Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6**

**”Таймеры-счетчики. Прерывания. Таймер-счетчик 0”**

**Вариант №7**

|  |  |  |  |
| --- | --- | --- | --- |
| Преподаватель |  | подпись, дата | Яблонский А.П. |
| Студент КИ19-09Б | 031940750  номер зач. книжки | подпись, дата | Кудрявцев Н.М. |

Красноярск 2022 г.

**ВВЕДЕНИЕ**

**Цель работы**: изучить операции по работе с таймером/счетчиком 0, работа с подпрограммами, выполнение обработки прерываний.

**Задание**: модифицировать программу, полученную в результате выполнения лабораторной работы №2, используя подпрограммы. Реализовать 2 режима управления светодиодом ШИМ и ЧИМ. Параметры модулированного сигнала изменяются на каждом такте таймера. Изменение параметров сигнала формируют пилообразный или треугольный сигнал в соответствии с вариантом.

При нажатии на кнопку Х осуществляется выбор режима:

1. Поочерёдно выводит данные из массива С на светодиодные индикаторы;

2. Запуск генерации ШИМ сигнала;

3. Запуск генерации ЧИМ сигнала.

Номер выбранного режима отображается на светодиодных индикаторах.

По нажатию на кнопку Y осуществляется запуск выбранного режима, при повторном нажатии осуществляется остановка выбранного режима.

Обработка нажатия кнопок осуществляется по заданным внешним прерываниям. Каждый из режимов выполняется в виде отдельной подпрограммы. Вывод массива С осуществляется по событиям от таймера.

Таблица 1 ─ Вариант на работу.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Вариант** | **Кнопка X** | **Кнопка Y** | **ШИМ** | **ЧИМ** |
| **7** | INT0 | INT1 | Пилообразный | Треугольный |

**Команды:**

Справка по Ассемблеру для Atmel AVR:

<https://dfe.karelia.ru/koi/posob/avrlab/avrasm-rus.htm>

sei ─ Установить флаг прерываний;

sbiw z, 1 ─ Вычесть константу **1** из слова **z**;

adiw z, 1 ─ Сложить константу **1** и слово **z**;

out TCCR0, temp ─ Запись **temp** в порт **TCCR0**;

subi index, 2 ─ Вычитание константы **index – 2**;

brcc PREVIOUS\_NOT\_LESS ─ Перейти, если перенос очищен;

brlo SORT ─ Перейти, если меньше;

.undef a\_elem ─ Обнулить переменную

.set is\_running\_bit\_index = 0 ─ Присваивает имени некоторое значение;

sbrs flags, x ─ Пропустить, если бит **x** в регистре **flags** установлен;

sbrc flags, x ─ Пропустить, если бит **x** в регистре **flags** очищен;

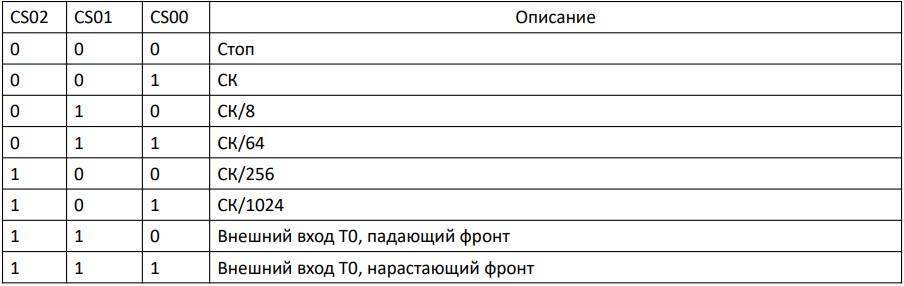
breq RESTART ─ Перейти, если равно

ldi temp, (1<<CS02)|(1<<CS00) ─ Работа с Timer0 (предделитель = 1024);

ldi temp, (1<<OCIE0)|(1<<TOIE0) ─ Разрешить прерывание по сравнению и переполнению;

**Управляющий регистр таймером/счетчиком 0 (TCCR0)**:

Биты 2-0 (**CS02**, **CS01** и **CS00** соотв.) определяют источник тактирования T/C0;

