|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

|  |  |
| --- | --- |
| Факультет | «Информатика и системы управления» (ИУ) |
| Кафедра | «Информационная безопасность» (ИУ8) |

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

**АППАРАТНЫЕ СРЕДСТВА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ**

«Битовый процессор 2»

|  |  |
| --- | --- |
| Преподаватель: Рафиков А.Г. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |
| Студент: Девяткин Е.Д., группа ИУ8-74 (4 курс) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

**Содержание**

[Цель работы 3](#_Toc208567293)

[Теоретическая часть 3](#_Toc208567294)

[Выполнение работы 6](#_Toc208567295)

[Задание 1 6](#_Toc208567296)

[Задание 2 – Битовые логические инструкции. 8](#_Toc208567297)

[Задание 3 – Тест битов. 9](#_Toc208567298)

[Задание 4 – Байтовые логические инструкции. 11](#_Toc208567299)

# Цель работы

Собрать в Proteus модель центра управления сигналами автомобиля, используя МК i8051. КЗ должен быть равен 90%. В качестве индикации должен быть выбран дисплей.

# Теоретическая часть

В программе используется таймер 0 микроконтроллера в режиме 1   
(16-битный), настроенный на генерацию прерываний с фиксированным периодом. Прерывание таймера – это автоматический вызов подпрограммы обработки (в данном случае TL0\_PROC) при переполнении таймера, что позволяет выполнять периодические действия без блокировки основной программы.

При инициализации (подпрограмма INIT) устанавливаются разрешения глобальных прерываний (EA) и прерываний от таймера 0 (ET0), а также загружаются начальные значения в регистры TL0 и TH0 для задания периода:

INIT:

MOV TMOD, #00000001B

MOV TL0, #0

MOV TH0, #-16

MOV SUB\_DIV, #244

MOV R5, #61

SETB ET0

SETB EA

SETB TR0

CLR RS

CLR E

CLR F0

MOV R7, #1

MOV F, #255

MOV R6, #48

RET

Значение -16 в TH0 задаёт начальное счётное значение таймера, при котором он переполняется каждые 16 машинных циклов, обеспечивая высокую частоту прерываний (~15,625 кГц при частоте кварца 12 МГц).

Для снижения частоты мигания используется программный делитель SUB\_DIV, который декрементируется при каждом прерывании и сбрасывается после достижения нуля:

TL0\_PROC:

MOV TL0, #0

MOV TH0, #-16 ; перезагрузка таймера

DJNZ SUB\_DIV, T0\_SERV ; уменьшить SUB\_DIV, если не ноль — перейти к T0\_SERV

MOV SUB\_DIV, #244 ; сбросить делитель

Когда SUB\_DIV достигает нуля, его биты используются для формирования двух условных уровней частоты: бит 0 (SUB\_DIV.0) обозначает высокую частоту (HI\_FREQ), а бит 7 (SUB\_DIV.7) – низкую (LO\_FREQ).

По условию скважность сигнала должна быть 30% (сигнал включён 30% времени, выключен – 70%). Для этого в программе анализируются младшие три бита регистра SUB\_DIV (биты 2, 1 и 0). Таблица истинности показывает, при каких комбинациях выход должен быть включён:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SUB\_DIV.2 | SUB\_DIV.1 | SUB\_DIV.0 | 1 \* 0 | (1 \* 0) + 2 | 60% |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

Реализация в программе:

MOV C, SUB\_DIV.0

ORL C, SUB\_DIV.1

ANL C, SUB\_DIV.2

MOV DIM, C

В обработчике прерывания TL0\_PROC сначала таймеру снова задают начальное значение, чтобы он продолжал отсчитывать одинаковые промежутки времени. Потом уменьшают счётчик SUB\_DIV, и по состоянию его битов определяют, должен ли сейчас гореть или гаснуть сигнал – этот результат временно сохраняют во флаге DIM.

Основной цикл программы пуст (SJMP $), так как вся логика управления световыми индикаторами реализована в фоновом режиме через прерывания таймера.

# Выполнение работы

## **Задание 1**

Для вывода значения Q на светодиод потребуется установить токоограничивающий резистор. Вычислим его сопротивление при схеме включения диода, изображенной на рисунке 3.

