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 №3**

дисциплина «Программирование микроконтроллеров»

тема: «Использование ЦАП и реализация АЦП последовательного приближения»

Выполнил: студент группы ВТ-31

Ковалёв И. Д.

Проверил: Гольцов Ю. А.

Белгород 2020

**Цель работы**

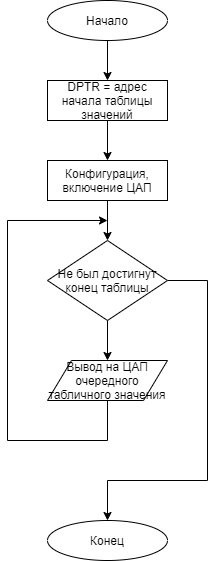
Изучить принципы работы с ЦАП в микроконтроллере 8051, изучить принцип работы АЦП последовательного приближения, и реализовать данный алгоритм на микроконтроллере 8051.

**Задание к работе**

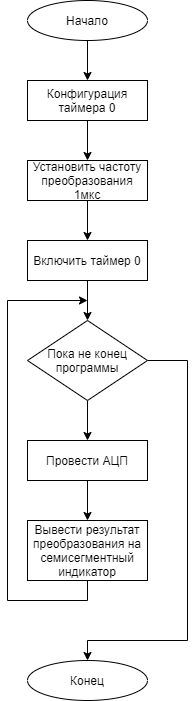
Создать программу для отрисовки на осциллографе, входящем в состав EdSim8051DI, графиков синусоиды, пилообразной волны и меандо. Создать программу, выполняющую преобразование уровня напряжения 0-5 вольт в числовое значение. Значение вывести на семисегментный индикатор. Программа должна использовать ЦАП и компаратор, входящие в симулятор EdSim51DI.

**Блок-схемы разработанных программ:**

**Задача 1:**

****

**Задача 2:**

****

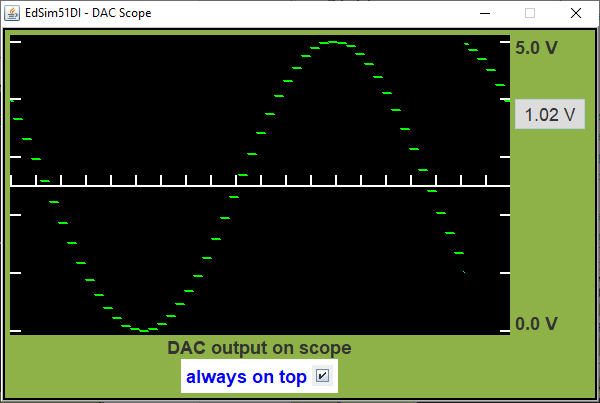
Вывод на семисегментный индикатор был реализован в лабораторной работе №2, потому блок-схему алгоритма ее работы приводить не будем, так как принцип действия не изменился:

* Получить число
* Перевести его в код сегмента
* Выбрать обновляемый сегмент по портам Р3.3 и Р3.4
* Вывести код сегмента на порт Р1

