https://doi.org/10.30857/2786-5371.2024.4.7

УДК 004.42

ПОСВІСТАК В. С., МІРОШНИЧЕНКО Д. В.
Київський національний університет технологій та дизайну, Україна

ВИКОРИСТАННЯ МОБІЛЬНОГО ЗВ'ЯЗКУ ДЛЯ ВІДДАЛЕНОГО КЕРУВАННЯ ДРОНОМ

Мета. *Розробка системи віддаленої комунікації між наземною станцією та дроном, що інтегрований із мікрокомп'ютером, із використанням мобільного інтернету та SMS-повідомлень.*

Методика. *Експерименти проводились на FPV-дроні, польотний контролер якого фізично інтегрований із мікрокомп'ютером Raspberry Pi 5 через протокол UART. У якості джерела мобільного зв'язку використано мобільний USB-модем ZTE MF79U та SIM-карту одного із українських операторів. Орендовано та налаштовано віртуальний виділений сервер із ОС Ubuntu. Налаштовано MAVlink-комунікацію між дроном та наземною станцією із використанням сервера-посередника та програмного забезпечення MAVLink Router. Налаштовано допоміжні systemd-служби та розроблено bash-скрипти на сервері та мікрокомп'ютері дрона для ініціації з'єднання при увімкненні та відновлення у випадку його втрати. У якості програмного забезпечення наземної станції для планування місій використано додаток QGroundControl. Мобільний модем налаштовано для можливості використання у якості серійного порта, що підтримує AT-команди. Розроблено скрипти на мові Python для моніторингу отриманих SMS-повідомлень та автовідповідача.*

Результати. *Реалізовано комунікацію дрона із наземною станцією через мобільний інтернет, розроблено скрипти для комунікації з використанням SMS-повідомлень.*

Наукова новизна. *За результатами дослідження розроблено інструменти, при комбінації яких можна знизити вірогідність втрати комунікації між дроном і наземною станцією.*

Практична значимість. *Результати можуть бути використані при реалізації системи віддаленої комунікації між наземною станцією та дроном, і при плануванні автономних польотних місій.*

Ключові слова: *дрон; БПЛА; мобільний модем; мобільний роутер; наземна станція; планування місій БПЛА; MAVLink; MAVLink Router; Raspberry Pi; AT-команди.*

Вступ. Класичне радіокерування дроном має свій ряд обмежень. Вибагливість до дистанції між дроном та ініціатором команд, чи то оператор, чи наземна станція з радіопередавачем робить керування на великих дистанціях проблематичним. Частково вирішити проблему дистанції можуть ретранслятори радіосигналу, проте вразливість до погодних умов, засміченість радіоканалів та інші радіоперешкоди у деяких випадках можуть призвести до втрати сигналу, а іноді і самого дрону. Для зменшення вірогідності виникнення вищезазначених проблем можливе використання резервних каналів комунікації, наприклад мобільного зв'язку – GSM/3G/4G/5G.

Постановка завдання. Метою даного дослідження є огляд реалізації резервних каналів комунікації на дроні з допомогою мобільного зв'язку – інтернету та SMS-повідомлень. Мережа мобільного зв'язку стрімко розвивається у світі, з'являються нові покоління з покращеною якістю та швидкодією. У більшості країн, включно із Україною, присутня велика кількість вишок мобільного зв'язку, що передбачає наявність сигналу на більшій частині території. Мобільне інтернет-з'єднання, за умови його стабільності, є перспективним видом зв'язку для застосування на дроні [1, 2]. У якості резервного каналу комунікації на випадок відсутності інтернету можливе застосування SMS-повідомлень. Вони є економним та більш надійним варіантом комунікації, але можуть бути застосовані лише в певних обмежених випадках. Варто зазначити, що мобільний зв'язок також може бути вразливим до перешкод, таких як погодні умови, радіоелектронна боротьба тощо [3], тому основною метою його

застосування є зменшення вірогідності втрати контролю над дроном у випадку застосування виключно радіосигналу.

Результати дослідження. Сфера дронових технологій та область їх застосування стрімко розвивається. Деякі польотні контролери дронів підтримують підключення додаткових радіомодулів телеметрії. Подібні модулі підключаються до самого польотного контролера та до пристрою із програмним забезпеченням наземної станції (рис. 1). Це дозволяє встановлювати між ними комунікацію, що відбувається через протокол MAVLink [4]. За умови наявності такого зв'язку, можливо керувати дроном із допомогою ПЗ, що підтримує MAVLink протокол, наприклад MAVProxy. Більш просунуте програмне забезпечення для наземних станцій (MissionPlanner, QGroundControl та інші) дозволяє планувати складні польотні місії по заданих GPS точках, площах інтересу тощо [5].

