Министерство образования Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по курсу «Микропроцессорные системы и микроконтроллеры»
на тему «МПС сбора и обработки информации»

Выполнил:
студент группы 14ВВ1
Орлов А. В.

Принял:
Бычков А. С.

Пенза 2017

Содержание

Введение

В современном мире информационные технологии имеют огромное значение. Они выполняют огромные объемы вычислений, но что же выполняет эти вычисления? Элементарные математические операции лежат в основе всех вычислений, и выполняет эти вычисления процессоры или микроконтроллеры. В данной курсовой работе будет использован микроконтроллер 8051, рассматриваемый на специальном тренажере EdSim51.

1 Постановка задачи

Требуется разработать МПС сбора и обработки информации, осуществляющую ввод информационного потока с одного периферийного устройства, обработку и выдачу на два других периферийных устройства.

Устройство ввода – универсальный асинхронный приемопередатчик (UART).

Устройство вывода – цифроаналоговый преобразователь (DAC).

Дополнительное устройство вывода – дисплей на семисегментных индикаторах.

Количество вводимых слов – 10.

Порядок ввода-вывода – FIFO.

Начальный адрес массива – 4Dh.

Тип данных – целые без знака.

Метод организации цикла – с предусловием.

Алгоритм обработки информационного массива – найти количество нечетных чисел, больших 50.

Язык программирования: ассемблер MCS51.

2 Описание модели лабораторного стенда и используемого периферийного оборудования

Для выполнения поставленной задачи был использован тренажер EdSim51, схема которого представлена на рисунке 1.

Электронный симулятор EdSim51 представляет собой программную модель лабораторного стенда, построенного на основе МК 8051 и содержащего набор внешних периферийных устройств, наиболее часто используемых при построении микропроцессорных систем управления. Электронный симулятор EdSim51 имеет в своем составе необходимый набор программных средств, предназначенных для разработки и отладки программного обеспечения МК.

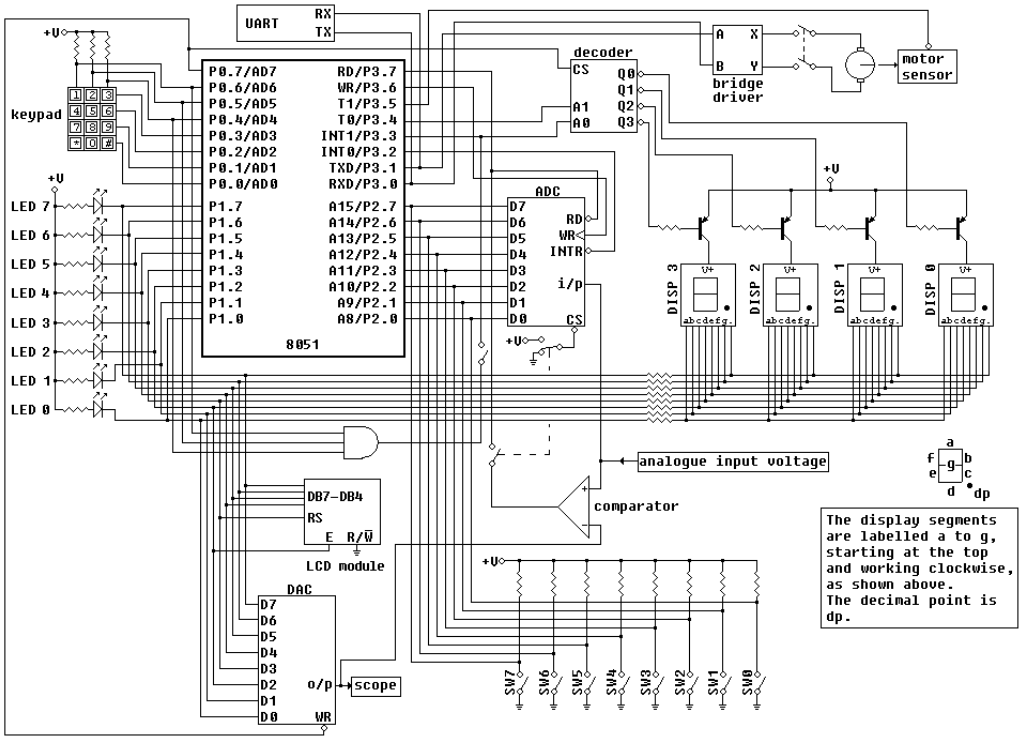


Рисунок 1 - Схема тренажера EdSim51

Чтобы выполнить поставленную задачу нам понадобиться:

- Микроконтроллер 8051;
- Цифроаналоговый преобразователь (DAC);
- Асинхронный приемопередатчик (UART)
- Семисегментная индикация.

