Цепочечные команды. Обработка строк.

План темы:

1. Назначение цепочечных команд.
2. Направление обработки цепочки.
3. Адреса операндов.
4. Префикс повторения.
5. Операция пересылки цепочек.
6. Операция сравнения цепочек.
7. Операция сканирования цепочек.
8. Загрузка элемента цепочки в аккумулятор.
9. Перенос элемента из аккумулятора в цепочку.
10. Ввод цепочки из порта ввода-вывода.
11. Вывод цепочки в порт ввода-вывода.

1. Назначение цепочечных команд.

Цепочечные команды **также называют командами** ***обработки строк символов*****.**

Цепочечные команды позволяют проводить действия над блоками памяти, представляющими собой последовательности элементов следующего размера:

- 8 бит — байт;
- 16 бит — слово;
- 32 бита — двойное слово.

1. Назначение цепочечных команд.

Всего в системе команд микропроцессора имеется семь *операций-примитивов* обработки цепочек.

Каждая из них реализуется в микропроцессоре тремя командами, в свою очередь, каждая из этих команд работает с соответствующим размером элемента — байтом, словом или двойным словом.

2. Направление обработки цепочки.

Особенность всех цепочечных команд в том, что они, кроме обработки текущего элемента цепочки, осуществляют еще и *автоматическое продвижение* к следующему элементу данной цепочки.

Возможны два направления обработки цепочки:

- от начала цепочки к ее концу, т.е. в направлении возрастания адресов;
- от конца цепочки к началу, т.е. в направлении убывания адресов.

2. Направление обработки цепочки.

Направление обработки цепочки определяется значением флага направления **df**:

- если **df = 0**, то обработка будет осуществляться в направлении возрастания адресов;
- если **df = 1**, то обработка будет идти в направлении убывания адресов.

Состояние флага **df** управляется командами:

- **CLD** (Clear Direction Flag) — Команда сбрасывает флаг направления df в 0.
- **STD** (Set Direction Flag) — Команда устанавливает флаг направления df в 1.

3. Адреса операндов.

Полные физические адреса для операндов цепочечных команд должны быть следующие:

- **адрес источника — пара ds:si;**
- **адрес приемника — пара es:di.**

Эти регистры заполняются нужными значениями перед использованием цепочечных команд.

В самих командах операнды можно не указывать!

4. Префикс повторения.

Перед цепочечной командой обычно указывается *префикс повторения:*

- **rep** - заставляет цепочечную команду циклически выполняться, пока *содержимое в cx не станет равным 0.*
- **repe** или **repz** - заставляют цепочечную команду циклически выполняться до тех пор, пока *содержимое cx не равно нулю или флаг zf равен 1.*
- **repne** или **repnz** - заставляют цепочечную команду циклически выполняться до тех пор, пока *содержимое cx не равно нулю или флаг zf равен 0.*

Таким образом, набор действий при выполнении цепочечной команды следующий:

- **Установить значение флага df в зависимости от того, в каком направлении будут обрабатываться элементы цепочки — в направлении возрастания (CLD) или убывания (STD) адресов;**
- **Загрузить указатели на адреса цепочек в памяти в пары регистров ds:si (источник) и es:di (приемник);**
- **Загрузить в регистр cx количество элементов, подлежащих обработке;**
- **Задать команду с нужным префиксом повторения (rep\repe\repne).**

5. Операция пересылки цепочек.

Производится копирование элементов из одной области памяти (цепочки) в другую. Размер элемента определяется применяемой командой:

movsb — переслать цепочку байт;

movsw — переслать цепочку слов;

movsd — переслать цепочку двойных слов.

6. Операция сравнения цепочек.

Производится сравнение элементов цепочки-источника с элементами цепочки-приемника:

cmpsb — сравнить строку байт;
cmpsw — сравнить строку слов;
cmpsd — сравнить строку двойных слов.

6. Операция сравнения цепочек.

С командой cmps можно использовать префикс повторения repe/repz или repne/repnz:

- **repe/repz — поиск несовпадающих элементов в цепочках, сравнение выполняется пока: не достигнут конец цепочки (cx<>0) и очередные элементы в цепочке одинаковы (zf=1);**
- **repne/repnz — поиск совпадающих элементов в цепочках, сравнение выполняется пока: не достигнут конец цепочки (cx<>0) и очередные элементы в цепочке разные (zf=0).**

7. Операция сканирования цепочек.

Производится поиск некоторого значения в области памяти. Искомое значение предварительно должно быть помещено в регистр al/ax/eax, адрес цепочки должен быть заранее сформирован в es:di:

scasb — сканировать цепочку байт;

scasw — сканировать цепочку слов;

scasd — сканировать цепочку двойных слов.

7. Операция сканирования цепочек.

С командой scas можно использовать префикс повторения repe/repz или repne/repnz:

- repe/repz — поиск несовпадающего с образцом в аккумуляторе элемента цепочки, сравнение выполняется пока: не достигнут конец цепочки (cx<>0) и очередной элемент в цепочке равен значению аккумулятора (zf=1);
- repne/repnz — поиск совпадающего с образцом в аккумуляторе элемента цепочки, сравнение выполняется пока: не достигнут конец цепочки (cx<>0) и очередной элемент в цепочке не равен значению аккумулятора (zf=0).

8. Загрузка элемента цепочки в аккумулятор

Эта операция позволяет извлечь элемент цепочки и поместить его в аккумулятор. Её удобно использовать вместе с поиском (сканированием) с тем, чтобы, найдя нужный элемент, извлечь его (например, для изменения), адрес цепочки должен быть заранее сформирован в **ds:si**:

lodsb — загрузить байт из цепочки в **al**;
lodsw — загрузить слово из цепочки в **ax**;
lodsd — загрузить двойное слово из цепочки в **eax**.

9. Перенос элемента из аккумулятора в цепочку.

Эта операция позволяет сохранить значение из аккумулятора в элементе цепочки. Её удобно использовать вместе с операцией поиска (сканирования) **scans** и загрузки **lods**, с тем, чтобы, *найдя нужный элемент, извлечь его в регистр и записать на его место новое значение*, адрес цепочки должен быть заранее сформирован в e**s:di**:

stosb - сохранить байт из al в цепочке;

stosw - сохранить слово из ax в цепочке;

stosd - сохранить двойное слово из eax в цепочке.

10. Ввод цепочки из порта ввода-вывода.

Эта операция позволяет произвести ввод цепочки элементов из порта ввода-вывода, номер которого указывается в регистре dx, адрес цепочки должен быть заранее сформирован в es:di:

insb — ввести из порта цепочку байт;
insw — ввести из порта цепочку слов;
insd — ввести из порта цепочку двойных слов.

11. Вывод цепочки в порт ввода-вывода.

Эта операция позволяет произвести вывод цепочки элементов в порт ввода-вывода, номер которого указывается в регистре **dx**, адрес цепочки должен быть заранее сформирован в **ds:si**:

outsb — вывести в порт цепочку байт;

outsw — вывести в порт цепочку слов;

outsd — вывести в порт цепочку двойных слов.