ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Русанов В.В., Шевелев М.Ю.

МИКРОПРОЦЕССОРНЫЕ УСТРОЙСТВА И СИСТЕМЫ

Руководство к выполнению лабораторных
работ для студентов
направления 210100 «Электроника и наноэлектроника»

Томск - 2012

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра промышленной электроники (ПрЭ)

УТВЕРЖДАЮ
Зав. каф. ПрЭ
_____ А.В. Кобзев

МИКРОПРОЦЕССОРНЫЕ УСТРОЙСТВА И СИСТЕМЫ

Руководство к выполнению лабораторных
работ для студентов
направления 210100 «Электроника и наноэлектроника»

Разработчики:
Доцент каф. ПрЭ
_____ М.Ю. Шевелев

Ст. преп. каф. ПрЭ
_____ В.В. Русанов

2012

Лабораторная работа № 1
ПРОГРАММНАЯ МОДЕЛЬ И СИСТЕМА КОМАНД ЛАБОРАТОРНОГО СТЕНДА SDK 1.1

Цель работы. Целью работы является приобретение студентами первоначальных навыков работы с микроконтроллерами семейства MSC-51 на примере лабораторного стенда SDK 1.1. Работа включает знакомство с архитектурой комплекса: микроконтроллером ADuC842, его программной моделью, встроенной периферией, а также с программными средствами разработки и отладки прикладных программ.

1 Общие сведения о лабораторном стенде SDK 1.1

Лабораторный стенд SDK 1.1 предназначен для освоения студентами архитектуры и методов проектирования, создания и комплексной отладки систем на базе микроконтроллеров семейства MSC-51. Схематическое изображение основных блоков приведено на рисунке 1.

Комплектность учебного лабораторного стенда SDK 1.1:

- лабораторный макет;
- блок питания лабораторного стенда;
- интерфейсный кабель для загрузки прикладных программ из компьютера в память стенда.

Состав лабораторного макета:

- микроконтроллер ADuC842BS;
- внешняя EEPROM объемом 256 байт;
- клавиатура АК1604А-WWB фирмы ACCORD;
- жидкокристаллический индикатор (ЖКИ) WH1602B-YGK-CP фирмы WinStar Display.
- часы реального времени PCF8583;
- 128K памяти внешней SRAM с возможностью расширения до 512K;
- набор сигнальных светодиодов.
- программируемая логическая интегральная схема (ПЛИС) MAX 8064.
- звуковой пьезокерамический излучатель;
- расширитель портов ввода-вывода.

2 Общий вид стенда SDK 1.1

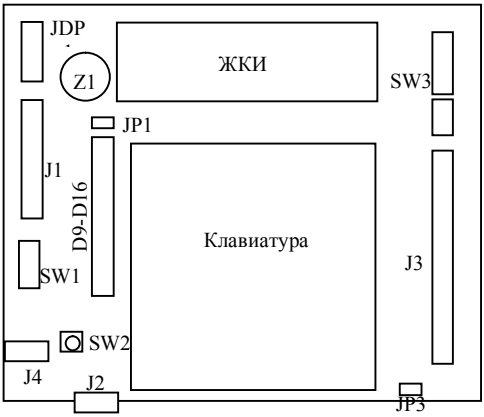

Рисунок 1 - Схематическое изображение лицевой панели стенда SDK

Таблица 1 – Расшифровка обозначений

Элемент	Описание
Z1	Звуковой пьезокерамический излучатель
SW1	Переключатель, замыкающий каналы 0 и/или 1 ЦАП на входы (0,1) каналов АЦП
SW2	Кнопка сброса RESET
SW3	Набор переключателей, замыкающих выводы J3 на корпус (переключение в лог. «0»)
J1	Выводы каналов АЦП и ЦАП
J2	Выводы JTAG-интерфейса ПЛИС MAX
J3	16 линий (HI, LO) параллельного порта ПЛИС MAX и 4 линии параллельного порта P3 МК ADuC842 (INT0, INT1, T0, T1)
JP1	Перемычка, замыкающая вывод PSEN МК ADuC842 на корпус для реализации функции программирования
JP3	Разъем подкл. внешней батареи питания часов PCF8583
JDP1	Разъем последовательного порта
D9-D16	Набор сигнальных светодиодов

3 Краткое описание основных модулей лабораторного стенда

Полное описание возможностей стенда приведено в «Описании лабораторного стенда SDK 1.1», прилагающегося к лабораторному стенду в печатном и электронном виде.