Рис. 1. **Модуль радіотелеметрії SiK**

У даній роботі розглядається реалізація комунікації через мобільний зв'язок на прикладі класичного FPV-дрона із польотним контролером SpeedyBee F405 V3 із прошивкою ArduPilot, що фізично з'єднаний із мікрокомп'ютером Raspberry Pi 5 через протокол UART [6]. На Raspberry Pi 5 було встановлено SD-карту із операційною системою Raspberry Pi OS, що базується на дистрибутиві Debian. Наявність USB-портів робить можливим використання мобільного USB-модема ZTE MF79U. Даний модем використовує RNDIS протокол від Microsoft, що полегшує його цільове використання –комп'ютер розпізнає його як інтерфейс провідного підключення до інтернету. На ОС Linux присутні RNDIS драйвери, таким чином для підключення до інтернету достатньо вставити модем у USB-порт на Raspberry Pi.

Для організації зв'язку між наземною станцією та дроном можливо використовувати додаток MAVLink Router [7], що представляє собою двохсторонній форвардинг MAVLink повідомлень. MAVLink Router необхідно скомпілювати та налаштувати як на дроні (для форвардингу повідомлень із польотного контролера на сервер через UART порти та навпаки), так і на сервері (для форвардингу повідомлень із дрона на наземну станцію та навпаки) – рис. 2.

Для комунікації сервера із дроном виділяється UDP порт, також MAVLink Router створює на окремому порті TCP-з'єднання, що можна використовувати у ПЗ наземної станції. Комунікація через TCP-порт відбувається після встановлення сесії. MAVLink Router підтримує налаштування портів та інших опцій через окремий конфігураційний файл – /etc/mavlink-router/main.conf. На рис. 3 показані налаштування UDP та TCP з'єднань на сервері. Це передбачає попередні налаштування фаєрволу з допомогою утиліти UFW – відкриті відповідні вихідні та вхідні з'єднання.

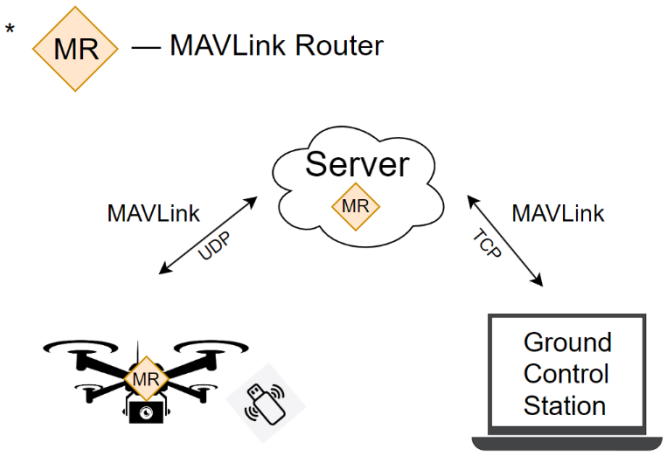

Рис. 2. **Схема комунікації між дроном і наземною станцією**

```
main@d-machine:~$ cat /etc/mavlink-router/main.conf
[General]
TcpServerPort=5761
MavlinkDialect=ardupilotmega
#DebugLogLevel=debug

[UdpEndpoint alpha]
Mode=server
Address=0.0.0.0
Port=14555
```

Рис. 3. **Налаштування MAVLink Router на сервері**

Для забезпечення коректної роботи запуск MAVLink Router можливо винести в окрему systemd службу. Таким чином можна налаштувати автозапуск при старті системи та аварійний перезапуск. Оскільки MAVLink Router передбачає підключення до інтернету, запускати його можна після успішної перевірки з'єднання із сервером Google з допомогою скрипта в умові ExecStartPre – рис. 4.

```
main@d-machine:~$ cat scripts/mavlink_router_config.sh
#!/bin/bash

mavlink-routerd -v
main@d-machine:~$ cat /etc/systemd/system/mavlink_router.service
[Unit]
Description=mavlink-router service
After=network.target

[Service]
ExecStartPre=/bin/bash -c 'until ping -c1 google.com > /dev/null 2>&1; do sleep 1; done'
ExecStart=/home/main/scripts/mavlink_router_config.sh
WorkingDirectory=/home/main
StandardOutput=inherit
StandardError=inherit
Restart=always
RestartSec=10
User=main

[Install]
WantedBy=multi-user.target
```

Рис. 4. **Конфігурація systemd служби для MAVLink Router на сервері**

Автозапуск при старті системи було активовано з допомогою команди systemctl enable. Перевірити статус служби можна командою systemctl status. При коректному налаштуванні служба буде запущеною і з активованим автозапуском (рис. 5).

