ЛАБОРАТОРНАЯ РАБОТА №2
УПРАВЛЕНИЕ ХОДОМ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА

ЦЕЛЬ РАБОТЫ: освоение способов адресации данных и команд управления ходом вычислений.

ОБЩИЕ СВЕДЕНИЯ

Лабораторная работа направлена на выработку навыков организации и управления ходом вычислительного процесса, использования команд условных и безусловных переходов, способов организации циклических вычислений.

Набор команд микроконтроллеров семейства MCS51 поддерживает следующие типы адресации данных .

Прямая адресация (Direct Addressing) – операнд определяется 8-битовым адресом, указанным в команде. Например: `mov A,45h` – скопировать в аккумулятор содержимое ячейки ОЗУ №45h. Эти команды могут занимать до трех байт в памяти программ.

Косвенная адресация (Indirect Addressing) – в команде указывается адрес РОН, содержащий адрес операнда. Допускается использовать только регистры R0 или R1 выбранного банка регистров и некоторые регистры специальных функций. Например: `mov A,@R1` – скопировать в аккумулятор содержимое ячейки ОЗУ, адрес которой хранится в регистре **R1**.

Регистровая адресация (Register Instruction) – в команде указываются адреса регистров **R0** … **R7** выбранного банка регистров. Выбор одного из четырех доступных регистровых банков осуществляется программированием битов селектора банка (**RS1**, **RS0**) в слове состояния процессора **PSW**. Например: `mov A,R1` – скопировать в аккумулятор содержимое регистра **R1**. Эти команды являются самыми короткими, они могут занимать всего один байт в памяти программ.

Непосредственная адресация (Immediate Constants) – операнд записан непосредственно в поле команды. Например: `mov A,#45h` – записать в аккумулятор число 45h. Считываемое значение хранится в памяти программ и не может быть изменено в ходе ее выполнения.

Индексная адресация (Indexed Aggressing) – число читается из памяти программ с использованием специального базового регистра (регистры **DPTR** или **PC**) и индексного регистра (обычно – аккумулятора). Адрес элемента вычисляется сложением содержимых базового и индексного регистров. Обычно используется при работе с таблицами данных, хранящихся в памяти программ. Например: `mov A,@A+DPTR` – записать в аккумулятор число из ячейки памяти программ, адрес которой вычисляется суммированием содержимого аккумулятора и регистра **DPTR**.

Некоторые из описанных выше типов адресации использовались при выполнении лабораторной работы №1.

В ходе работы по программе осуществляется последовательное выполнение команд, хранящихся с памяти программ. Команда может содержать операнды и в

зависимости от типа адресации команда может занимать в памяти программ от 1 до 3 байт. Первый байт любой команды содержит код операции, второй и третий байты содержат либо адреса операндов, либо непосредственно операнд. Адрес текущей ячейки памяти определяется содержимым регистра счетчика команд **PC,** которое постоянно увеличивается на 1 и указывает сначала на код команды, а затем на операнды. Таким образом, содержимое **PC** изменяется автоматически в зависимости от кода последней выполненной команды.

Однако иногда возникает необходимость нарушить естественный ход выполнения команд. Для этого используются команды переходов. Команда `jmp address` - команда безусловного перехода (сокращение слова jump – прыжок) к точке программы, указанной в поле `address`. Непосредственное указание адреса при программировании неудобно, поэтому компиляторы допускают использования в поле `address` строковой метки. Метка предшествует исполняемой команде и отделяется от нее символом **:** (двоеточие). При компиляции программы метка будет заменена соответствующим адресом исполняемой команды.

ЗАДАНИЕ И ВАРИАНТЫ ВЫПОЛНЕНИЯ РАБОТЫ

Написать программу, реализующую циклическое увеличение (уменьшение) двухбайтового числа, хранящегося в ОЗУ. Увеличение (уменьшение) осуществляется на однобайтовую величину, последовательно считываемую из массива данных, хранящихся в памяти программ.

Считываемые из памяти программ данные используются для вычисления результата, если выполнено условие, указанное в задании. Критерий прекращения вычислений также задается заданием.

Рекомендуемые значения для массива данных: 209, 78, 203, 251, 146, 225, 170, 91, 15,92, 58, 55, 217, 39, 162, 23, 112, 8, 227, 200, 17,116, 200, 64, 105.

Варианты выполнения лабораторной работы представлены в таблице 4.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. В соответствии с вариантом задания написать программу работы микроконтроллера на языке Ассемблер MCS51. Для этого следует запустить программу MCStudio и выбрать *Создать новый проект или файл -> Создать проект->Ok* с выбором личной папки и оригинальным именем проекта. В качестве модели микроконтроллера рекомендуется выбрать Intel 80C51 или Atmel AT89S8252.

В открывшемся главном окне проекта набрать текст программы. Сохранить текст на диске компьютера.

2. Нажать Ctrl-F9 или выбрать *Проект ->Компилировать->Ok* и выполнить компиляцию программы (т.е. перевод текстового файла в двоичный код для микроконтроллера). Исправить обнаруженные компилятором синтаксические ошибки.

Таблица 4 – Варианты выполнения лабораторной работы

№ варианта	Операция	Условие использования байта данных	Критерий прекращения вычислений
1	увеличение	>20	Операция выполнена 10 раз
2	увеличение	>40	Операция выполнена 12 раз
3	увеличение	>50	Сумма >1024
4	увеличение	<250	Операция выполнена 7 раз
5	увеличение	<150	Сумма >500
6	увеличение	четное число	Операция выполнена 5 раз
7	увеличение	четное число	Сумма >2000
8	увеличение	четное число	Операция выполнена 7 раз
9	увеличение	четное число	Сумма >800
10	увеличение	четное число	Операция выполнена 10 раз
11	уменьшение	нечетное число	Сумма <-1024
12	уменьшение	нечетное число	Операция выполнена 11 раз
13	уменьшение	нечетное число	Сумма <-512
14	уменьшение	нечетное число	Операция выполнена 9 раз
15	уменьшение	нечетное число	Сумма <-2048
16	уменьшение	нечетное число	Операция выполнена 6 раз
17	уменьшение	>45	Сумма <-400
18	уменьшение	>100	Сумма <-850
19	уменьшение	>80	Операция выполнена 7 раз
20	уменьшение	<200	Сумма <-1500

3. Проверить работу программы на симуляторе. Для этого запустить симуляцию, выбрав *Выполнение -> Запустить симуляцию* и нажать F9 (дважды). При запущенной симуляции на экране отображается окно **Выполнение программы** с круговой диаграммой хода машинного времени. Открыть окно **РПД – просмотр**, выбрав *Вид -> Резидентная память данных*. В ячейках резидентной памяти данных считать содержимое результата вычислений. Полученные численные значения должны соответствовать числу, вычисленному на калькуляторе по заданному алгоритму. Сохранить проект и продемонстрировать работу программы преподавателю.

4. Оформить отчет о работе.

СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать :

1. Титульный лист
2. Цель работы и конкретный вариант задания
3. Листинг программы работы микроконтроллера
4. Результаты работы программы в виде скриншотов соответствующих окон системы моделирования
5. Выводы по работе

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какое число разрядов имеет регистр **DPTR**?
2. Какое число разрядов имеет регистр **PC**?
3. В чем различие а применении имен аккумулятора **A** и **ACC**?
4. Что обозначает команда `cjne @R0,#40,Stop` ?
5. Что обозначает команда `jmp @A+DPTR` ?