ISSN 2347 - 3983

Volume 8. No. 8, August 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter10482020.pdf
https://doi.org/10.30534/ijeter/2020/104882020

Creation of a Rotor-Type UAV with Flight Controllers, Based On a ATmega2560 and STM32f405 Microprocessors

Serhii Lienkov[1], AlexanderMyasischev[2], Larysa Komarova[3], Nataliia Lytvynenko[4], Viktor Shvab[5], Olexander Lytvynenko[6]

[1]Research Center, Military Institute of Taras Shevchenko National University of Kyiv, Kyiv, Ukraine, lenkov_s@ukr.net
[2]Department of Telecommunications and Radio Engineering, Khmelnitsky National University, Khmelnitsky, Ukraine, alex56ma@gmail.com
[3]Odessa National Academy of Telecommunications, Kyiv, Ukraine, lacosta_k@ukr.net
[4]Research Center, Military Institute of Taras Shevchenko National University of Kyiv, Kyiv, Ukraine, n123n@ukr.net
[5]Research Center, Military Institute of Taras Shevchenko National University of Kyiv, Kyiv, Ukraine, shvab_v_k@ukr.net
[6]Research Center, Military Institute of Taras Shevchenko National University of Kyiv, Kyiv, Ukraine, s63010566s@gmail.com

ABSTRACT

In this article, the budgetary unmanned aerial vehicles (UAVs) of the rotary type (the hexacopter, the quadcopter) based on flight controllers with ATmega2560, STM32F405 microcontrollers have been developed. A comparison is made of their capabilities, using the most popular firmware developed for them. The firmware was adjusted depending on the sensors, used in the flight controllers. The designed quadcopters, hexacopters are capable of performing the following flight modes. The holding horizontal - uses the gyroscope and the accelerometer. The maintaining a given altitude - uses a barometric sensor. The holding position - GPS receiver is additionally used. The returning to take-off mode and programmed flight along waypoints marked on the map before the UAV launch. A magnetometer is additionally used here. It has been experimentally established, that the hexacopter is more stable in thevflight compared to the quadcopter based on MegapirateNG ver.3.1.5R2 and ArduCopter 3.2.1 firmware. It has been shown, that for large quadcopters with frame sizes larger, than 450mm, INAV firmware is less preferable, than Ardupilot firmware. The built copters can be used for photo and video shooting of terrain with the radius of up to 3.5 km with lithium-ion cells with a capacity of 6000 mAh. They have the flight duration of 15-20 minutes and the average speed of 20 km/h.

Key words: OMNIBUSF4V3, INAV, GPS receiver, STM32F405, GLONASS, Ardupilot, MegapirateNG, ATmega2560, ArduCopter, MPU6050, MS5611 / BMP180.

1. INTRODUCTION

The great interest are unmanned the flying robots [1], in particular of the rotary type (the quadrocopters, the hexacopters, the orthocopters), intended for terrain exploration, photo and video filming, and rescue operations. The advantages have the hexacopters, the orthocopters in comparison with quadrocopters due to stability in flight, greater lift and reliability [2]. For example, the failure of one motor allows the ortho and hexacopter to continue their route. In this case, the quadcopter falls. However, the quadcopters are cheaper, and with further improvements in flight controllers and software, they are becoming more widespread. Therefore, in this article, the attention is paid to the creation and experimental study of the hexacopter and the quadrocopter built on different flight controllers and firmware. What unites them is that these unmanned aerial vehicles (UAVs) must be assembled from fairly common budget components, use free software products open to correction and support the following flight modes [2, 3]:

1. Holding the horizon. The gyroscope and the accelerometer are used.
2. Maintaining the given height. The barometric sensor is required.
3. Holding the position. The GPS receiver is being used.
4. The mode of returning to the starting point on command from the control panel, in case of loss of communication with the control equipment or battery discharge. The magnetometer is additionally required here.
5. The flight mode along the trajectory specified on the map. All of the sensors listed above are used.

The video transmitter, installed on the hexacopter, allows to perform FPV flights [4].

2. METHODOLOGY

For the stable UAV flight, performing the listed flight modes, the flight controllers with software are used. In addition to the microcontroller, the gyroscope, accelerometer, barometer, magnetometer, and GPS receiver are installed on it [4-6]. The most common flight controllers are MultiWii, APM 2.x, Pixhawk, SPRacingF3, OMNIBUSF4V3, SPRacing F7 and others [6-9], that can be used open for firmware patching Ardupilot [2], cleanflight, INAV [9, 10], betafight. These firmwares are the software for microcontrollers, that use such mathematical models, as the PID regulators [11], the Kalman filter, the complementary filter, the dynamic Notch filter, etc. For ensuring stable UAV flight, performing specified flight modes, the firmware is adjusted selection of parameters depending on the geometry of the copter, the installed propulsion system, sensors, speed parameters and also partially the program code [12]. The aim of the article is to create a budget (no more, than $ 120- $ 150) rotary UAVs and an experimental study of their capabilities, based on ATmega2560 and STM32F405 microcontrollers and the firmware of the Ardupilot family [2], the cleanflight (INAV) [10]. The method for solving the problem is the design and improvement of the flight controllers for copters, as well as the adjustment of freely distributed firmware to perform the above flight modes as a result of numerous flight tests.