3.1 Микроконтроллер ADuC842BS

Микроконтроллер ADuC842BS является клоном Intel 8051 со встроенной периферией.

Основные характеристики.

3.1.1 8051-совместимое ядро:

- рабочая частота 11.0592 МГц;
- три 16-битных таймера счетчика;
- 32 программируемых линий порта ввода-вывода, организованных как четыре 8-битных параллельных порта;
- порт 3 с высокой нагрузочной способностью;
- 9 источников прерываний.

3.1.2 Встроенная аналоговая периферия:

- 8-канальный прецизионный 12-битный АЦП со скоростью выборки 200 кбит/с (в режиме ПДП);
- Два 12-битных ЦАП (код-напряжение);
- внутренний температурный сенсор.

3.1.3 Память:

- 8 Кбайт внутренней Flash памяти программ;
- 640 байт программируемого EEPROM со страничной организацией (256 страниц по 4 байта);
- 256 байт внутренней памяти данных;
- адресное пространство внешней памяти 16 Мбайт;
- встроенный программатор (работающий через UART).

3.1.4 Питание:

- напряжение питания 3 или 5 В;
- 3 режима управления питанием (нормальный, холостой, пониженый);

3.1.5 Встроенная цифровая периферия:

- асинхронный двунаправленный последовательный порт (UART);
- аппаратные интерфейсы I^2C, SPI;
- монитор источника питания;
- сторожевой таймер WatchDogTimer (WDT).

3.2 Внешняя EEPROM

EEPROM – перепрограммируемое электрически стираемое постоянное запоминающее устройство. Объем памяти EEPROM, установленной в стенде SDK 1.1, составляет 128 байт. Микросхема EEPROM взаимодействует с процессором посредством интерфейса I^2C.

Таблица 3.1 – Адрес устройства I^2C EEPROM

Бит	7	6	5	4	3	2	1	0
Значение	1	0	1	0	0	0	1	R/W

3.3 Матричная клавиатура АК1604A-WWB

Клавиатура организована в виде матрицы 4x4. Доступ к колонкам и рядам организован как чтение/запись регистра КВ, размещенного во внешней памяти (в ПЛИС) по адресу 80000H. Значение после сброса равно 00000000B.

Таблица 3.2 – Структура регистра КВ

Бит	7	6	5	4	3	2	1	0
Операция	R	R	R	R	W	W	W	W
Тетрада	СТРОКА				СТОЛБЕЦ			

Работа с клавиатурой производится по следующему алгоритму: в строки записываются все «единицы», в столбцы выдается «бегущий нуль», а затем производится чтение строк. Если в какой-либо строке обнаружен «нуль», следовательно, была нажата какая-то клавиша. Зная номер столбца, в который был выдан «бегущий нуль» и номер строки, в которой обнаружен «нуль» при чтении, можно однозначно определить, какая именно была нажата клавиша.

3.4 ЖКИ WH1602B-YGK-CP

Жидкокристаллический индикатор работает в текстовом режиме и способен отображать две строки по 16 знаков каждая.

Чтение/запись в регистр данных ЖКИ производится по адресу 80001H внешней памяти (ПЛИС). Запись в регистр управления ЖКИ производится по адресу 80006H внешней памяти.

Полная информация о работе с ЖКИ приведена в «*Описании лабораторного стенда SDK 1.1»* (стр. 14-16, 32-45).

3.5 Часы реального времени PCF8583

PCF8583 – часы\календарь с памятью объемом 256 байт, работающие от внешнего кварцевого резонатора с частотой 32,768 кГц. Питание осуществляется ионистором (0,1 Ф). Из 256 байт памяти собственно часами используются только первые 16 (восемь постоянно обновляемых регистров-защелок на установку/чтение даты/времени и восемь - на будильник). Остальные 240 байт доступны для хранения данных пользователя. Точность измерения времени – до сотых долей секунды. Взаимодействие с процессором осуществляется через интерфейс I^2C.

