Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Оренбургский государственный университет»

Кафедра промышленной электроники
и информационно-измерительной техники

С. А. Сильвашко, В. Н. Булатов

МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ КОНТРОЛЯ И УПРАВЛЕНИЯ

Методические указания

Рекомендовано к изданию редакционно-издательским советом федерального государственного бюджетного образовательного учреждения высшего образования «Оренбургский государственный университет» для обучающихся по образовательной программе высшего образования по направлению подготовки 11.04.04 Электроника и наноэлектроника

Оренбург
2018

УДК 004.38(076.5)
ББК 32.971.3я7
С 36

Рецензент – доцент, кандидат технических наук А. В. Хлуденев

Сильвашко, С. А.

С 36 Микропроцессорные системы контроля и управления : методические указания / С. А. Сильвашко, В. Н. Булатов; Оренбургский гос. ун-т. – Оренбург : ОГУ, 2018. – 51 с.

В методических указаниях изложена методика выполнения обучающимися лабораторных работ, практических занятий и расчетно-графической работы, предусмотренных рабочей программой дисциплины «Микропроцессорные системы контроля и управления». Приведены варианты индивидуальных заданий и краткие сведения о программных и технических средствах, используемых при выполнении этих заданий.

Методические указания предназначены для обучающихся по образовательной программе высшего образования по направлению подготовки 11.04.04 Электроника и наноэлектроника.

УДК 004.38(076.5)
ББК 32.971.3я7

Содержание

Введение

Современные высокотехнологичные производства (в области машиностроения, энергетики, пищевой промышленности и т. п.) немыслимы без использования автоматизированных систем управления технологическими процессами (АСУТП) или объектами (АСУТО), неотъемлемой составной частью которых являются информационно-измерительные системы (ИИС). В том или ином виде ИИС находят и самостоятельное применение. Примером могут служить автоматизированные системы контроля и управления (АСКиУ). Особый интерес представляют современные системы, реализованные на основе микроконтроллеров (программируемых логических контроллеров – ПЛК) или однокристальных микро-ЭВМ, которые известны как микропроцессорные системы контроля и управления (МСКиУ). Такие системы обеспечивают возможность одновременного измерения большого числа параметров контролируемого объекта (объектов), а также обработки результатов измерений. В качестве основных достоинств МСКиУ можно выделить их многофункциональность, долговременную метрологическую стабильность, возможность быстрой перенастройки посредством изменения программного обеспечения и др.

Практикум, предусмотренный по дисциплине «Микропроцессорные системы контроля и управления», направлен на приобретение обучающимися способности обеспечивать программную реализацию алгоритмов обработки результатов измерений в реальном времени. Для выполнения лабораторных работ и заданий на практических занятиях используются как программные средства компьютерного моделирования, так и лабораторное оборудование – учебно-отладочные стенды УМК-48.

Задания, предусмотренные практикумом, обучающиеся выполняют как в часы, отведенные для аудиторных занятий, так и в часы самостоятельной работы.

Методические указания предназначены для обучающихся по образовательной программе магистратуры по направлению подготовки 11.04.04 Электроника и наноэлектроника.

1 Лабораторные работы

1.1 Лабораторная работа № 1. Исследование функционирования микроконтроллера устройства сопряжения с объектом при вводе-выводе данных

Цель лабораторной работы – развитие у обучающихся способности к проведению экспериментальных исследований с применением микропроцессорных информационно-измерительных систем.

Аудиторное время, отводимое на занятие – 8 академических часов.

1.1.1 Моделирование микроконтроллеров в среде Multisim

Особенности использования программы схемотехнического моделирования Multisim при исследовании аналоговых и простых цифровых электронных устройств изложены в [7, С. 124 – 159]. Остановимся только на вопросе моделирования устройств, элементом которых является микроконтроллер.

Модели микроконтроллеров, которые можно программировать, размещены в библиотеке элементов под названием MCU. В составе этой библиотеки четыре типа микроконтроллеров: 8051, 8052, PIC16F84 и PIC16F84A. В дальнейшем при выполнении лабораторных работ будем использовать микроконтроллер 8051. Рассмотрим особенности создания проекта на основе микроконтроллера [6, 8].

На начальном этапе в окно схемы (на рабочий лист) необходимо поместить условное графическое обозначение (УГО) микроконтроллера, выбранного из библиотеки MCU одним из способов:

1) через меню ***Place*** → ***Component*** → ***MCU*** → **805x** → **8051**;

2) через панель компонентов, используя кнопку  , а затем выбирая в открывшемся окне **805x** → **8051**.

После размещения УГО микроконтроллера в окне схемы открывается окно мастера создания проекта с MCU «***MCU Wizard***». Проект создается за три шага. На **первом шаге** (рисунок 1) предлагается выбрать место для размещения рабочей об-

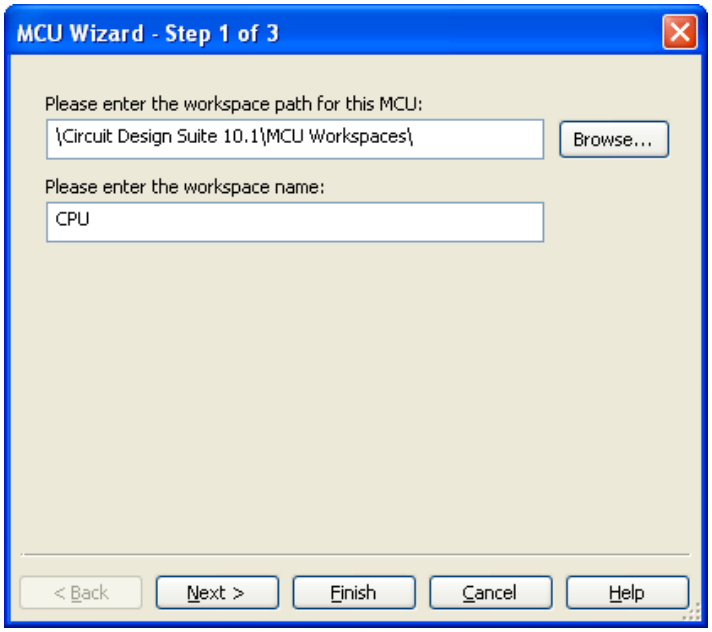


Рисунок 1

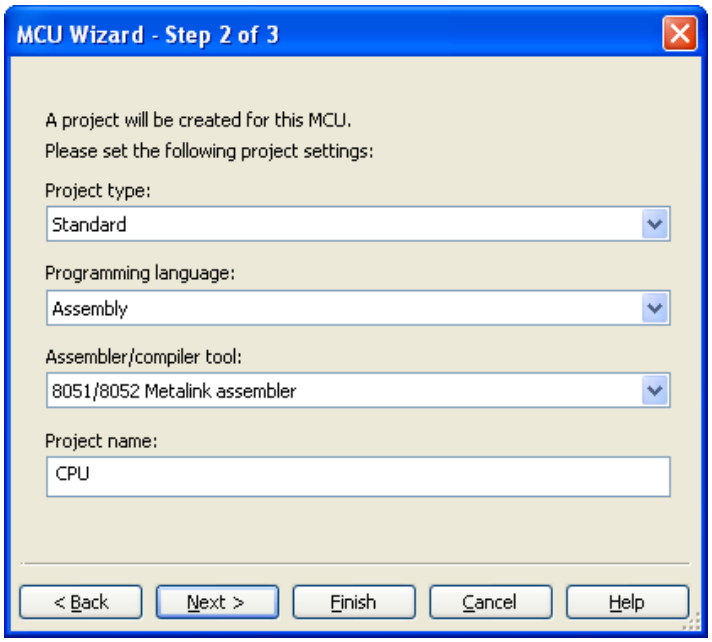


Рисунок 2

ласти (папки) проекта (по умолчанию – C:\Documents and Settings\User\Мои документы\National Instruments\Circuit Design Suite 10.1\MCU Workspaces\) и названия рабочей области (папки).

После заполнения полей окна первого шага следует нажать на кнопку <**Next**>, в результате чего откроется окно **второго шага** создания проекта (рисунок 2). В этом окне выбирается: **тип проекта** – стандартный (Standard) или с использованием внешнего шестнадцатеричного файла (Use External Hex File); **язык программирования** – C или ассемблер (Assembly); **имя проекта** (по умолчанию – project1). При выполнении лабораторных работ рекомендуется использовать ***стандартный*** тип проекта, а в качестве языка программирования – ***ассемблер***. В этом случае по умолчанию задается компилятор **8051/8052 Metalink assembler**. После нажатия на кнопку <**Next**> откроется окно **третьего шага** создания проекта (рисунок 3). В этом окне предлагается либо создать пустой проект (Create empty project), то есть **без программного файла**, либо использовать **исходный программный файл** (Add source file, рисунок 4), указав его имя.

Работа с мастером завершается нажатием на кнопку <**Finish**>. После этого в окно схемы помещается УГО микроконтроллера и появляется еще одна вкладка (с именем программного файла), при нажатии на которую открывается окно для ввода

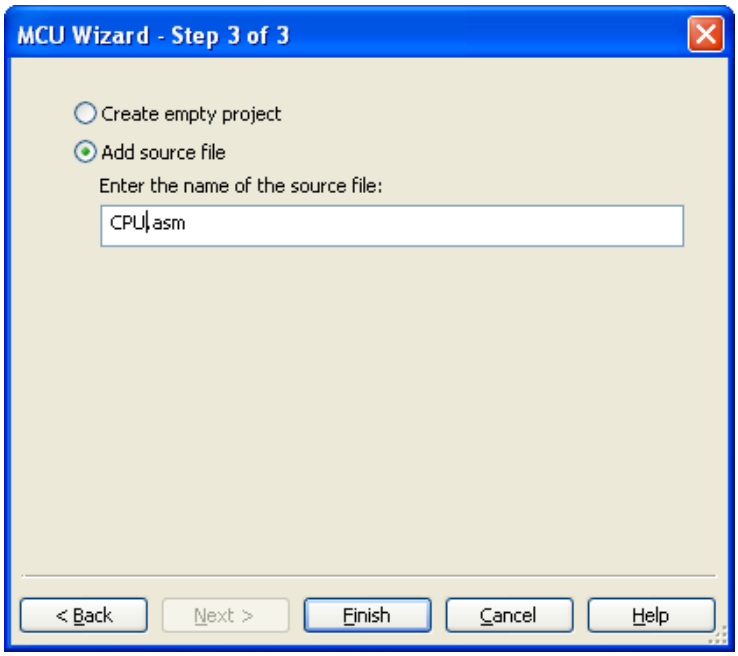


Рисунок 3

текста программы на языке ассемблер (рисунок 4).

В окне ***Design Toolbox*** (в левой части окна программы) на вкладке ***Hierarchy*** отображается структура созданного проекта. Открыть **программный файл** (файл с расширением **ASM**) после открытия проекта можно двойным щелчком левой кнопки мыши по его названию в окне ***Design Toolbox***.

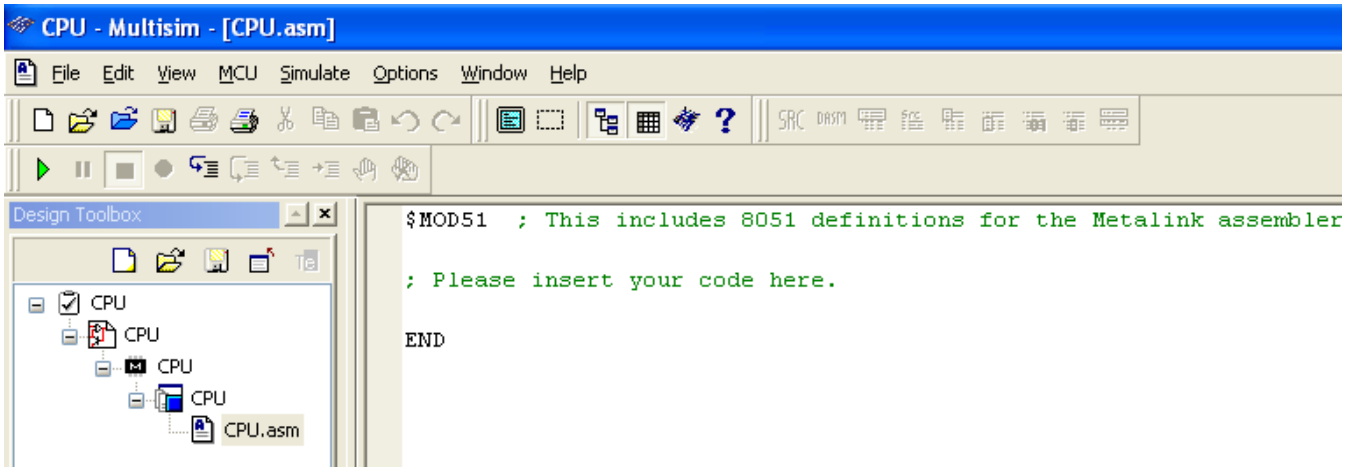
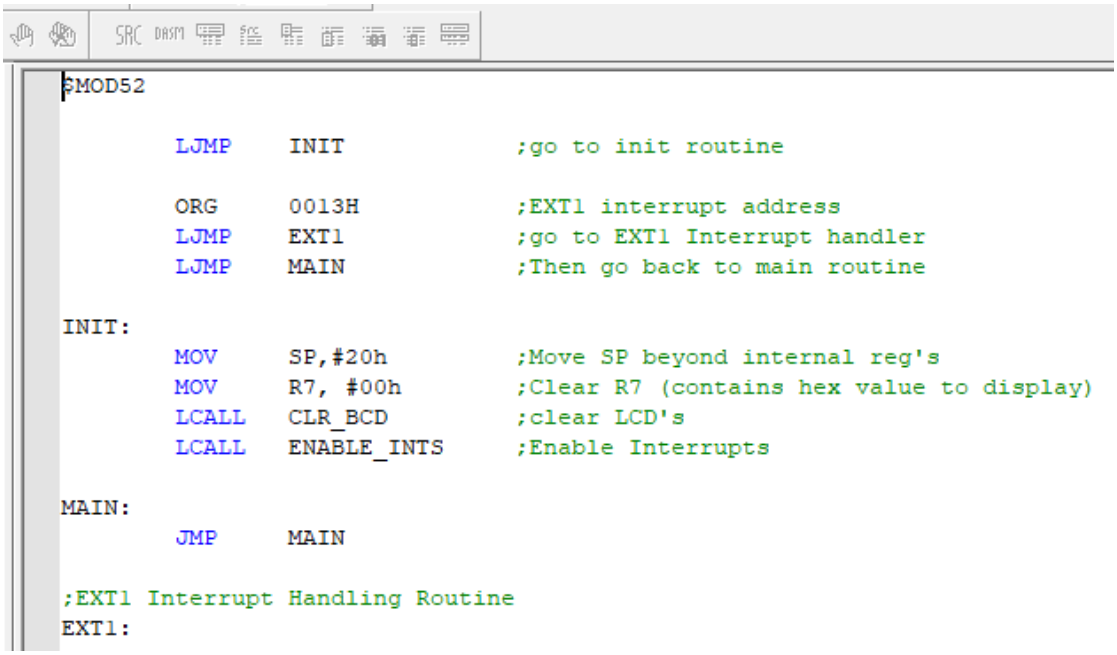


Рисунок 4

После завершения процедуры создания проекта с использованием микроконтроллера (то есть после выполнения третьего шага) можно приступить к созданию схемы, добавляя в окно схемы необходимые элементы. Кроме этого за основу можно взять уже готовую схему из тех, что размещены в папке C:\Documents and Settings\All Users\Документы\National Instruments\Circuit Design Suite 10.1\samples\MCU Sample Circuits\805x Samples\, скопировав содержимое файла в окно схемы документа и отредактировав схему с учетом решаемой задачи моделирования.