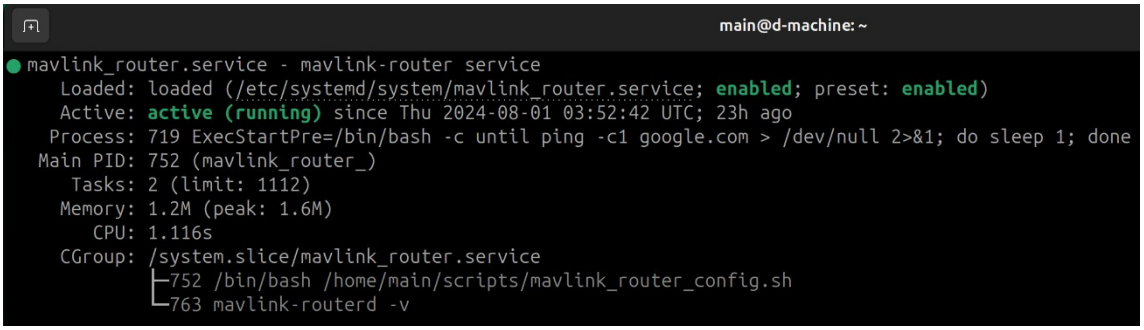

Рис. 5. **Статус systemd служби для MAVLink Router на сервері**

Налаштування MAVLink Router на мікрокомп'ютері Raspberry Pi, який інтегровано з дроном, буде подібним до сервера, крім конфігураційного файлу, де буде налаштований міст між серійним UART портом та UDP з'єднанням – див.

Рис. 6 – Baud rate – параметр, що визначає швидкість передачі даних через інтерфейс UART, вимірюється в бітах на секунду. Важливо, щоб його конфігурація співпадала із значенням, встановленим в налаштуваннях польотного контролера у прошивці ArduPilot – SERIALx_BAUD, його можна встановити через додаток Mission Planner у вкладці Full Parameter List.

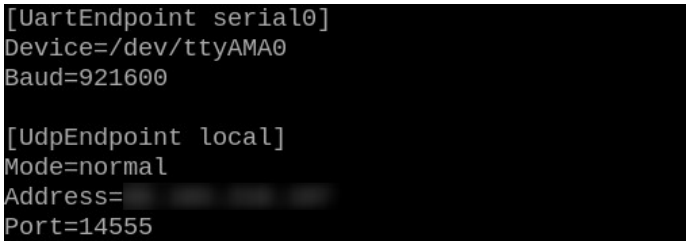

Рис. 6. **Конфігурація мосту між UART та UDP на Raspberry Pi**

Тепер за умови коректно працюючих служб MAVLink Router на Raspberry Pi та на сервері повинен відбуватися успішний обмін даними, про це свідчать логи служби на сервері, що можна переглянути з допомогою команди journalctl – рис. 7.

В налаштуваннях додатку для наземної станції QGroundControl необхідно додати підключення за протоколом TCP до нашого сервера, із відповідною IP-адресою та портом 5761. Після підключення акумуляторної батареї до дрона запускається Raspberry Pi та мобільний USB-модем. В середньому успішний запуск служби MAVLink Router займає до хвилини після підключення батареї, після чого повинно бути встановлене успішне з'єднання із наземною станцією. Додаток QGroundControl інформує про підключення дрона голосовим сповіщенням та зміною статусу підключення із Disconnected на Ready To Fly або Not Ready, залежно від режиму польоту та відповідних умов готовності. На головному екрані додатку показано карту, статус датчиків дрона, готовність, режим польоту, рівень заряду батареї та інші параметри. Дрон оснащено GPS-модулем, його позиція буде відображена на карті за умови наявності GPS-сигналу. Можливо змінити режим польоту, запустити мотори та запланувати політ по заданій місії – рис. 8.