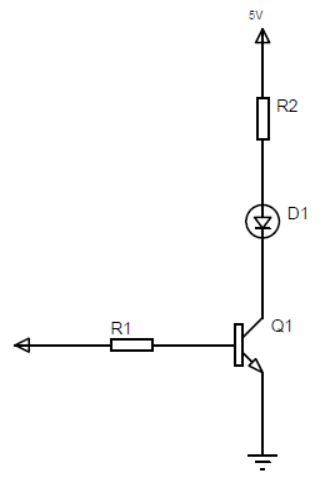


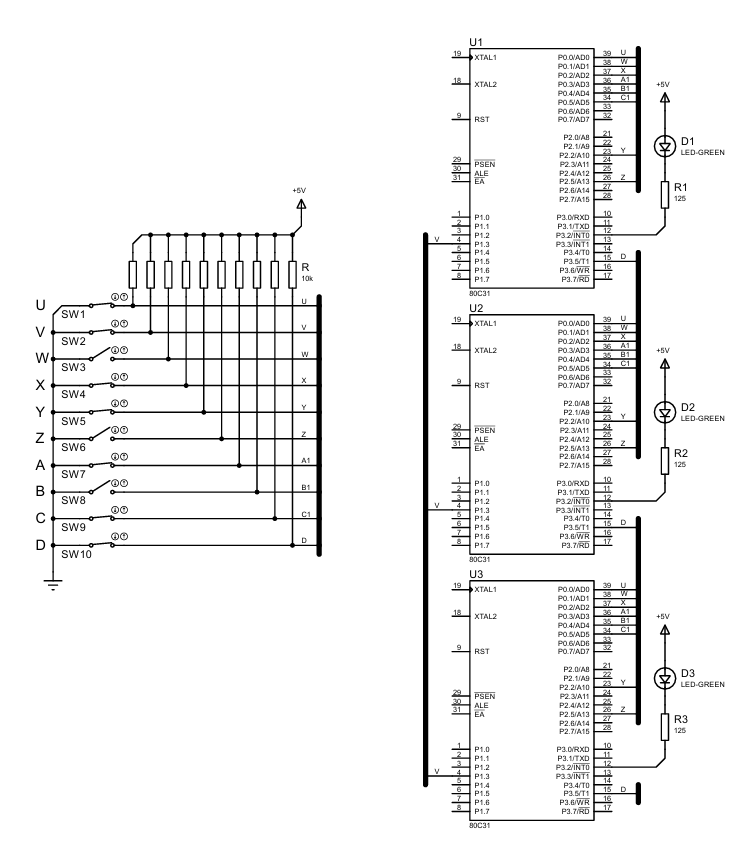
Рис. 3 – Схема включения диода.

Напряжение питания , напряжение на транзисторе , напряжение на диоде , силу тока примем равной . Значение сопротивления равно .

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Q | U | V | W | X | Y | Z | A | B | C | D |
| P3.2 | 20H.3 | P1.3 | 22H.0 | 28H.2 | P2.2 | P2.5 | 28H.5 | 21H.4 | 25H.0 | P3.5 |

Q=(WV+X\*/Z)+(A+C)(U+B)

Схема



Диод загорается, когда на входе 0. Чтобы диод загорался в программах считается /Q. Поэтому когда Q=1, диод будет гореть.

## **Задание 2 – Битовые логические инструкции.**

Суть метода: используя логические выражения, вычисляется функция по действиям. В этом методе используется бит переноса C.

; $NOMOD51

; $INCLUDE (8051.MCU)

Q BIT P3.2

Uq BIT 20H.3

Vq BIT P1.3

Wq BIT 22H.0

Xq BIT 28H.2

Yq BIT P2.2

Zq BIT P2.5

Aq BIT 28H.5

Bq BIT 21H.4

Cq BIT 25H.0

Dq BIT P3.5

U\_ BIT P0.0

W\_ BIT P0.1

X\_ BIT P0.2

A\_ BIT P0.3

B\_ BIT P0.4

C\_ BIT P0.5

F1 BIT 20H.0

org 0h

MAIN:

MOV C, Vq

ANL C, W\_

MOV F0, C

MOV C, X\_

ANL C, /Zq

ORL C, F0

MOV F0, C

MOV C, A\_

ORL C, C\_

MOV F1, C

MOV C, U\_

ORL C, B\_

ANL C, F1

ORL C, F0

MOV Q, C

CPL Q

END

## **Задание 3 – Тест битов.**

Суть метода: строится блок-схема по данной функции, которая показывает переходы исходя из равенства переменной нулю или единице. По нарисованной блок-схеме пишется код с условными переходами. JNB – переход при условии, что переменная равна 0. JB – переход при условии, что переменная равна 1.