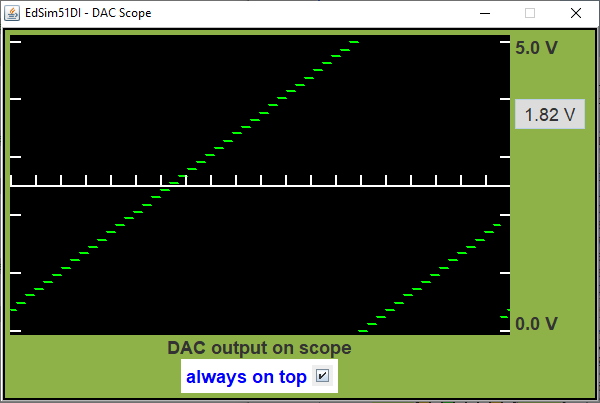
**Скриншоты разработанной программы:**

**Задача 1:**

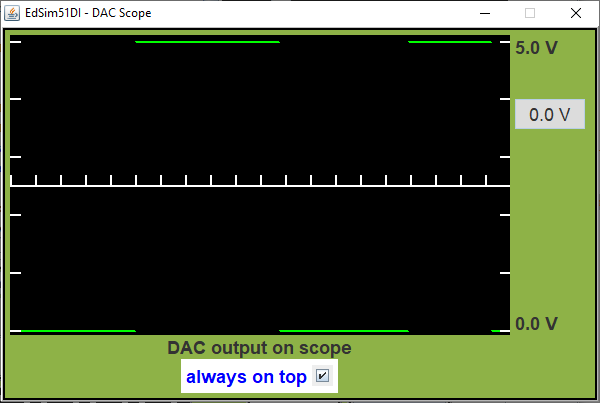
Синусоида:



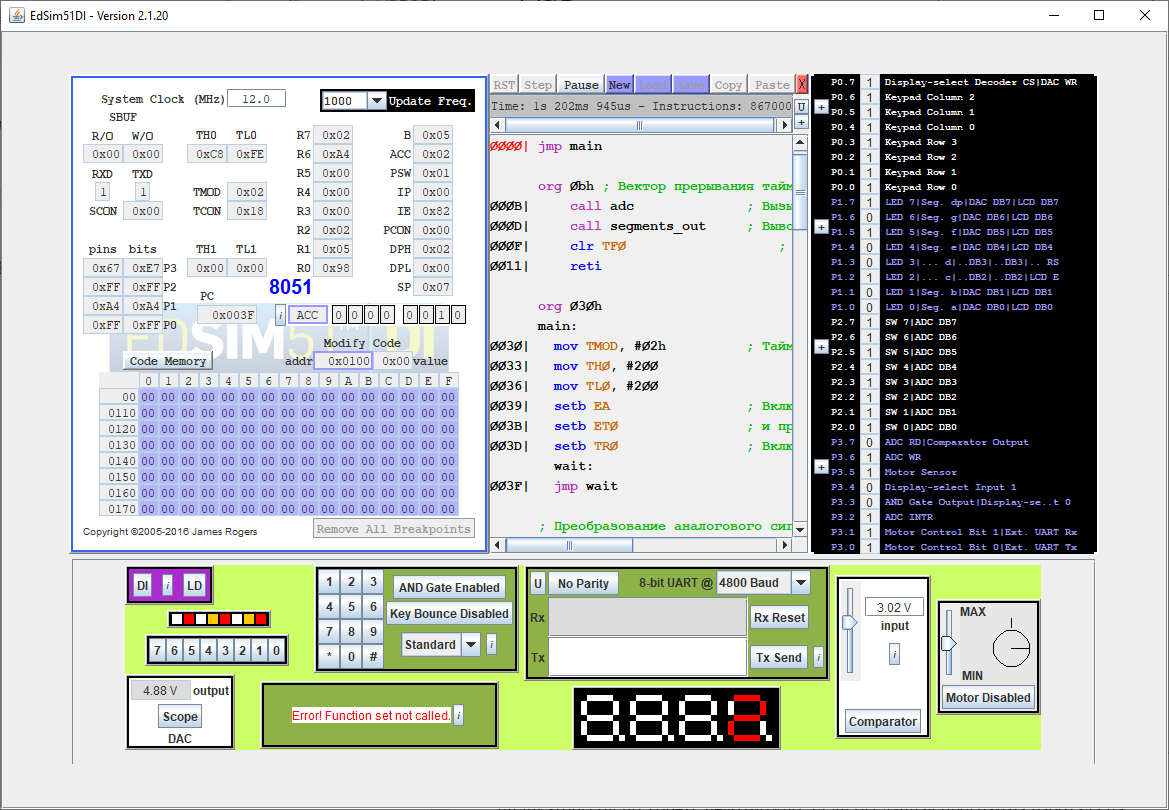
Пила:



Меандр:



**Задача 2:**



Поскольку обновление ЖКИ происходит посегментно, уловить момент, когда все индикаторы будут гореть невозможно, поэтому результат предоставим несколькими скриншотами:







**Вывод:** в процессе выполнения были изучены принципы работы с ЦАП в микроконтроллере 8051, принцип работы АЦП последовательного приближения, а так же был реализован АЦП последовательного приближения на микроконтроллере 8051.