The solution to this problem consists of three parts. The first part is the construction of the hexacopter and the quadrocopter based on the flight controller from the Mega2560 Pro board (the ATmega2560 microcontroller). The second part is using the APM 2.6 / 2.8 controller to create the drone. The third part is setting up the flight controller based on the STM32F405 microcontroller. The first and second parts are based on the firmware of the Ardupilot family, the third one is based on the firmware of the cleanflight (INAV) family.

Let's consider the construction of the hexacopter [4] based on the Mega2560 Pro board, that is a smaller version of the well-known Arduino Mega2560 R3 controller with similar characteristics, performance and all Arduino Mega ports (Fig. 1).

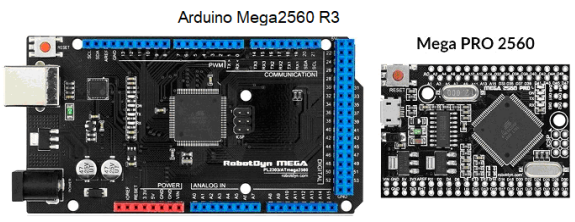

Figure. 1: Comparison of dimensions of Arduino Mega2560 R3 and Mega2560 Pro

For assembling a full-fledged flight controller, in addition to the Mega2560 Pro, the following boards are connected: GY-521 – the 6-axis gyroscope-accelerometer MPU6050, GY-273 – the 3-axis compass HMC5883L, GY-63/GY-68 – the barometer/altimeter MS5611/BMP180, module GY-NEO6MV2 – the GPS receiver u-blox NEO-6M (Fig. 2). The GY-521, GY-63/GY-68, GY-273 sensors are connected to the Mega2560 Pro via the I2C bus, the GPS receiver - GY-NEO6MV2 to the second serial interface RX2TX2. Instead of the MS5611 sensor, a cheaper and less accurate BMP180 barometric sensor (GY-68 board) can be used. The HMC5883L sensor must be installed further 25 cm from the motors and 15 cm above the plane of the motors. It was found, that such an arrangement deviates the compass readings at maximum engine speed by no more, than 2.0-2.5 degrees. When installed in the plane of the motors, the deviation of the compass will be more than 25 degrees.

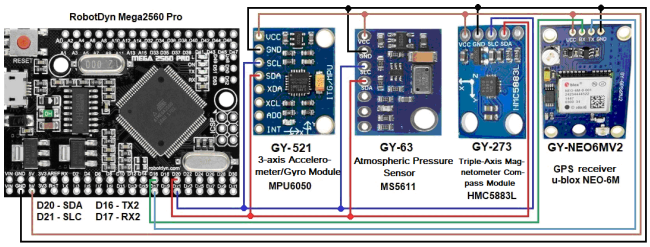

Figure 2: Connecting sensors to the microcontroller

The UAV is controlled by using the FlySky FS-I6 six-channel equipment [13] and the FS IA10 ten-channel receiver. To expand the control capabilities, the FlySky FS-I6 equipment was reprogrammed for 10 channels [13]. For example, it allowed to combine two switches to set six modes of operation. Figure 3 shows the FS IA10 receiver and its connection to the MEGA2560 PRO.

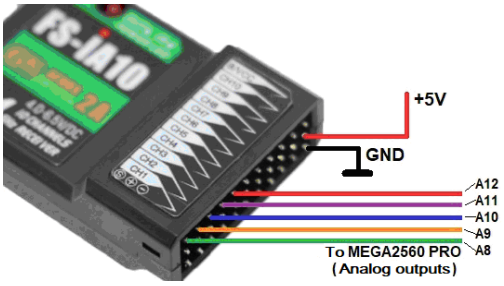

Figure 3: Connecting channels of the Fs Ia10 receiver to analog outputs of the Mega2560 Pro

The motors are connected to the flight controller via ESC controls. They are also used to power the flight controller and receiver. In the Fig. 4 the connection of the sixth motor to the flight controller via ESC are shown. The other five motors are connected in the same way.