Таблица 3.3 – Адрес устройства I^2C PCF8583

Бит	7	6	5	4	3	2	1	0
Значение	1	0	1	0	0	0	1	R/W

3.6 Линейка светодиодов

Линейка светодиодов состоит из восьми светодиодов и предназначена для отображения цифровых сигналов. Вывод на светодиоды производится путем записи байта в регистр SV, размещенный во внешней памяти (в ПЛИС) по адресу 80007H. Светящийся светодиод соответствует логической «единице».

Значение регистра после аппаратного сброса микроконтроллера равно 00000000B.

4 Распределение памяти в SDK 1.1

Память в SDK 1.1 распределяется следующим образом:

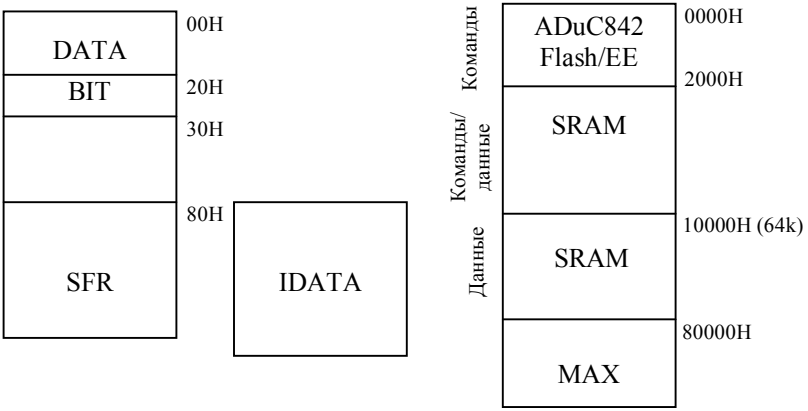

Рисунок 2 – Карта памяти SDK 1.1

Стандартная для архитектуры 8051 структура внутренней памяти представлена четырьмя банками по восемь регистров общего назначения (00H-07H, 08H-0FH, 10H-17H, 18H-1FH), битовым сегментом (20H-2FH), свободным участком (30H-7FH), областью размещения SFR (регистров специальных функций) (80H-FFH), доступной при прямой адресации и свободной областью 80H-FFH, доступной при косвенной адресации.

Внешняя память SDK 1.1 разбита на следующие области: ADuC842 Flash/EE, SRAM, MAX.

4.1 ADuC842 Flash/EE

Это область, в которой размещена таблица векторов прерываний и резидентный загрузчик файлов в формате HEX в память SRAM.

4.2 SRAM

Статическая память SRAM в SDK 1.1 имеет страничную организацию (макс. восемь страниц по 64 Кбайт) и условно разделена на две области. Первая занимает младшие 64 Кбайт (адреса 0H-0FFFFH - страница 0) и доступна для выборки команд микроконтроллером ADuC842. Таким образом,

программы могут располагаться только в этих младших 64 Кбайт адресного пространства (за вычетом 8 Кбайт Flash-памяти ADuC842, которая отображается в самые младшие адреса (0000H-1FFFH). Фактически для размещения программ доступно 56 Кбайт статической памяти). Остальные страницы доступны только для размещения данных. Для адресации ячейки памяти определенной страницы необходимо записать номер страницы в регистр DPP ADuC842 (адрес 84H в области SFR ADuC842).

4.3 Программируемая логическая микросхема MAX8064

В младших адресах восьмой страницы адресного пространства (80000H-80007H) располагается восемь ячеек-регистров ПЛИС MAX8064. Эта область предназначена для взаимодействия с периферийными устройствами.

5 Карта портов ввода-вывода

В стенде SDK 1.1 ввод-вывод данных осуществляется с помощью портов микроконтроллера и портов микросхемы ПЛИС, которая имеет восемь регистров, отображаемых во внешнее адресное пространство процессора.

Карта портов ввода-вывода приведена в «Описании лабораторного стенда SDK 1.1».

6 Основные этапы работы со стендом

Для программирования стенда может использоваться любой транслятор ассемблера или Си для ядра MK51. В данном лабораторном цикле используются пакеты как для DOS, так и для Windows.

Основные этапы программирования стенда:

- подготовка программы в текстовом редакторе или в среде программирования;
- транслирование исходного текста и получение загрузочного HEX-модуля программы;
- подготовка и загрузка HEX-модуля в стенд через интерфейс RS232 с помощью резидентного загрузчика WSD. Перед загрузкой программы в стенд необходимо установить перемычку JP1 и нажать кнопку Reset на стенде;

- передача управления загруженной программе. Для запуска загруженной программы на выполнение необходимо удалить перемычку JP1 и нажать кнопку «Reset».

