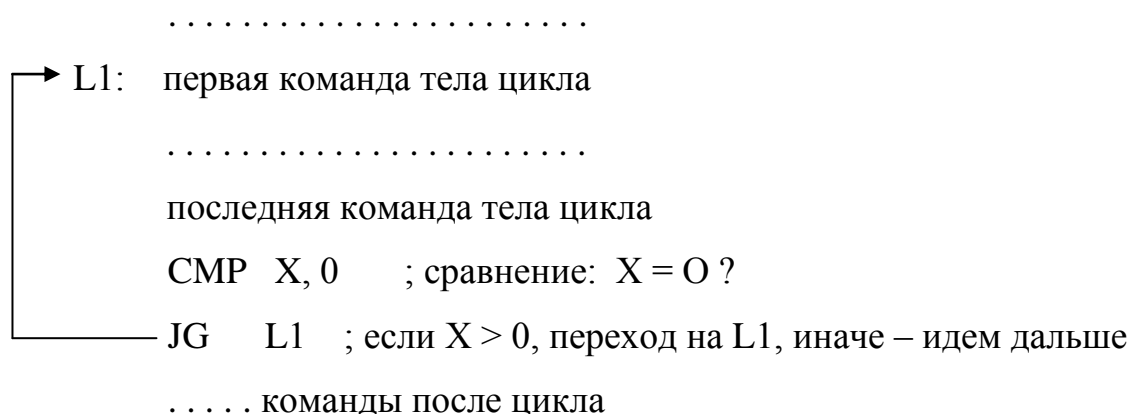


8. Организация циклов

Рассмотрим способы реализации разных циклов в программах на языке ассемблера. Начнем с циклов с неизвестным числом повторений, которые, как известно, управляются соответствующими условиями. Наиболее просто реализуются циклы с постусловием, которые в языках высокого уровня имеют, например, следующий вид:

repeat действия **until** ($X > 0$);

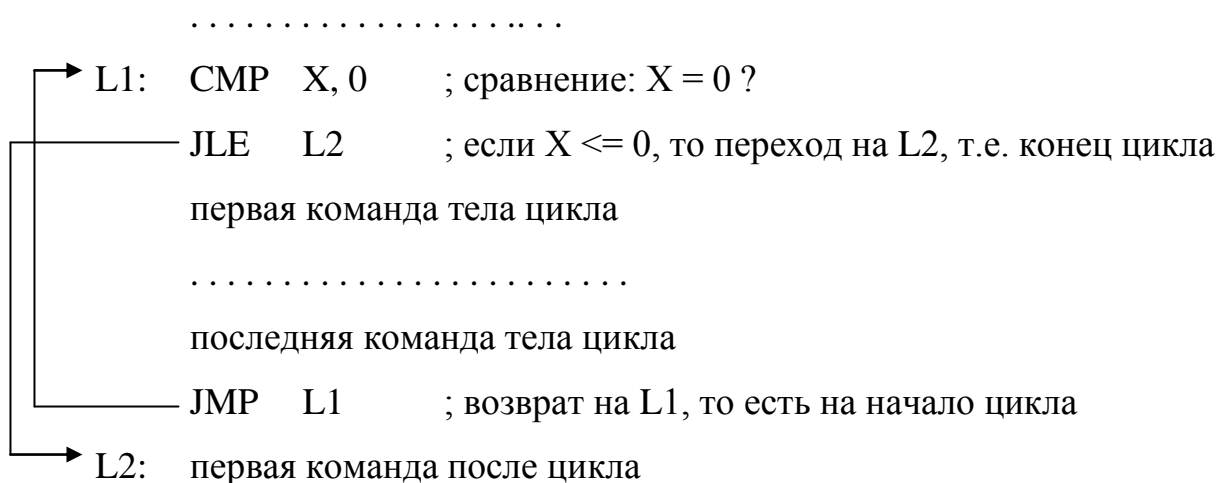
Такой цикл реализуется с помощью одной команды условного перехода:



Немного сложнее реализуется цикл с предусловием вида

while ($X > 0$) **do** действия;

Здесь приходится использовать две команды перехода – одну условную и одну безусловную.