Чтобы выполнять моделирование электронного устройства с использованием схемы, созданной в схемном окне, необходимо **подготовить программный файл** на ассемблере. С этой целью нужно активировать вкладку программного файла и, используя команды микроконтроллера 8051 [1, 3], написать текст программы. После написания программы на ассемблере файл целесообразно сохранить.

На рисунке 5 в качестве примера приведен фрагмент текста программы на ассемблере.



```
$MOD52

        LJMP    INIT            ;go to init routine

        ORG     0013H           ;EXT1 interrupt address
        LJMP    EXT1            ;go to EXT1 Interrupt handler
        LJMP    MAIN            ;Then go back to main routine

INIT:
        MOV     SP,#20h         ;Move SP beyond internal reg's
        MOV     R7, #00h        ;Clear R7 (contains hex value to display)
        LCALL   CLR_BCD         ;clear LCD's
        LCALL   ENABLE_INTS     ;Enable Interrupts

MAIN:
        JMP     MAIN

;EXT1 Interrupt Handling Routine
EXT1:
```

Рисунок 5

Кроме команд микроконтроллера в тексте программы могут быть использованы и другие элементы, например:

– метки (label) – для организации переходов в программе. После идентификатора метки ставится двоеточие («**INIT:**», «**MAIN:**», «**EXT1:**»);

– комментарии – для пояснения тех или иных фрагментов программного кода. Перед комментарием ставится точка с запятой («**; go to init routine**»).

После того, как программный файл подготовлен, его необходимо **откомпилировать**. Для этого используется команда меню **Build**: ***MCU → MCU 8051 U1 → Build***. Результат компиляции отображается в окне, которое открывается в нижней части окна программы (под открытым окном программного файла). Если в процессе

компиляции обнаружены ошибки, то в окне сообщений указываются номера строк, в которых они имеют место. Для отображения номеров строк служит команда меню ***MCU → Show lines numbers***. Если ошибки в программном файле отсутствуют, то можно выполнять моделирование работы схемы.

Моделировать работу схемы с микроконтроллером можно двумя способами:

– в автоматическом (непрерывном) режиме, как и для любого аналогового или простого цифрового устройства. В этом случае моделирование начинается, например, выбором команды меню ***Simulate → Run***, либо использованием кнопки  на панели моделирования;

– в пошаговом режиме с использованием отладчика. Для пошагового моделирования используется команда меню ***MCU → Step into*** (горячая клавиша <**F11**>) или кнопка  на панели моделирования.

Использование пошагового режима позволяет контролировать изменение данных в различных регистрах (например, в аккумуляторе, портах и т. п.), внутренней памяти микроконтроллера (RAM, ROM) и во внешней памяти, если она используется. Просмотр памяти микроконтроллера активируется командой меню ***MCU → MCU Windows…*** (или ***MCU → MCU 8051 U1 → Memory view***). После выбора указанной команды открывается окно ***MCU Windows*** (рисунок 6), в котором нужно выбрать желаемый тип памяти (установить флажок) и нажать на кнопку <**OK**>. В результате этих действий откроется окно, отображающее состояние выбранной памяти (рисунок 7). На рисунке 7 показана область окна, в которой отображены часть регистров специальных функций (SFR) и область с начальными адресами оперативного запоминающего устройства (RAM). Для просмотра других регистров и областей внутренней памяти используются полосы прокрутки окна.

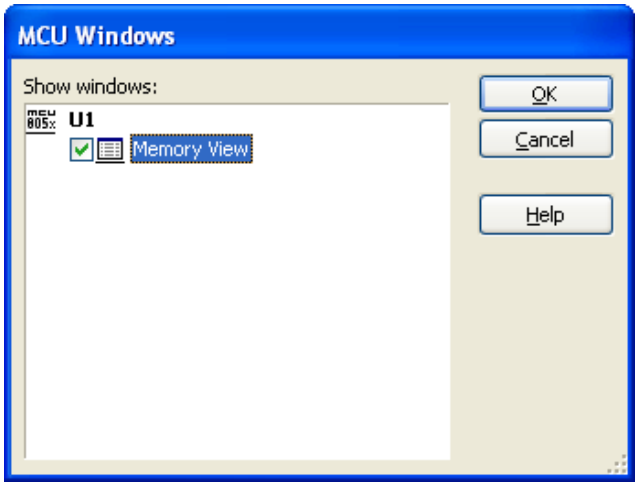


Рисунок 6

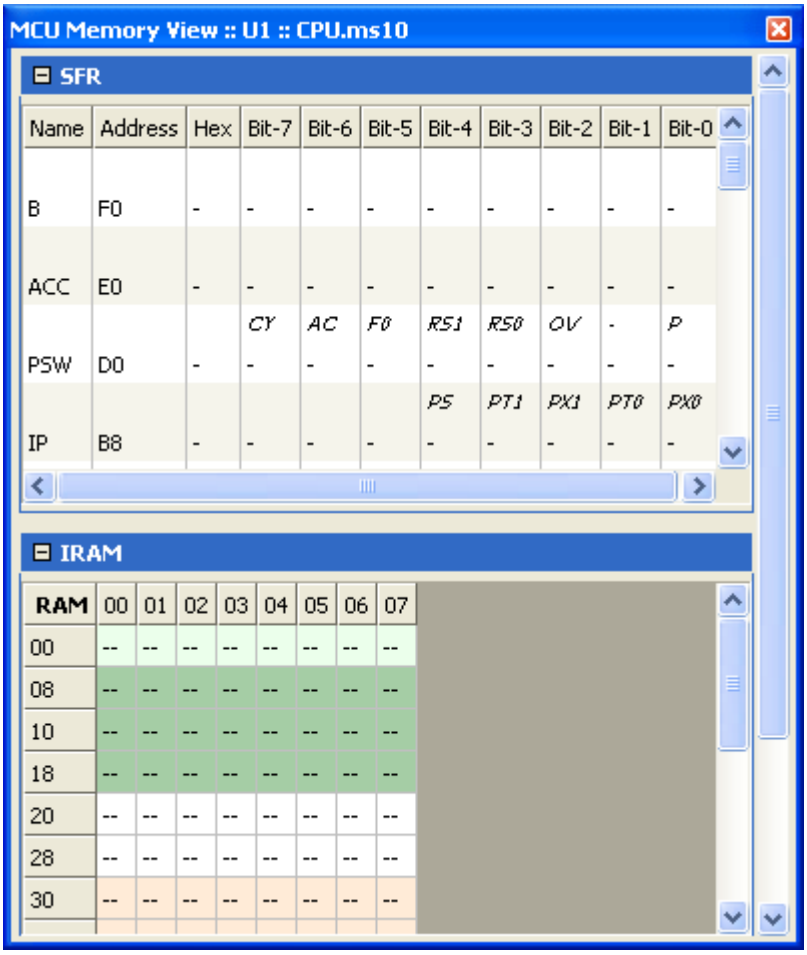


Рисунок 7

1.1.2 Задание для подготовки к лабораторной работе

1.1.2.1 Изучить архитектуру микроконтроллера 8051 [3, С. 3 – 26].

1.1.2.2 Ознакомиться с командами пересылки данных, битовых операций и передачи управления микроконтроллера 8051 [3, С. 32 – 35, 44 – 51].

1.1.2.3 Составить алгоритм функционирования микроконтроллера 8051 для преобразования аналогового напряжения в цифровой код с помощью восьмиразрядного аналого-цифрового преобразователя (АЦП) и передачи цифрового кода на световые индикаторы (пробники) (рисунок 9).

1.1.2.4 Подготовить в текстовом редакторе текст программы на ассемблере для реализации разработанного алгоритма.

1.1.3 Задание на лабораторную работу

1.1.3.1 Исследовать работу восьмиразрядного аналого-цифрового преобразователя.

1.1.3.2 Исследовать функционирование микроконтроллера 8051 при вводе-выводе данных.

1.1.4 Порядок выполнения лабораторной работы

1.1.4.1 Собрать схему, представленную на рисунке 8.

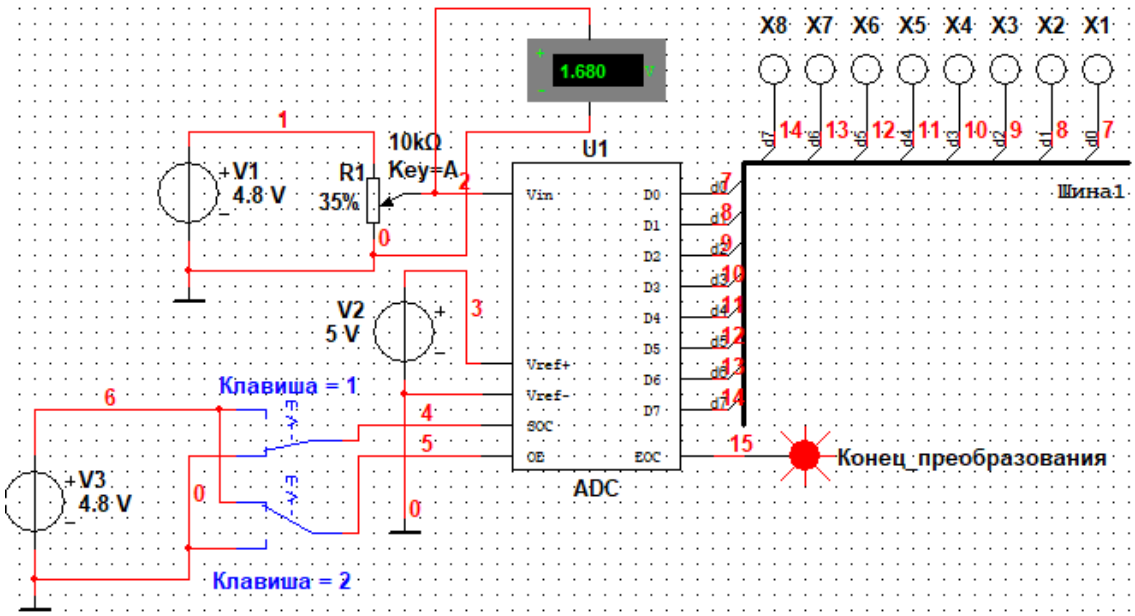


Рисунок 8

Примечание – Назначение выводов виртуального АЦП приведено в таблице 1

Таблица 1 – Назначение выводов

Обозначение	Вход/ выход	Назначение
V_{in}	I	Аналоговое входное напряжение
V_{ref+}.	I	Опорное напряжение (+)
V_{ref-}	I	Опорное напряжение (-)
D0 – D7	O	Цифровые выходы
SOC	I	Начать преобразование
EOC	O	Конец преобразования
OE	I	Разрешение цифрового вывода

Проанализировать функционирование АЦП в зависимости от логических уровней на его входах ***SOC*** и ***OE***.

Установить напряжения источников *V*1 и *V*2 в соответствии с таблицей 3. Выполнить моделирование. Изменяя сопротивление резистора *R*1 в диапазоне от 0 до 10 кОм с шагом 5 %, заносить в таблицу (таблица 2) показания вольтметра и двоичный код, сформированный на выходе АЦП.

Таблица 2 – Результаты моделирования

Сопротивление, %	Напряжение $U_{вх}$, В	Двоичный код	Напряжение измеренное, В	Относительная погрешность, %
0				
5				
10				
…				
100				

Используя формулу

$$U_{изм} = \frac{D_{10}}{256} \cdot \left(V_1 + \left| -V_2 \right| \right) , \quad (1)$$

где $D_{(10)}$ − значение выходного кода АЦП, представленного в десятичной системе счисления; V_1 и V_2 – опорные напряжения V_{ref+} и V_{ref-} соответственно, В, вычислить значения напряжений, соответствующие полученным двоичным кодам. Результаты вычислений занести в столбец «Напряжение измеренное» таблицы 2.

Вычислить относительную погрешность измерения для каждого значения входного напряжения по формуле:

$$\delta_U = \frac{U_{изм} - U_{вх}}{U_{вх}} \cdot 100\% . \quad (2)$$

Результаты вычислений занести в соответствующий столбец таблицы 2.

Используя данные, расположенные в столбцах «Напряжение $U_{вх}$» и «Двоичный код» таблицы 2, построить передаточную характеристику исследуемого АЦП

(зависимость вида $D_{(10)} = f(U_{вх})$. Оценить линейность полученной характеристики. Сформулировать выводы на основе результатов проведенных исследований.

1.1.4.2 Собрать схему для проведения исследований, представленную на рисунке 9.

