Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ОТЧЕТ
по лабораторной работе №1
на тему

АСИНХРОННАЯ ДВУНАПРАВЛЕННАЯ ПОБАЙТНАЯ ПЕРЕДАЧА ДАННЫХ

Выполнил студент группы № 050503 Казак И. А

Преподаватель Марцинкевич В. А.

Минск 2022

1 КОД ПРОГРАММЫ

1.1 Пакет serial

```
package serial

import (
    "github.com/tarm/serial"
)

type Port struct {
    Name       string
    Baund      int
    SerialPort *serial.Port
}

func Read(p Port) (int, string, error) {
    buf := make([]byte, 200480)
    n, err := p.SerialPort.Read(buf)
    if err != nil {
        return 0, "", err
    }
    return n, string(buf), nil
}

func Write(p Port, text string) (int, error) {
    n, err := p.SerialPort.Write([]byte(text))
    if err != nil {
        return 0, err
    }
    return n, nil
}

func InitPort(p *Port) error {
    c := &serial.Config{Name: p.Name, Baud: p.Baund}
    temp, err := serial.OpenPort(c)
    p.SerialPort = temp
    return err
}

func Close(p Port) error {
    if err := p.SerialPort.Close(); err != nil {
        return err
    }
    return nil
}
func ChangeSpeed(p *Port) error {
    err := Close(*p)
    if err != nil {
        return err
}
```

```
    err = InitPort(p)
    if err != nil {
        return err
    }
    return nil
}
```

1.2 Пакет main

```
package main

import (
    "fmt"
    "io"
    "math/rand"
    "os"

    "time"
    "toks/serialDriver/serial"
)

var done = make(chan struct{})

var writer io.Writer = os.Stdout
var bytes int

func main() {
    var s string

    portWrite := &serial.Port{Name: "/dev/ttys003", Baund: 9600}
    portRead := &serial.Port{Name: "/dev/ttys004", Baund: 50}

    serial.InitPort(portRead)
    defer serial.Close(*portRead)

    go send(portWrite, portRead)

    buff := make([]byte, 1)
    for {
        os.Stdin.Read(buff)
        switch buff[0] {
        case 113:
            close(done)
            time.Sleep(100 * time.Microsecond)
            return
        case 99:

            writer = io.Discard
            fmt.Printf(«Write:\n1-115200\n2-57600\n3-38400\n
                       4-19200\n5-9600\n6-4800\n7-2400\n8-1200\n
```

```
                        9-600\n10-300\n11-200\n12-150\n13-134\n
                        14-110\n15-75\n16-50\n")
            fmt.Scan(&s)
            setSpeed(portWrite, s)

            fmt.Printf(«Read:\n1-115200\n2-57600\n3-38400\n
                       4-19200\n5-9600\n6-4800\n7-2400\n8-1200\n
                       9-600\n10-300\n11-200\n12-150\n13-134\n
                       14-110\n15-75\n16-50\n")
            fmt.Scan(&s)
            setSpeed(portRead, s)

            writer = os.Stdout
        }
    }

}

func send(p *serial.Port, pr *serial.Port) {

    serial.InitPort(p)
    defer serial.Close(*p)

    for {
        select {
        case <-done:
            return
        default:
            var err error
            var text string
            for i := 0; i < 1024; i++ {
                text += fmt.Sprint(rand.Intn(10))
            }

            bytes, err = serial.Write(*p, text)
            if err != nil {
                fmt.Fprintf(os.Stderr, "Error while writing:
                            %v", err)
                return
            }
            fmt.Fprintln(writer, "Sended text: ", text, bytes)
            read(pr)
        }

    }

}

func read(p *serial.Port) {

    var readBytes int
    for {
```

```
        select {
        case <-done:
            n, text, err := serial.Read(*p)
            fmt.Println(n)
            if err != nil {
                fmt.Fprintf(os.Stderr, "Error while reading:
%v", err)
                return
            }
            fmt.Fprintln(writer, "Readed text: ", text, " Bytes:
", n)
            return
        default:
            n, text, err := serial.Read(*p)
            readBytes += n
            if err != nil {
                fmt.Fprintf(os.Stderr, "Error while reading:
%v", err)
                return
            }
            fmt.Fprintln(writer, "Readed text: ", text, " Bytes:
", n)
        }
        if readBytes == bytes {
            return
        }
    }

}

func setSpeed(p *serial.Port, s string) {
    switch s {
    case "1":
        p.Baund = 115200
        serial.ChangeSpeed(p)
    case "2":
        p.Baund = 57600
        serial.ChangeSpeed(p)
    case "3":
        p.Baund = 38400
        serial.ChangeSpeed(p)
    case "4":
        p.Baund = 19200
        serial.ChangeSpeed(p)
    case "5":
        p.Baund = 9600
        serial.ChangeSpeed(p)
    case "6":
        p.Baund = 4800
        serial.ChangeSpeed(p)
    case "7":
        p.Baund = 4800
```

```
            serial.ChangeSpeed(p)
        case "8":
            p.Baund = 2400
            serial.ChangeSpeed(p)
        case "9":
            p.Baund = 1200
            serial.ChangeSpeed(p)
        case "10":
            p.Baund = 600
            serial.ChangeSpeed(p)
        case "11":
            p.Baund = 300
            serial.ChangeSpeed(p)
        case "12":
            p.Baund = 150
            serial.ChangeSpeed(p)
        case "13":
            p.Baund = 134
            serial.ChangeSpeed(p)
        case "14":
            p.Baund = 110
            serial.ChangeSpeed(p)
        case "15":
            p.Baund = 75
            serial.ChangeSpeed(p)
        case "16":
            p.Baund = 50
            serial.ChangeSpeed(p)
        }
}
```

2 ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

1. Разработать программный модуль реализации процедуры передачи (приема) байта информации через последовательный интерфейс.
2. В программах синхронно изменить скорости передачи и приема байта до минимальной и максимальной. Проверить функционирование звена приемопередачи.
3. Установить различные скорости для приемника и передатчика. Проверить функционирование звена приемопередачи.