3 Описание алгоритма

Согласно требованиям задания, вводимая информация должна обрабатываться по следующему алгоритму: найти количество нечетных чисел, больших 50. Блок схема для данного алгоритма представлена на рисунке 2.

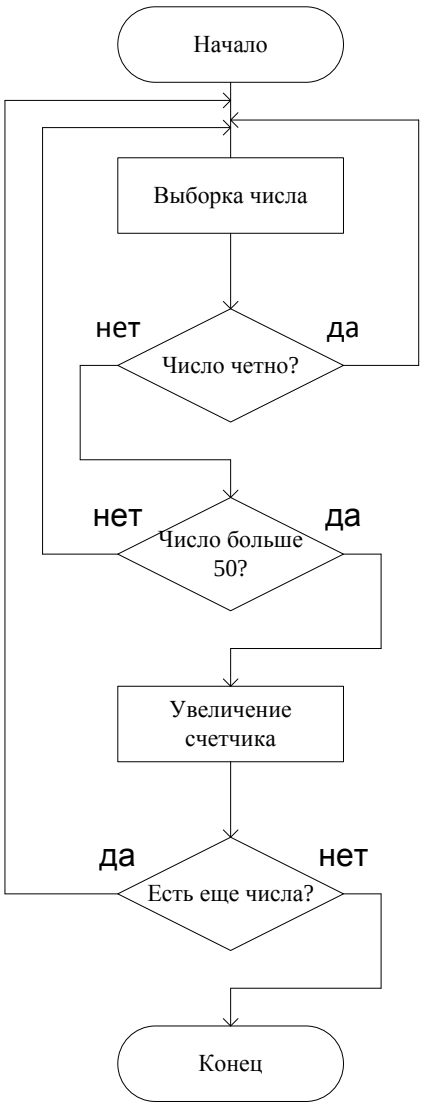


Рисунок 2 – Блок-схема алгоритма обработки информации

4 Описание программы

Написанная программа работает следующим образом.

Сначала пользователь вводит последовательно 10 чисел в память микроконтроллера с помощью универсального асинхронного приемопередатчика (UART). Считывается ASCII код символа. Затем с помощью метода преобразования получается число, которое и записывается в память. Процесс повторяется 10 раз.

После программа начинает обрабатывать заданную последовательность чисел с помощью алгоритма описанного в пункте 3, она считает количество нечетных чисел, больших 50. Результат также помещается в память.

Затем происходит вывод последовательности чисел цифроаналоговый преобразователь.

В завершение результат работы алгоритма выводиться на семисегментную индикацию.

5 Руководство оператора

Процесс взаимодействия пользователя с программой происходит следующим образом.

Пользователь должен запустить программу, затем последовательно ввести 10 чисел через пробел в текстовое поле для UART. После чего, убедившись, что скорость записи установлена как 19200 бод, нажать на кнопку “TxSend”. Программа самостоятельно считает 10 чисел в память и выведет их в порядке FIFO на цифроаналоговый преобразователь, а результат алгоритма на семисегментную индикацию.

6 Контрольный просчёт

На вход устройства UART для контрольного примера было подано 10 чисел в шестнадцатиричной системе (11 22 33 44 55 66 77 88 99 FF).

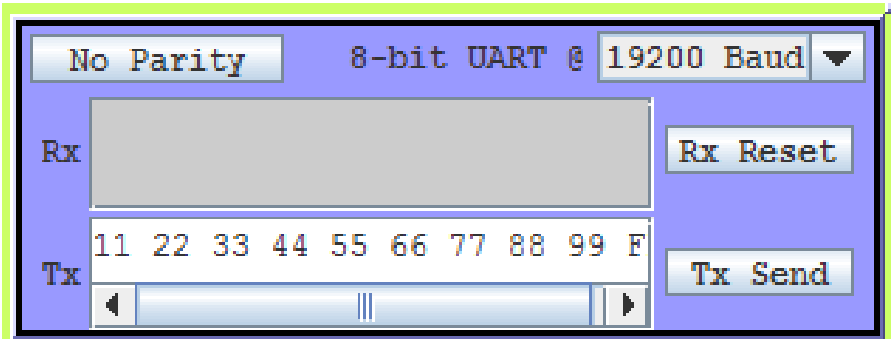


Рисунок 3 - Пример подачи чисел на UART

По заданию необходимо найти количество нечетных чисел, больших 50. В таком случае по заданию из ряда программой должно быть выбрано 5 чисел: 33, 55, 77, 99 и FF, при переводе в десятичную систему: 51, 85, 119, 153 и 255 соответственно. Значит выбранные значения подходят под условия, согласно номера варианта курсового проекта. Остальные значения отбрасываем, потому что они меньше 50 или четны (Меньше 50: 11, 22 Четные: 22, 44, 66, 88). Однако все значения, введенные с UART, отображаются на устройстве вывода. По заданию курсовой работы – это цифроаналоговый преобразователь (DAC).

