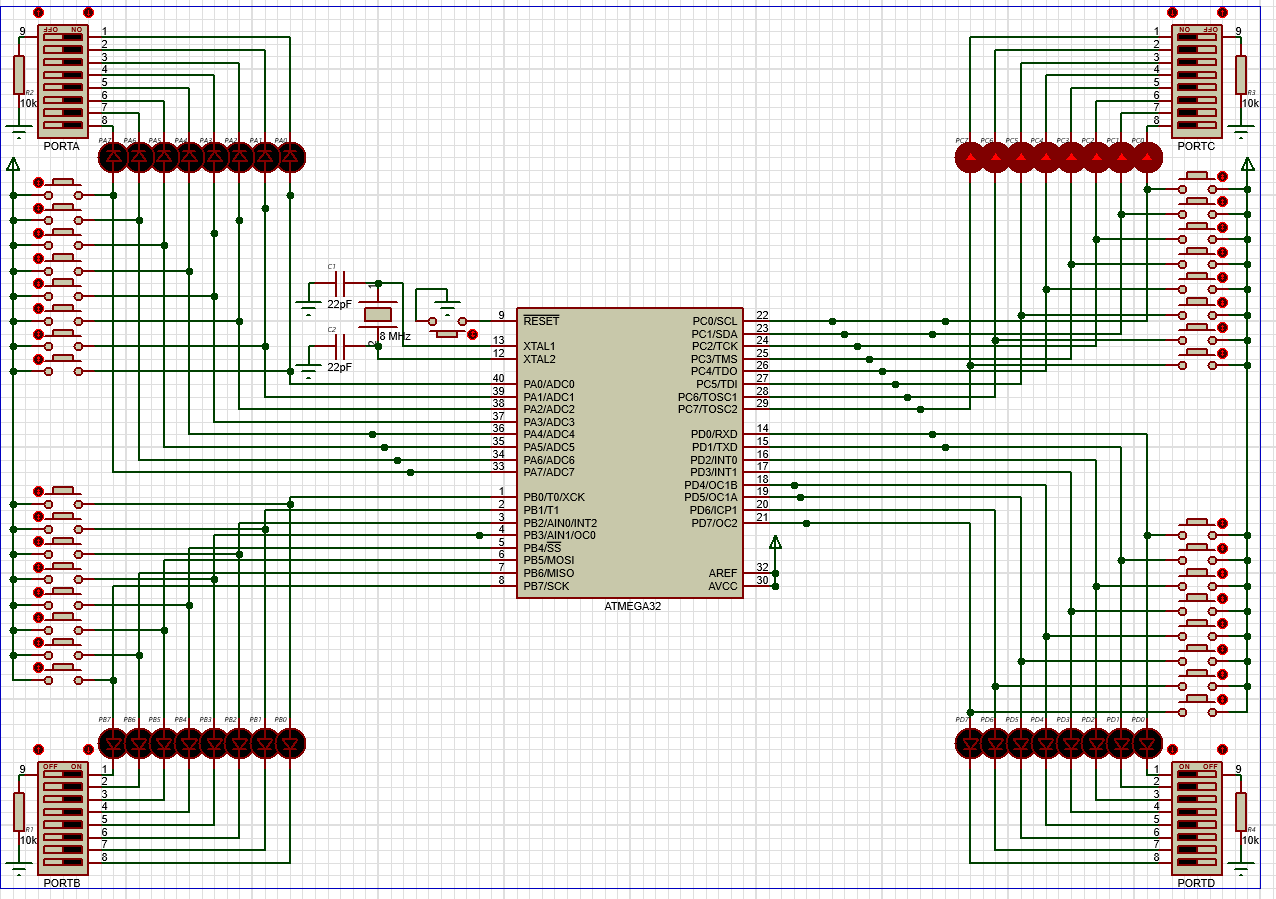
# Формулировка задания

Регистры PORTA–PORTC выполняют роль новогодней гирлянды. Есть следующие 3 режима работы:

* на каждом регистре два состояния – вывод чисел 0xFF и 0x00, смена состояний с частотой x;
* на каждом регистре два состояния – вывод чисел 0xAA и 0x55, смена состояний с частотой x;
* три состояния, при которых на одном из регистров PORTA - PORTC выводится 0xFF, на остальных 0x00, переключение состояний с частотой x.