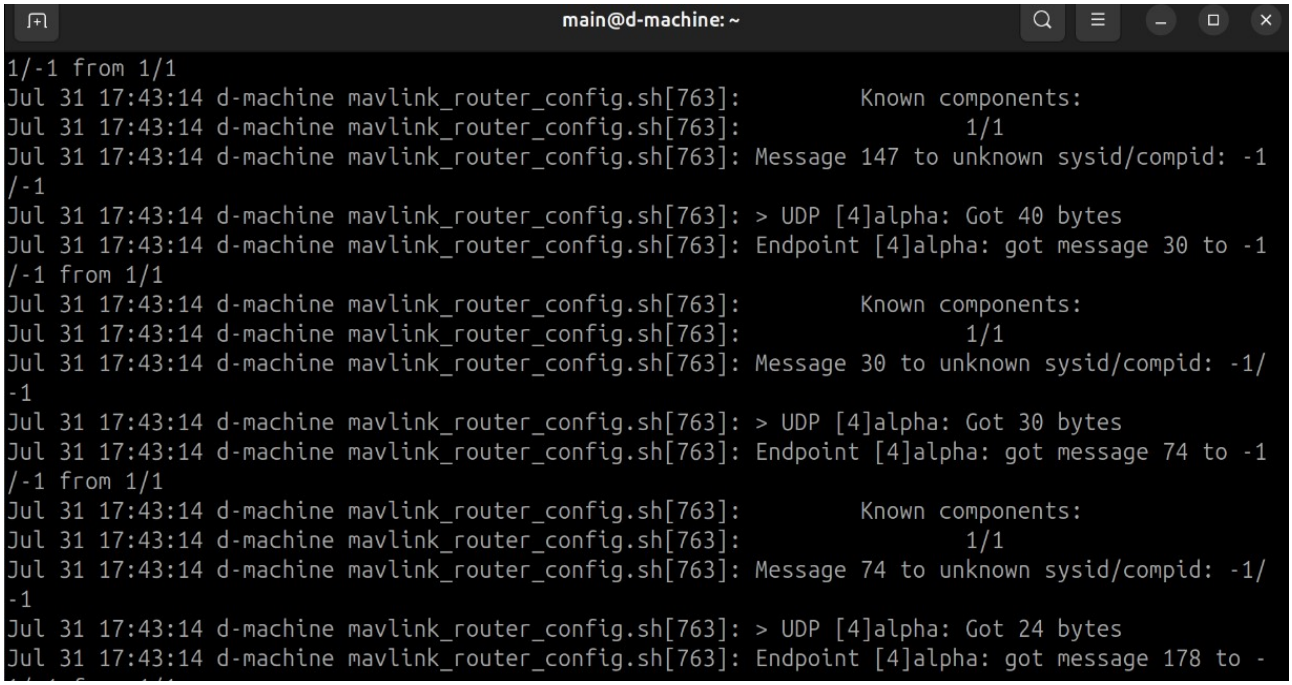

Рис. 7. **Логи служби MAVLink Router на сервері при успішному обміні даних**

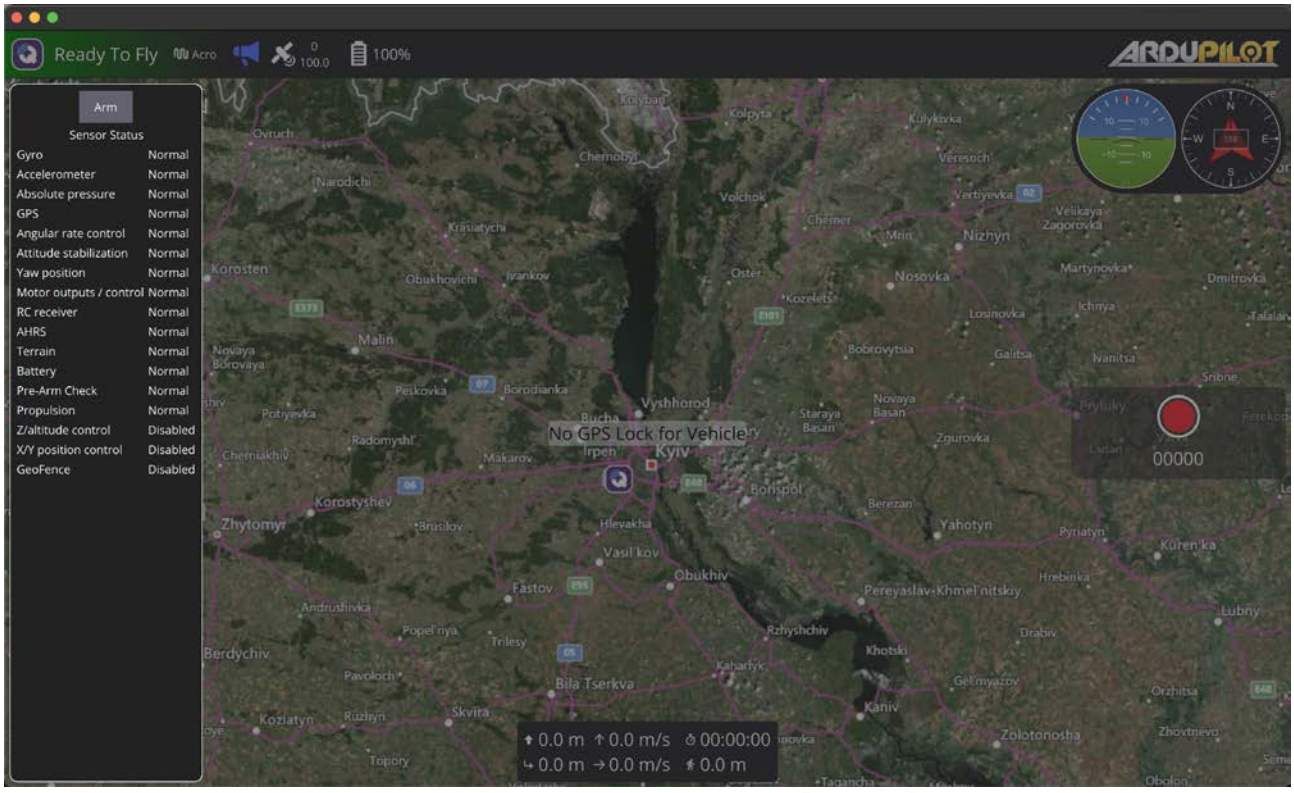

Рис. 8. **Успішне підключення дрона у QGroundControl, статус датчиків**

Після запуску моторів статус змінюється на Armed, а у випадку втрати сигналу – на Communication Lost. В налаштуваннях QGroundControl відображено статистику надісланих, отриманих та втрачених повідомлень MAVLink – рис. 9.