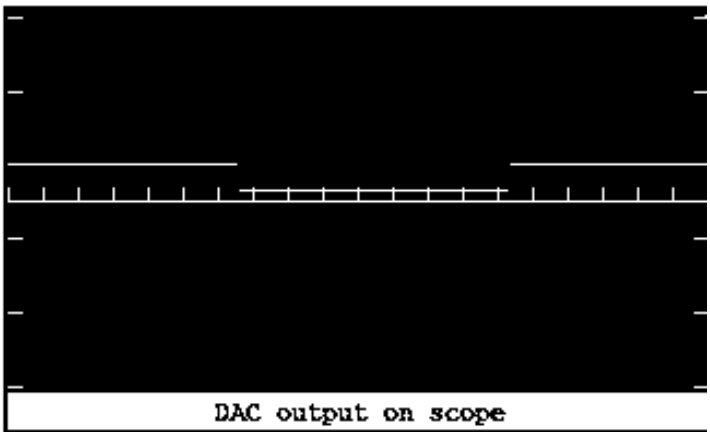
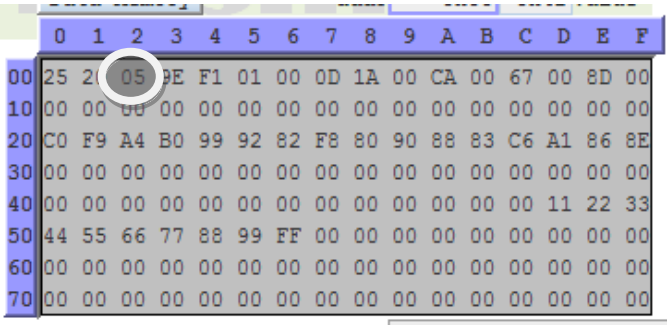


Рисунок 4 - Вывод результата на ЦАП

Среди регистров интересен R2, именно в нём накапливается количество нечетных чисел больших 50, которые нужно вывести согласно заданию.



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	25	2	05	9E	F1	01	00	0D	1A	00	CA	00	67	00	8D	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	C0	F9	A4	B0	99	92	82	F8	80	90	88	83	C6	A1	86	8E
30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
40	00	00	00	00	00	00	00	00	00	00	00	00	00	11	22	33
50	44	55	66	77	88	99	FF	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Рисунок 5 - Регистры

После запуска программы с 22 мсек на семисегментом индикаторе появляется результат. Как можно заметить, он сходится с расчётными значениями.

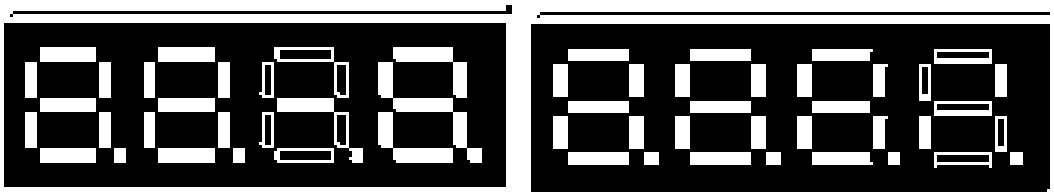


Рисунок 6 - Результат вывода на семисегментный индикатор

7 Тестирование и отладка

В процессе написания программы были найдены и исправлены все возникающие ошибки. Результаты работы программы совпали с результатами расчетов. Из этого можно сделать вывод, что написанная программа работает верно.

Заключение

При выполнении данной курсовой работы были получены навыки работы с микроконтроллером, вывода информации на семисегментную индикацию и цифроаналоговый преобразователь, освоены принципы работы интерфейса UART.

В результате отладки и тестирование были обнаружены и исправлены все возникшие ошибки. Программа выдает корректный результат при всех заданных наборах данных.