Регистры PD4-PD5 отображают номер режима работы (1-3), регистры PD6-PD7 отображают номер состояния в конкретном режиме (1-2 / 1-3), регистры PD0-PD1 отображают номер элемента x в множестве частот. Смена режима работы должна производиться циклически с помощью кнопки PD2 (прерывание INT0), при этом номер активного состояния должен сохраняться. С помощью кнопки PD3 (прерывание INT1) циклически изменяется величина x в следующем множестве {0,25 Гц; 0,5 Гц; 1 Гц}. Изменения должны отображаться на регистре PORTD и сохраняться во внутренний EEPROM МК в момент нажатия на кнопки PD2 и PD3. Исходное положение: первый режим и первое состояние (0xFF на регистрах PORTA-PORTC), величина x извлекается из EEPROM.

# Схема лабораторной установки



# Блок-схема алгоритма работы программы

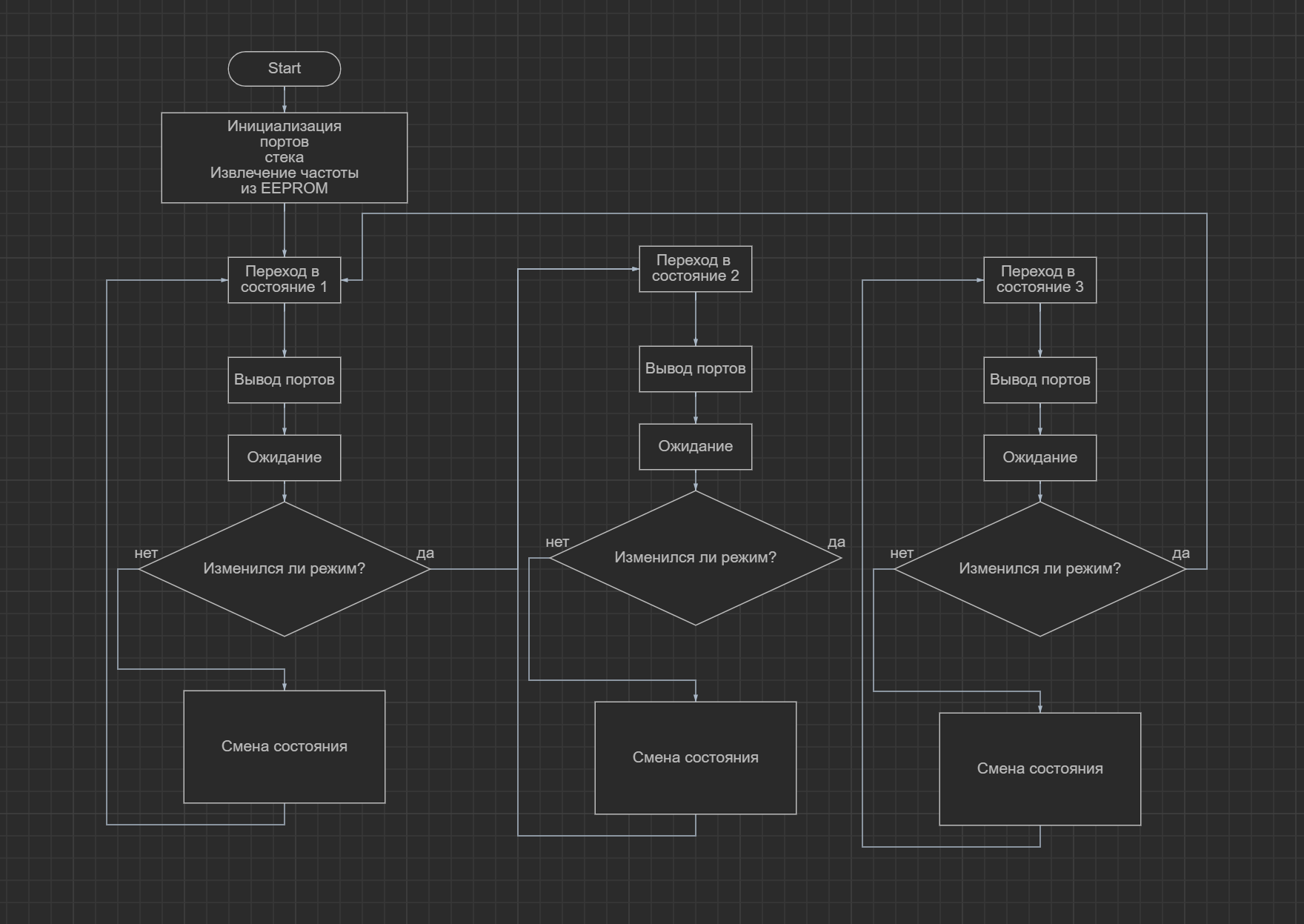


Рисунок 1 – Блок-схема алгоритма работы программы

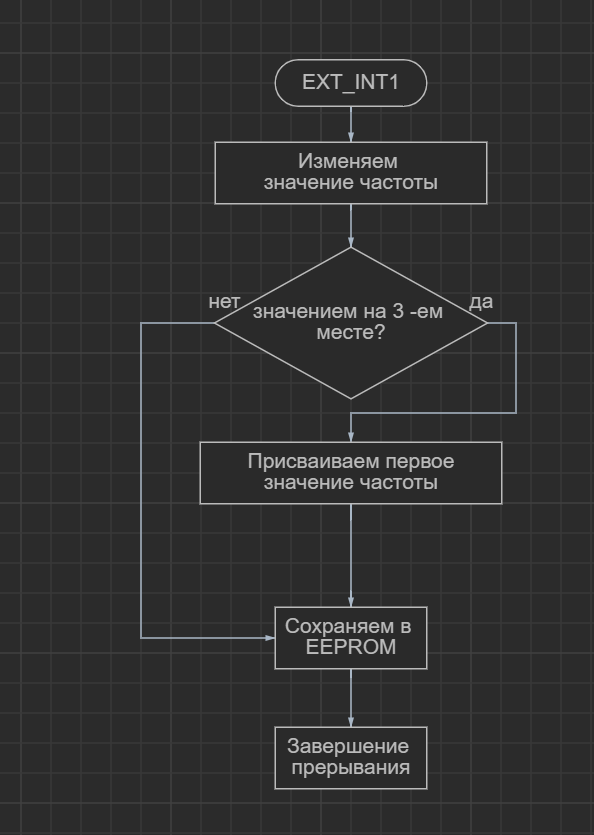


Рисунок 2 – Блок-схема алгоритма работы прерывания “EXT\_INT1”

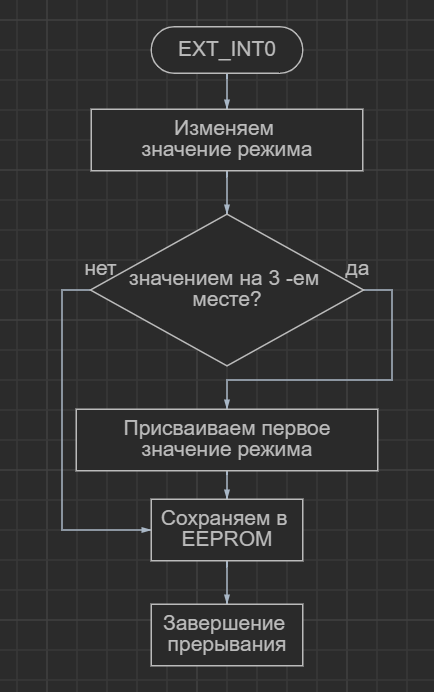


Рисунок 3 – Блок-схема алгоритма работы прерывания “EXT\_INT0”

# Временные диаграммы логических сигналов на портах МК (фрагмент)

# Алгоритм выполнения задействованных команд (конструкций) ассемблера

OUT P, Rr – запись содержимого Rr в регистр ввода-вывода P

LDI Rd, K – загрузка константы

CPI Rd, K – сравнение регистра с константой

BREQ k – переход если равно (флаг нулевого значения Z=1)

BRNE k – переход если неравно (флаг нулевого значения Z=0)