****

**Регистр маски прерывания таймеров/счетчиков (TIMSK):**

**OCIE0** (бит 1): Разрешение прерывание по сравнению;

**TOIE0** (бит 0): Разрешение прерываний по переполнению счета;

**Регистр флагов таймеров/счетчиков (TIFR):**

**OCF0** (бит 1): Флаг о совпадении при сравнении;

**TOV0** (бит 0): Флаг о переполнении;

**Регистр данных таймера/счетчика 0 (TCNT0)**;

**Регистр сравнения выхода таймера/счетчика 0 (OCR0)**;

**Ход работы:**

.def count = r16 ; счетчик

.def count2 = r17 ; второй счетчик

.def a\_elem = r18 ; текущий элемент массива A

.def b\_elem = r19 ; текущий элемент массива B

.def c\_length = r20 ; длина заполненной части массива C

.def previous\_elem = r21

.def current\_elem = r22

.def index = r23

.set start\_max = 255

.set sizeAB = 10 ; размер массивов A и B

.dseg ; указатель на сегмент данных

.org 0x60 ; начальный адрес

mas\_a: .byte sizeAB

mas\_b: .byte sizeAB

.org 0x7b ; выделим под массив C 10 байт

mas\_c: .byte 10

;------------------------------------------------------;

.macro load\_x

ldi xl, low(@0) ; X

ldi xh, high(@0)

.endm

.macro load\_y

ldi yl, low(@0) ; Y

ldi yh, high(@0)

.endm

.macro load\_z

ldi zl, low(@0) ; Z

ldi zh, high(@0)

.endm

;------------------------------------------------------;

.cseg ; указатель на сегмент кода

.org $000 rjmp INIT

.org INT0addr rjmp P\_INT0

.org INT1addr rjmp P\_INT1

.org OC0addr rjmp T\_OC\_SECOND

.org OVF0addr rjmp T\_OWF\_Select

INIT:

load\_x mas\_a

ldi a\_elem, 4 ; переменная для заполнения массива А

ldi count, sizeAB

MASA:

st x+, a\_elem

inc a\_elem

dec count

brne MASA

load\_y mas\_b

ldi b\_elem, 16

ldi count, sizeAB – 5 ; пропускаем парк элементов чтобы массивы отличались

MASB: ; заполняем массив B

st y+, b\_elem

dec b\_elem

dec count

brne MASB

;------------------------------------------------------;

; Формируем регистровые пары

load\_x mas\_a

load\_y mas\_b

load\_z mas\_c

ldi c\_length, 0

ldi count, sizeAB

A\_LOOP:

ld a\_elem, x+

ldi count2, sizeAB

load\_y mas\_b

B\_LOOP:

ld b\_elem, y+

cp a\_elem, b\_elem

brne NOT\_EQUAL

st z+, a\_elem

inc c\_length

NOT\_EQUAL:

dec count2

brne B\_LOOP

dec count

brne A\_LOOP

;------------------------------------------------------;

; Сортировка убывание

load\_z mas\_c

SORT:

ld previous\_elem, z+

ld current\_elem, z

inc index

cp previous\_elem, current\_elem

brsh PREVIOUS\_NOT\_LESS ; если previous\_elem >= current\_elem

PREVIOUS\_LESS:

st z, previous\_elem ; поменять значения местами в памяти

st -z, current\_elem

sbiw z, 1 ; декремент указателя на 1

subi index, 2

brcc PREVIOUS\_NOT\_LESS ; если перенос установлен (индекс меньше 0), то обнулить

ldi index, 0

adiw z, 1 ; инкремент указателя на 1

PREVIOUS\_NOT\_LESS:

cp index, c\_length

brlo SORT ; если индекс меньше длинны, то новая итерация

.undef a\_elem

.undef b\_elem

.undef previous\_elem

.undef current\_elem

;------------------------------------------------------;