Список используемых источников

1. http://www.edsim51.com/8051Notes/8051/serial.html
2. http://www.edsim51.com/examples.html
3. http://www.edsim51.com/simInstructions.html#lcdModuleInstructionSet

Приложение А
Листинг программы

ПриложениеА.1
Файл «cod.asm»

```
    CALL ZapolneniyeTablici
    MOV R2,#00H      ; Результат
    MOV R0, #4DH ; Адрес начала массива
    MOV R1, #0AH ; Счетчик элементов
    CALL Input       ; Вызов функции получения значений с
UART


    MOV R0, #4DH ; Адрес начала массива
    MOV R1, #0AH ; Счетчик элементов
    CALL calculating; Нахождение количества нечетных
чисел больше 50


    MOV R0, #4DH ; Адрес начала массива
    MOV R1, #0AH ; Счетчик элементов
    CALL DACout  ; Вывод последовательности на ЦАП



    CALL PrepareR2ToSSI   ;готовим r2 к выводу бьем его
на части
OutSSD:
    CALL OutputSSI       ; Вывод результата на семи
сегментную индикацию
    JMP OutSSD

;-----------------------------------------------------
---------------------
```

```
; Функция считывания чисел
Input:

    CLR SM0
    SETB SM1     ; установка последовательного порта в 8-битный UART режим
                 ;(скорость в бодах задается от таймера 1)

    SETB REN     ; включение последовательного порта получателя

    MOV A, PCON
    SETB ACC.7
    MOV PCON, A      ; установка бита SMOD (старший бит регистра Power CONtrol) отвечающего за частоту срабатывания таймера
                 ;если SMOD установлен то таймер считает с частотой не (system frequency)/32 а (system frequency)/16

    MOV TMOD, #20H       ; установка таймера 1 в 8битный автоперезагружаемый периодический режим
    MOV TH1, #0FDH       ; таймер будет переполняться каждые 3 мкс
                 ; формула подсчета для UART: TH1 = 256 - ((system frequency / (12 * 16)) / baud)=256-2.99 частота ошибки сост 0.01 1%
    MOV TL1, #0FDH       ; первый раз таймер переполнится четез 3мкс
    SETB TR1     ; запуск таймера 1
    MOV R1, #4DH     ; начальный адрес массива
    MOV R6, #0h          ;0-1й полубайт получен с UART ,1-2й
```

```
    MOV R5, #0Ah      ;счетчик чисел (не символов) полученных с UART
again:
    JNB RI, $         ;если считать с UART нечего - ожидание
    CLR RI            ;флаг RI аппаратно устанавливается в 1 когда байт информации получен по UART
                      ; программно сбрасываем чтобы понять когда принимать следующий байт
    MOV R7, SBUF      ;SBUF хранит байт полученный по UART

label:
    CJNE R7, #0DH, convert    ;если чисел пришло меньше 10 - обрабатываем те что есть
                      ;любая последовательность символов вводимых в текстовое поле UART дополняется символом \r (код 0Dh)
                      ;таким образом, проверяется на конец строки
    JMP ExitFromFunction
;конвертируем ASCII код символа пришедшего с UART в число
convert:
    MOV A,#21h
    SUBB A, R7
    JNC itsSpace      ;число < 21 - пробел
    MOV A,#40h
    SUBB A ,R7
    JC itsSymbol      ;число > 40 - символ
    MOV A,R6      ;2й полубайт?
    JZ itsFirstByteNumber;число 0..9
    MOV R6,#0
    MOV A,R7      ;0..9
    ADD A,#11010000b ;-30h (или + 208)
```

```
    ADD A,@R1
    MOV @R1,A
    JMP again

itsFirstByteNumber:       ;1й полубайт
    MOV @R1,#0        ;обнуление старых значений памяти
    INC R6
    MOV A,R7
    ADD A,#11010000b ;-30h (или + 208)
    SWAP A            ;делается полубайт старшим
    MOV @R1, A
    JMP again

itsSpace:             ;пробел
    INC R1
    DEC R5
    MOV A, R5
    JZ calculating
    JMP again

itsSymbol:            ;A B C D E F
    MOV A,R6     ;2й полубайт? в R6 хранится флаг - 0 1й
полубайт 1-2й
    JZ itsFirstByteSymbol
    MOV R6,#0         ;значит следующим будет 1й полубайт