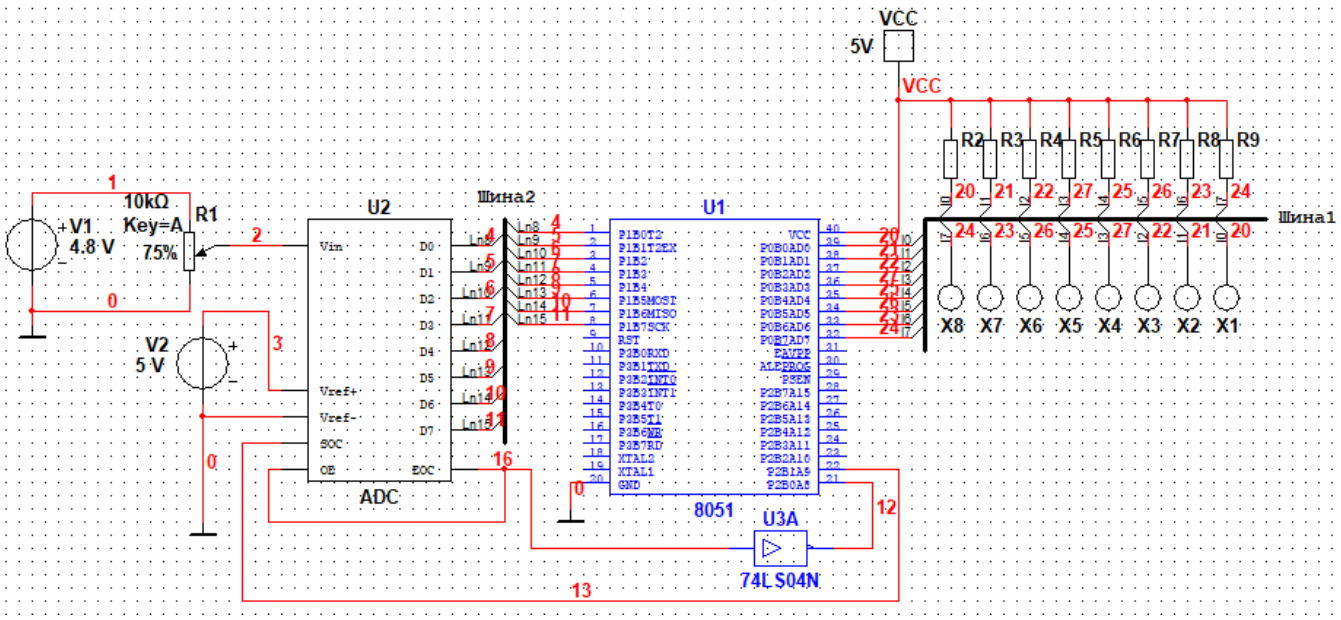


Рисунок 9

Создать программный файл.

Выполнить моделирование. Изменяя напряжение на входе АЦП, контролировать изменение двоичного кода на его выходе. Сравнить двоичные коды для трех значений входных напряжений, полученные в процессе моделирования, с соответствующими кодами, полученными при выполнении п. п. 1.1.4.1. Сформулировать выводы.

1.1.4.3 Выполнить моделирование в пошаговом режиме. Проанализировать изменение в процессе моделирования двоичных данных в аккумуляторе, портах P0, P2, в ОЗУ и ПЗУ микроконтроллера.

Сформулировать выводы.

1.1.5 Варианты индивидуальных заданий

Таблица 3 – Варианты индивидуальных заданий

Номера вариантов	Напряжение *V*1, В	Напряжение *V*2, В
1	4,9	5
2	8	10
3	9,6	10
4	4,5	5
5	4,3	5
6	4,7	5
7	9,8	10
8	4,8	5
9	5	5
10	9,1	10
11	9,5	10
12	9,7	10
13	4,4	5
14	9,9	10
15	4,6	5

1.1.6 Вопросы для самоконтроля

1.1.6.1 Какой объем внутренней памяти программ (ROM) микроконтроллера 8051?

1.1.6.2 Какой объем внутренней памяти данных (RAM) микроконтроллера 8051?

1.1.6.3 Для чего используют порты P0 и P2 микроконтроллера? Поясните особенности использования порта P0.

1.1.6.4 Поясните назначение портов P1 и P2 микроконтроллера.

1.1.6.5 Поясните особенности использования микроконтроллера при последовательном вводе-выводе данных.

1.1.6.6 Поясните особенности использования команд битовых операций.

1.1.6.7 Для чего используют регистры специальных функций?

1.1.6.8 Для чего используется регистр слова состояния микроконтроллера (PSW)?

1.1.6.9 Как организовать цикл в программе, написанной на ассемблере?

1.2 Лабораторная работа № 2. Обработка измерительной информации с помощью МСКиУ в реальном времени

Лабораторная работа имеет целью развитие у обучающихся навыков программирования микроконтроллера для решения задач обработки измерительной информации в реальном времени.

Аудиторное время, отводимое на занятие – 8 академических часов.

1.2.1 Краткие сведения о микроконтроллере 8051

В микроконтроллере 8051 реализованы команды арифметических операций, которые можно использовать при обработке измерительной информации в реальном времени. Мнемоника этих команд приведена в таблице 4.

Таблица 4 – Команды арифметических операций

Мнемоника	Описание
ADD A, Rn	(A) ← (A) + (Rn)
ADD A, addr	(A) ← (A) + (addr)
ADD A, @Ri	(A) ← (A) + ((Ri))
ADD A, #direct8	(A) ← (A) + #direct8
ADDC A, Rn	(A) ← (A) + (Rn) + (C)
ADDC A, addr	(A) ← (A) + (addr) + (C)
ADDC A, @Ri	(A) ← (A) + ((Ri)) + (C)
ADDC A, #direct8	(A) ← (A) + #direct8 + (C)
DAA	Десятичная коррекция аккумулятора
SUBB A, Rn	(A) ← (A) - (Rn) - (C)
SUBB A, addr	(A) ← (A) - (ad) - (C)
SUBB A, @Ri	(A) ← (A) - ((Ri)) - (C)
SUBB A, #direct8	(A) ← (A) - #direct8 - (C)
INC A	(A) ← (A) + 1INC Rn(Rn) < (Rn) + 1
INC Rn	(Rn) ← (Rn) + 1
INC addr	(addr) ← (addr) + 1
INC @Ri	((Ri)) ← ((Ri)) + 1
INC DPTR	(DPTR) ← (DPTR) + 1
DEC A	(A) ← (A) - 1
DEC Rn	(Rn) ← (Rn) - 1
DEC addr	(addr) ← (addr) - 1
DEC @Ri	((Ri)) ← ((Ri)) - 1
MUL AB	(B)(A) ← (A) x (B)
DIV AB	(B).(A) ← (A) / (B)

Результат выполнения команд сложения (ADD, ADDC), вычитания (SUBB), умножения (MUL) и деления (DIV) влияет на флаги слова состояния микроконтроллера (PSW) следующим образом:

– флаг переноса C устанавливается в единицу при переносе из разряда D7 в том случае, если результат операции не помещается в восемь разрядов; флаг дополнительного переноса AC устанавливается в единицу при переносе из разряда D3 в командах сложения и вычитания и служит для реализации десятичной арифметики. Этот признак используется командой DAA;

– флаг OV устанавливается в единицу при переносе из разряда D6 в случае, если результат операции не помещается в семь разрядов и восьмой не может быть интерпретирован как знаковый. Этот признак служит для организации обработки чисел со знаком;

– флаг четности P устанавливается и сбрасывается аппаратно. Если число единичных битов в аккумуляторе нечетно, то P = 1, в противном случае P = 0.

Назначение битов регистра PSW поясняется в таблице 5.

Таблица 5 – Регистр состояния микроконтроллера

Обозначение	Бит	Описание
C	PSW.7	Флаг переноса. Устанавливается и сбрасывается аппаратно или программно при выполнении арифметических и логических операций
AC	PSW.6	Флаг вспомогательного переноса. Устанавливается и сбрасывается только аппаратно при выполнении команд сложения и вычитания и сигнализирует о переносе или займе в бите 3
F0	PSW.5	Флаг 0. Может быть установлен, сброшен или проверен программой как флаг, специфицируемый пользователем
RS1	PSW.4	Выбор банка регистров. Устанавливается и сбрасывается программно для выбора рабочего банка регистров
RS0	PSW.3	
OV	PSW.2	Флаг переполнения. Устанавливается и сбрасывается аппаратно при выполнении арифметических операций
-	PSW.1	Не используется
P	PSW.0	Флаг четности. Устанавливается и сбрасывается аппаратно в каждом цикле и фиксирует нечетное/четное число единичных битов в аккумуляторе, т. е. выполняет контроль по четности

При сложении двух положительных чисел выход результата сложения за пределы одного байта приводит к установке флага переноса C в единицу. К

аналогичному результату приводит выход **результата вычитания** двух чисел за пределы представления положительного числа (что связано с заимствованием единицы старшего разряда). Это свойство можно использовать при сравнении двух положительных чисел.

Чтобы контролировать значение флага переноса, используют **битовые операции**. Система команд микроконтроллера 8051 включает 12 команд операций над битами (основные из которых приведены в таблице 6), позволяющих выполнять операции: сброс, установку, инверсию, логическое И (&) и логическое ИЛИ (|). В качестве операндов в таких командах могут выступать 128 бит из **внутренней памяти данных** микроконтроллера (начиная с адреса 20h и заканчивая адресом 2Fh), а также **регистры специальных функций** (SFR) (таблица 7), допускающие адресацию отдельных битов.

Таблица 6 – Команды битовых операций

Мнемоника	Описание
CLR C	(C) ← 0
CLR bit	(bit) ← 0
SETB C	(C) ← 1
SETB bit	(bit) ← 1
CPL C	(C) ← ~(C)
CPL bit	(bit) ← ~(bit)
ANL C, bit	(C) ← (C) & (bit)
ORL C, bit	(C) ← (C) | (bit)
MOV C, bit	(C) ← (bit)
MOV bit, C	(bit) ← (C)

Кратко рассмотрим особенности использования некоторых регистров специальных функций. **Аккумулятор** (A) является **источником операнда** при выполнении арифметических, логических операций и ряда операций передачи данных. В него же записывается результат операции. Кроме того, некоторые операции можно выполнить **только с использованием аккумулятора** (например, операции сдвигов, проверку на нуль, формирование флага паритета и т. п.). При обращении к отдельным битам аккумулятора (при реализации битовых операций) в программе на ассемблере используется обозначение ACC. Например, **ACC.3** означает третий бит аккумулятора.

В распоряжении программиста имеются четыре банка по 8 **регистров общего назначения** (R0 – R7). Выбор того или иного банка осуществляется с помощью битов RS0 и RS1 регистра PSW. Комбинации значений указанных битов при выборе банка приведены в таблице 8.

Таблица 7 – Регистры специальных функций

Обозначение	Описание	Адрес
A	Аккумулятор	0E0H
B	Регистр-расширитель аккумулятора	0F0H
PSW	Слово состояния программы	0D0H
SP	Регистр-указатель стека	81H
DPTR	Регистр-указатель данных (DPH)	83H
	(DPL)	82H
P0	Порт 0	80H
P1	Порт 1	90H
P2	Порт 2	0A0H
P3	Порт 3	0B0H
IP	Регистр приоритетов прерываний	0B8H
IE	Регистр маски прерываний	0A8H
TMOD	Регистр режима таймера/счетчика	89H
TCON	Регистр управления/статуса таймера	88H
TH0	Таймер 0 (старший байт)	8CH
TL0	Таймер 0 (младший байт)	8AH
TH1	Таймер 1 (старший байт)	8DH
TL1	Таймер 1 (младший байт)	8BH
SCON	Регистр управления приемопередатчиком	98H
SBUF	Буфер приемопередатчика	99H
PCON	Регистр управления мощностью	87H

Таблица 8 – Битовые комбинации для выбора банка регистров

RS1	RS0	Банк	Границы адресов
0	0	0	00h-07h
0	1	1	08h-0Fh
1	0	2	10h-17h
1	1	3	18h-lFh

Среди регистров специального назначения есть регистры, выполняющие **функции адресации данных, находящихся в памяти**. К ним относятся 8-разрядный **указатель стека** (SP) и **регистр-указатель** DPTR. Указатель стека может адресовать любую область внутренней памяти данных микроконтроллера, при этом содержимое этого регистра **инкрементируется** перед выполнением команд

PUSH и CALL и **декрементируется** после выполнения команд POP и RET. **В процессе инициализации микроконтроллера после сигнала RST в указатель стека автоматически загружается код 0x07**. Это значит, что если программа не переопределяет содержимое указателя стека, то первый элемент данных в стеке будет располагаться в ячейке памяти 0x08. При загрузке данных в стек адрес растет вверх, поэтому если в программе используются банки 1, 2 или 3, то указатель стека следует проинициализировать адресом из неиспользуемой области памяти, например 0x30, чтобы не перезаписать содержимое регистров одного из банков.

Двухбайтный регистр-указатель данных DPTR обычно используется для фиксации 16-разрядного адреса в операциях с обращением к внешней памяти. При работе с DPTR допускается использование старшего и младшего байтов регистра (DPH и DPL соответственно).

В таблице 9 приведены команды передачи управления. С их помощью можно программировать разветвляющиеся и циклические вычислительные процессы.

Таблица 9 – Команды передачи управления

Мнемоника	Описание
LJMP addr l6	Длинный безусловный переход по всем адресам памяти
AJMP addr ll	Безусловный переход в пределах страницы 2 Кб
SJMP label	Безусловный переход на метку label в пределах страницы 256 байт
JMP @A+DPTR	Безусловный переход по косвенному адресу
JZ label	Переход на метку label, если нуль
JNZ label	Переход на метку label, если не нуль
JC label	Переход на метку label, если бит переноса установлен
JNC label	Переход на метку label, если бит переноса не установлен
JB bit, label	Переход на метку label, если бит установлен
JNB bit, label	Переход на метку label, если бит не установлен
JBC bit, label	Переход на метку label, если бит установлен с очисткой бита
DJNZ Rn, label	Переход на метку label, если содержимое Rn не равно
ODJNZ addr, label	Переход на метку label, если содержимое addr не равно
OCJNE A, addr, label	Сравнение аккумулятора с байтом и переход на метку label, если не равно
CJME A, #direct8, label	Сравнение аккумулятора с константой и переход на метку label, если не равно
CJNE Rn, #direct8. label	Сравнение регистра с константой и переход на метку label, если не равно
CJNE @Ri, #direct8, label	Сравнение байта памяти с константой и переход на метку label, если не равно

Продолжение таблицы 9

Мнемоника	Описание
LCALL addr l6	Длинный вызов подпрограммы по всем адресам памяти
ACALL addr ll	Вызов подпрограммы в пределах страницы 2 Кб
RET	Возврат подпрограммы
RETI	Возврат подпрограммы обработки прерывания
NOP	Пустая операция

Кратко поясним работу некоторых команд передачи управления.