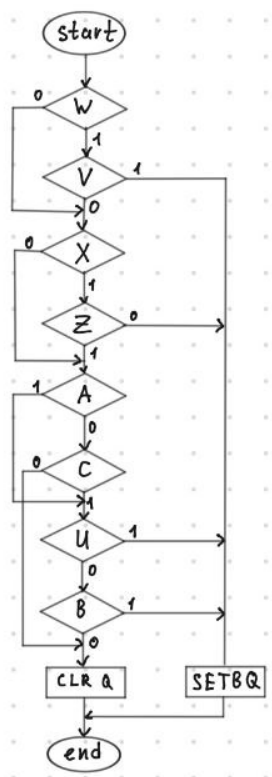


Рис. 4 – Блок-схема.

; $NOMOD51

; $INCLUDE (8051.MCU)

Q BIT P3.2

Uq BIT 20H.3

Vq BIT P1.3

Wq BIT 22H.0

Xq BIT 28H.2

Yq BIT P2.2

Zq BIT P2.5

Aq BIT 28H.5

Bq BIT 21H.4

Cq BIT 25H.0

Dq BIT P3.5

U\_ BIT P0.0

W\_ BIT P0.1

X\_ BIT P0.2

A\_ BIT P0.3

B\_ BIT P0.4

C\_ BIT P0.5

org 0h

MAIN:

JNB W\_, TEST\_X ;W=0

JB W\_, TEST\_V ;W=1

TEST\_V: JNB Vq, TEST\_X

JB Vq, SET\_Q

TEST\_X: JNB X\_, TEST\_A

JB X\_, TEST\_Z

TEST\_Z: JNB Zq, SET\_Q

JB Zq, TEST\_A

TEST\_A: JNB A\_, TEST\_C

JB A\_, TEST\_U

TEST\_C: JNB C\_, CLR\_Q

JB C\_, TEST\_U

TEST\_U: JNB U\_, TEST\_B

JB U\_, SET\_Q

TEST\_B: JNB B\_, CLR\_Q

JB B\_, SET\_Q

SET\_Q: ; Установить Q в 1 (но делаем обратное, т.к. при 0 включается диод)

CLR Q

JMP MAIN

CLR\_Q:

SETB Q

JMP MAIN

END

## **Задание 4 – Байтовые логические инструкции.**

Суть метода: используется байтовые команды микроконтроллера. Для написания кода также следует ориентироваться по блок-схеме. Чтобы проверить состояние нужного бита, байт загружается в аккумулятор A, и с помощью операции ANL A, #mask (логическое И с маской) выделяется нужный бит. Затем проверяется состояние аккумулятора (флаг Z в PSW) командами условного перехода JZ/JNZ. По сути, это аналог второго метода.

; $NOMOD51

; $INCLUDE (8051.MCU)

Q BIT P3.2

Uq BIT 20H.3

Vq BIT P1.3

Wq BIT 22H.0

Xq BIT 28H.2

Yq BIT P2.2

Zq BIT P2.5

Aq BIT 28H.5

Bq BIT 21H.4

Cq BIT 25H.0

Dq BIT P3.5

U\_ BIT P0.0

W\_ BIT P0.1

X\_ BIT P0.2

A\_ BIT P0.3

B\_ BIT P0.4

C\_ BIT P0.5

org 0h

MAIN:

MOV A, P0 ;для W

ANL A, #00000010B

JZ TEST\_X

JNZ TEST\_V

TEST\_V: MOV A, P1

ANL A, #00001000B

JZ TEST\_X

JNZ SET\_Q

TEST\_X: MOV A, P0

ANL A, #00000100B

JZ TEST\_A

JNZ TEST\_Z

TEST\_Z: MOV A, P2

ANL A, #00100000B

JZ SET\_Q

JNZ TEST\_A

TEST\_A: MOV A, P0

ANL A, #00001000B

JZ TEST\_C

JNZ TEST\_U

TEST\_C: MOV A, P0

ANL A, #00100000B

JZ CLR\_Q

JNZ TEST\_U

TEST\_U: MOV A, P0

ANL A, #00000001B

JZ TEST\_B

JNZ SET\_Q

TEST\_B: MOV A, P0

ANL A, #00010000B

JZ CLR\_Q

JNZ SET\_Q

SET\_Q: ; Установить Q в 1 (но делаем обратное, т.к. при 0 включается диод)

CLR Q

JMP MAIN

CLR\_Q:

SETB Q

JMP MAIN

END