7 Запись программы на языке ассемблера и ее трансляция с помощью макроассемблера x8051

В данной работе используется макроассемблер x8051.

Последовательность действий при работе с макроассемблером (пункты 2 и 3 приведены для удобства отладки программы).

1) создать текст исходной программы на ассемблере в любом текстовом редакторе (например test.asm);

2) для удобства дальнейшей работы создать исполняемый файл compile.bat, содержащий следующий текст:

========= текст файла compile.bat ==========

```
x8051 -q test -d
pause
xlink xlink.lnk
pause
```

========= конец текста файла compile.bat

3) создать файл xlink.lnk, содержащий текст *:

========= текст файла xlink.lnk ==========

```
test.obj
_
_
_
_
h
```

========= конец текста файла xlink.lnk ==========

* Примечание: в тексте использован знак «_», а не «–».

Важное замечание: при компиляции ассемблер выдает сообщение вида:

```
x     xxxx   xx xx xx                    mov  84H, 0xxH
***** NON-EXISTENT  INTERNAL  RAM  ADRESS *****.
```

Ассемблеру неизвестен микроконтроллер ADuC и его адрес из области регистров специальных функций, поэтому он выдает сообщение, что адрес 84H не является адресом регистра SFR. Это не ошибка, а только предупреждение. HEX-файл будет создан.

4) запустить на выполнение файл compile.bat. При отсутствии ошибок на экране появится надпись «Press any key». После нажатия «ENTER» на экране появятся сообщения линкера Link.exe и снова надпись «Press any key». После нажатия «ENTER» в директории с исходным asm-файлом появится файл test.hex.

5) получить доступ к COM порту компьютера. Для этого в командной строке необходимо набрать следующую строку:

net use com1: \\tsclient\com1

6) загрузить полученный HEX-файл в стенд, используя программу WSD.exe.

Для этого необходимо выполнить следующие действия:

- соединить лабораторный стенд с компьютером посредством кабеля и включить питание стенда;
- перевести микроконтроллер ADuC стенда в режим программирования. Для этого установить перемычку JP1 и нажать на стенде кнопку «Reset». Микроконтроллер будет ожидать загрузки программы по последовательному порту;
- выбрать в списке установленных программ компьютера (меню «Пуск – Программы») пункт ADuC, а в нем – WSD. На дисплее появится окно программы;
- нажать кнопку «Configuration» и в открывшемся окне установить следующие параметры:

  Port: Com1
  Crystal Frequency: 11,059 MHz crystal
  Erase Mode: Erase the CODE ONLY
  Download Mode: Download CODE ONLY
  Security Mode: снять все флажки
  Run From Start 0 Hex (и снять все флажки в окне Run)

- нажать кнопку ОК и вернуться в окно программы WSD.

- нажать клавишу RESET в окне программы. В окне ниже появится надпись Configuration: COM1, 9600 baud RESETTING PART: Ok.

- нажать клавишу Download и загрузить программу в HEX-формате в стенд.

- удалить на стенде перемычку JP1 и нажать на стенде кнопку RESET. Пользовательская программа начнет выполняться.

8 Программа работы

8.1. Создать программу test.asm. Данная программа реализует эффект «ёлочка», отображая его на светодиодах стенда.

```
================ файл test.asm ================
        ORG     2100H   ; Нач. адрес размещения программы
START:  MOV     DPTR, #0007h ; установка номера регистра
светодиодов
        MOV     84H,#08h ; Установка номера страницы
внешней памяти (в данном случае это регистры ПЛИС)
        MOV     A,R6
        INC     A       ; Увеличение счетчика на 1
        MOV     R6,A
        MOVX    @DPTR,A   ; выдача данных в порт
светодиодов
        CALL    DELAY ; вызов задержки
        JMP     START  ; зацикливание
DELAY:
        DJNZ    R2,$    ; задержка с помощью вложенных
циклов
        DJNZ    R3,DELAY
        RET             ; Возврат из подпрограммы DELAY
END
============== Конец файла test.asm ================
```

Загрузить данную программу в стенд и пронаблюдать реализуемый эффект. Что нужно сделать, чтобы изменить скорость переключения эффекта?