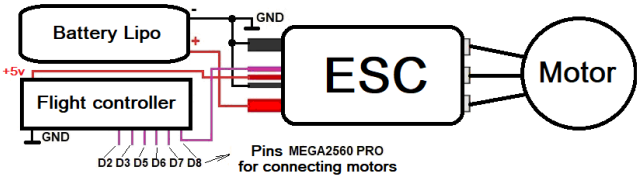

Figure 4: Connecting the sixth motor to the flight controller

The motors M1, M2, M3, M4, M5, M6 are connected to the terminals D2, D3, D5, D6, D7, D8, respectively. The hexacopter is equipped with six budget motors of the A2212/13T 1000V model, each of them with 10 "x4.5" propellers develops a thrust of up to 0.7 kg. The total maximum thrust is 4.2 kg. The location of the motors on the hexacopter, their numbering and direction of rotation, and a photo of the assembled flight controller are shown in Fig. 5.

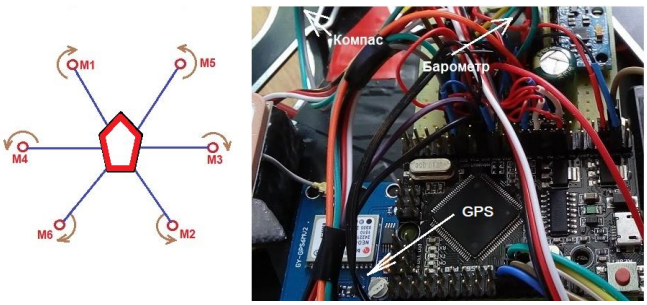

Figure 5: Numbering of motors on the hexacopter and photo of the flight controller

For the hexacopter's function, the flight controller must be flashed. It uses MegapirateNG firmware ver.3.1.5R2. It is configured depending on the sensors and aircraft frame used. The generation was performed on the computer with Windows 10. The following sequence of steps is used. A folder is created on the D:\drive to host the firmware, for example, 2018_megapirate. The firmware is copied from the address: https://github.com/MegaPirateNG/ardupilot-mpng/tree/mpng-3.1-beta to the created directory. After unzipping, the ardupilot-mpng-mpng-3.1-beta directory is automatically created, that contains the text of the programs for compilation in the Arduino IDE ver. 1.0.3. The modified Arduino IDE (file ArduPilot-Arduino-1.0.3-gcc-4.8.2-windows.zip) is copied from the address: http://firmware.ardupilot.org/Tools/Arduino/under the arducopter. This file is unpacked in the folder for firmware. Since this modified Arduino IDE is patched, except for the arducopter, nothing, but the firmware, will not be compiled. MHV_AVR_Tools_20131101.exe is copied from the address: http://firmware.ardupilot.org/Tools/Arduino/ - a set of tools used by Arduino IDE ver. 1.0.3 to build and install the firmware. There are the updated GCC compiler and the avrdude programmer here. The file runs like a normal program. The pde.jar file is copied to the ArduPilot-Arduino-1.0.3-windows \ lib directory from the root of the folder with the text of the arducopter programs ardupilot-mpng-mpng-3.1-beta. Next, the Arduino IDE is launched from the ArduPilot-Arduino-1.0.3-windows folder. The target platform is installed – the compilation for the MegaPirateNG firmware. For copy the firmware to the microcontroller without errors, the avrdude.conf file is copied from the folder with AVRtools installed in Program files, for example, from C:\Program Files (x86)\MHV AVR Tools\bin to the folder:
C:\2018_megapirate\ArduPilot-Arduino-1.0.3-gcc-4.8.2-windows\-Arduino-1.0.3-windows \ hardware\ tools\avr\etc.
The Arduino IDE is launched from C:\2018_megapirate\ArduPilot-Arduino-1.0.3-gcc-4.8.2-windows\ArduPilot -Arduino-1.0.3-windows folder. A sketch of an arducopter is selected: File->Folder with sketches->ArduCopter. The controller type is checked in the Arduino IDE: Service->Board->Arduino Mega 2560 or Mega ADK and the port to that the flight controller is connected.
Next, the firmware is configured. The APM_Config.h file is edited. The board is selected:
#define MPNG_BOARD_TYPE CRIUS_V1.
The parameters of the GPS device are entered:
//GPS port speed (Serial2) 38400 by default
#define SERIAL2_BAUD 38400
//GPS driver selection
#define GPS_PROTOCOL GPS_PROTOCOL_AUTO.
If there is no GPS, the GPS_PROTOCOL_NONE is set. In this case, the hexacopter will not return to the starting point and will not fly over the points in automatic mode.
The frame of the copter is configured is hexacopter:
#define FRAME_CONFIG HEXA_FRAME.
Some operating modes of the firmware are prohibited:
#define CLI_ENABLED DISABLED//disable the CLI (command-line-interface) to save 21K of flash space
#define LOGGING_ENABLED DISABLED//disable dataflash logging to save 11K of flash space
#define OPTFLOW DISABLED//disable optical flow sensor to save 5K of flash space
#define CONFIG_SONAR DISABLED//disable sonar to save 1k of flash.
The above changes are valid, when using the relatively expensive and high-precision MS5611 barometer. If the BMP180 barometer is used, it is necessary to make changes additionally in the config.h file by commenting out two lines:
#if MPNG_BOARD_TYPE == HK_RED_MULTIWII_PRO || MPNG_BOARD_TYPE == BLACK_VORTEX
define CONFIG_IMU_TYPE CONFIG_IMU_ITG3200
define CONFIG_BARO AP_BARO_BMP085
#elif MPNG_BOARD_TYPE == PARIS_V5_OSD
define CONFIG_IMU_TYPE CONFIG_IMU_ITG3200
define CONFIG_BARO AP_BARO_MS5611
define CONFIG_MS5611_SERIAL
AP_BARO_MS5611_I2C