В команде **SJMP** указан относительный адрес перехода. Величина смещения label рассматривается как число со знаком, поэтому переход возможен в пределах от –128 до +127 байт **относительно адреса команды**, следующей за командой SJMP.

Команда косвенного перехода **JMP @A+DPTR** позволяет вычислять адрес перехода в процессе выполнения самой программы.

С помощью команд условного перехода можно проверять следующие условия:

– **JZ** – аккумулятор содержит нулевое значение;

– **JNZ** – аккумулятор содержит ненулевое значение;

– **JC** – бит переноса C установлен (C = 1);

– **JNC** – бит переноса C не установлен (C = 0);

– **JB** – прямо адресуемый бит равен 1 (например, P1.2 = 1);

– **JNB** – прямо адресуемый бит равен 0 (например, ACC.5 = 0);

– **JBC** – прямо адресуемый бит равен 1 и очищается при выполнении команды.

Команда **DJNZ** предназначена для организации программных циклов. Указанные в команде регистр **Rn** или байт по адресу **addr** содержат счетчик повторений цикла, а смещение **label** – относительный адрес перехода к началу цикла. При выполнении команды содержимое счетчика уменьшается на 1 и проверяется на 0. Если значение содержимого счетчика не равно 0, то осуществляется переход на начало цикла, в противном случае выполняется следующая команда.

Команда **CJNE** удобна для реализации процедур ожидания событий. Операнды команды сравниваются между собой, после чего, в зависимости от результата

сравнения, выполняется либо переход на метку **label**, либо выполняется следующая команда.

Действие команд вызова процедур полностью аналогично действию команд безусловного перехода – за исключением того, что они сохраняют в стеке адрес возврата. Команда возврата из подпрограммы **RET** восстанавливает из стека значение содержимого счетчика команд, а команда возврата из процедуры обработки прерывания **RETI**, кроме того, разрешает прерывания.

Работа микроконтроллера в системе реального времени связана, как правило, с обработкой событий, генерируемых внешними устройствами (объектами контроля), и установкой временных зависимостей между событиями в системе. Для этих целей в микроконтроллере служит система прерываний. Алгоритм обработки прерывания при обнаружении запроса прерывания показан на рисунке 10 [1].

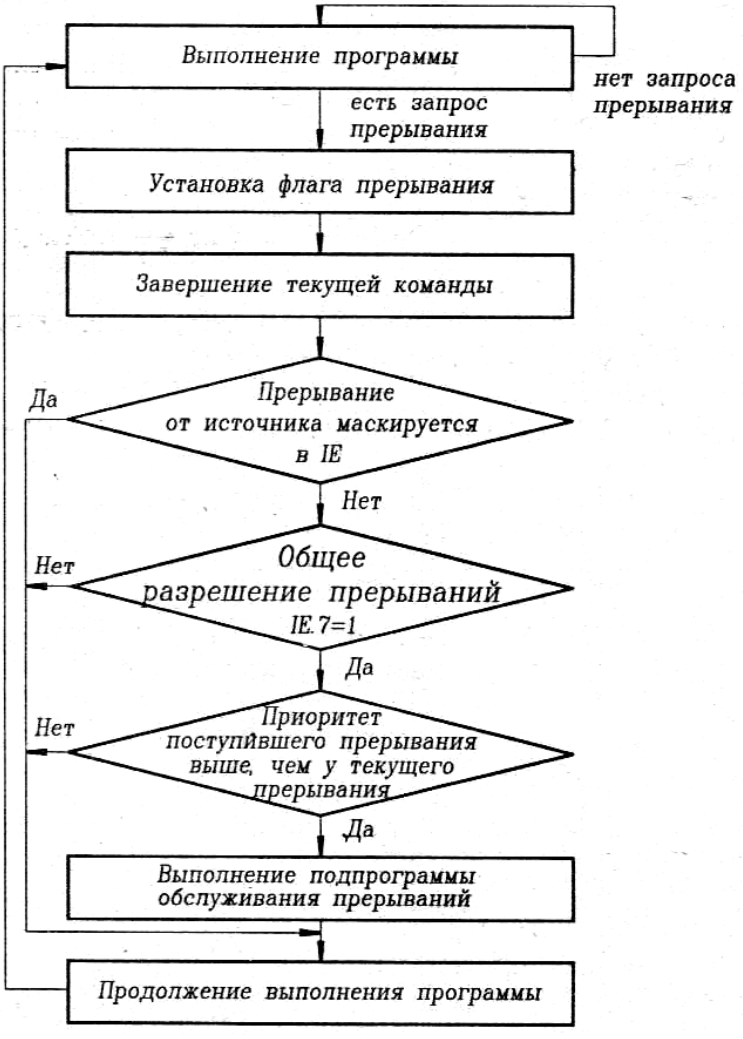


Рисунок 10

На рисунке 11 показаны все возможные источники прерывания в базовой версии микроконтроллера 8051 [1]:

– два внешних прерывания, инициированных сигналами на входах INT0 (вывод P3.2) и INT1 (вывод P3.3);

– два прерывания таймеров – TF0 и TF1;

– прерывание последовательного порта (TI, RI).

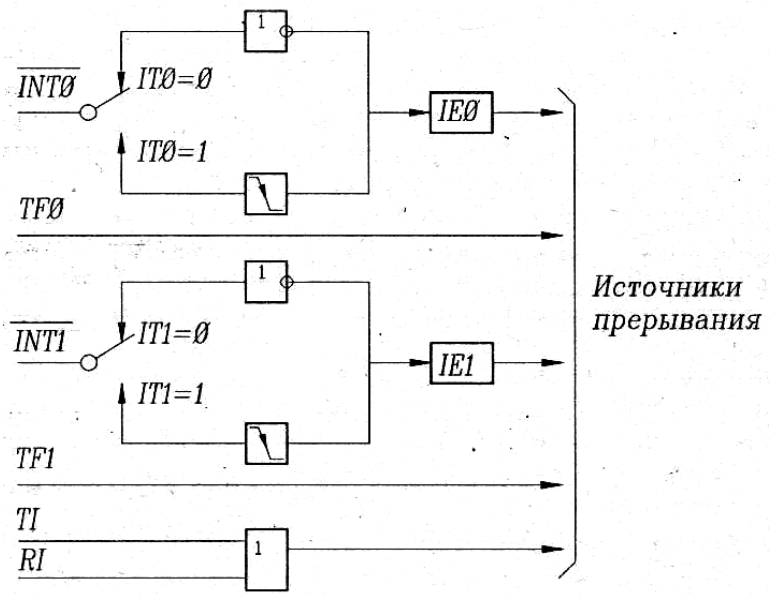


Рисунок 11

Очередность выполнения двух и более одновременно поступивших прерываний определяется их **приоритетами**. Приоритеты прерываний на рисунке 11 расставлены сверху вниз. Прерывание INT0 имеет наивысший приоритет. Самый низкий приоритет в микроконтроллере 8051 имеет прерывание последовательного порта (сигналы RI, TI).

Для обработки прерывания должна быть предусмотрена соответствующая подпрограмма обслуживания прерывания (программа – обработчик прерывания). При этом стартовый адрес обработчика прерывания в памяти микроконтроллера зависит от источника прерывания:

– для прерывания INT0 – **0x003**;

– для прерывания INT1 – **0x013**;

– для прерывания таймера TF0 – **0x00B**;

− для прерывания таймера TF1 – **0x01B**;

− для прерывания последовательного порта – **0x023**.

Для изменения приоритетов выполнения прерываний следует устанавливать специальные флаги в регистре приоритетов IP (таблица 10).

Таблица 10 – Регистр приоритетов

Обозначение	Бит регистра IP	Описание
PX0	IP.0	Бит приоритета внешнего прерывания 0. Установка/сброс программой для назначения прерыванию INT0 высшего/низшего приоритета
PT0	IP.1	Бит приоритета таймера 0. Установка/сброс программой для назначения прерыванию от таймера 0 высшего/низшего приоритета
PX1	IP.2	Бит приоритета внешнего прерывания 1. Установка/сброс программой для назначения прерыванию INT1 высшего/низшего приоритета
PT1	IP.3	Бит приоритета таймера 1. Установка/сброс программой для назначения прерыванию от таймера 1 высшего/низшего приоритета
PS	IP.4	Бит приоритета последовательного порта. Установка/сброс программой для назначения прерыванию последовательного порта высшего/низшего приоритета

Микроконтроллер реагирует на прерывания только в том случае, если будет установлен флаг соответствующего прерывания в регистре TCON (таблица 11).

Таблица 11 – Биты прерываний в регистре TCON

Обозначение	Бит регистра	Описание
IT0	TCON.0	Бит управления типом прерывания INT0, допускает программную установку. Бит IT0 = 1, если прерывание должно инициироваться перепадом сигнала 1 - 0 на выводе INT0/P3.2. Если IT0 = 0, то прерывание инициируется низким уровнем сигнала на выводе INT0
IE0	TCON.l	Флаг прерывания INT0. Устанавливается по перепаду 1 - 0 сигнала на выводе INT0. Сбрасывается аппаратно при обслуживании прерывания
IT1	TCON.2	Бит управления типом прерывания INT1, допускает программную установку. Бит IT1 = 1, если прерывание должно инициироваться перепадом сигнала 1 - 0 на выводе INT0/P3.3. Если IT1 = 0, то прерывание инициируется низким уровнем сигнала на выводе INT1
IE1	TCON.3	Флаг прерывания INT1. Устанавливается по перепаду 1 - 0 сигнала на выводе INT1. Сбрасывается аппаратно при обслуживании прерывания
TF0	TCON.5	Флаг переполнения таймера T0. Устанавливается аппаратно и сбрасывается при обслуживании прерывания
TF1	TCON.7	Флаг переполнения таймера T1. Устанавливается аппаратно и сбрасывается при обслуживании прерывания

Любое из прерываний будет вызвано, если его вызов разрешен путем установки соответствующих битов в регистре IE (таблица 12).

Таблица 12 – Биты разрешения прерываний

Обозначение	Бит регистра IE	Описание
EA	IE.7	Разрешение/запрет прерываний от всех источников
EX0	IE.0	Разрешение/запрет прерывания по входу INT0
ET0	IE.1	Разрешение/запрет прерывания таймера 0
EX1	IE.2	Разрешение/запрет прерывания по входу INT1
ET1	IE.3	Разрешение/запрет прерывания таймера 1
ES	IE.4	Разрешение/запрет прерывания последовательного порта

В таблице 12 флаг EA разрешает работу всех прерываний, для каждого из которых должен быть установлен соответствующий бит. Например, в фрагменте программного кода

SETB EX1

SETB EA

разрешается работа прерывания по входу INT1.

Внешние прерывания INT0 и INT1 в микроконтроллере 8051 могут быть вызваны **уровнем** или **переходом сигнала** из «1» в «0» на входах микроконтроллера в зависимости от значений управляющих битов IT0 и IT1 в регистре TCON. Если соответствующий бит IT0 или IT1 установлен (то есть IT0 = 1 или IT1 = 1), то прерывание вызывается по перепаду от «1» к «0» на соответствующем входе INTx (x = 0, 1) микроконтроллера. Если эти биты сброшены (то есть IT0 = 0 или IT1 = 0), то прерывание INTx вызывается при наличии **низкого уровня** на соответствующем входе. При возникновении внешнего прерывания INTx устанавливается соответствующий **флаг IEx** в **регистре TCON**, инициирующий **вызов соответствующей подпрограммы обслуживания прерывания** (программы – обработчика прерывания). **Сброс флага** IEx выполняется **аппаратно** в случае, если прерывание было вызвано перепадом сигнала на входе INTx. Если же вход прерывания INTx настроен на срабатывание по низкому уровню, то **условие выхода из прерывания должно быть прописано в программе – обработчике** данного прерывания.

Внешние прерывания INT0 и INT1 (если они разрешены) могут вызываться одним из трех способов:

– установкой низкого уровня сигнала на соответствующем выводе микроконтроллера;

– программной установкой битов P3.2 и (или) P3.3;

– установкой флага прерывания IE0 или IE1.

На рисунке 12 представлен пример вызова прерывания INT0 путем установки флага IE0 [5].

```
cseg at 0
        jmp start
int0Isr:
        org 03h
        cpl P1.7
        reti
start:
        mov SCON, #50h
        clr RI
        orl PCON, #80h
mov TH1, #0F3h ;скорость обмена 9600 бод при тактовой частоте кристалла 24.0 МГц
        orl TMOD, #20h
        setb TR1
        setb EX0
        setb EA
        setb IT0
again:
        jnb RI, $
        mov A, SBUF
        clr RI
        cjne A, #0dh, skip
        setb TCON.1
skip:
        jmp again
end
```

Рисунок 12

Обработчик прерывания INT0 инвертирует бит 7 порта P1 каждый раз, когда вызывается данное прерывание. Вызов прерывания INT0 осуществляется при приеме символа с кодом 0Dh с последовательного порта, после чего выполняется команда **setb TCON.1**, устанавливающая флаг прерывания IE0. Этот флаг сбрасывается

аппаратно при обработке прерывания, поэтому очищать его в данном случае не нужно.

Подобным образом (устанавливая флаги прерываний) можно инициировать вызов любого из прерываний.

Примечание – Программа – обработчик прерывания должна всегда заканчиваться командой **RETI**.