8.2. Создать программу Keyboard.asm. Данная программа демонстрирует работу с клавиатурой и светодиодами стенда.

```
================ файл Keyboard.asm ================
        ORG     2100H   ; Нач. адрес размещения программы
        MOV     84H,#08h    ; Установка номера страницы внешней памяти
M1:     MOV     DPTR, #0000h ; Установка номера регистра клавиатуры
        MOV     A,#11111110B    ;
        MOVX    @DPTR,A     ; Установка строк в единицы, и выдача «0» в один столбец;
        NOP                 ; Задержка
        MOVX    A,@DPTR     ; Чтение регистра клавиатуры
        MOV     DPTR,#07H ; Установка номера регистра светодиодов
        MOV     @DPTR,A     ; Вывод
        JMP     M1
END
================ конец файла Keyboard.asm ===========
```

Загрузить программу в стенд и нажать последовательно несколько клавиш на клавиатуре стенда. Что будет отображаться на светодиодах? Объяснить, почему. Что произойдет, если на клавиатуре нажать одновременно две клавиши.

8.3. В соответствии с заданным преподавателем вариантом написать, отладить и запустить на выполнение программу:

а) после нажатия кнопки Reset на светодиодах реализован эффект «бегущий огонек». При нажатии на клавишу «2» клавиатуры огонек бегает в 2 раза быстрее. При нажатии на «1» огонек опять бегает с заданной скоростью.

б) после нажатия кнопки Reset на светодиодах реализован эффект «елочка». При нажатии на клавишу «3» реализуется инверсный эффект «елочка». При нажатии на «4» - эффект исходный;

в) после нажатия кнопки Reset на светодиодах реализован эффект «бегущий огонек». При нажатии на клавишу «5» огонек меняет направление. При нажатии на «6» огонек опять бежит в прежнюю сторону;

г) после нажатия кнопки Reset на светодиодах реализован эффект «бегущий огонек». При каждом нажатии на клавишу «*» огонек меняет направление;

д) после нажатия кнопки Reset на светодиодах реализован эффект «бегущий огонек». При нажатии на клавишу «7» огонек начинает бегать через один. При нажатии на «8» огонек бегает опять как в начале.

е) после нажатия кнопки Reset на светодиодах реализован эффект «бегущий огонек». При нажатии на клавишу «9» реализуется эффект «инверсный бегущий огонек». При нажатии на клавишу «0» - исходный эффект.

9 Контрольные вопросы

9.1 Привести команду чтения данных из расширенной области ОЗУ МК ADuC842 (адреса 80H-FFH).

9.2 Почему для записи программ в SRAM доступно только адресное пространство с 2000H по 0FFFFH?

9.3 Каким образом можно обратиться к адресам выше 0FFFFH? Привести фрагмент программы.

9.4 Можно ли обратиться напрямую к портам МК ADuC842 в составе лабораторного стенда SDK 1.1? Если да, то к каким именно?

Лабораторная работа № 2
ИССЛЕДОВАНИЕ РАБОТЫ ЗНАКОГЕНЕРИРУЮЩЕГО ЖИДКОКРИСТАЛЛИЧЕСКОГО ИНДИКАТОРА (ЖКИ)

Цель работы: ознакомиться с принципом работы знакогенерирующего ЖКИ, основанном на контроллере HD44780, научиться программировать контроллер HD44780 и выводить информацию на ЖКИ;

1 Общие сведения о ЖКИ

Контроллер HD44780 фирмы Hitachi фактически является промышленным стандартом и широко применяется при производстве алфавитно-цифровых ЖКИ-модулей. Аналоги этого контроллера или совместимые с ним по интерфейсу и языку выпускают множество зарубежных фирм. Еще большее число фирм производят ЖКИ-модули на базе данных контроллеров. Эти модули можно встретить в самых разнообразных устройствах.

Алфавитно-цифровые ЖКИ-модули представляют собой недорогое и удобное решение, позволяющее сэкономить время и ресурсы при разработке новых изделий, при этом обеспечивают отображение большого объема информации при хорошей различимости и низком энергопотреблении.

Контроллер HD44780 потенциально может управлять двумя строками по 40 символов в каждой (для модулей с четырьмя строками по 40 символов используются два одинаковых контроллера), при матрице символа 5 х 7 точек.