#else
define CONFIG_IMU_TYPE
CONFIG_IMU_MPU6000_I2C
// # define CONFIG_BARO AP_BARO_MS5611
// # define CONFIG_MS5611_SERIAL
AP_BARO_MS5611_I2C
#endif.

Next, the control channels for the receiver are configured from the control panel. For this, the ardupilot-mpng-mpng-3.1-beta\libraries\AP_HAL_MPNG\RCInput_MPNG.cpp file is used. The editing of the ardupilot-mpng-mpng-3.1-beta\libraries\AP_HAL_MPNG\RCInput_MPNG.cpp file is done from a third-party editor (eg NotePad ++). If the receiver outputs a PPM signal, no editing is done. If the receiver emits a PWM signal, as in our case, then the line of the RCInput_MPNG.cpp file changes:
#define SERIAL_PPM SERIAL_PPM_ENABLED
on the
#define SERIAL_PPM SERIAL_PPM_DISABLED.

After the above changes, the firmware is compiled in the Arduino IDE ver. 1.0.3 with the subsequent loading of its microcontroller.

Further configuration of the flight controller firmware is performed, using the Mission Planner software, for example, version 1.3.15 [14]. The calibration of the accelerometer, compass, radio equipment is done [2]. The Failsafe [2] must be configured, otherwise, if communication with the control equipment is lost, the copter flies away in accordance with the remaining parameters of the radio receiver. The PID parameters of the controller are corrected as described in [2]. In Fig. 6 the route of the hexacopter's flight in automatic mode along waypoints, generated in the Mission Planer is shown.

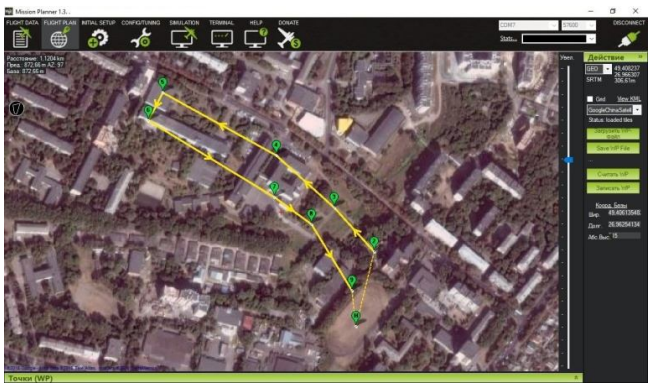

Figure 6: Flight route of the hexacopter in the automatic mode of the mission planer program

In Fig. 7 a photo of an assembled hexacopter with an F550 frame and the quadrocopter with an F450 frame according to the diagram above are shown. However, the standard Arduino Mega2560 R3 controller and the BMP180 barometric sensor are used as the flight controller for the quadcopter. The reference [15] presents the quadrocopter's tests.

Figure 7: Photo of the hexacopter and quadrocopter with an ATmega2560 Based Flight Controller

The flight tests showed, that for the controllers based on the Arduino mega2560 and the considered modified MegapirateNG firmware, the insufficient landing accuracy was observed at the launch point (the error up to 2m), and some "floating" altitude hold (up to 0.5m). In the next experiment, the APM 2.6 flight controller with ArduCopter 3.2.1 firmware was used. The reviewed copters were also equipped with a Firefly q6 video camera, a TS832 video transmitter (for FPV flight [6]) and telemetry (the transceivers, that are connected to the flight controller and computer). Using telemetry and the Mission Planer program, the copter movement overlaid on the map was displayed on the computer within a radius of up to 1 km, and the main telemetry data - flight altitude, battery voltage, satellites, etc. In the Fig.8 the hexacopter with the APM 2.6 flight controller and the flight path telemetry data displayed in the Mission Planner online during flight are shown.