Прерывания от таймеров вызываются при переполнении регистров таймера (TH0 – TL0 для таймера 0, TH1 – TL1 для таймера 1). Для обоих таймеров при достижении значений 0xFF в THx – TLx (x = 1, 2) следующий инкремент таймера вызовет установку флага TFx, что инициирует прерывание. **Флаги прерывания** таймеров T0 и T1 размещены **в регистре TCON**, при этом TF0 соответствует биту TCON.5, а TF1 – биту TCON.7. **Флаги запросов прерывания** от таймеров TF0 и TF1 **сбрасываются автоматически** при передаче управления подпрограмме обслуживания.

Прерывание последовательного порта вызывается установкой одного из флагов – TI или RI (рисунок 11). Флаги запросов прерывания RI и TI устанавливаются логикой последовательного порта аппаратно, но сбрасываться должны программой. Прерывания могут быть вызваны или отменены программой, так как все перечисленные флаги программно доступны.

Все операции по обработке прерываний выполняет логика прерываний микроконтроллера. **Флаги прерываний опрашиваются в каждом машинном цикле**, при этом **анализ приоритетов прерываний выполняется в течение следующего машинного цикла**. Логика прерываний аппаратно сформирует дальний вызов LCALL соответствующей программы – обработчика прерывания, если только этот вызов не блокируется по одной из следующих причин:

– в данный момент обслуживается запрос прерывания с приоритетом равным или более высоким, чем текущий;

– текущий машинный цикл не является последним в цикле выполняемой команды;

– в данный момент выполняется команда RETI или любая команда, которая обращается к одному из регистров – IE или IP.

Если флаг прерывания был установлен, но по любой из перечисленных причин прерывание не было обслужено, а флаг к моменту окончания блокировки был сброшен, то запрос прерывания теряется.

1.2.2 Задание для подготовки к лабораторной работе

1.2.2.1 Изучить структуру внутренней памяти, назначение регистров специальных функций и систему команд микроконтроллера 8051 [3, С. 5 – 11, 27 – 50].

1.2.2.2 Разработать алгоритм функционирования микроконтроллера 8051 в режиме контроля уровня аналогового напряжения (для схемы, приведенной на рисунке 13) с учетом индивидуального задания (таблица 13). На световые индикаторы (пробники) выводить информацию в зависимости от соотношения уровня входного напряжения и заданного уровня порога: зеленый – напряжение в норме, красный – напряжение выше нормы (или ниже нормы – в зависимости от задания). Подготовить текст программы на ассемблере для реализации разработанного алгоритма.

1.2.2.3 Разработать алгоритм преобразования аналогового напряжения в цифровой код с помощью восьмиразрядного АЦП под управлением микроконтроллера 8051 по прерыванию, поступающему на вход INTx, и передачи цифрового кода на световые индикаторы (пробники) (рисунок 14). Подготовить текст программы на ассемблере для реализации разработанного алгоритма. Варианты использования портов P2 и P3 представлены в таблице 14.

1.2.3 Задание на лабораторную работу

1.2.3.1 Исследовать функционирование микроконтроллера в режиме контроля параметра объекта в реальном времени.

1.2.3.2 Исследовать функционирование микроконтроллера в режиме контроля параметра объекта в реальном времени при наличии внешних прерываний.

1.2.4 Порядок выполнения лабораторной работы

1.2.4.1 Собрать схему для проведения исследований, представленную на рисунке 13 (с учетом использования разрядов порта P2 в соответствии с таблицей 13).

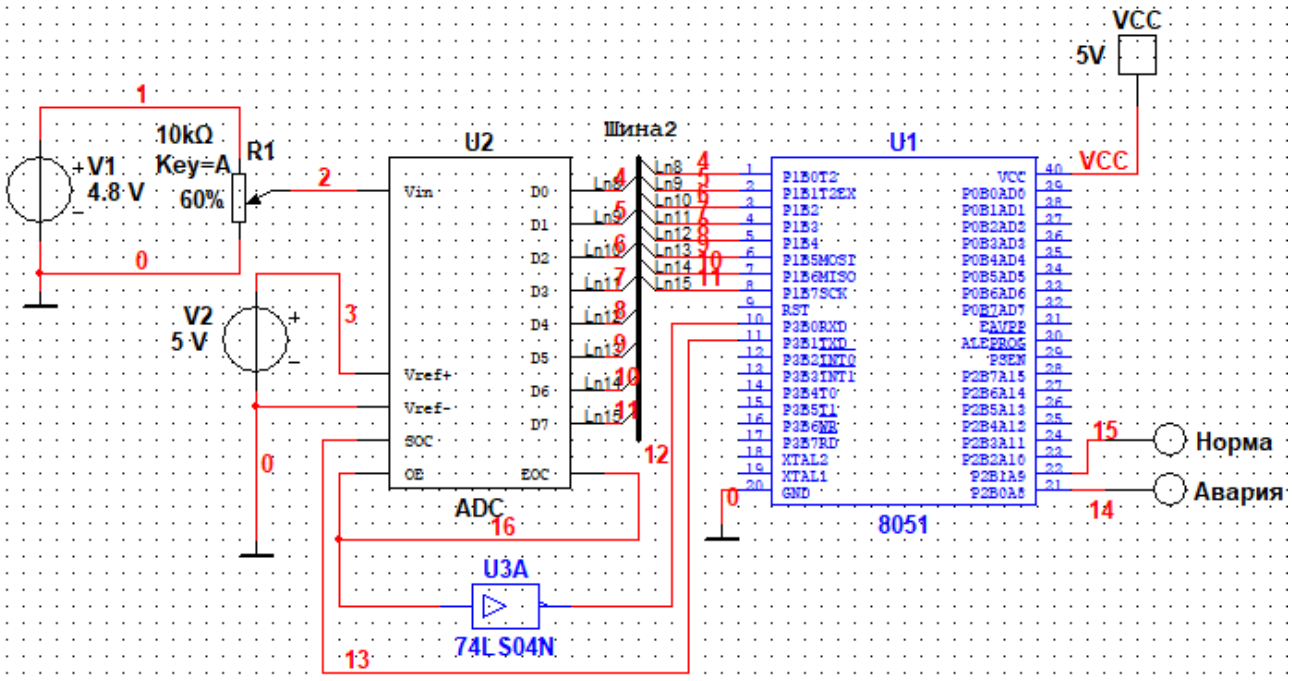


Рисунок 13

Создать программный файл (на основе результатов выполнения п. п. 1.2.2.2). Значение двоичного кода, соответствующего норме входного напряжения, использовать из результатов выполнения лабораторной работы № 1.

Выполнить моделирование. Плавно изменяя напряжение на входе АЦП, следить за отображением информации на световых индикаторах. Сформулировать выводы о возможности использования микроконтроллера 8051 для контроля параметра объекта в реальном времени.

1.2.4.2 Собрать схему для проведения исследований, представленную на рисунке 14 (с учетом использования разрядов портов P2 и P3 в соответствии с таблицей 14).

Создать программный файл (на основе результатов выполнения п. п. 1.2.2.3).

Выполнить моделирование. Проверить работу микроконтроллера при наличии и отсутствии прерывания на входе.

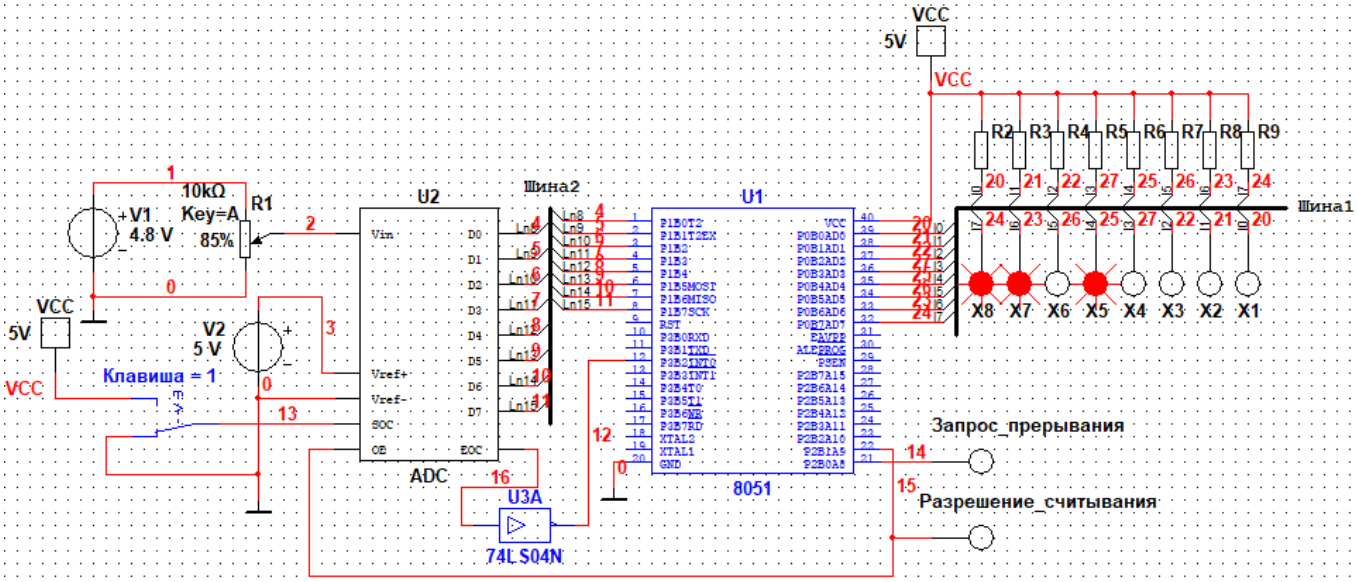


Рисунок 14

Сформулировать выводы о возможности использования микроконтроллера 8051 для обработки входной информации по внешнему прерыванию.

1.2.5 Варианты индивидуальных заданий

Таблица 13 – Варианты индивидуальных заданий

Номера вариантов	Напряжение *V*1, В	Напряжение *V*2, В	Контроль напряжения		Разряды порта P2	
			выше нормы	ниже нормы	норма	авария
1	4,9	5	+		P2.0	P2.1
2	8	10	+		P2.0	P2.2
3	9,6	10		+	P2.0	P2.3
4	4,5	5		+	P2.0	P2.4
5	4,3	5		+	P2.0	P2.5
6	4,7	5	+		P2.0	P2.6
7	9,8	10		+	P2.0	P2.7
8	4,8	5	+		P2.1	P2.0
9	5	5	+		P2.2	P2.0
10	9,1	10		+	P2.3	P2.0
11	9,5	10		+	P2.4	P2.0
12	9,7	10	+		P2.5	P2.0
13	4,4	5	+		P2.6	P2.0
14	9,9	10	+		P2.7	P2.0
15	4,6	5		+	P2.2	P2.5
Примечание – В качестве нормы контролируемого напряжения принять уровень, соответствующий 50 % от максимального уровня входного напряжения.						

Таблица 14 – Варианты индивидуальных заданий

Номера вариантов	Напряжение *V*1, В	Напряжение *V*2, В	Вход прерывания	Разряды порта P2	
				запрос	разрешение
1	4,9	5	INT0	P2.0	P2.1
2	8	10	INT1	P2.0	P2.2
3	9,6	10	INT1	P2.0	P2.3
4	4,5	5	INT0	P2.0	P2.4
5	4,3	5	INT0	P2.0	P2.5
6	4,7	5	INT1	P2.0	P2.6
7	9,8	10	INT0	P2.0	P2.7
8	4,8	5	INT1	P2.1	P2.0
9	5	5	INT1	P2.2	P2.0
10	9,1	10	INT0	P2.3	P2.0
11	9,5	10	INT0	P2.4	P2.0
12	9,7	10	INT0	P2.5	P2.0
13	4,4	5	INT1	P2.6	P2.0
14	9,9	10	INT0	P2.7	P2.0
15	4,6	5	INT1	P2.2	P2.5
Примечание – На световой индикатор «Запрос прерывания» выводится информация после появления перепада 1 – 0 на входе INTx.					

1.2.6 Вопросы для самоконтроля

1.2.6.1 При каком условии устанавливается флаг переноса C?

1.2.6.2 При каком условии устанавливается флаг OV?

1.2.6.3 Как устанавливается флаг четности P?

1.2.6.4 Какую функцию в микроконтроллере 8051 выполняет регистр PSW? Пояснить назначение отдельных битов регистра PSW.

1.2.6.5 Что может быть использовано в качестве операндов в командах битовых операций?

1.2.6.6 Для чего в микроконтроллере 8051 используется аккумулятор?

1.2.6.7 Для чего предназначены команды передачи управления?

1.2.6.8 Для каких целей в микроконтроллере служит система прерываний?

1.2.6.9 Назовите источники прерываний в микроконтроллере 8051.

1.2.6.10 Проведите ранжирование прерываний микроконтроллера 8051 в порядке убывания их приоритетов.

1.2.6.11 Поясните алгоритм обработки запроса прерывания.

2 Практические занятия

2.1 Практическое занятие № 1. Ввод и обработка измерительной информации в узле микропроцессорной системы контроля и управления

Целью занятия является формирование способности обеспечивать программную реализацию алгоритмов обработки результатов измерений.

Аудиторное время, отводимое на занятие – 6 академических часов.

2.1.1 Краткие сведения об интерпретаторе виртуальной машины

Интерпретатор ВМ используется для составления программ управления учебно-отладочным стендом УМК-48 [2]. Программа на интерпретаторе может быть написана в любом текстовом редакторе, представляющем символы в WIN-коде (например, в приложении для ОС MS Windows «Блокнот»). При написании текста программы **каждой команде отводится отдельная строка**. Если в программе предусмотрены переходы, то в качестве адресных меток используются номера строк (при этом в тексте программы номера строк не записываются). **Первая строка программы имеет номер «0»**.

Для правильного функционирования устройства под управлением программы, написанной на интерпретаторе, в тексте программы **не должно быть пустых строк**, а после последнего символа в строке **не должно быть пробелов**.

Числовые данные, используемые в программе в качестве аргументов, **представляются в десятичном коде**.

В тексте программы на интерпретаторе ВМ можно использовать аргументы и команды, список которых представлен в таблицах 15 и 16. При записи аргументов и команд можно использовать как заглавные, так и строчные буквы латинского алфавита.

Примечание – Не допускается в записи одного оператора использовать одновременно и заглавные и строчные буквы, например Set.