RJMP k – относительный безусловный переход

JMP k – прямой безусловный переход

RCALL k – относительный вызов подпрограммы

IN Rd, P – считывание значения порта в регистр

OR Rd, Rr – логическое ИЛИ двух регистров

RET – возврат из подпрограммы

SEI – глобальное разрешение прерываний

ADD Rd, Rr – сложение двух регистров

MOV Rd, Rr – копирование регистра Rr в Rd

SUBI Rd, K – вычитание константы из регистра

INC Rd – инкремент

RETI – возврат из обработчика прерывания (установка флага I)

NOP – пустая команда

# Результаты работы

Была создана программа для реализации гирлянды. Она имеет три состояния, переключения между которыми происходят с частотой X Гц. Регистры PD4-PD5 отображают номер режима работы (1-3), регистры PD6-PD7 отображают номер состояния в конкретном режиме (1-2 / 1-3). Смена режима работы производиться циклически с помощью кнопок PD2 (прерывание INT0, смена частоты) и PD3 (прерывание INT1, смена режима), Исходное частота извлекается из EEPROM.

# Ответы на контрольные вопросы

1. Как реализуется подсистема прерываний в микроконтроллере AVR?

Прерывание прекращает нормальный ход программы для выполнения приоритетной задачи, определяемой внутренним или внешним событием микроконтроллера. При возникновении прерывания микроконтроллер сохраняет в стеке содержимое счетчика команд PC и загружает в него адрес соответствующего вектора прерывания. По этому адресу, как правило, находится команда безусловного перехода к подпрограмме обработки прерывания. Последней командой подпрограммы обработки прерывания должна быть команда RETI, которая обеспечит возврат в основную программу и восстановление предварительно сохраненного счетчика команд.

Младшие адреса памяти программ отведены под таблицу векторов прерывания. Каждому прерыванию соответствует адрес в этой таблице. Положение вектора в таблице также определяет и приоритет соответствующего прерывания: чем меньше адрес, тем выше приоритет прерывания.

2. Как программно разрешить или запретить выполнение прерываний?

Независимо от того активированы ли определённые прерывания, микроконтроллер не начнёт обработку этих прерываний, пока в «бит всеобщего разрешения прерываний» (Global Interrupt Enable bit в регистре состояния SREG) не будет записана логическая единица. Чтобы запретить выполнение прерываний в этот бит надо поставить логический ноль.

3. Какие источники прерываний есть в микроконтроллерах AVR?

RESET – сигнал сброса

INT0 - Внешний запрос на прерывание по входу INT0

INT1 - Внешний запрос на прерывание по входу INT1

INT2 - Внешний запрос на прерывание по входу INT2

TIMER2\_COMP - Совпадение с регистром сравнения таймера T/C2

TIMER2\_OVF - Переполнение счётчика T/C2

TIMER1\_CAPT - Захват по таймеру T/C1

TIMER1\_COMPA - Совпадение с регистром сравнения A таймера T/C1

TIMER1\_COMPB - Совпадение с регистром сравнения B таймера T/C1

TIMER1\_OVF - Переполнение счётчика T/C1

TIMER0\_COMP - Совпадение с регистром сравнения таймера T/C0

TIMER0\_OVF - Переполнение счётчика T/C0

SPI\_STC - Передача данных по интерфейсу SPI завершена

UART\_RXC - Приём данных приёмопередатчиком UART завершён

UART\_UDRE - Регистр данных UART пуст

UART\_TXC - Передача данных приёмопередатчиком UART завершена

ADC - Завершено преобразование АЦП

EE\_RDY - EEPROM готов

ANA\_COMP - Прерывание от аналогового компаратора

TWI - Прерывание от интерфейса I2C

SPM\_RDY - Запись программной памяти (Flash) готова

4. Как настраиваются внешние прерывания?

За управление внешними прерываниями в ATmega32 отвечают 4 регистра:

GICR (он же GIMSK) – запрет/разрешение прерываний по сигналам на входах INT0, INT1;

MCUCR – выбор условия срабатывания прерываний int0 и int1;

GIFR – управление внешними прерываниями;

Регистр GICR:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INT1 | INT0 | - | - | - | - | - | - |

INTx=1: прерывания по сигналу на входы INTx разрешены.

Регистр MCUCR:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SE | SM2 | SM1 | SM0 | ISC11 | ISC10 | ISC01 | ISC00 |