.def mode = r18

.def flags = r19

.def compare\_number = r21

.def start\_number = r22

.def temp = r24

.def temp2 = r25

.set is\_running\_bit\_index = 0

.set is\_decreasing\_bit\_index = 1

.set is\_running\_flag = 1 << is\_running\_bit\_index

.set is\_decreasing\_flag = 1 << is\_decreasing\_bit\_index

clr temp ; выкл таймер

out TCCR0, temp

ldi temp, low(RAMEND) ; настройка стека

out SPL, temp

ldi temp, high(RAMEND)

out SPH, temp

ser temp

out DDRB, temp ; настройка светодиодов

out PORTB, temp

clr temp

out DDRD, temp ; настройка кнопок

sei ; разрешить прерывания

ldi temp, 0b00001111 ; прерывания по нарастающему фронту для INT0 и INT1

out MCUCR, temp

ldi temp, 0b11000000 ; разрешить прерывания INT0 и INT1

out GICR, temp

clr flags

ldi mode, 1 ; начальный номер режима

out PORTB, mode

MAIN:

out PORTB, mode

sbrs flags, is\_running\_bit\_index

rjmp main

sbrc mode, 0

call FIRST

sbrc mode, 1

call SECOND

sbrc mode, 2

call THIRD

rjmp MAIN

FIRST:

ldi temp, (1<<TOIE0) ; вкл прерывания для таймера 0

out TIMSK, temp

mov count, c\_length

load\_z mas\_c

ldi temp,(1<<CS02)|(1<<CS00) ; предделитель

out TCCR0,temp

loop\_1:

sbrc flags, is\_running\_bit\_index ; выключение

rjmp loop\_1

clr temp ; выкл таймер

out TCCR0, temp

ret

;------------------------------------------------------;

; Пилообразный ШИМ

SECOND:

ldi temp, 0x01 ; загрузка числа сравнения

out OCR0, temp

ldi temp, (1<<OCIE0)|(1<<TOIE0) ; разрешить прерывание по сравнению и переполнению

out TIMSK, temp

ldi temp, (1<<CS01) | (0<<CS00) ; предделитель

out TCCR0, temp ; старт таймера

clr count ; начальное значение диодов

clr compare\_number ; число сравнения

loop\_2:

sbrc flags, is\_running\_bit\_index ; выключение

rjmp loop\_2

clr temp ; выкл таймер

out TCCR0, temp

ret

; Треугольный ЧИМ

THIRD:

clr count ; начальное значение диодов

clr start\_number ; начальное число отсчёта

clr temp

out TCNT0, temp

ldi temp, (1<<TOIE0) ; разрешить прерывание по переполнению

out TIMSK, temp

ldi temp, (1<<CS02) | (0<<CS00) ; предделитель

out TCCR0, temp ; старт таймера

loop\_3:

sbrc flags, is\_running\_bit\_index ; выключение

rjmp loop\_3

clr temp ; выкл таймер

out TCCR0, temp

ret

P\_INT0: ; кнопка X выбор режима

lsl mode

andi flags, ~is\_running\_flag

sbrc mode, 3

ldi mode, 1

reti

P\_INT1: ; кнопка Y запуск/остановка режима

; если запущен, то выключить и наоборот

ori flags, is\_running\_flag

reti

T\_OWF\_Select:

sbrc mode, 0

rjmp T\_OWF\_FIRST

sbrc mode, 1

rjmp T\_OWF\_SECOND

sbrc mode, 2

rjmp T\_OWF\_THIRD

T\_OWF\_FIRST:

ld temp, z+ ; загрузить элемент массива С

com temp

out PORTB, temp ; вывести его на светодиоды

dec count

breq RESTART

reti

; если конец массива, то начинаем заново

RESTART:

mov count, c\_length

load\_z mas\_c

reti

T\_OC\_SECOND:

com count ; включение светодиодов

out PORTB, count

reti

; Пилообразный ШИМ

T\_OWF\_SECOND:

com count ; выключение светодиодов

out PORTB, count

dec compare\_number

out OCR0, compare\_number

reti

; Треугольный ЧИМ

T\_OWF\_THIRD:

com count ; включение/выключение светодиодов

out PORTB, count

cpi start\_number, 0x00

in temp2, SREG

sbrc temp2, SREG\_Z ; пропустить, если не равно 0

sbr flags, is\_decreasing\_flag

cpi start\_number, 0xff

in temp2, SREG

sbrc temp2, SREG\_Z ; пропустить, если не равно 255

cbr flags, is\_decreasing\_flag

sbrs flags, is\_decreasing\_bit\_index

dec start\_number

sbrc flags, is\_decreasing\_bit\_index

inc start\_number

out TCNT0, start\_number

reti

// Бесконечный цикл

inf:

rjmp inf