Таблица 15 – Список аргументов интерпретатора ВМ

Аргумент	Комментарий
S0, S1, S2, S3, S4, S5, S6, S7	Входные переменные дискретных величин (состояние тумблеров): 1 – включен, 0 – выключен
ACP1, ACP2, ACP3, ACP4	Входные переменные аналоговых величин АЦП (значение принадлежит интервалу [0 – 15])
D0, D1, D2, D3, D4, D5, D6, D7	Выходные переменные дискретного типа (состояние светодиодных индикаторов): 1 – включен, 0 – выключен, 2 – мигание
D_ALL	Выходная переменная дискретного типа (состояние светодиодных индикаторов) (значение принадлежит интервалу [0 – 255])
CAP1, CAP2	Выходные переменные аналогового типа, значениями которых управляет пользователь (ЦАП) (значение принадлежит интервалу [0 – 15])
BEEP	Код микроконтроллера BEEP – звукового сигнала длительностью 1 с.
WAIT	Ожидание (задержка в секундах)
INIT	Код микроконтроллера RESET

Таблица 16 – Список команд интерпретатора ВМ

Оператор	Операнд A1	Операция	Примеры
IF A1=A2 THEN A3 ELSE A4 IF A1=A2 THEN A3	S0, S1, S2, S3, S4, S5, S6, S7, ACP1, ACP2, ACP3, ACP4	Если условие выполняется, то происходит переход на метку A3, а если условие не выполняется, то переход на метку A4; если ELSE отсутствует и условие не выполняется, то происходит переход на оператор, следующий за оператором IF	IF S0 = 1 THEN 12 ELSE 3 IF S0 = 0 THEN 12 IF ACP1 = 1 2 THEN 12 ELSE 3 (если на выходе АЦП от 1 до 2, то...) IF ACP1 = 0 2 THEN 12 (если на выходе АЦП от 0 до 2, то...)
SET A1=A2	D0, D1, D2, D3, D4, D5, D6, D7, CAP1, CAP2, D_ALL	Оператор установки выходных и внутренних переменных	SET D0 = 1 SET D7 = 0 SET D_ALL = 255
GOTO A1	Целое число от 0 до 1000	Оператор безусловного перехода	GOTO 0 GOTO 10
WAIT=A1	Целое число от 0 до 3600	Оператор остановки на определенное время (в секундах)	WAIT = 3 WAIT = 60

Примечание – Значение выходной переменной дискретного типа D_ALL формируется как десятичное число для байта данных. Например, если D7…D0 = 11000110 (то есть требуется обеспечить свечение светодиодов с номерами 1, 2, 6 и 7), то D_ALL = 2^7 + + 2^6 + 2^2 + 2^1 = 128+64+4+2 = 198.

Более полную информацию о командах интерпретатора ВМ можно получить из [2, С. 256 – 261].

Текст программы на интерпретаторе ВМ необходимо сохранить в виде файла с расширением **TXT**.

Для проверки функционирования устройства сопряжения с объектом (УСО) под управлением разработанной программы необходимо включить питание учебно-отладочного стенда УМК-48 и запустить программу *WorkspaceDemo*. Открыть в программе файл, содержащий текст программы. Вид окна программы *WorkspaceDemo* с текстом программы на интерпретаторе ВМ показан на рисунке 15.

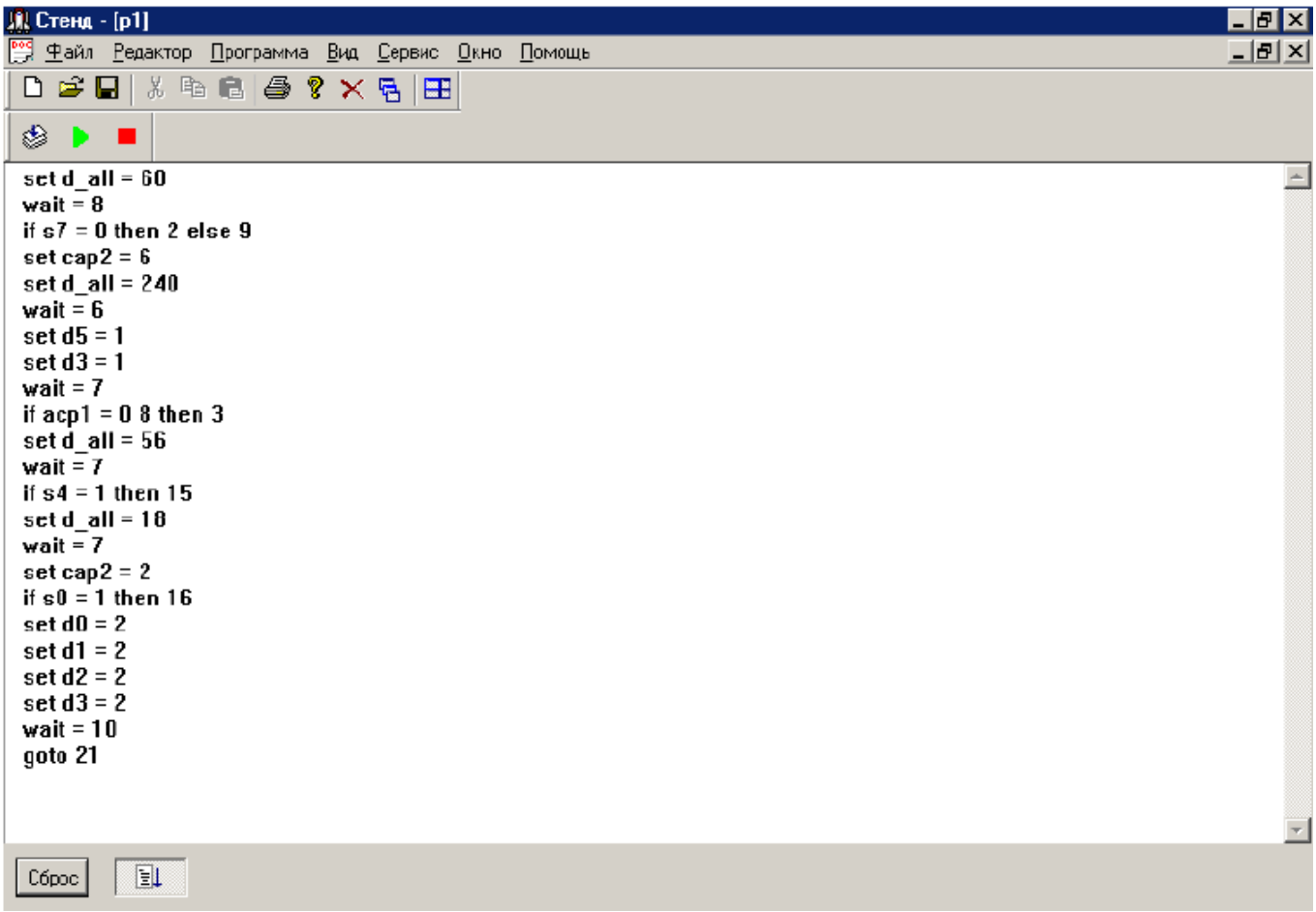


Рисунок 15

Запуск программы, открытой в *WorkspaceDemo*, осуществляется нажатием на кнопку  . Прервать выполнение программы можно нажатием на кнопку  . В случае обнаружения при компиляции программы ошибок, их можно исправить непосредственно в программе *WorkspaceDemo*.

2.1.2 Задание для подготовки к практическому занятию

2.1.2.1 Ознакомиться с устройством учебно-отладочного стенда УМК-48 и порядком использования управляющей программы *WorkspaceDemo* [2, С. 155 – 164].

2.1.2.2 Изучить аргументы и команды интерпретатора виртуальной машины (п. 2.1.1 методических указаний).

2.1.2.3 Разработать алгоритм управления узлом микропроцессорной системы контроля и управления (МСКиУ) в соответствии с индивидуальным заданием (таблица 17).

2.1.3 Задание на практическое занятие

2.1.3.1 Исследовать работу узла МСКиУ при вводе и выводе информации.

2.1.3.2 Используя алгоритм, разработанный при подготовке к занятию, написать программу на интерпретаторе виртуальной машины. Проверить функционирование узла МСКиУ под управлением разработанной программы.

2.1.4 Порядок выполнения заданий

2.1.4.1 Запустить программу *WorkspaceDemo.* В случае успешной загрузки, обнаружения и инициализации УСО в нижней части окна программы высвечивается зеленый кружок, рядом с которым отображается имя устройства последовательного ввода-вывода (порт COM1…COM4).

Примечание – В случае красного цвета кружка или других сообщений о неблагополучном запуске программы *WorkspaceDemo* сообщить руководителю занятия.

2.1.4.2 Открыть окно «***Показания датчиков***», для чего в меню ***Файл*** программы выбрать команду ***Новый***. В открывшемся окне «***Добавить окно***» выбрать опцию ***Показания датчиков*** и нажать на кнопку <**OK**>. Вид окна «***Показания датчиков***» представлен на рисунке 16.

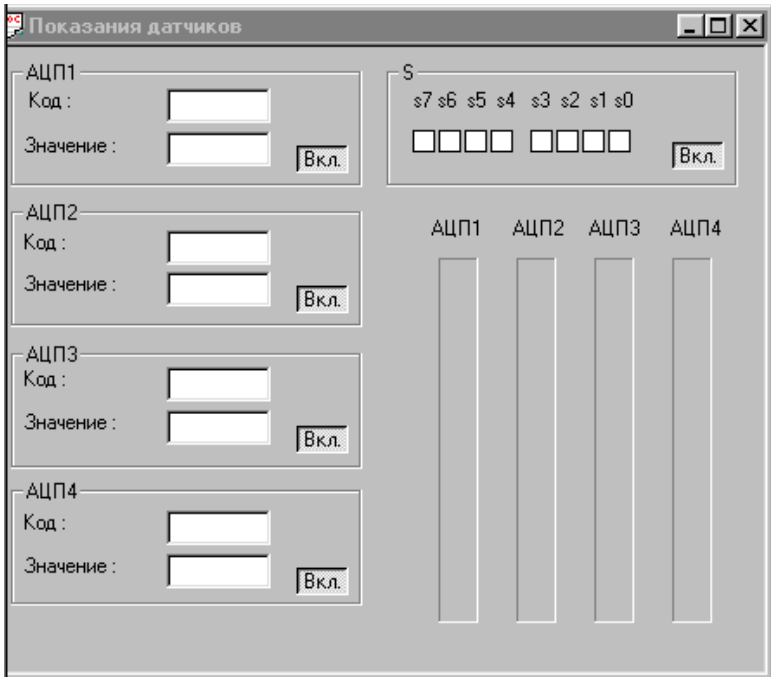


Рисунок 16

Вращая вал резистивного датчика угла положения учебно-отладочного стенда УМК-48, связанного с первым аналого-цифровым преобразователем (АЦП1), снимать значения двоичного кода в поле ***Код*** области **АЦП1** окна «***Показания датчиков***». Установить связь между изменением кода на единицу младшего разряда и приращением аналоговой угловой величины. Оценить разрешение исследуемого АЦП.

Изменяя положение концевых датчиков (S0…S7), наблюдать за состоянием полей в области **S** окна «***Показания датчиков***». Появление флажка «√» в поле свидетельствует о вводе дискретной информации в УСО.

2.1.4.3 Открыть окно «***Установка выходных величин***» (рисунок 17), для чего в меню ***Файл*** программы выбрать команду ***Новый***. В открывшемся окне «***Добавить окно***» выбрать опцию ***Установка выходных величин*** и нажать на кнопку <**OK**>.

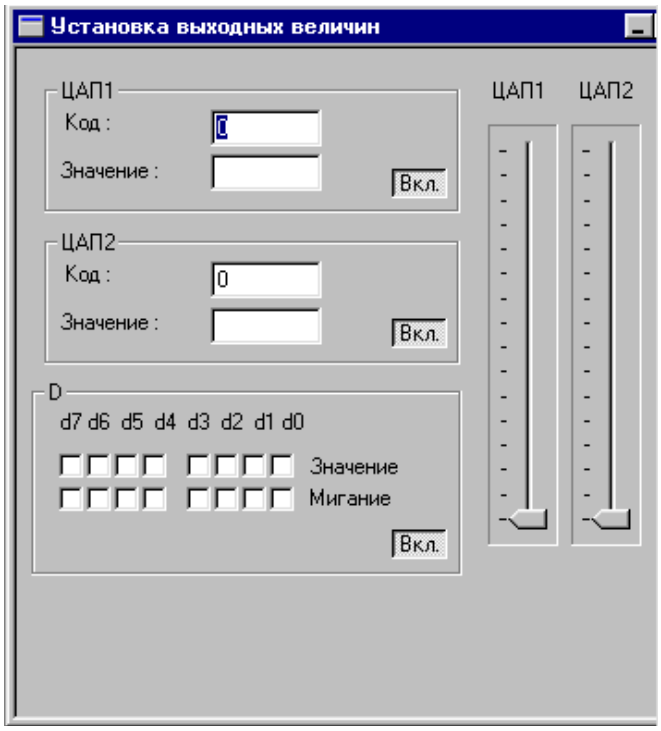


Рисунок 17

Подключить к клеммам ***UвыхЦАП1*** на учебно-отладочном стенде УМК-48 вольтметр.

Перемещая «движок» задания кода ЦАП1 в окне «***Установка выходных величин***», изменять значение двоичного кода на входе ЦАП. Значение кода отображается в поле ***Код*** области **ЦАП1** окна. На шкале вольтметра наблюдать изменение напряжения на выходе ЦАП при изменении входного кода.

Установить связь между изменением кода на единицу младшего разряда и приращением выходного напряжения ЦАП. Оценить разрешение исследуемого ЦАП.

С помощью указателя курсора мыши изменять двоичный код в полях ***Значение*** области **D**. Наличие флажка «√» в поле свидетельствует о вводе логической единицы, отсутствие метки – о вводе логического ноля. Вывод дискретной информации контролировать с помощью светодиодных индикаторов на учебно-отладочном стенде УМК-48.

Сформулировать выводы о возможностях УСО по вводу и выводу аналоговой и дискретной информации.

2.1.4.4 Закрыть окна «***Показания датчиков***» и «***Установка выходных величин***». Используя команду ***Файл → Новый***, открыть окно «***Добавить окно***» и выбрать опцию ***Текст программы***. В главном окне программы появится область для набора текста программы на интерпретаторе ВМ. Если файл с текстом управляющей программы подготовлен заранее (например, в программе «Блокнот»), то загрузить его в окно программы *WorkspaceDemo* можно командой ***Файл → Открыть***.