Существует несколько более-менее стандартных форматов ЖКИ-модулей: 8 х 2, 16 х 1, 16 х 2, 16 х 4, 20 х 1, 20 х 2, 20 х 4, 24 х 2, 40 х 2, 40 х 4. Есть и другие форматы, встречающиеся достаточно редко.

Полное описание режимов работы ЖКИ приведено в «Описании лабораторного стенда SDK 1.1», прилагающегося к лабораторному стенду в печатном и электронном виде.

2 Подключение ЖКИ

Для соединения ЖКИ-модуля с управляющим МК используется параллельная синхронная шина, насчитывающая:

- 8 или 4 (выбирается программно) линий данных DB0..DB7;
- линию выбора операции R/W;
- линию выбора регистра RS;
- линию синхронизации E.

Кроме линий управляющей шины имеются две линии для подачи напряжения и линия для подачи напряжения драйвера ЖКИ – V_0.

3 Краткое описание процедуры записи информации в ЖКИ для 8-разрядной шины:

1. Установить значение линии RS (0 или 1).
2. Установить значение линии R/W = 0.
3. Вывести значение байта данных на линии шины DB.
4. Установить линию E = 1.
5. Установить линию E = 0.
6. Установить линии шины DB = 1.

4 Краткое описание процедуры чтения информации из ЖКИ для 8-разрядной шины:

1. Установить значение линии RS (0 или 1).
2. Установить значение линии R/W = 1.
3. Установить линию E = 1.
4. Считать значение байта данных с линий шины DB.
5. Установить линию E = 0.
6. Установить значение линии R/W = 0.

5 Рекомендации по работе с ЖКИ.

1. После включения питания необходимо провести **процедуру инициализации** индикатора. Эту процедуру рекомендуют выполнять производители контроллера HD44780.

Последовательность действий при процессе инициализации:

1) выдержать паузу не менее 15 мс между установлением рабочего напряжения и работой с контроллером ЖКИ;

2) в регистр команд записать управляющее слово 30H, которое будет настраивать работу ЖКИ, не проверяя значение флага занятости ЖКИ BF.

3) выполнить задержку не менее, чем на 4,1 мс;

4) снова записать в регистр команд управляющее слово 30H, не проверяя флаг BF;

5) выполнить задержку не менее, чем на 100 мкс;

6) снова записать в регистр команд управляющее слово 30H, не проверяя флаг BF.

В результате этих действий дисплей выйдет на нормальный режим работы из любого состояния.

Так как на момент включения индикатор ничего не отображает, то необходимо хотя бы включить изображение, установив флаг D.

Пример широко распространенной последовательности для инициализации ЖКИ: 38H, 0CH, 06H. Первая команда устанавливает режим отображения 2-х строк с матрицей 5x7 точек и работу с 8-разрядной шиной данных; вторая команда включает изображение на экране ЖКИ без отображения курсоров; третья команда устанавливает режим перемещения курсора слева направо после вывода каждого символа.

```
; ==== Подпрограмма записи в ЖКИ для 8-битной передачи ===

; В регистре R0 находятся данные для вывода на ЖКИ
; @DPTR – порт DB ЖКИ
; F0 – бит для установки типа команды ( F0 = 1 - запись в р-р данных DR, F0 = 0 - запись регистр команд IR)

SAVE_LCD:
    PUSH   ACC
    MOV    84H,#08H ; Устанавливаем 8-ю страницу внешней памяти
    MOV    DPTR,#0001H ; Устанавливаем адрес р-ра шины данных ЖКИ
    MOV    A,R0
    MOVX   @DPTR,A ; Выставляем данные на шину данных ЖКИ
    MOV    DPTR,#0006H ; Устанавливаем адрес регистра управления ЖКИ
    MOV    A,#00000001B ; RS = 0, R/W = 0, E = 1
    JNB    F0,S_LCD1 ; Если <>1, то пишем в р-р команд
```

```
    MOV    A,#00000101B ; RS = 1, R/W = 0, E = 1
S_LCD1:
    MOVX   @DPTR,A    ; Выводим данные в ЖКИ
    CALL   DELAY
    MOV    A,#0H    ; Устанавливаем бит E = 0
    MOVX   @DPTR,A    ; Завершаем вывод данных в ЖКИ
    POP    ACC
    RET
```

Пример использования подпрограммы SAVE_LCD для записи в регистр команд IR.

```
    CLR    F0
    MOV    R0,#38H
    CALL   SAVE_LCD
    CALL   DELAY
```

Пример использования подпрограммы SAVE_LCD для записи в регистр данных DR.

```
    SETB   F0
    MOV    R0,#38H
    CALL   SAVE_LCD
    CALL   DELAY
```

6 Программа работы

6.1. Реализовать программу инициализации ЖКИ-модуля. В результате загрузки программы в стенд ЖКИ должен перейти в режим развертки двух строк.