    MOV A,R7
    ADD A,#11001001b ;чтобы преобразовать код буквы (A B
C...F) в числовой эквивалент нужно прибавить 201 (или -
41h)
    ADD A,@R1
    MOV @R1,A
    JMP again
```

```
itsFirstByteSymbol:       ;1й полубайт
    MOV @R1,#0        ;обнуляются старые значения памяти
    INC R6            ;следующим будет 2й полубайт
    MOV A,R7
    ADD A,#11001001b ;чтобы преобразовать код буквы (A B
C...F) в числовой эквивалент нужно прибавить 201 (или -
41h)
    SWAP A            ;делается полубайт старшим
    MOV @R1, A
    JMP again


;-----------------------------------------------------
----------------------
ExitFromFunction:
    ret
; Поиск и подсчет необходимых чисел
calculating:
    DEC R0
    MOV R2, #00H ; Счетчик количества нечетных чисел
CYCL:
; Предусловие
    MOV A, R1
    JZ STOP
    INC R0        ; Следующий элемент массива
    DEC R1        ; Уменьшение счетчика цикла на 1
    MOV A, #32H       ; Выборка числа
    SUBB A, @R0       ; Вычитание чисел
    JNC CYCL          ; Если число больше 50, то следущая
итерация
    MOV A, @R0        ; Выборка числа
    MOV B, #02H
```

```
    DIV AB
    MOV A, B          ; Остаток помещаем в A
    JZ CYCL      ; Если остаток ноль (число четное), то
следующая итерация
    INC R2            ; Увеличение счетчика нечетных чисел
R2
    JMP CYCL          ; Переход на следующую итерацию


STOP:
    RET


;-------------------------------------------------------
----------------------


DACout:
    CLR P0.7
    MOV A, R1         ; Количество выводимых чисел
    JZ DACStop        ; Если счетчик 0, то выход
    DEC R1            ; Уменьшение счетчика


    MOV P1, @R0
    CALL delay
    INC R0
    JMP DACout
DACStop:
SETB P0.7
RET


;-------------------------------------------------------
------
PrepareR2ToSSI:
    MOV A,R2
```

```
    SWAP A
    ANL A,#00001111b
    ADD A,#20h
    MOV R1,A     ;R1-старшая часть

    MOV A,R2
    ANL A,#00001111b
    ADD A,#20h
    MOV R0,A     ;R0-младшая часть


ret

; Вывод на ССИ
OutputSSI:
    CLR P3.4
    CLR P3.3     ;выбор ССИ 0
    MOV P1, @R0      ;подача младшего символа
    call delay
    MOV P1,#0FFh     ;гашение символа

    SETB P3.3        ;выбор ССИ 1
    MOV P1, @R1      ;подача старшего символа
    call delay
    MOV P1,#0FFh     ;гашение символа


ret

;---------------------------------
```

```
; Функция задержки
delay:
    MOV R3, #0FFH
    DJNZ R3, $
    RET

;-------------------------------------------------------
----------------------
ZapolneniyeTablici:
    MOV R1,#20h      ;заполнение таблицы
    MOV @R1, #11000000b;0
    INC R1
    MOV @R1, #11111001b;1
    INC R1
    MOV @R1, #10100100b;2
    INC R1
    MOV @R1, #10110000b;3
    INC R1
    MOV @R1, #10011001b;4
    INC R1
    MOV @R1, #10010010b;5
    INC R1
    MOV @R1, #10000010b;6
    INC R1
    MOV @R1, #11111000b;7
    INC R1
    MOV @R1, #10000000b;8
    INC R1
    MOV @R1, #10010000b;9
    INC R1
    MOV @R1, #10001000b;A
```

```
    INC R1
    MOV @R1, #10000011b;b
    INC R1
    MOV @R1, #11000110b;C
    INC R1
    MOV @R1, #10100001b;d
    INC R1
    MOV @R1, #10000110b;E
    INC R1
    MOV @R1, #10001110b;F
ret

;--------------------------
```

Приложение В
Схема электрическая принципиальная

Приложение С
Перечень элементов

Поз. обозн.	Наименование	Кол.	Примечание
	Конденсаторы		
C1,C2	К50-16-16В-20мкФ	2	
	Микросхемы		
DD1	MCS8051	1	Импорт
DD2	КР1561ИД7	1	
	Цифроаналоговый преобразователь		
HG1	MCP3201	1	Импорт
	Семисегментная индикация		
HG2	DA56	1	Импорт
	Резисторы		
R1…R8	С2-29в-0.125-2 кОм ± 1%	8	
R9,R10	С2-29в-0.125-10 кОм ± 1%	2	
	Транзисторы		
VT1,VT2	КТ343Г	2	
	Разъёмы		
XP1	WF-R-8	1	Импорт