Figure 8: The hexacopter with APM 2.6 flight controllers and its flight by telemetry

The source [16] presents a video, demonstrating the automatic flight of the hexacopter. The tests have shown, that in this case, fixing the position of the copter during flight and landing is more accurate, than for the Arduino-based controller and the MegapirateNG firmware.

Let's consider the case, when the STM32F405 microcontroller is used as a flight controller. The difference from the ATmega2560 is that the STM32F405 family of microcontrollers operates at 168 MHz, has 1 MB of the program memory, 192 KB static memory and an integrated floating point coprocessor. The ATmega2560 has parameters: 16 MHz, 256 KB, 8 KB, no coprocessor. Therefore, the flight controller, based on STM32F405, should be much faster and provide the high-quality, stable flight of the copter. The weak link here may be sensors, that aren't able to work at the speeds

of the microcontroller and the firmware isn't of high quality. The following components were used to build the experienced quadcopter:

1. The frame size 450 mm.
2. The motors A2212 / 1000 with ESC regulators for 30A.
3. The propellers 10x45 inches.
4. The flight controller OMNIBUSF4V3 [17] based on STM32F405 [9] LQFP64 microcontroller (168Mhz, 1M Flash, 192kB SRAM) with built-in gyroscope, MPU6000 accelerometer and BMP280 barometer.
5. The compass HMC5883L connected to the I2C bus.
6. The GPS receiver GY-NEO6MV2.

In Figure 9 the connection to the OMNIBUSF4V3 of the compass, the GPS receiver, the PPM control receiver is shown. The compass should be above the plane of the propellers' rotation at the height of at least 15 cm to reduce interference with the motors. For this controller, it's possible to connect a radio control receiver via the SBUS bus that will give 10 control channels compared to 8 via PPM. However, it will result in the use of an additional UART port on the microcontroller. When choosing PPW or SBUS, the jumper must be installed, that is located in the upper right part of the controller.

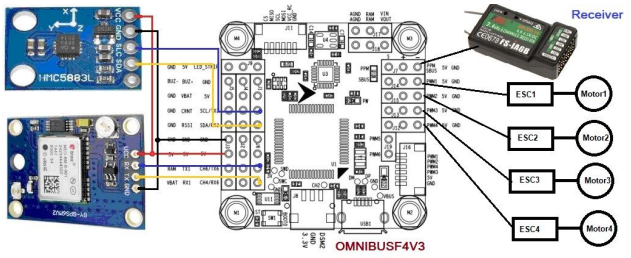

Figure 9: Connecting to the Omnibusf4v3 compass, the GPS receiver, the Fs-Ia6b control receiver

The controller uses the INAV ver.2.2.1 firmware, that is copied from the site: https://github.com/iNavFlight/inav/releases/download/2.2.1/inav_2.2.1_OMNIBUSF4V3.hex, and the INAV ver. 1 from: https://github.com/iNavFlight/inav-configurator/releases/download/2.2.1/INAV-Configurator_win32_2.2.1.zip. The configurator is used for the OMNIBUSF4V3 firmware and for setting up the quadcopter for the given configuration and flight modes.

The copter is configured after starting the configurator and sequentially entering the tabs to set the required parameters [8]. In the Ports tab, on the UART6 line, in the Sensors position, GPS with the transmission rate of 38400 is set. In the Mixer tab, the Quad X drone type is selected. After exiting each tab, the changes in parameters are saved (save button). In the Configuration tab, the Sensors must be MPU6000 (Accelerometer), HMC5883 (Magnetometer), BMP280 (Barometer). In the Board and Sensor Alignment section, the MAG alignment is set - CW 90. It corresponds to the position of the compass rotated 90 degrees. The Receiver Mode - PPM RX input - set the receiver operating mode. In the GPS section, GPS is turned on and the UBLOX protocol is installed. In the ESC / Motor Features section, "Enable motor and servo output" is selected. The protocol is set, for example, ONESHOT125. In the PID tuning tab, the PID parameters of the controller are preset. More precise tuning is carried out in the test flight. The Motors tab controls, what direction the motors rotate in according to the configurator drawing. For changing the direction of rotation the connection of two of the three motor wires to the ESC controller is changed. The flight modes of the quadcopter are set, described in [18-21]. In the Modes tab [20, 21] and in INAV ver.2.2.1 it is possible to set up a flight mission – the Mission Control (flight along the given trajectory with indicating waypoints, Fig. 10). Here you can select a map area. The waypoints are indicated by clicking on the map. Each waypoint after the second click on it with the mouse displays its coordinates with the parameters of the flight height above it and the speed. These values are editable. If you need to return to the starting point with automatic landing, check the RTH at the end of the mission and Landing. The generated route is recorded by the Save mission to FC and Save Eeprom mission commands. The waypoint flight is performed only if the radio control switch is set to NAV WP flight mode in the Modes tab.