Цикл с известным числом повторений, т.е. управляемый счетчиком числа повторений, реализуется по следующей схеме:

- выбираем регистр для хранения счетчика и загружаем в него начальное значение числа повторов
- выполняем тело цикла
- уменьшаем значение в регистре-счетчике на 1
- сравниваем значение в регистре-счетчике с 0
- если значение в регистре-счетчике не равно 0, то переходим на первую команду тела цикла

Фрагмент программного кода с использованием регистра-счетчика CX и числом повторений цикла, равным 100:

```
.....  
MOV  CX, 100    ; (CX) = 100  
  
L1:  первая команда тела цикла  
  
.....  
      последняя команда тела цикла  
DEC  CX    ; (CX) = (CX) - 1  
CMP  CX, 0  ; (CX) = 0 ?  
JNE  L1    ; если (CX) ≠ 0, то переход на L1, иначе – выход  
      первая команда после цикла
```

Поскольку действия по организации цикла достаточно стандартны, для упрощения программирования были введены специальные команды группы LOOP. Основная команда этой группы имеет мнемонику LOOP и заменяет последние 3 команды. В качестве регистра-счетчика всегда используется регистр CX:

```
MOV  CX, 100  
L1:  начало тела цикла  
.....  
LOOP L1
```

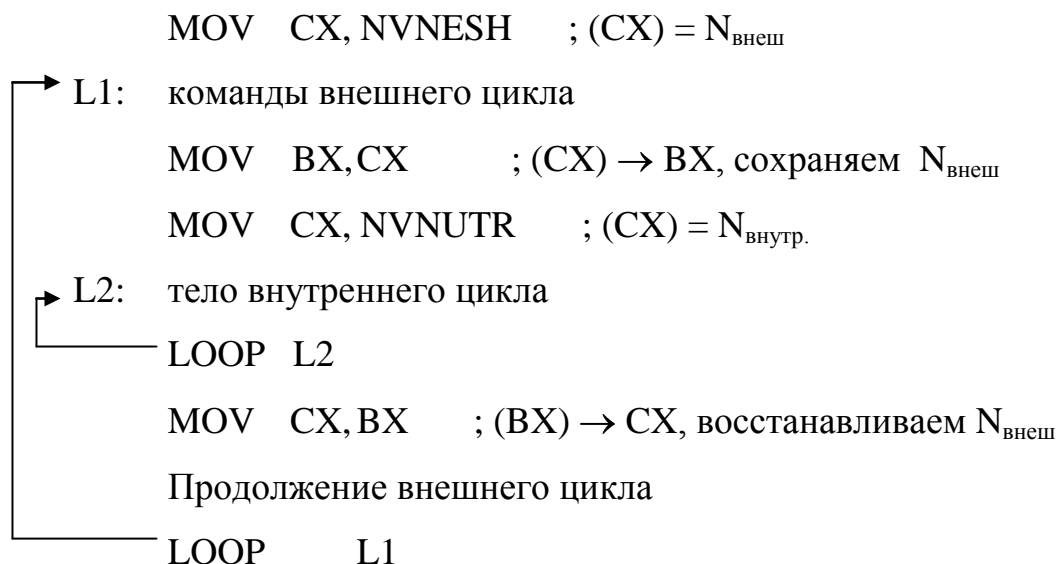
Преимуществом данного способа является более короткий код и более быстрое выполнение. Недостаток – использование только регистра CX.

Полезными разновидностями команды LOOP являются команды LOOPE и LOOPNE. Они каждый раз проверяют еще и значение флага ZF, что позволяет преждевременно заканчивать цикл. Команда LOOPE позволяет повторить цикл, если $(CX \neq 0)$ и $(ZF=1)$. Команда LOOPNE позволяет повторить цикл, если $(CX \neq 0)$ и $(ZF=0)$.

Особенно полезна вторая команда - LOOPNE, т.к. она позволяет найти в наборе элементов первое совпадение с заданным значением. Перед LOOPNE надо поставить сравнение по команде CMP, которое при совпадении значений установит флаг $ZF = 1$ и LOOPNE прекратит цикл.

При использовании команд LOOPE и LOOPNE надо после цикла проверять причину завершения цикла, анализируя флаг ZF с помощью команд условного перехода типа JE (или JZ) и JNE (JNZ).

Иногда в ассемблерных программах приходится реализовывать вложенные циклы. Без использования команды LOOP это приводит к необходимости работы с двумя регистрами-счетчиками. Если же использовать команду LOOP, то для внешних и внутренних счетчиков надо использовать один и тот же регистр CX, поэтому перед входом во внутренний цикл его надо сохранять и потом восстанавливать (например – в другом регистре или в области памяти). Схема реализации



Очевидно, что возможны и комбинированные реализации, когда внешние и внутренние циклы реализуются по-разному, т.е. без команды LOOP или с ее использованием.

Практические задания к теме №8.

Задание 1. Написать программы для реализации следующих циклов:

- цикл с постусловием
- цикл с предусловием

Задание 2. Написать программы, которые реализуют:

- цикл со счетчиком без использования команды LOOP
- цикл со счетчиком с использованием команды LOOP

Задание 3. Написать программы, которые реализуют вложенные циклы четырьмя способами:

- оба – без использования команды LOOP
- оба – с использованием команды LOOP
- внешний – с помощью команды LOOP, внутренний – без нее
- внешний – без команды LOOP, внутренний – с командой LOOP