По ранее разработанному алгоритму управления узлом МСКиУ написать текст программы. Проверить наличие ошибок в программе можно с помощью компилятора. Для запуска компилятора используют команду ***Компилировать*** из меню ***Программа***. При работе компилятора программа выполняется в пошаговом режиме с одновременной проверкой каждой строки на наличие ошибок. Редактирование программы и исправление ошибок допускается только в режиме «***Остановить***», перейти к которому можно либо нажатием на кнопку  на панели инструментов, либо командой ***Программа → Остановить***.

В начале программы предусмотреть инициализацию УСО. С этой целью выходным переменным дискретного типа (всем d_i) и выходным переменным аналогового типа (всем cap_i) присвоить нулевые значения.

Запустить программу на выполнение и проверить реакцию УСО на поступающие команды.

Чтобы убедиться в том, что программа управления функционированием УСО написана правильно, необходимо осуществить прогон программы по всем ветвям алгоритма, образованным ветвлениями.

2.1.5 Варианты индивидуальных заданий

Таблица 17 – Варианты индивидуальных заданий

Номера вариантов	Номер шага			
	1	2	3	4
1	acp1 < 5 → 2, иначе → 3	d2 = 1	d0…d3 = 1; t = 10 с	
2	s2 = 1 → 2, иначе → 3	cap1 = 6,	d1,d3 = 0,1,0…	

Продолжение таблицы 17

Номера вариантов	Номер шага			
	1	2	3	4
3	acp1 < 2 → 2, иначе → 4	acp2 < 5 → 3, иначе → 4	d7 = 1; t = 10 с	d0…d3 = 1,0,1… t = 10 с
4	acp2 < 8 → 2, иначе → 3	cap1 = 2	d0,d3 = 1,0,1… t = 10 с	
5	s0 = 1 → 2, иначе → 4	s3 = 1 → 3, иначе → 4	cap2 = 1; t = 10 с	d2,d3 = 1,0,1… t = 10 с
6	d6,d7 = 1,0,1… t = 3 с	s4 = 1 → 3, иначе → 4	cap1 = 3; t = 10 с	d6,d7 = 1,0,1… t = 10 с
7	acp2 < 8 → 2, иначе → 3	cap 2 = 9; t = 4 с	d3…d6 = 1,0,1… t = 10 с	
8	acp1 < 3 → 3	s0 = 1 → 4	cap1 = 4; t = 4 с	d0,d2 = 1,0,1… t = 10 с
9	s5 = 1 → 3	s1 = 1 → 4	cap2 = 2; t = 3 с	d1,d5 = 1; t = 10 с
10	acp2 < 5 → 3	s3 = 0 → 4	d2 = 1; t = 3 с	d1,d4 = 1,0,1… t = 10 с
11	s7 = 0 → 2, иначе → 4	acp1 < 6 → 4	s2 = 1 → 4, иначе → 2	d2,d3 = 1,0,1… t = 10 с
12	acp2 < 10 → 3	acp1 < 10 → 4; иначе → 3	d4…d7 = 1,0,1… t = 10 с	d0…d3 = 1,0,1… t = 10 с
13	s1 = 1 → 3	s0 = 1 → 4	cap1 = 4; t = 4 с	cap1 = 6; t = 10 с
14	d2,d6 = 1; t = 3 с	s2=0 → 4	d1…d5 = 1,0,1… t = 3 с	cap2 = 2; t = 10 с
15	s4 = 0 → 4	acp2 < 4 → 4	cap1 = 6; t = 4 с	d2,d3 = 1,0,1… t = 10 с

2.1.6 Вопросы для самоконтроля

2.1.6.1 Какую функцию в составе МСКиУ выполняет УСО? Что входит в состав УСО?

2.1.6.2 Какие функции выполняет микроконтроллер (микро-ЭВМ) в составе узла МСКиУ?

2.1.6.3 Какие задачи возлагают на программное обеспечение узла МСКиУ?

2.1.6.4 Зачем нужен в составе программного обеспечения узла МСКиУ встроенный интерпретатор?

2.1.6.5 Что можно использовать в качестве аргументов в командах, записанных на интерпретаторе ВМ?

2.1.6.6 Какие операторы можно использовать при записи команд на интерпретаторе ВМ?

2.2 Практическое занятие № 2. Организация обмена данными управляющей ЭВМ и микроконтроллера узла МСКиУ

Целью занятия является формирование навыков производства измерений параметров удаленного объекта в реальном времени.

Аудиторное время, отводимое на занятие – 4 академических часа.

2.2.1 Организация обмена данными между микроконтроллером узла МСКиУ и управляющей ЭВМ

При обмене данными между микроконтроллером и управляющей ЭВМ взаимодействуют два интеллектуальных объекта, при этом управляющая ЭВМ является ведущим объектом, а микроконтроллер – ведомым. Это значит, что микроконтроллер должен понимать определенный набор команд, поступающих от управляющей ЭВМ. Такой набор обычно составляет от одного до нескольких десятков одно- и двухбайтовых команд. **Информационно-структурная схема** узла МСКиУ, при его функционировании под управлением ЭВМ вышестоящего уровня, представлена на рисунке 18.

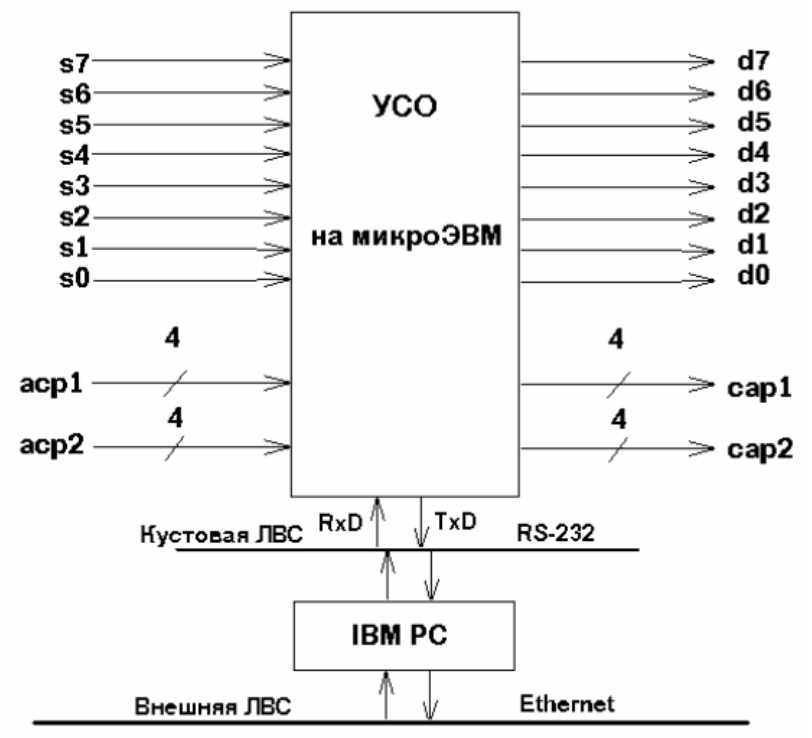


Рисунок 18

Каналом связи между управляющей ЭВМ и микроконтроллером УСО, как правило, является последовательный интерфейс с одним из установленных стандартом протоколов (например, RS-232 или С-2). Независимо от того, какое из вычислительных устройств является ведущим, а какое ведомым, **прием** осуществляется всегда по **аппаратному прерыванию**, а **передача** – по **программному**.

На управляющую ЭВМ возлагают следующие функции:

– обеспечение помощи оператору в установке и настройке аппаратуры контроля;

– задание системе контроля исходных параметров (задание исходных режимов);

– выдача информации о ходе контроля в реальном масштабе времени;

– фиксация сбоев и возникающих ошибок при функционировании аппаратуры контроля;

– выдача в виде печатного документа протокола контроля объекта.

Кроме этого, как правило, обеспечивается возможность оперативно вносить исправления и добавления в программу контроля как в процессе отладки, так и в процессе ее прохождения на управляющую ЭВМ.

Для внесения предусмотренных технологией управления поправок в ход выполнения программ в состав программного обеспечения встраиваются **интерпретаторы** с минимально необходимым набором операторов, которые не требуют от оператора глубоких профессиональных знаний в области программирования.

На рисунке 19 показано типовое техническое решение последовательного канала связи управляющей ЭВМ и микроконтроллера на основе ОЭВМ.

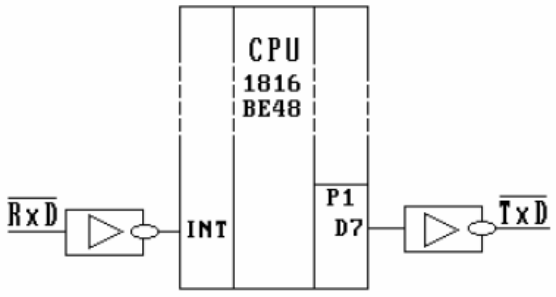


Рисунок 19

На рисунке 19 инвертирующими элементами являются: слева – приемник сигналов из линии с протоколом RS-232; справа – передатчик в линию с протоколом RS-232. Как видно из рисунка, особенностью использования микро-ЭВМ является то, что при подключении ее к последовательному интерфейсу отсутствует контроллер последовательного интерфейса. В этом случае обмен данными через последовательный интерфейс может быть реализован только программными средствами. Тактирование входных битов последовательного кода при преобразовании его в параллельный код (и наоборот) осуществляется с помощью внутреннего таймера, программно настроенного на длительность интервала между двумя соседними битами.

Подпрограмма ввода вызывается по аппаратному прерыванию (при появлении на входе **INT** логического нуля, что соответствует началу стартового бита последовательной посылки). Затем вход INT **маскируется**, и с него через фиксированные интервалы времени, определяемые задержкой (TIM), вводятся биты (как правило, 8 бит), из которых командами логического сдвига формируется байт входных данных. Перед началом ввода осуществляется задержка в половину заданного интервала для обеспечения считывания бита данных по его центру. После ввода байта данных вход INT демаскируется и микро-ЭВМ снова готова принять по прерыванию из канала связи следующую посылку.

Одной из основных задач микроконтроллера УСО является выполнение команд, выдаваемых со стороны управляющей ЭВМ. Чтобы не путать собственные (машинные) команды микроконтроллера с командами, поступающими со стороны управляющей ЭВМ, используют так называемые **макрокоманды управления**. Система макрокоманд, используемых для работы с учебно-отладочным стендом УМК-48, приведена в таблице 18.

На команду ввода микроконтроллер УСО выдает байт данных, содержание которого показано на рисунке 20.

Разряды байта информации							
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	x	x	x	x
				(Данные из УСО)			

Рисунок 20

Таблица 18 – Коды макрокоманд управления УСО

Вид команды	Разряды байта информации								Приемник данных в УСО
	D7	D6	D5	D4	D3	D2	D1	D0	
Output: (вывод данных в УСО)	0	0	0	0	Выводимые данные в ЦАП или на световые индикаторы				LED3…LED0 (индикаторы l3…l0)
			0	1					LED7…LED4 (индикаторы l7…l4)
			1	0					DAC1 (ЦАП1)
			1	1					DAC2 (ЦАП2)
	D7	D6	D5	D4	D3	D2	D1	D0	Источник данных в УСО
Input: (вывод команды в УСО на передачу данных из источника УСО	0	1	0	0	0	x	x	x	ADC1 (АЦП1)
			0	0	1	(произвольные состояния)			ADC2 (АЦП2)
			0	1	0				ADC3 (АЦП3)
			0	1	1				ADC4 (АЦП4)
			1	0	0				Keys 3...0 (переключатели s3...s0)
			1	0	1				Keys 7...4 (переключатели s7...s4)
			1	1	x				Не используется

Обмен данными между персональным компьютером и микроконтроллером УСО в УМК-48 обеспечивается посредством программы **MONITOR.COM**. Команда должна содержать, в том числе, адрес COM-порта последовательного ввода-вывода: 3F8 для COM1; 2F8 для COM2. Команды записывают с использованием шестнадцатеричного кода.

Пример 1 – Используя порт COM1, вывести команду в микроконтроллер УСО для вывода на светодиодные индикаторы l3…l0 данных 0000.

Байт данных для ввода в УСО в двоичном коде имеет вид: 00000000.

Команда в программе **MONITOR.COM** в этом случае будет иметь вид:

o 3F8 00.

Пример 2 – Используя порт COM2, считать данные с АЦП4.

Байт данных для ввода в УСО в двоичном коде имеет вид: 01011xxx, например, 01011000. В этом случае команда в программе **MONITOR.COM** будет иметь вид:

o 2F8 58.

Для считывания в персональный компьютер данных от УСО о состоянии АЦП4 через порт COM2 используется команда:

i 2F8.

2.2.2 Задание для подготовки к практическому занятию

2.2.2.1 Изучить коды макрокоманд управления микроконтроллером УСО со стороны управляющей ЭВМ (персонального компьютера) [2, С. 163, 164].

2.2.2.2 Подготовить макрокоманды для обмена данными между микроконтроллером УСО и управляющей ЭВМ в соответствии с индивидуальным заданием (таблица 19).

2.2.3 Задание на практическое занятие

2.2.3.1 Исследовать работу устройства сопряжения с объектом при получении команд от управляющей ЭВМ.

2.2.4 Порядок выполнения заданий

2.2.4.1 Для выполнения заданий практического занятия:

1) перезагрузить систему с ядром DOS и вызвать программу-оболочку Volkov Commander (VC);

2) используя команду **MODE**, настроить COM-порт, к которому подключен узел МСКиУ. С этой целью в командной строке VC записать и выполнить команду:

mode com1: **9600**,**n**,**8**,**1**,

где 9600 – скорость приема-передачи порта COM1 (бит/с), n – отсутствие контрольного бита, 8 – число бит в посылке, 1 – длина стопового бита.