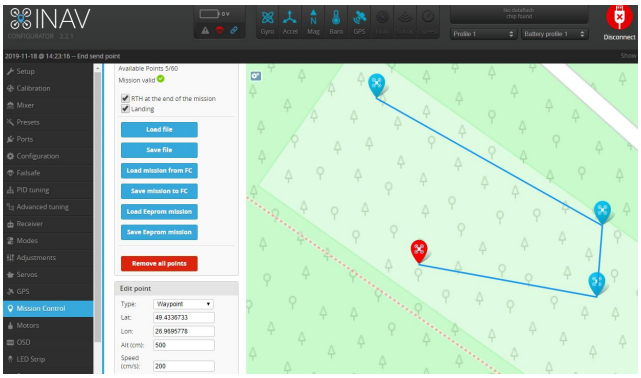

Figure 10: Formation of the flight mission by waypoints

The Calibration tab calibrates the Accelerometer and Compass. The Accelerometer calibration scheme is shown in the figure on this tab. To do this, press the Calibrate Accelerometer button beforehand. When calibrating the compass, the quadcopter must be rotated in 6 axes for no more than 30 seconds. It is preferable to perform calibration in the field by moving the sticks on the control panel according to the rule: left stick up and to the right, right stick down with waiting for 2-3 seconds. After that, the copter rotates along 6 axes.

When flying the copter in automatic mode, the parameters are configured in the Advanced tuning tab. The configuration is carried out in two sections: Multirotor Navigation Settings and RTH and Landing Settings. User Control Mode: Altitude. It is when the drone is tracking less of its position via satellites than Cruise. For example, the remote control switches to the

flight mode NAV POSHOLD - holding the position. If you move forward with the remote control stick, the coordinates from the satellite will not be perceived by the copter. If the stick is moved to the neutral position, the copter will determine the coordinates from the satellite and will return to the position, when the stick took the neutral position. At high speed, it is possible to overshoot the position and the copter will return back to the point, when the stick has taken a neutral position while twitching up to loss of balance and an accident for copters with the frame larger, than 450 mm. This mode is used for small and fast drones.

Cruise. When the sticks are moved from the neutral position, the copter constantly monitors its coordinates in the flight and if the sticks return to the neutral position, the copter stops immediately. This mode isn't very dynamic due to the low speed of the GPS receiver and is used for large quadcopters.

Max. navigatoin speed is the maximum speed in navigation mode, cm/s (when the NAV POSHOLD mode is set).

Max. CRUISE speed is the maximum speed in the Cruise mode, cm/s.

Max. navigator climp rate is the maximum ascent speed in the navigation mode, cm/s.

Max. ALTHOLD climp rate is the maximum ascent speed in the mode of keeping the height, cm/s.

Multirotor max. banking angle [degrees] is the maximum tilt angle of the aircraft in degrees in the navigation mode.

Use mid. throttle for ALTHOLD. When enabled, the altitude hold mode is set, when the throttle stick is in the middle position.

Hover throttle is a number, proportional to the rotational speed of the motors at an average throttle stick.

In the RTH and Landing Settings section, the parameters for returning to the starting point are set, that can be left unchanged.

The peculiarity of the INAV firmware is the changes in the P, I, D parameters during flight, that allows you to adjust the PID controller to ensure maximum flight stability of the copter. To do this, use the Adjustments tab. Two channels are selected on the control panel - a three-position switch and a channel, connected to a variable resistor that changes the pulse values from 1000 to 2000. The changing of the parameter values is performed using the three-position switch. In the middle position, the parameter doesn't change. The upper position decreases the parameter. The lower one increases. A buzzer is connected to the flight controller. When the parameter is decreased by one unit, the buzzer gives a single signal. When zoomed in, double. The channel with the resistor is used to select the parameter. If the rotation of the resistor is marked into 3 identical parts, then three parameters can be changed by setting the resistors to the selected position (Fig. 11).

