МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г.Шухова)

Лабораторная работа №1

дисциплина «Программирование микроконтроллеров»

тема: «Считывание нажатий клавиатуры с помощью микроконтроллера»

Выполнил: студент группы ВТ-31

Новожен Н.В.

Проверил: Гольцов Ю. А.

Белгород 2020

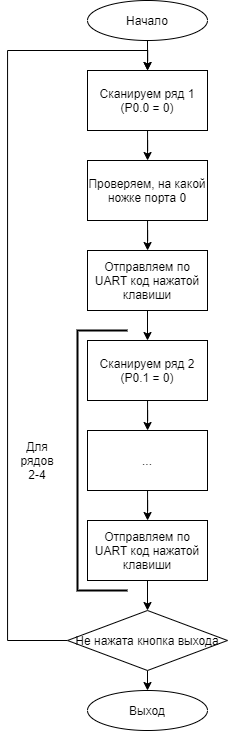
**Цель работы:** изучить принципы работы **UART** и организацию прерываний в микроконтроллерах серии **MCS-51**.

Задание к работе:

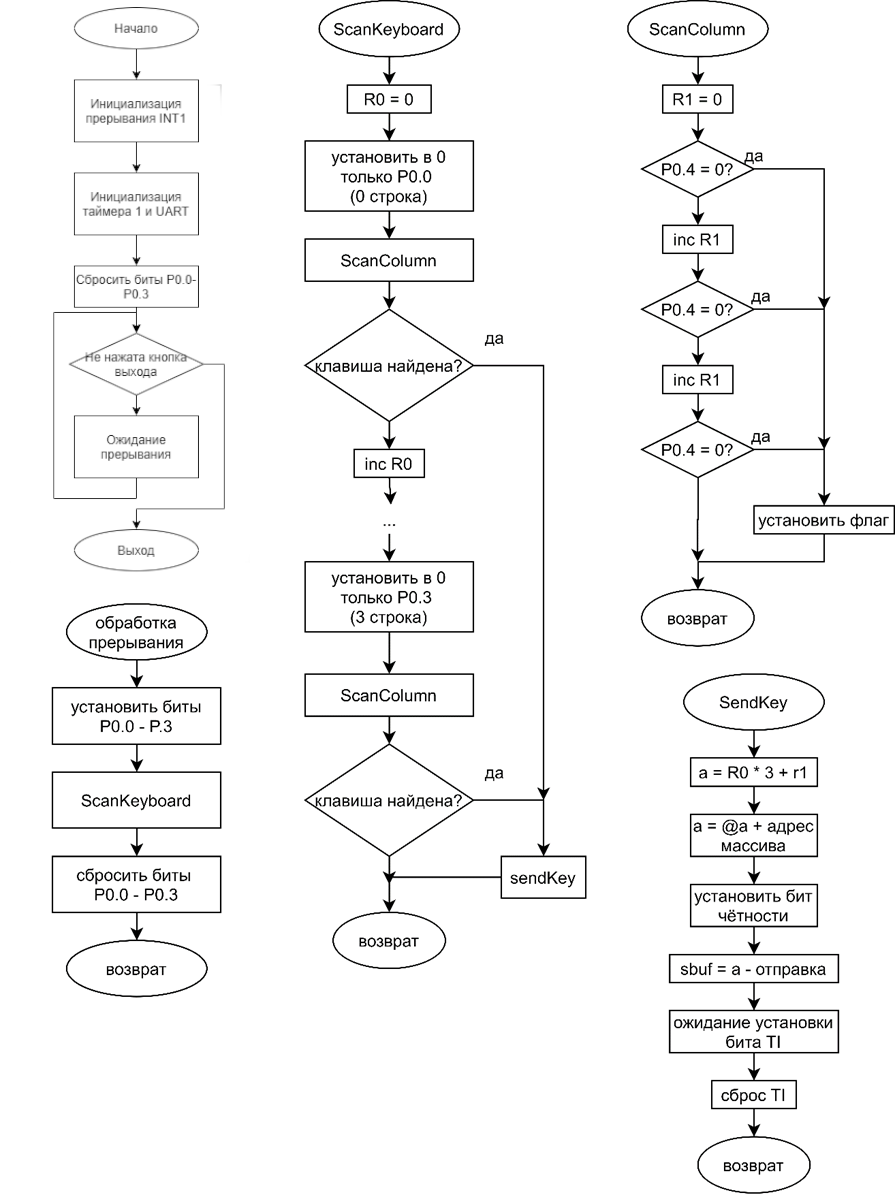
1. Создать программу для считывания скан-кода с клавиатуры, перевод его в ASCII код, и передачи полученного символа в терминал через UART путем постоянного опроса портов Р0, к которому подключена клавиатура.
2. Создать программу для считывания скан-кода с клавиатуры, перевод его в ASCII код, и передачи полученного символа в терминал через UART используя прерывания клавиатуры.

