Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**

**”Встроенный контроллер прерываний”**

**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 обработки внешних прерывания, регистры настройки и управления внешних прерываний.

**Задание**: необходимо разработать программу, в которой реализовать обработку внешних прерываний. Запрограммировать обработку внешнего прерывания на конкретное действие в соответствии с вариантом задания.

Таблица 1 – Вариант задания.

|  |  |  |  |
| --- | --- | --- | --- |
| **Вариант** | **Действие** | **Фронт генерации прерывания INT0** | **Фронт генерации прерывания INT1** |
| 7 | Изменение скорости моргания индикаторов: INT1 – быстрое моргание, INT0 – медленное | Нарастающий | Падающий |

**Команды:**

Справка по Ассемблеру для Atmel AVR:

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

**.def temp = r16** ─ Назначить регистру **r16** символическое имя “**temp**”;

**.cseg** ─ Определяет начало программного сегмента;

**.org 0x00** ─ Переходим на ячейку памяти по адресу, равному **0x00**;

**clr temp** ─ Очистить регистр;

**ser temp** ─ Установить регистр;

**sei** ─ Установить флаг прерываний;

**rcall** ─ Относительный вызов подпрограммы;

**ret** ─ Возврат из подпрограммы;

**reti** ─ Возврат из прерывания;

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

**ldi temp, 0x3c** ─ Записываем константу **0xC3** в регистр temp;

**inc temp** ─ Инкрементировать значение регистра **temp**;

**dec** **temp** ─ Декрементировать значение регистра **temp**;

**brne dly** ─ Перейти, если не равно;

**rjmp loop** ─ Относительный переход;

**com temp** ─ Побитная Инверсия;

Адрес вершины стека хранится в константе **RAMEND**, которая определена в подключаемом файле для данной модели микроконтроллера.

Младший байт константы RAMEND надо загрузить в регистр **SPL** (Stack Pointer Low), а старший - в регистр **SPH** (Stack Pointer High). Это установит указатель стека на конец ОЗУ. В регистры **SPL** и **SPH** необходимо загружать адрес вершины стека (адрес верхней границы ОЗУ).

**DDRX** ─ Это регистр направления порта (Регистр управления). Порт в конкретный момент времени может быть либо входом, либо выходом (но для состояния битов **PIN** это значения не имеет. Читать из PIN реальное значение можно всегда).

* DDRxy=0 ─ вывод работает как ВХОД.
* DDRxy=1 ─ вывод работает на ВЫХОД.

**PORTX** ─ Режим управления состоянием вывода (Регистр выходных данных). Когда мы настраиваем вывод на вход, то от **PORT** зависит тип входа;

1. Когда ножка настроена на **выход**, то значение соответствующего бита в регистре PORTx определяет состояние вывода:

* Если **PORTxy=1,** то на выводе лог1;
* Если **PORTxy=0,** то на выводе лог0.

1. Когда ножка настроена на **вход**, то…

* Если **PORTxy=0**, то вывод в режиме **Hi-Z**;
* Если **PORTxy=1,** то вывод в режиме **PullUp** с подтяжкой резистором в 100к до питания.

**MCUCR** (регистр управления микроконтроллером) ─ конфигурационный регистр для выходов INT0 и INT1;

Основной регистр управления прерываниями – **GICR.**

Регистр **GICR** управляет маскированием внешних прерываний и размещением таблицы векторов прерываний.

Формат регистра GICR:

* бит 7 — INT1, маска внешнего прерывания INT1;
* бит 6 — INT0, маска внешнего прерывания INT0;

**Ход работы:**

Ссылка на задание: [тут](https://e.sfu-kras.ru/pluginfile.php/2646532/mod_resource/content/2/Lab%204%20%D0%9F%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F.pdf);

Ссылки на лекцию: [тут](https://e.sfu-kras.ru/pluginfile.php/2646355/mod_resource/content/2/%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D1%8F%205%20%D0%9F%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F%20%D0%B8%20%D0%9F%D0%94%D0%9F.pdf).

//.include "m32adef.inc"

.def temp = r16

.def diods = r17

.def interruption\_count = r18

.cseg

.org 0x00 rjmp INIT

.org 0x02 rjmp P\_INT0 ; настраиваем вектора прерываний

.org 0x04 rjmp P\_INT1

INIT:

clr interruption\_count ; очищаем количество прерываний

ldi temp, low(RAMEND) ; настраиваем стек

out SPL, temp

ldi temp, high(RAMEND)

out SPH, temp

ser temp ; устанавливаем temp в единицы

out DDRB, temp ; настраиваем порт B на вывод

out PORTB, temp

sei

ldi temp, 0b00001011 ; настраиваем прерывания по нарастающиму и падающему фронту

out MCUCR, temp

ldi temp, 0b11000000 ; разрешаем внешние прерывания 0 и 1

out GICR, temp

ldi diods, 0xff ; гасим светодиоды

ldi r21,120 ; задаем начальную задержку для моргания

MAIN:

COM diods

out PORTB, diods ; вывод на светодиоды

rcall Delay

MOV r20, r21

rjmp MAIN

Delay:

m20:ldi r30,255

m10:dec r30

brne m10

dec r20

brne m20

ret

P\_INT0:

ldi r21,255

reti

P\_INT1:

ldi r21,50

reti