У випадку втрати інтернет-підключення зв’язок із наземною станцією буде перервано. У якості резервного каналу зв’язку можна використовувати SMS-повідомлення. Оскільки для

інтернет-підключення використовується мобільний модем із SIM-картою, є можливість використовувати його і для SMS-повідомлень, і немає необхідності придбання окремого GSM-модуля для Raspberry Pi. Низькорівнева комунікація модемів та телефонів із SIM-картами відбувається з допомогою AT-команд – спеціальної мови команд, розробленої у 1981 році для модемів [8]. За замовчуванням ZTE MF79U знаходиться в режимі RNDIS + Virtual CD-ROM (AT+ZMODE=0) [9], при підключенні до комп'ютера з'являється інтерфейс інтернет-підключення та віртуальний дисковод. Про це свідчить результат команди lsusb, із стандартним productId модема, 1405 – рис. 10.

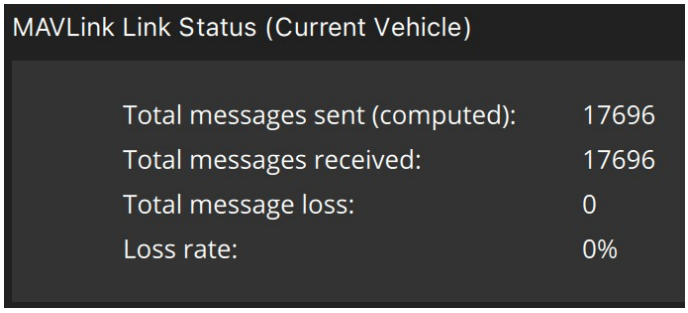

Рис. 9. **Статистика повідомлень MAVLink**

```
master@machine:~$ lsusb | grep ZTE
Bus 001 Device 018: ID 19d2:1405 ZTE WCDMA Technologies MSM DEMO Mobile Boardband
```

Рис. 10. **Результат команди lsusb для стандартного режиму модема ZTE MF79U**

Для переключення в режим із AT портами необхідно інсталювати утиліту sg3-utils, визначити назву віртуального пристрою CD-ROM та надіслати на нього SCSI-послідовність відповідного переключення (рис. 11).

```
master@machine:~$ virtual_cdrom_name=$(sg_scan -i 2>/dev/null | grep -1 "ZTE.*USB SCSI CD-ROM" |
 head -1 | cut -d ":" -f1)
master@machine:~$ echo $virtual_cdrom_name
/dev/sg2
master@machine:~$ sg_raw -n $virtual_cdrom_name 99 00 00 00 00 00
NVMe Result=0x0
```

Рис. 11. **Переключення модему в режим із AT портами**

Після переключення режиму productId модему зміниться на 1602. AT-порти мають з'явитися із порядковими іменами ttyUSB* у папці /dev/. Якщо порти не з'являються автоматично, необхідно активувати серійні порти для модему власноруч (рис. 12).

```
master@machine:~$ modprobe usbserial
master@machine:~$ modprobe option
master@machine:~$ echo '19d2 1602 ff' | sudo tee /sys/bus/usb-serial/drivers/option1/new_id
19d2 1602 ff
```

Рис. 12. **Активація серійних портів модему**

Після цього з'являться порти, на які можна надсилати AT команди через командний інтерфейс, використовуючи утиліту minicom чи інші (рис. 13).

```
19d2 1602 ff
master@machine:~$ ls /dev | grep ttyUSB
ttyUSB0
ttyUSB1
master@machine:~$ minicom -D /dev/ttyUSB0
```

Рис. 13. **Список серійних портів, запуск командного інтерфейсу minicom**

Поточний режим модему має серійні порти, але протокол RNDIS відсутній. Режим AT+ZMODE=1 підтримує і серійні порти, і RNDIS протокол, тому дану команду необхідно надіслати на серійний порт модему. При успішній зміні режиму порт надсилає відповідь “ОК”. Після цього необхідно перевстановити модем у USB-порт, productId поточного режиму – 0536. У випадку відсутності серійних портів, їх активація відбувається як вказано вище із новим productId пристрою. Вищезазначені дії було винесено в окремий bash-скрипт для зручного налаштування модему (рис. 14).