6.2 Вывести на индикатор строку, содержащую свою фамилию.

7 Контрольные вопросы.

7.1 Привести функциональную схему подключения ЖКИ к микроконтроллеру семейства МК51.

7.2 Привести функциональную схему подключения ЖКИ к питанию.

7.3 Чем осуществляется регулировка контрастности ЖКИ?

7.4 Пояснить, что такое флаг занятости BF и для чего он нужен.

ЛАБОРАТОРНАЯ РАБОТА № 3. ИССЛЕДОВАНИЕ РЕЖИМОВ РАБОТЫ ПОСЛЕДОВАТЕЛЬНОГО ПОРТА (UART)

Цель работы:

Изучение различных режимов работы последовательного порта.

1 Описание лабораторной работы

Таблица 1. Регистр управления/статуса UART

Сим-вол	Позиция	Имя и назначение
SM0 SM1	SCON.7 SCON.6	Биты управления режимом работы UART. Устанавливаются / сбрасываются программно (см. Примечание)
SM2	SCON.5	Бит управления режимом UART. Устанавливается программно для запрета приема сообщения, в котором девятый бит имеет значение 0
REN	SCON.4	Бит разрешения приема. Устанавливается / сбрасывается программно для разрешения/запрета приема последовательных данных
TB8	SCON.3	Передача бита 8. Устанавливается/сбрасывается программно для задания девятого передаваемого бита в режиме UART-9 бит
RB8	SCON.2	Прием бита 8. Устанавливается/сбрасывается аппаратно для фиксации девятого принимаемого бита в режиме UART-9 бит
TI	SCON. 1	Флаг прерывания передатчика. Устанавливается аппаратно при окончании передачи байта. Сбрасывается программно после обслуживания прерывания
RI	SCON.0	Флаг прерывания приемника. Устанавливается аппаратно при приеме байта. Сбрасывается программно после обслуживания прерывания

Примечание.

SM0	SM1	Режим работы UART
0	0	Сдвигающий регистр расширения ввода/вывода
0	1	UART-8 бит. Изменяемая скорость передачи
1	0	UART-9 бит. Фиксированная скорость передачи
1	1	UART-9 бит. Изменяемая скорость передачи

2 Скорость приема/передачи

Скорость приема/передачи, т.е. частота работы UART в различных режимах, определяется различными способами.

В режиме 0 частота передачи зависит только от резонансной частоты кварцевого резонатора $f_0 = f$ рез / 12. За один машинный цикл последовательный порт передает один бит информации.

В режимах 1, 2 и 3 скорость приема/передачи зависит от значения управляющего бита SMOD в регистре специальных функций PCON.

В режиме 2 частота передачи определяется выражением $f_2 = (2^{SMOD}/64)f$ рез. Иными словами, при SMOD = 0 частота передачи равна (1/64) f рез, а при SMOD = 1 равна (1/32) f рез.

В режимах 1 и 3 в формировании частоты передачи кроме управляющего бита SMOD принимает участие таймер 1. При этом частота передачи зависит от частоты переполнения (OVT1) и определяется следующим образом: $f_{1,3} = (2^{SMOD}/32)f_{OVT1}$. Прерывание от таймера 1 в этом случае должно быть заблокировано. Сам TCNT1 может работать и как таймер, и как счетчик событий в любом из трех режимов. Однако наиболее удобно использовать режим таймера с автоперезагрузкой (старшая тетрада TMOD = 0010B). При этом частота передачи определяется выражением

$f_{1,3} = (2^{SMOD}/32)*(f$ рез / 2) /(256 - (TH1)).

В табл. 2 приводится описание способов настройки T/C1 для получения типовых частот передачи данных через UART.