**Листинг программ:**

**Задача 1:**

org 0h

jmp main

org 00Bh ; Вектор прерываний таймера 0

mov A, #0

movc A, @A+DPTR ; Смещаемся по таблице

mov P1, A ; Выводим значение таблицы

inc DPL ; Увеличиваем младшую часть DPTR, чтобы не выходить за границы таблицы (Например, с 0100h до 01FFh)

clr TF0 ; Сбрасываем флаг прерывания

reti

org 30h

main:

mov TMOD, #02h ; Таймер 0, режим 3, автоматическая перезагрузка

setb EA ; Включаем глобальные прерывания

setb ET0 ; и прерывания от таймера 0

mov TH0, #00FFh ; Настраиваем частоту вызова прерываний по таймеру

mov TL0, #00FFh

mov DPTR, #0100h ; Начальный адрес таблицы в DPTR

clr P0.7 ; Включаем ЦАП

setb TR0 ; Включаем таймер

wait:

jmp wait

org 0100h

db 0x80, 0x92, 0xa5, 0xb6, 0xc7, 0xd6, 0xe3, 0xee, 0xf6, 0xfc, 0xff, 0xff, 0xfd, 0xf8, 0xf0, 0xe6, 0xd9, 0xcb, 0xbb, 0xa9, 0x97, 0x84, 0x71, 0x5f, 0x4d, 0x3c, 0x2d, 0x1f, 0x14, 0xb, 0x4, 0x1, 0x0, 0x1, 0x6, 0xd, 0x17, 0x23, 0x30, 0x40, 0x51, 0x63, 0x76, 0x89, 0x9c, 0xae, 0xbf, 0xcf, 0xdc, 0xe8, 0xf2, 0xf9, 0xfe, 0xff, 0xfe, 0xfb, 0xf4, 0xeb, 0xe0, 0xd2, 0xc3, 0xb2, 0xa0, 0x8e, 0x7b, 0x68, 0x56, 0x44, 0x34, 0x26, 0x19, 0xf, 0x7, 0x2, 0x0, 0x0, 0x3, 0x9, 0x11, 0x1c, 0x29, 0x38, 0x49, 0x5a, 0x6d, 0x80, 0x92, 0xa5, 0xb6, 0xc7, 0xd6, 0xe3, 0xee, 0xf6, 0xfc, 0xff, 0xff, 0xfd, 0xf8, 0xf0, 0xe6, 0xd9, 0xcb, 0xbb, 0xa9, 0x97, 0x84, 0x71, 0x5f, 0x4d, 0x3c, 0x2d, 0x1f, 0x14, 0xb, 0x4, 0x1, 0x0, 0x1, 0x6, 0xd, 0x17, 0x23, 0x30, 0x40, 0x51, 0x63, 0x76, 0x89, 0x9c, 0xae, 0xbf, 0xcf, 0xdc, 0xe8, 0xf2, 0xf9, 0xfe, 0xff, 0xfe, 0xfb, 0xf4, 0xeb, 0xe0, 0xd2, 0xc3, 0xb2, 0xa0, 0x8e, 0x7b, 0x68, 0x56, 0x44, 0x34, 0x26, 0x19, 0xf, 0x7, 0x2, 0x0, 0x0, 0x3, 0x9, 0x11, 0x1c, 0x29, 0x38, 0x49, 0x5a, 0x6d, 0x7f, 0x92, 0xa5, 0xb6, 0xc7, 0xd6, 0xe3, 0xee, 0xf6, 0xfc, 0xff, 0xff, 0xfd, 0xf8, 0xf0, 0xe6, 0xd9, 0xcb, 0xbb, 0xa9, 0x97, 0x84, 0x71, 0x5f, 0x4d, 0x3c, 0x2d, 0x1f, 0x14, 0xb, 0x4, 0x1, 0x0, 0x1, 0x6, 0xd, 0x17, 0x23, 0x30, 0x40, 0x51, 0x63, 0x76, 0x89, 0x9c, 0xae, 0xbf, 0xcf, 0xdc, 0xe8, 0xf2, 0xf9, 0xfe, 0xff, 0xfe, 0xfb, 0xf4, 0xeb, 0xe0, 0xd2, 0xc3, 0xb2, 0xa0, 0x8e, 0x7b, 0x68, 0x56, 0x44, 0x34, 0x26, 0x19, 0xf, 0x7, 0x2, 0x0, 0x0, 0x3, 0x9, 0x11, 0x1c, 0x29, 0x38, 0x49, 0x5a, 0x6d, 0x7f

