Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №1

«Последовательный порт»

Выполнил: Проверил:

Студент группы 050503 Преподаватель

Латфулин В.Р. Одинец Д.Н.

Минск, 2022

1. Постановка задачи

Разработать программный модуль реализации процедуры передачи (приёма) байта информации через последовательный интерфейс.

Программа должна демонстрировать программное взаимодействие с последовательным интерфейсом с использованием следующих механизмов:

1. Прямое взаимодействие с портами ввода-вывода (wirte, read)
2. Использование BIOS прерывания 14h
3. Работа с COM-портом через регистры как с устройствами ввода-вывода.
4. Алгоритм

Программа состоит из нескольких подпрограмм (частей программы), представляющих собой некоторые функции. К ним относятся функции:

* Инициализация порта
* Запись байта информации в порт
* Чтение байта информации из порта
* Вывод результата на экран

1. Листинг программы

Далее приведены листинги программ, реализующие различные механизмы передачи (приёма) информации через последовательный интерфейс.

3.1. Листинг программы, взаимодействующей с портами ввода-вывода.

from traceback import print\_tb

import serial

import time

def ports():

firstPortsDevTtys = serial.Serial('/dev/ttys001')

secondPortsDevTtys = serial.Serial('/dev/ttys002')

print("first com port is open: " + str(firstPortsDevTtys.isOpen()))

print("second com port is open: " + str(secondPortsDevTtys.isOpen()))

firstPortsDevTtys.write(b'sun')

time.sleep(1)

time.sleep(1)

data = secondPortsDevTtys.read\_all()

print("second ports read: " + data)

firstPortsDevTtys.close()

print("first com port is open: " + str(firstPortsDevTtys.isOpen()))

secondPortsDevTtys.close()

print("second com port is open: " + str(secondPortsDevTtys.isOpen()))

if \_\_name\_\_ == '\_\_main\_\_':

ports()

3.2. Листинг программы, использующей BIOS прерывание 14h.

.model small   
.stack 100h   
  
.data   
  
Error\_Write db "Write error!",0Dh,0Ah,'$'   
Error\_Read db "Read error!",0Dh,0Ah,'$'   
             
.code   
  
start:   
; connect data   
mov ax,DGROUP   
mov ds,ax   
; send byte   
mov dx,0   
mov ah,1   
mov al,'a'   
int 14h   
mov al,'r'   
  
;check if cool   
cmp ah,7   
je W\_Error   
jmp W\_Cont   
W\_Error:   
  mov dx,offset Error\_Write   
  mov ah,9   
  int 21h   
  mov ax,4C00h   
  int 21h   
  ret   
W\_Cont:   
; read   
xor dx,dx   
mov dx,1   
mov ah,02h   
int 14h   
;check if read   
cmp ah,0   
jne R\_Error   
jmp R\_Continue   
R\_Error:   
  mov dx,offset Error\_Read   
  mov ah,9   
  int 21h   
  mov ax,4C00h   
  int 21h   
  ret   
R\_Continue:   
  mov ah,2   
  mov dl,al   
  int 21h   
  mov ax,4C00h   
  int 21h   
  
end start

3.3. Листинг программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.

.model small   
.stack 100h   
.data   
  
.code   
start:   
  
  
mov dx,3F8h   
mov al,'r'   
out dx,al   
  
mov al,'a'   
  
mov dx,2F8h   
  
in al,dx   
mov ah,2   
mov dl,al   
int 21h   
  
mov ax,4C00h   
int 21h   
end start

1. Тестирование программ

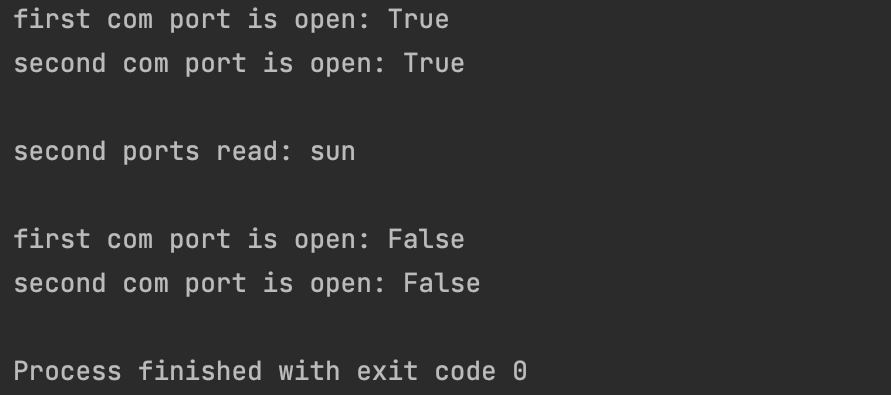


Рисунок 4.1 – Результат работы программы, взаимодействующей с портами ввода-вывода.



Рисунок 4.2 – Результат работы программы, использующей BIOS прерывание 14h.



Рисунок 4.3 – Результат работы программы, работающей с COM-портами через регистры как с устройствами ввода-вывода.

1. Заключение

В ходе лабораторной удалось передать 1 байт информации через последовательный порт с использованием различных механизмов.

Для эмуляции COM портов использовался socat, для эмуляции DOS используется QEMU и dosbox на хосте macOs Big Sur.