$ setup_dongle_mode.sh ×

home > master > Documents > dev > sandbox > phd > usb_dongle > $ setup_dongle_mode.sh

```
1  #!/bin/bash
2
3  echo 'Determining virtual CDROM name..'
4  cdrom_name=$(sg_scan -i 2>/dev/null | grep -1 "ZTE.*USB SCSI CD-ROM" | head -1 | cut -d ":" -f1)
5  echo $cdrom_name
6
7  echo 'Changing USB doungle mode..'
8  sg_raw -n $cdrom_name 99 00 00 00 00 00
9
10 echo 'Creating ttyUSB devices..'
11 modprobe usbserial
12 modprobe option
13 echo '19d2 1602 ff' > /sys/bus/usb-serial/drivers/option1/new_id
14 ls /dev | grep ttyUSB*
15
16 echo "Executing AT commands.."
17 echo "AT+ZMODE=1" | /home/home/bin/atinout - /dev/ttyUSB0 -
18 sleep 2
19 echo "AT+SOFTRESET" | /home/home/bin/atinout - /dev/ttyUSB0 -
20 sleep 2
21
22 echo "Finished, please unplug and plug in the dongle"
23 exit 0
```

Рис. 14. **Скрипт для налаштування режиму модему RNDIS + AT**

Після активації серійних портів є можливість виконання низькорівневих AT команд на модемі, що дозволяє моніторити отримані смс, та надсилати смс у відповідь [10, 11]. Python-модуль pyserial реалізує можливість комунікації із серійними портами [12]. Для зручного парсингу SMS-повідомлень необхідно перейти у текстовий режим з допомогою команди AT+CMGF=1, та встановити набір символів GSM командою AT+CSCS=”GSM”. AT+CPMS дозволяє обрати бажане місце зберігання SMS- повідомлень при читанні, видаленні, надсиланні та отриманні нових повідомлень [13]:

- SM – пам'ять на SIM-карті;
- ME – пам'ять модему;
- MT – об’єднана пам'ять SM та ME;
- BM, SR, TA – службова пам'ять для різних цілей.

AT+CSMP налаштовує параметри текстового режиму, необхідні для отримання та надсилання повідомлень. Для початкового налаштування модему та моніторингу реалізовано окремі функції (рис. 15).

monitor_messages.py > ...

```python
import time

from at import invoke_at_silent
from sms_message import extract_sms_messages

def setup_at(serial):
    invoke_at_silent(serial, 'AT+CMGF=1')
    invoke_at_silent(serial, 'AT+CSCS="GSM"')
    invoke_at_silent(serial, 'AT+CPMS="SM","SM","SM"')
    invoke_at_silent(serial, 'AT+CSMP=17,167,0,0')


def monitor_sms_messages(serial, on_message_received, polling_interval_ms=500):
    is_first_iteration = True
    cached_messages = []
    while True:
        try:
            invoke_at_silent(serial, 'AT+CMGL="ALL"')
            time.sleep(polling_interval_ms / 1000)
            messages = extract_sms_messages(serial.read(serial.in_waiting).decode())
            new_messages = set(messages) - set(cached_messages)
            cached_messages = messages

            if is_first_iteration:
                is_first_iteration = False
                continue

            if len(new_messages) == 0:
                print("No new messages")
                continue

            for msg in new_messages:
                on_message_received(msg)
        except Exception as e:
            print(f"Error occurred: {e}")
```

Рис. 15. **Функції для початкового налаштування модему та моніторингу SMS-повідомлень**

Варто зазначити, що навіть із наявністю SD-карти у модемі ZTE MF79U, через специфіку прошивки пристрою ME або MT пам'ять працює некоректно, не зберігаючи SMS-повідомлення взагалі. Єдиний можливий варіант – зберігати повідомлення у пам'яті SM, що має 10 вільних місць для повідомлень, тому можливо реалізувати автоматичне очищення після прочитання. Результатом команди AT+CMGL=”ALL” є список рядків із значеннями, розділеними комою. Для парсингу повідомлень реалізовано окремі функцію та клас (рис. 16 та рис. 17).

```python
def extract_sms_messages(at_cmgl_output):
    line_to_ignore_pattern = [
        # "\\+CMGL.*",
        "\\+ZEPCG.*",
        "\\+CMTI.*",
        "OK",
        "\\>.*",
    ]
    sms_metadata_line_pattern = "\\+CMGL.*"

    metadata = []
    messages = []

    for line in at_cmgl_output.split('\n'):
        line = line.replace('\r', '')
        should_ignore = False
        for regex in line_to_ignore_pattern:
            if re.match(regex, line):
                should_ignore = True
                break

        if should_ignore:
            continue

        if re.match(sms_metadata_line_pattern, line):
            metadata.append(line)
        else:
            messages.append(line)

    metadata = list(filter(None, metadata))
    messages = list(filter(None, messages))
    messages_by_metadata = zip_longest(metadata, messages)
    return [SmsMessage(metadata, message) for metadata, message in messages_by_metadata]
```

Рис. 16. **Парсинг SMS-повідомлень**

```python
class SmsMessage:
    def __init__(self, metadata_line, message):
        print(metadata_line)
        values = metadata_line.split(',')
        self.status = values[1].replace('"', '')
        self.sender = values[2].replace('"', '')
        self.date = f"{values[4]} {values[5]}".replace('"', '')
        self.message = message

    def __eq__(self, other):
        return self.sender == other.sender and self.date == other.date and self.message == other.message

    def __hash__(self):
        return hash((self.sender, self.date, self.message))
```

Рис. 17. **Структура класу для SMS-повідомлень**

Для експериментів реалізовано окрему функцію для надсилання повідомлень. В основному скрипті вона використовується у відповідь на повідомлення від «надійного відправника», що є верифікованим номером (рис. 18).

