state help you to change the value of the density displayed on LCD.

Our code use the following formula to calculate the airspeed which depends on the two pressures that the sensor is measuring and density.

$$V = \sqrt{\frac{2P}{\rho}} \tag{6}$$

4. RESULTS AND CONCLUSION

In this paper we presented the design of a low-cost embedded system design to measure in real-time the airspeed of the aircraft which was tested in a wind tunnel at Clinceni presented in "Fig. 7" and "Fig. 8". After we analyzed the results we discovered that our module have small errors introduced by the air density which was not measured very accurate with our equipment.

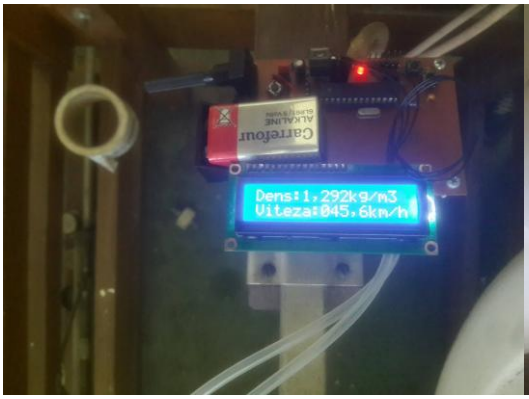

FIG. 6

FIG. 7

When the wind tunnel reached a flow velocity of 60-70 km/h, the tested system crashed and started to show false data, "Fig. 9". One major reason for this malfunction could be the usage of an simplified formula (5) which does not take into account all the parameters needed for high flow velocities (1). In order to solve this issue we will take in consideration the static pressure in the future.

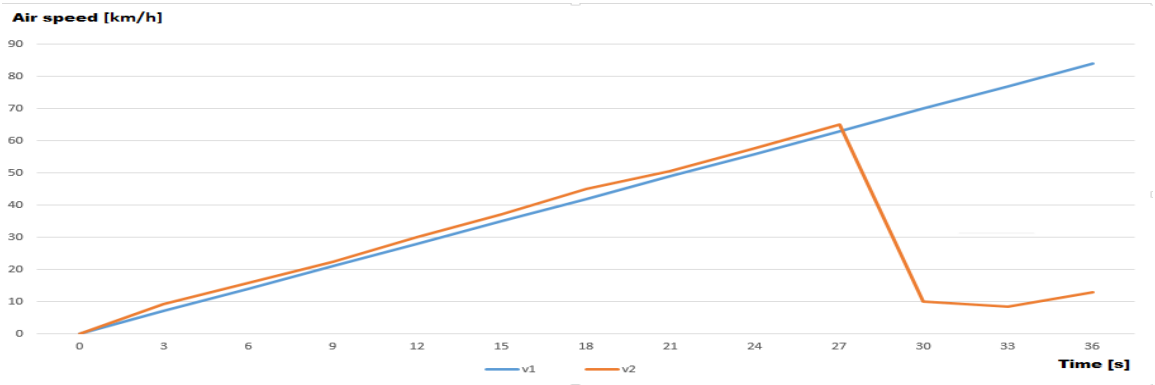

FIG. 8

The system is easy to use thanks to the graphical user interface which is design to accept new features. The future work assumes to equip our board with an wireless module and internet connection in order to be able to update in real-time and autonomous the density of the air and temperature depending on the altitude.

REFERENCES

[1] *Air data system calibration using GPS velocity information*, written by: Youngshin Kang, Sam-Ok Koo, Bum-Jin Park.
[2] *Experimental investigation to calibrate pitot-tube by laser doppler anemometer*, written by: L.-S. Cui, H.-M. Hu, C.-H. Li
[3] https://www.thebalance.com/basic-flight-instruments-the-airspeed-indicator-282607
[4] https://www.engineersgarage.com/articles/pressure-sensors-types-working
[5] http://www.experimentalaircraft.info/flight-planning/aircraft-navigation-speed.php