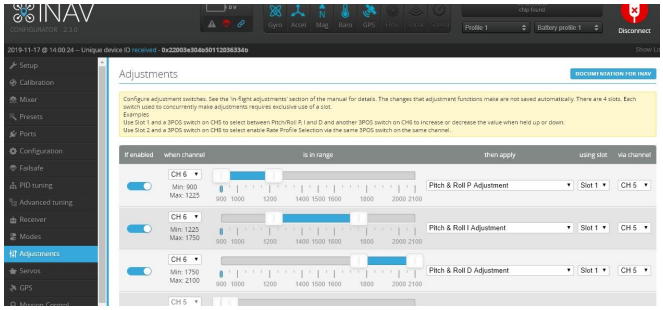

Figure 11: Setting of the parameters' P, I, D change during the flight

As shown in the Fig. 11, the CH6 channel uses the variable resistor. There is the three position switch on the CH5. If the knob of the resistor (CH6) is in the left position, the three position switch changes the P parameter in Pitch and Roll at the same time. When the resistor knob is in the middle position, the values of the parameter I also change by Pitch and Roll simultaneously. After the flight, before disconnecting the battery, to save the changed PID parameters on the control panel, you must lower the sticks down and spread them in the different directions. It should be noted that it is important to adjust the Pos XY P Adjustment, Vel XY P Adjustment, Vel XY D Adjustment parameters to ensure smooth flight of the aircraft in the navigation mode (NAV POSTHOLD, NAV RTH) and waypoints (NAV WP). Therefore, after adjusting the PID regulator, the Adjustments tab was used in the work to experimentally adjust these parameters.

When installing navigation equipment on the copter, the copter can only be armored, when connected to such a number of satellites, that are indicated in the Advanced tuning tab in the Min GPS satellites for the valid fix parameter, for example, 6 satellites. For arming without satellites, use the command set nav_extra_arming_safety = OFF, that is entered in the CLI tab. In this case, the aircraft only holds the altitude and horizon.

A photo of the experimental quadcopter with the OMNIBUSF4V3 controller is shown in Figure 12. The compass is placed outside the GPS receiver housing to reduce malfunctions, hovering and ensure straight-line movement.

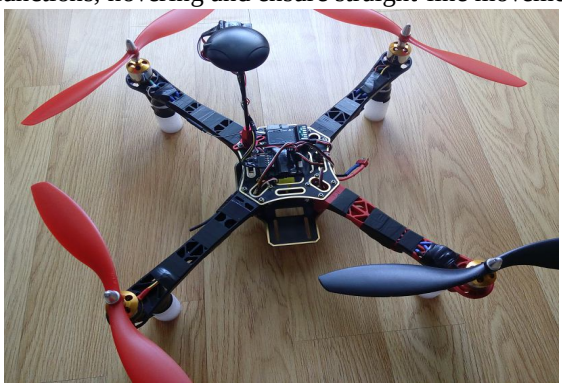

Figure 12: Photo of the experimental copter with the Omnibusf4v3 Controller

When launching of the copter, it was experimentally found for INAV firmware that its PID controllers are very sensitive to ESC controls and motors. For example, the installation of the ESC regulators built on different element bas, but with the same recoil current 30A and the same firmware Simonk showed, that with the sharp increase in the throttle, the copter flips along Roll and slightly along Pitch. With the smooth increase in the throttle, the copter rises steadily.

4. CONCLUSION

Based on the flight tests:

1. The budget fully automatic quadrocopters (up to $ 120) and hexacopter (up to $ 150) are designed and manufactured for taking photos and videos of terrain with the radius of up to 3.5 km with lithium-ion cells with the capacity of 6000 mAh. They have the flight duration of 15-20 minutes and the average speed of 20 km/h.
2. The possibility of creating the flight controller based on the RobotDyn Mega2560 Pro and the Arduino Mega2560 R3 and upgraded MegapirateNG firmware are established for building of the budget quadrocopter and hexacopter, that fly automatically along a set trajectory, with the return to the start point in case of the communication loss and FPV flight.
3. The built copters based on the APM 2.6 / 2.8 flight controller and the ArduCopter 3.2.1 firmware showed a slight improvement in the altitude and the position retention during tests, than the copters with RobotDyn Mega2560 Pro and Arduino, however, the cost of the flight controller doubled.
4. An insignificant difference in the using of the MS5611 and the BMP180 barometers during the altitude hold mode for the MegapirateNG 3.1.5R2 firmware is shown experimentally. Therefore, it is advisable to use the cheaper BMP180 barometer for similar tasks.
5. The possibility of using the INAV firmware with the OMNIBUSF4V3 flight controller based on the STM32F405 microcontroller starting from version 1.9.2 for flight along the given trajectory, that can be formed from the maximum of 60 waypoints, has been investigated.
6. For the INAV firmware, the PID controllers are found to be very sensitive to the ESC controls and motors. For example, the installation of the ESC regulators, built on different element base, but with the same recoil current 30A and the same firmware Simonk showed, that with a sharp increase in the throttle, the copter flips along Roll and slightly along Pitch, that leads to the copter falling.
7. It has been experimentally established, that for the INAV firmware one of the important parameters in the navigation mode is the Position XY - P. For example, when it is reduced from the default values, in the Cruise mode the copter flies very smoothly, but searches for a waypoint for a long time, that leads to a sharp increase flight time. The increasing of this parameter will result in abrupt and unstable flight and possible crash of the aircraft.
8. The INAV firmware problems are fixed in setting up of the HMC5883L magnetometer to ensure straight-line flight of the copter in the Cruise mode. However, if the axis of the magnetometer (X-axis) is directed in the direction of flight, a smaller error (5 degrees) can be achieved. However, for this it is necessary to use the external magnetometer, and the GPS, installing in the receiver housing, turn off.