Скрипт моніторить повідомлення, у випадку отримання повідомлення від надійного відправника – спрацьовує автовідповідач із еквівалентним текстом – див. логи скрипту на рис. 19.

main.py > ...

```python
import serial

from monitor_messages import monitor_sms_messages, setup_at
from send_sms import send_sms
from utils import clear_sms_memory

trusted_sender = "+380501234567"
port = "/dev/ttyUSB0"
serial = serial.Serial(port, 460800, rtscts=True, dsrdtr=True, timeout=5)


def handle_sms_message(msg):
    if msg.sender != trusted_sender:
        print(f"Received message from untrusted sender {msg.sender}")
        return

    print(f"Received message from {msg.sender}: {msg.message}")
    send_sms(serial, msg.sender, msg.message)


try:
    clear_sms_memory(serial)
    setup_at(serial)
    monitor_sms_messages(serial, lambda msg: handle_sms_message(msg))
finally:
    serial.close()

```

Рис. 18. **Основний скрипт для моніторингу повідомлень та автовідповідача**

```
No new messages
AT+CMGL="ALL"
No new messages
AT+CMGL="ALL"
+CMGL: 1,"REC UNREAD","+380[redacted]","","24/08/02,11:35:46+12"
Received message from +380[redacted]: Test
AT+CSMP=17,167,0,0
AT+CMGS="+380[redacted]"
Test
```

Рис. 19. **Логи основного скрипту при спрацюваннні автовідповідача**

При реалізації комунікації із дроном можливо визначити певний формат повідомлень-команд, важливо враховувати що SMS-повідомлення технічно підтримує до 160 символів, чого достатньо для критично важливих команд, аж до задання списку координатів для польоту. У випадку потенційного перевищення ліміту можливо вдосконалити формат команд для його надсилання декількома SMS-повідомленнями. Запуск скриптів моніторингу SMS-повідомлень можливо організувати у вигляді systemd служби з автозапуском, по аналогії із вищезазначеними службами для MAVLink Router.

Висновки. У цій роботі була досліджена реалізація комунікації між дроном та наземною станцією із використанням MAVLink Router та сервера-посередника. У якості джерела інтернет-підключення на дроні було використано мобільний USB-модем із SIM-картою. Даний модем було налаштовано для активації серійних портів, що підтримують AT команди, які необхідні для отримання та надсилання SMS-повідомлень у якості резервного каналу зв'язку для дрона. Реалізовано скрипти на мові Python для моніторингу повідомлень та автовідповідача.

Необхідні подальші дослідження для визначення оптимального формату команд, що будуть надсилатись із допомогою SMS-повідомлень. При цьому варто враховувати обмеження у 160 символах та конкретні потреби у функціоналі дрона.

Є потреба у використанні підпису MAVLink повідомлень, доступному у MAVLink v2, що покращує безпеку MAVLink комунікації. Також необхідні дослідження та експерименти у питаннях безпеки для уникнення можливих атак із метою перехоплення дрона чи переривання його зв'язку із наземною станцією.

References

1. Lin, X. et al. (2019). Mobile Network-Connected Drones: Field Trials, Simulations, and Design Insights. *IEEE Vehicular Technology Magazine*, Vol. 14, No. 3, P. 115–125, DOI: 10.1109/MVT.2019.2917363.

2. Burke, P. J. (2019). A Safe, Open Source, 4G Connected Self-Flying Plane With 1 Hour Flight Time and All Up Weight (AUW) <300 g: Towards a New Class of Internet Enabled UAVs. *IEEE Access* (Institute of Electrical and Electronics Engineers (IEEE), Vol. 7, P. 67833–67855, DOI: 10.1109/access.2019.2917851.

3. Ramirez, D. J. (2014). Interference mitigation techniques for 4G networks. Supélec.

4. Messages (common) MAVLink Developer Guide. *mavlink.io*. URL: https://mavlink.io/en/messages/common.html.

5. Ramirez-Atencia, C., Camacho, D. (2018). Extending QGroundControl for Automated Mission Planning of UAVs. *Sensors* (MDPI AG), Vol. 18, No. 7, P. 2339, DOI: 10.3390/s18072339.

6. Communicating with Raspberry Pi via MAVLink – Dev documentation. *ardupilot.org*. URL: https://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html.