Таблица 2. Настройка таймера 1 для управления частотой работы UART

Частота приема/передачи (BAUD RATE)	Частота резонатора, МГц	SMOD	C/T	Режим	Перезагружаемое число
Режим 0, макс: 1 МГц	12	X	X	X	X
Режим 2, макс: 375 кГц	12	1	X	X	X
Режимы 1,3:					
62,5 кГц	12	1	0	2	0FFH
19,2 кГц	11,059	1	0	2	0FDH
9,6 кГц	11,059	0	0	2	0FDH
4,8 кГц	11,059	0	0	2	0FAH
2,4 кГц	11,059	0	0	2	0F4H
1,2 кГц	11,059	0	0	2	0E8H
137,5 Гц	11,059	0	0	2	1DH
110 Гц	6	0	0	2	72H
110 Гц	12	0	0	1	0FEEBH

Примечание. В лабораторном стенде $f_{рез}$ = 11,059 МГц

3 Описание тестовых подпрограмм

3.1 Тестовая подпрограмма передачи байта данных по последовательному порту в режиме 1.

Число 0FH передается по последовательному порту и по срабатыванию прерывания от передатчика отображается на светодиодах.

```
; ========== Текст подпрограммы Transmitter ==============
        ORG     0
        JMP     START ; Переход на основную программу, минуя область векторов прерываний
        ORG     23H ; Адрес вектора прерывания от UART
        MOV     84H,#08H
        MOV     DPTR,#0007H
        MOVX    @DPTR,A ; Вывод на светодиоды переданного байта данных
```

```
        RETI  ; Выход из подпрограммы обслуживания прерывания
START:  MOV   IE,#10010000B ; Снятие общей блокировки прерываний и разрешение прерывания от UART
        MOV   SCON,#01110000B ; Установка режима UART
        MOV   TH1,#0FDH ; Число для перезагрузки
        SETB  TR1 ; Запуск таймера на счет
        MOV   A,#00001111B
        MOV   SBUF,A ; Отправление байта по UART
        JMP   $
; ========= Конец текста подпрограммы Transmitter =======
```

3.2 Тестовая подпрограмма приема байта данных по последовательному порту в режиме 1.

```
; ========= Текст подпрограммы Receiver ==============
        ORG   0
        JMP   START ; Переход на основную программу, минуя область векторов прерываний
START:  MOV   SCON,#01110000B ; Установка режима UART
        MOV   TH1,#0FDH ; Число для перезагрузки
        SETB  TR1 ; Запуск таймера на счет
        JNB   RI,$ ; Ожидание приема байта данных из UART
        MOV   A,SBUF
        MOV   84H,#08H
        MOV   DPTR,#07H
        MOVX  @DPTR,A ; Вывод полученного байта на светодиоды
        JMP   $
; ========= Конец текста подпрограммы Receiver =========
```

4 Программа работы.

4.1. Записать в один стенд тестовую программу Receiver, в другой стенд – Transmitter, соединить стенды интерфейсным кабелем и пронаблюдать на светодиодах порта принимающего стенда переданное число.

4.2. Реализовать программу двунаправленной передачи данных в первом режиме UART по следующему алгоритму:

Переданное число со стенда 1 (с отображением на светодиодах) высвечивается на светодиодах стенда 2. После задержки в одну секунду принятое число умножается на 2, отображается на светодиодах стенда 2, передается обратно в стенд 1 и отображается на светодиодах стенда 1.

4.3. Реализовать программу межконтроллерного обмена:

Переданный по последовательному порту в режиме 3 байт данных из стенда 1 в стенд 2 циклически сдвигается с частотой 1 герц в сторону, задаваемую битом TB8 (0 – сдвиг вправо, 1 – сдвиг влево). Значение бита TB8 меняется при каждом нажатии клавиши «1» на клавиатуре стенда 1

5 Контрольные вопросы

5.1 Пояснить преимущества приема и передачи данных по последовательному порту с использованием прерываний.

5.2 Вектора прерывания от приемника и передатчика совпадают. Как определить, от чего произошел запрос прерывания: от приемника или передатчика?

5.3 Модифицировать подпрограмму Receiver и привести текст подпрограммы приема по UART одного байта с использованием прерывания от приемника.

5.4 Можно ли UART MK51 использовать для связи с компьютером? Если да, то привести пример настройки последовательного порта.