**Задача 2:**

org 0h

jmp main

org 0bh ; Вектор прерывания таймера 0

call adc ; Вызываем АЦП

call segments\_out ; Выводим на семисегментник

clr TF0 ; Опускаем флаг прерывания флага 0

reti

org 030h

main:

mov TMOD, #02h ; Таймер 0, режим 3, автоматическая перезагрузка

mov TH0, #200

mov TL0, #200

setb EA ; Включаем глобальные прерывания

setb ET0 ; и прерывания от таймера 0

setb TR0 ; Включаем таймер

wait:

jmp wait

; Преобразование аналогового сигнала в цифру

; R0, R1 - для последовательного приближения

adc:

clr P0.7 ; Включаем ЦАП

; Начальное возможное напряжение

mov R0, #0h ; 0000 0000

mov R1, #080h ; 1000 0000

mov A, R1

loop\_adc:

orl A, R0 ; Нынешнее значение + 1 новый бит

mov P1, A

jnb P3.7, continue ; Сохранить новый бит, если компаратор дал 1

mov R0, A

continue:

mov A, R1

rr A

clr ACC.0

mov R1, A

jnz loop\_adc

ret

; Вывод результата АЦП на семисегментник

; На вход: А - результат АЦП

; r2 - 3 число

; r1 - 2 число

; A - 1 число, в регистры не заносим

segments\_out:

; Получаем отдельные цифры результата АЦП

mov A, R0

mov B, #10

div AB

mov R2, B

mov B, #10

div AB

mov R1, B

; Выводим 1 число на семисегментник

clr p0.7

call to\_segment ; Преобразуем цифру в код для семисегментника

clr p3.3

setb p3.4

mov P1, R6

; Выводим 2 число на семисегментник

mov A, R1

setb p0.7

call to\_segment ; Преобразуем цифру в код для семисегментника

setb p3.3

clr p3.4

mov P1, R6

; Выводим 3 число на семисегментник

mov A, R2

setb p0.7

call to\_segment ; Преобразуем цифру в код для семисегментника

clr p3.3

clr p3.4

mov P1, R6

setb P0.7

ret

; Преобразование числа в код для семисегментного индикатора

; Вход: А - преобразуемое число

; Выход: R6 - код для семисегментника

; 0200h - адрес начала таблицы значений

to\_segment:

mov r7, a ; Как же все таки хочется push A...

mov DPTR, #0200h ; Указываем на начало таблицы

MOVC A,@A+DPTR ; Сдвигаем на А по таблице

cpl A ; Инвертируем

mov R6, A ; Возвращаем значение

mov a, r7 ; Восстанавливаем значение А

ret

; Таблица с кодами чисел для семисегментника

org 0200h

db 3Fh, 06H, 5Bh, 4Fh, 66h, 6Dh, 7Dh, 07h, 7Fh, 6Fh