7. MAVLink Router. *GitHub*. Apr. 25, 2023. URL: https://github.com/mavlink-router/ mavlink-router.

8. Telit Wireless Solutions (2006). AT Commands Reference Guide.

9. Opensource. *opensource.ztedevices.com*. URL: https://opensource.ztedevices.com/

Література

1. Lin X. et al. Mobile Network-Connected Drones: Field Trials, Simulations, and Design Insights. *IEEE Vehicular Technology Magazine*. 2019. Vol. 14, No. 3. P. 115–125. DOI: 10.1109/MVT.2019.2917363.

2. Burke P. J. A Safe, Open Source, 4G Connected Self-Flying Plane With 1 Hour Flight Time and All Up Weight (AUW) <300 g: Towards a New Class of Internet Enabled UAVs. *IEEE Access* (Institute of Electrical and Electronics Engineers (IEEE). 2019. Vol. 7. P. 67833–67855. DOI: 10.1109/access.2019.2917851.

3. Ramirez D. J. Interference mitigation techniques for 4G networks. Supélec, 2014.

4. Messages (common) MAVLink Developer Guide. *mavlink.io*. URL: https://mavlink.io/en/messages/common.html.

5. Ramirez-Atencia C., Camacho D. Extending QGroundControl for Automated Mission Planning of UAVs. *Sensors* (MDPI AG). 2018. Vol. 18, No. 7. P. 2339. DOI: 10.3390/s18072339.

6. Communicating with Raspberry Pi via MAVLink – Dev documentation. *ardupilot.org*. URL: https://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html.

7. MAVLink Router. *GitHub*. Apr. 25, 2023. URL: https://github.com/mavlink-router/ mavlink-router.

8. Telit Wireless Solutions. AT Commands Reference Guide. 2006.

9. Opensource. *opensource.ztedevices.com*. URL: https://opensource.ztedevices.com/

10. Send SMS using AT commands. *www.smssolutions.net*. URL: https://www.smssolutions.net/tutorials/gsm/sendsmsat/
11. Receiving SMS messages using AT commands. *www.smssolutions.net.* URL: https://www.smssolutions.net/tutorials/gsm/receivesmsat/
12. pySerial. *GitHub*. Mar. 14, 2022. URL: https://github.com/pyserial/pyserial.
13. SMS Tutorial: Preferred Message Storage (AT+CPMS). *www.developershome.com.* URL: https://www.developershome.com/sms/cpmsCommand.asp.

10. Send SMS using AT commands. *www.smssolutions.net*. URL: https://www.smssolutions.net/tutorials/gsm/sendsmsat/
11. Receiving SMS messages using AT commands. *www.smssolutions.net.* URL: https://www.smssolutions.net/tutorials/gsm/receivesmsat/
12. pySerial. *GitHub*. Mar. 14, 2022. URL: https://github.com/pyserial/pyserial.
13. SMS Tutorial: Preferred Message Storage (AT+CPMS). *www.developershome.com.* URL: https://www.developershome.com/sms/cpmsCommand.asp.

POSVISTAK VALERII
Postgraduate Student,
Faculty of Mechatronics and Computer Technologies,
Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0009-0001-7785-1378
E-mail: valposv@gmail.com

MIROSHNYCHENKO DMYTRO
Postgraduate Student,
Faculty of Mechatronics and Computer Technologies,
Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0009-0002-0904-1645
E-mail: dmiroshnycheko@gmail.com

POSVISTAK V. S., MIROSHNYCHENKO D. V.
Kyiv National University of Technologies and Design, Ukraine

USAGE OF MOBILE NETWORK FOR REMOTE DRONE

Purpose. *Developing a system of remote communication via mobile internet and SMS messages between a ground control station and a drone integrated with a microcomputer.*

Methodology. *Experiments were conducted on an FPV-drone with a flight controller that is physically integrated with a Raspberry Pi 5 microcomputer via UART protocol. As a mobile network source a mobile USB-dongle with a Ukrainian SIM-card was used. An Ubuntu virtual private server was rented and set up. MAVLink communication between the drone and a ground control station was set up using MAVLink Router application and a middleware server. Helper systemd services and bash scripts were set up on the server and the drone microcomputer to initiate a connection on system start and in case of failure. As a ground control system QGroundControl was used. Mobile USB-dongle was set up to support serial ports which accept AT commands. Python scripts to monitor and send SMS messages were developed.*

Findings. *Implemented communication between a drone and a ground control station via mobile network.*

Originality. *According to the results of the research, developed instruments with combination of which the risk of communication loss between a drone and a ground control station can be reduced.*

Practical value. *The results can be used to implement a remote communication system between a drone and a ground control station and to plan autonomous flight missions.*

Keywords*: drone; UAV; mobile modem; mobile router; mobile USB-dongle; ground control station; UAV mission planning; MAVlink; MAVLink Router; Raspberry Pi; AT commands.*