Примечание – Информацию о номере COM-порта, к которому подключен последовательный интерфейс связи с УСО, можно получить, запустив программу *WorkspaceDemo*;

3) запустить программу **MONITOR.COM** (находится в каталоге C:\VC\);

4) посредством инструкций программы **MONITOR.COM** произвести вывод команд управления узлом МСКиУ и команд ввода информации со стороны УСО, подготовленных при выполнении п. п. 2.2.2.2.

Убедиться в правильности выполнения макрокоманд микроконтроллером УСО.

Результаты выполнения заданий представить преподавателю.

2.2.5 Варианты индивидуальных заданий

Таблица 19 – Варианты индивидуальных заданий

Номера вариантов	Вывод данных	Ввод данных
1	l1,l3 = 1; DAC1 = 5	Keys 7…4
2	l4,l5 = 1; DAC2 = 12	ADC1
3	DAC1 = 13	Keys 3…0; ADC2
4	DAC1 = 6	Keys 7…4; ADC2
5	l0,l3 = 1; DAC2 = 15	Keys 3…0
6	DAC2 = 2	Keys 7…4; Keys 3…0;
7	l2,l3 = 1; DAC2 = 7	ADC1
8	l0…l3 = 1; DAC1 = 10	Keys 7…4
9	l4,l6 = 1	ADC1, ADC2
10	DAC1 = 9; l1,l2 = 1;	Keys 3…0
11	l1,l3 = 1; l5,l7 = 1;	ADC2
12	DAC1 = 3; DAC2 = 6	Keys 7…4
13	DAC1 = 11	ADC1; Keys 7…4
14	DAC2 = 12	Keys 7…4; Keys 3…0;
15	l0,l2 = 1; l5,l6 = 1	Keys 7…4

2.2.6 Вопросы для самоконтроля

2.2.6.1 Как используется аппаратное прерывание в системе обмена данными между микроконтроллером УСО и управляющей ЭВМ?

2.2.6.2 Как используются таймерные программы в системе обмена данными между микроконтроллером УСО и управляющей ЭВМ?

2.2.6.3 По какому интерфейсу осуществляется обмен данными между микроконтроллером УСО и управляющей ЭВМ?

2.2.6.4 Как реализуется преобразование из последовательного кода в параллельный при обмене данными между микроконтроллером УСО и управляющей ЭВМ?

2.2.6.5 Как реализуется преобразование данных из параллельного кода в последовательный?

2.2.6.6 Для чего служат буферы обмена данными?

2.2.6.7 Какие функции возлагаются на управляющую ЭВМ?

2.2.6.8 Дайте характеристику программному обеспечению управляющей ЭВМ.

2.2.6.9 С какой целью разрабатываются макрокоманды управления УСО?

3 Расчетно-графическая работа

Тема расчетно-графической работы (РГР): «Разработка и программная реализация алгоритма обработки результатов измерений в реальном времени».

Целью выполнения РГР является развитие у обучающихся способности обеспечивать программную реализацию алгоритмов обработки результатов измерений в реальном времени.

3.1 Задание для выполнения расчетно-графической работы

1 По индивидуальным заданиям (таблица 20) составить схему алгоритма функционирования УСО.

2 На основе схемы алгоритма написать программу на интерпретаторе виртуальной машины (ВМ), используя таблицы со списком аргументов интерпретатора (таблица 15) и списком команд интерпретатора (таблица 16).

Примечание – Нумерация строк в тексте программы начинается с нуля.

3 Используя учебно-отладочный стенд УМК-48 [2, С. 155 – 164], проверить функционирование УСО под управлением разработанной программы.

Таблица 20 – Варианты индивидуальных заданий

Номера вар-ов	Шаг					
1	1	2	3	4	5	6
	d5…d2=1; t=5	s3=1→6; (→7)	cap1=6	d3…d0=1; t=7	d6,d0=1; t=10	acp1=0…4 →3
	Шаг					
	7	8	9	10	11	12
	d4…d1=1; t=6	s5=1→10	d5,d0=1; t=7	cap1=2	s0=1→11	d3…d0=0, 1,0...
2	Шаг					
	1	2	3	4	5	6
	cap1=6; t=4	d7…d4=1; t=4	d3,d0=1; t=15	acp1=0…6 →4	d4…d1=1; t=3	s7=1→8
	Шаг					
	7	8	9	10	11	12
	d3…d1=1; t=5	d6,d4=1; t=8	acp1=0…3 →7	d4…d0=1; t=5	s1=1→11	d5,d3=0,1, 0…

Продолжение таблицы 20

Номера вар-ов	Шаг					
3	1	2	3	4	5	6
	s6=0→1; (→5)	cap2=9	d3…d1=1; t=10	d7,d0=1; t=6	acp1=0…8 →2	d4…d1=1; t=6
	Шаг					
	7	8	9	10	11	12
	d6…d3=1; t=5	s7=0→8	d3…d0=1; t=9	d6,d2=1; t=7	acp1=0...7 →11	d7…d4=0, 1,0...
4	Шаг					
	1	2	3	4	5	6
	acp1=0…2 →3	d5…d2=1; t=3	s6=0→1	d7,d0=1; t=6	cap2=8	d3…d0=1; t=4
	Шаг					
	7	8	9	10	11	12
	s3=0→6	cap2=4	d6…d4=1; t=8	d3,d0=1; t=10	acp1=0...5 →11	d6,d2=0,1, 0…
5	Шаг					
	1	2	3	4	5	6
	d5,d2=1; t=8	acp1=0…7 →1	d4…d0=1; t=8	s5=0→6; (→7)	d5,d0=1; t=4	s4=0→5
	Шаг					
	7	8	9	10	11	12
	d7…d4=1; t=5	d5,d1=1; t=6	d4,d0=1; t=7	cap2=9	s7=0→11	d7…d4=0, 1,0...
6	Шаг					
	1	2	3	4	5	6
	d7…d5=1; t=4	s7=1→4	d5…d3=1; t=4	d5,d3=1; t=4	acp1=0…3 →4	d6…d3=1; t=4
	Шаг					
	7	8	9	10	11	12
	s7=0→6; (→9)	cap1=7	d3…d0=1; t=4	d7,d3=1; t=4	s3=0→8	d2,d0=0,1, 0…
7	Шаг					
	1	2	3	4	5	6
	d4…d2=1; t=5	d7…d5=1; t=5	s7=0→1	d3…d0=1; t=3	d6,d1=1; t=3	acp1=0…3 →4
	Шаг					
	7	8	9	10	11	12
	d4…d0=1; t=6	s4=0→7; (→10)	d5,d0=1; t=4	s0=0→9	d6,d2=1; t=7	d3…d1=0, 1,0...
8	Шаг					
	1	2	3	4	5	6
	s2=0→1	d5,d0=1; t=6	cap1=9	d3…d1=1; t=3	s3=1→7	cap2=7
	Шаг					
	7	8	9	10	11	12
	d6…d3=1; t=5	acp1=0…5 →7	d4…d1=1; t=3	cap2=6	s4=0→10	d7,d5= 0,1,0…

Продолжение таблицы 20

Номера вар-ов	Шаг					
9	1	2	3	4	5	6
	acp1=1...2 →4	d6…d3=1; t=6	s6=0→2; (→5)	cap1=6	d3…d0=1; t=6	d5,d3=1; t=5
	Шаг					
	7	8	9	10	11	12
	d4…d2=1; t=5	s3=0→6	d7,d5=1; t=6	cap2=5	s1=1→11	d5…d2=0, 1,0...
10	Шаг					
	1	2	3	4	5	6
	d4,d0=1; t=8	acp1=0…7 →1	d6…d3=1; t=7	s7=1→6	d5,d3=1; t=6	s5=0→5; (→8)
	Шаг					
	7	8	9	10	11	12
	d5,d0=1; t=4	s3=0→7	d6…d4=1; t=5	d6,d2=1; t=7	s1=0→10	d4,d2=0,1, 0…
11	Шаг					
	1	2	3	4	5	6
	d3…d0=1; t=7	d6,d0=1; t=10	acp1=0…4 →2	d4…d1=1; t=6	s5=0→4	d5,d0=1; t=7
	Шаг					
	7	8	9	10	11	12
	cap1=2	s0=1→7; (→10)	cap2=5	d7,d5=1; t=6	s6=0→9	d6,d3=0,1, 0…
12	Шаг					
	1	2	3	4	5	6
	d3,d0=1; t=15	acp1=0…2 →2	d4…d1=1; t=3	s7=0→4	d3…d1=1; t=5	d6,d4=1; t=8
	Шаг					
	7	8	9	10	11	12
	acp1=6…9 →7	d4…d0=1; t=5	s1=1→8	s0=0→9	d6,d2=1; t=7	d3…d1=0, 1,0...
13	Шаг					
	1	2	3	4	5	6
	d5…d2=1; t=3	s6=0→1	d7,d0=1; t=6	cap2=8	d3…d0=1; t=4	s3=0→6
	Шаг					
	7	8	9	10	11	12
	cap2=4	d6…d4=1; t=8	d3,d0=1; t=10	acp1=0...5 →8	s1=0→11	d4,d1=0,1, 0…
14	Шаг					
	1	2	3	4	5	6
	s7=0→1	d3…d0=1; t=3	d6,d1=1; t=3	acp1=0…3 →4	d4…d0=1; t=6	s4=0→5; (→8)
	Шаг					
	7	8	9	10	11	12
	d5,d0=1; t=4	s0=0→7	d6,d2=1; t=7	cap1=9	s5=0→9	d7…d4=0, 1,0...

Продолжение таблицы 20

Номера вар-ов	Шаг					
15	1	2	3	4	5	6
	cap1=8	d3…d0=1; t=4	s3=0→2	cap2=4	d6…d4=1; t=8	d3,d0=1; t=10
	Шаг					
	7	8	9	10	11	12
	acp1=0...5 →7	s4=0→6; (→10)	d5,d0=1; t=4	s0=0→9	d6,d2=1; t=7	d3…d1=0, 1,0...

Примечания

1 Запись в скобках «→*n*» указывает номер шага (строки в тексте программы) *n* перехода при невыполнении условия перехода.

2 Состояние «=0,1,0…» – режим мерцания (поочередное включение и выключение светодиодных индикаторов).

3 Переменная t = *x* – время задержки в секундах.

4 Значения выходных переменных d_i устанавливать командой D_ALL.

5 В начале алгоритма предусмотреть инициализацию УСО (все $d_i = 0$, $cap_i = 0$).

6 Завершить алгоритм бесконечным циклом мерцания заданных индикаторов.

3.2 Вопросы для самоконтроля

3.3.1 Поясните особенности программирования микропроцессорной системы устройства сопряжения с объектом контроля.

3.3.2 Какой язык программирования используется при разработке программного обеспечения микропроцессорной системы на первом уровне программирования?

3.3.3 Назовите основные виды подпрограмм, используемых в микропроцессорной системе УСО.

3.3.4 Для решения каких задач в составе программного обеспечения микропроцессорной системы УСО используется интерпретатор?

3.3.5 Что может быть использовано в качестве аргументов при написании программы на интерпретаторе виртуальной машины?

3.3.6 Приведите примеры операторов условного и безусловного переходов интерпретатора виртуальной машины.

3.3.7 Для чего используют операторы SET и WAIT?

3.3.8 Поясните особенности записи текста программы на интерпретаторе виртуальной машины.

Список использованных источников

1 Боборыкин, А. В. Однокристальные микроЭВМ / А. В. Боборыкин, Г. П. Липовецкий, О. Н. Оксинь и др. – М. : МИКАП, 1994. – 400 с. – ISBN 5-85959-030-X.

2 Булатов, В. Н. Элементы и узлы информационных и управляющих систем (основы теории и синтеза) [Электронный ресурс] : учеб. пособие / В. Н. Булатов; М-во образования Рос. Федерации, Гос. образоват. учреждение высш. проф. образования «Оренбург. гос. ун-т». – Электрон. текстовые дан. (1 файл: 3.84 Мб). – Оренбург : ГОУ ОГУ, 2002. – Загл. с тит. экрана. – Adobe Acrobat Reader 5.0. – Режим доступа: http://artlib.osu.ru/site_new/find-book.

3 Дьяков, И. А. Микропроцессорные системы. Архитектура микроконтроллеров семейства MCS-51 [Электронный ресурс] / И. А. Дьяков ; М-во образования и науки Рос. Федерации, Федеральное гос. бюджетное образоват. учреждение высш. проф. образования «Тамбовский гос. техн. ун-т». – Тамбов : Издательство ФГБОУ ВПО «ТГТУ», 2014. – 79 с. – Режим доступа: http://biblioclub.ru/index.php?page=book&id=277684. – ЭБС «Университетская библиотека онлайн» (14.01.2018).

4 Каспер, Э. Программирование на языке Ассемблера для микроконтроллеров семейства i8051 / Э. Каспер. – М. : Горячая линия – Телеком, 2004. – 191 с. – ISBN 5-93517-104-X.

5 Магда, Ю. С. Микроконтроллеры серии 8051 : практический подход / Ю. С. Магда. – М. : ДМК Пресс, 2008. – 228 с. – ISBN 5-94074-394-3.

6 Макаренко, В. Моделирование работы микроконтроллеров в программе NI Multisim [Электронный ресурс] /В. Макаренко, А. Бабко. – Режим доступа: http://www.ekis.kiev.ua/UserFiles/Image/pdfArticles/V.Makarenko_Babko_MODELLING_OF_MICROCONTROLLERS_in_Multisim_EKIS_4_2012.pdf.

7 Сильвашко, С. А. Программные средства компьютерного моделирования элементов и устройств электроники : учебное пособие / С. А. Сильвашко, С. С. Фролов; Оренбургский гос. ун-т. – Оренбург : ОГУ, 2014. – 170 с. – ISBN 978-5-4417-0454-0.

8 Шегал, А. А. Применение программного комплекса Multisim для проектирования устройств на микроконтроллерах [Электронный ресурс] : лабораторный практикум / А. А. Шегал ; М-во образования и науки Рос. Федерации, Уральский федеральный ун-т им. первого Президента России Б. Н. Ельцина ; науч. ред. В. И. Иевлев. – Екатеринбург : Издательство Уральского университета, 2014. – 116 с. – ISBN 978-5-7996-1117-0. – Режим доступа: http://biblioclub.ru/index.php?page=book&id=276471. ЭБС «Университетская библиотека онлайн». (14.01.2018).