REFERENCES

1. A. Boyko. **Fields of application of drones,** available at: http://robotrends.ru/robopedia/oblasti-primeneniya-besp ilotnikov.
2. **Ardupilot,** available at: https://ardupilot.org/.
3. **Modes,** available at: https://github.com/iNavFlight/inav/wiki/Modes.
4. A. A. Myasishchev. **Using the ROBOTDYN MEGA2560 PRO board to build a hexacopter flight controller**, *Bulletin of the Khmelnytsky National University. Technical sciences*, № 3, pp. 171–179, 2018.
5. S. Lienkov, A. Myasischev, O. Banzak, Y. Husak, I. Starynski. **Use of rescue mode for UAV on the basis of STM32 microcontrollers.** *International Journal of Advanced Trends in Computer Science and Engineering*, ISSN 2278-3091, Vol. 9, No.3, pp. 3506-3513, 2020, https://doi.org/10.30534/ijatcse/2020/156932020.
6. S. A. Shvorov, N. A. Pasichnyk, S. D. Kuznichenko, I. V. Tolok., S. V. Lienkov, L. A. Komarova. **Using UAV during Planned Harvesting by Unmanned Combines,** *IEEE 5th International Conference Actual Problems of Unmanned Aerial Vehicles Developments*, ISBN: 978-172812592-3, pp. 252-257, 2019.
7. Flight controller, available at: https://ru.wikipedia.org/wiki/Полётный_контроллер.
8. A. A. Myasishchev. **Possibilities of the Cc3d Flight Controller with the INAV Firmware**, *Visnik KhNU. Technical sciences*, №1, pp. 129-13, 2019.
9. F1, F3, F4 and F7 Flight Controller Differences Explained, available at: https://oscarliang.com/f1-f3-f4-flight-controller.
10. INAV, available at: https://github.com/iNavFlight/inav/wiki.
11. Quadcopter PID Explained, available at: https://oscarliang.com/quadcopter-pid-explained-tuning /.
12. S. Lienkov, G. Zhyrov, O. Sieliukov, I. V. Tolok, A.-S. M. Talib, I. V. Pampukha. **Calculation of Reliability Indicators of Unmanned Aerial Vehicle Class 'μ' taking into account Operating Conditions at the Design Stagempukha,** *IEEE 5th International Conference Actual Problems of Unmanned Aerial Vehicles Developments*, ISBN: 978-172812592-3, pp. 52-56, 2019.
13. FlySky-i6-Mod-10ch, available at: https://github.com/benb0jangles/FlySky-i6-Mod-.
14. Mission Planner Home, available at: http://ardupilot.org/planner/.

15. Flying on the points of the quadcopter on the Arduino Mega with firmware MegapirateNG 3.1.5R2, available at: https://www.youtube.com/watch?v=cBdmjWUiCZA.
16. Demonstration of automatic flight of a hexacopter on APM 2.6 with a camera Firefly q6, available at: https://www.youtube.com/watch?v=bLMkhIbNo08.
17. OMNIBUS F4V3, available at: http://nic.vajn.icu/PDF/radio-controlled/OMNIBUS_F4_V3.pdf.
18. A. Kumar, S. Yoon. **Development of Fast and Soft Landing System for Quadcopter Drone using Fuzzy Logic Technology,** *International Journal of Advanced Trends in Computer Science and Engineering*. Vol. 9, No.1, pp. 624-629, 2020, https://doi.org/10.30534/ijatcse/2020/87912020.
19. Gennady G. Kalach and Gennady P. Kalach. **Navigation System Based on the Fuzzy Logic Expert System**, *International Journal of Advanced Trends in Computer Science and Engineering*. Vol. 8, No.6, pp. 2693-2698, 2019, https://doi.org/10.30534/ijatcse/2019/02862019.
20. A. A. Myasishchev. **Features of Determining Parameters of PID Regulator for UAV Firmware on the Example of INAV and PC Omnibusf4v3**, available at: https://www.researchgate.net/publication/337917482_OSOBENNOSTI_OPREDELENIA_PARAMETROV_PID_REGULATORA_DLA_PROSIVOK_BPLA_NA_PRIMERE_INAV_I_PK_OMNIBUSF4V3.