**Блок-схемы разработанных программ**

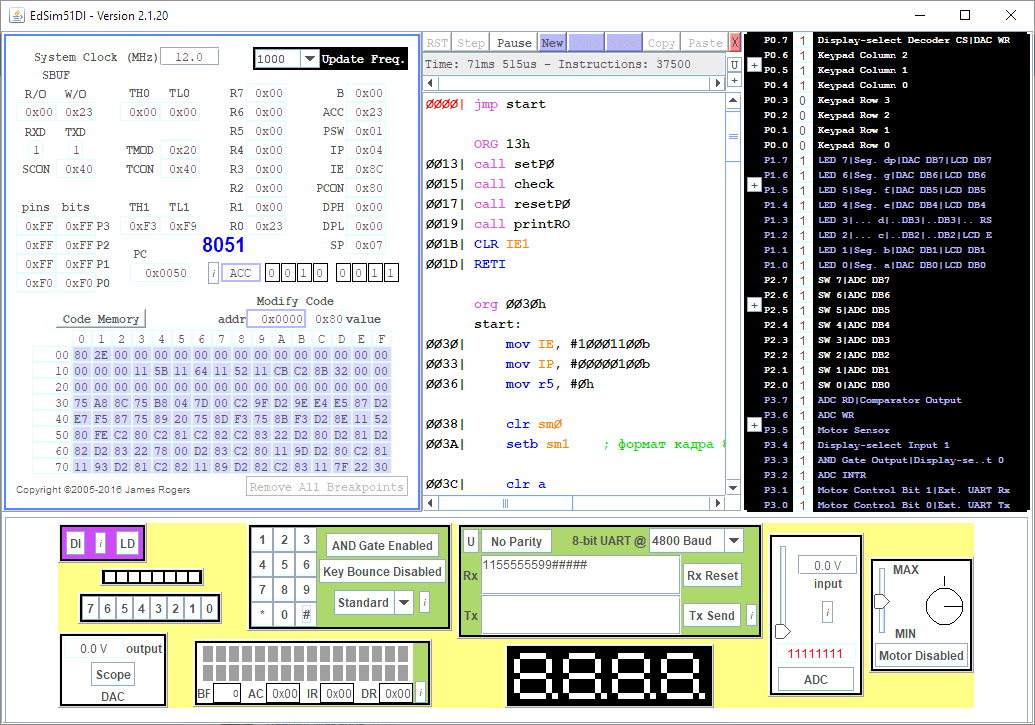
**Задача 1**



**Задача 2**



**Результаты работы программы на симуляторе EdSim51DI:**



**Вывод:** в процессе выполнения лабораторной работы были изучены принципы работы UART, опроса клавиатуры методом полинга и организация прерываний в микроконтроллерах серии MCS-51.

**Задача №1**

; Символы клавиатуры

org 500h

db 031h, 032h, 033h, 043h, 044h, 034h, 035h, 036h, 045h, 046h, 037h, 038h, 039h, 047h, 048h, 041h, 030h, 042h, 049h, 04Ah

org 0h

loop1:

; Для сканирования выставим 1 на всем порте 0, кроме последнего бита

mov p0, #11111110b

mov a, p1

; Если кнопки не нажаты, пропускаем

cjne a, 0ffh, j1

jmp j1\_

j1:

; Определяем нажатую кнопку

jnb p1.0, j10

jnb p1.1, j11

jnb p1.2, j12

jnb p1.3, j13

jnb p1.4, j14

; Берем индекс нажатой клавиши

j10:

mov a, #0

jmp j1end

j11:

mov a, #1

jmp j1end

j12:

mov a, #2

jmp j1end

j13:

mov a, #3

jmp j1end

j14:

mov a, #4

j1end:

; Отправляем в UART символ с индексом a

mov dptr, #500h

movc a, @a + dptr

mov sbuf, a

mov sbuf, #10

mov sbuf, #13

j1\_:

; Аналогично первому варианту, но сканируем следующий бит

mov p0, #11111101b

mov a, p1

cjne a, 0ffh, j2

jmp j2\_

j2:

jnb p1.0, j20

jnb p1.1, j21

jnb p1.2, j22

jnb p1.3, j23

jnb p1.4, j24

j20:

mov a, #0

jmp j2end

j21:

mov a, #1

jmp j2end

j22:

mov a, #2

jmp j2end

j23:

mov a, #3

jmp j2end

j24:

mov a, #4

j2end:

mov dptr, #505h

movc a, @a + dptr

mov sbuf, a

mov sbuf, #10

mov sbuf, #13

j2\_:

mov p0, #11111011b

mov a, p1

cjne a, 0ffh, j3

jmp j3\_

j3:

jnb p1.0, j30

jnb p1.1, j31

jnb p1.2, j32

jnb p1.3, j33

jnb p1.4, j34

j30:

mov a, #0

jmp j3end

j31:

mov a, #1

jmp j3end

j32:

mov a, #2

jmp j3end

j33:

mov a, #3

jmp j3end

j34:

mov a, #4

j3end:

mov dptr, #50Ah

movc a, @a + dptr

mov sbuf, a

mov sbuf, #10

mov sbuf, #13

j3\_:

mov p0, #11110111b

mov a, p1

cjne a, 0ffh, j4

jmp j4\_

j4:

jnb p1.0, j40

jnb p1.1, j41

jnb p1.2, j42

jnb p1.3, j43

jnb p1.4, j44

j40:

mov a, #0

jmp j4end

j41:

mov a, #1

jmp j4end

j42:

mov a, #2

jmp j4end

j43:

mov a, #3

jmp j4end

j44:

mov a, #4

j4end:

mov dptr, #50Fh

movc a, @a + dptr

mov sbuf, a

mov sbuf, #10

mov sbuf, #13

j4\_:

; Повторяем опрос

ljmp loop1

end

**Задача №2**

org 500h

; Символы клавиатуры

db 23h, 30h, 02ah, 39h, 38h, 37h, 36h, 35h, 34h, 33h, 32h, 31h

ORG 0

JMP main

; обработчик прерывания int1

org 13h ; int1

setb p0.0

setb p0.1

setb p0.2

setb p0.3

call scanKeyBoard

CLR p0.0

CLR p0.1

CLR p0.2

CLR p0.3

reti

org 030h

main:

; Настройка работы UART

CLR SM0

SETB SM1 ; 2 режим, 8 бит, стоп/старт, четность

MOV A, PCON

SETB ACC.7

MOV PCON, A ; 7 бит в PCON установлен в 1, множитель скорости 2

; Настройка таймера

; При переполнении записывается из TL1 в TH1 (M1 = 1, M0 = 0)

MOV TMOD, #20H

MOV TH1, #243

MOV TL1, #243

SETB TR1 ; запуск таймера 1

; Настройка прерывания

SETB IT1 ; по срезу

SETB EX1 ; прерывание от INT1

SETB EA ; разрешить все прерывания

mov dptr, #500h

; Устанавливаем в порте 0, чтобы ждать прерывания при нажатии

CLR p0.0

CLR p0.1

CLR p0.2

CLR p0.3

; Бесконечный пустой цикл

sjmp $

; Сканирование по рядам

scanKeyBoard:

mov r0, #0

mov p0, #11111110b

CALL scanColumn

JB F0, gotKey

inc r0

mov p0, #11111101b

CALL scanColumn

JB F0, gotKey

inc r0

mov p0, #11111011b

CALL scanColumn

JB F0, gotKey;

inc r0

mov p0, #11110111b

CALL scanColumn

JB F0, gotKey

ret

gotKey:

call sendKey

ret

; Сканирование столбца

scanColumn:

mov r1, #0

JNB P0.4, gotCol

INC R1

JNB P0.5, gotCol

INC R1

JNB P0.6, gotCol

INC R1

clr f0

RET ; ключ не найден

gotCol:

SETB F0 ; ключ найден, устанавливаем пользовательский флаг

ret

; Отправка нужного символа

sendKey:

mov a, r0

mov b, #3

mul ab

add a, r1

movc a, @a + dptr

MOV C, P

MOV ACC.7, C ; бит чётности

MOV SBUF, A

JNB TI, $ ; ожидание пока установится бит TI

CLR TI

ret