

## Лекция 11

### Раздел 3. Процедуры и макрокоманды

#### 3.1 Процедуры и особенности их использования

Процедурой (или подпрограммой) называется фрагмент программы, к которому можно перейти, и из которого можно вернуться туда, откуда осуществляется переход.

Переход к процедуре называется вызовом, а переход назад - возвращением. Возвращение осуществляется к команде, следующей после вызова.

Эти функции выполняют две команды:

CALL (call a procedure) и

RET (return from procedure - вернуться из процедуры).

Команда CALL имеет формат:

**CALL имя**

где **имя** - это имя процедуры, которая вызывается.

Сама же процедура, как отмечалось, имеет вид:

```
NAME PROC
.....
RET
NAME ENDP
```

Имя процедуры считается меткой, которая принадлежит к первой команде процедуры. Поэтому имя процедуры можно отмечать в командах перехода, который означает переход на первую команду процедуры.

В отличие от языков высокого уровня, имена в процедурах (переменные, метки) не локализируются внутри их, следовательно должны быть уникальными для соответствующего сегмента.

Где размещать процедуру? Где угодно. Но так, чтобы процедура (подпрограмма) не выполнялась, если к ней не обращаются. Поэтому можно рекомендовать 3 варианта:

1. В одном сегменте после основной программы;
2. В одном сегменте перед точкой входа;
3. В отдельном сегменте.

Тогда можно в конце основной программы поставить команду FINISH.

A SEGMENT	B SEGMENT	C SEGMENT
BEG: <осн. прог.> FINISH		
Процедура	Процедура	Процедура
A ENDS END BEG	BEG: <осн. прог.> B ENDS END BEG	C ENDS D SEGMENT BEG:<осн. прог.> D ENDS END BEG

Если процедура предназначена для использования и в других сегментах, то она имеет атрибут дистанции **FAR**. Главная процедура также должна иметь атрибут **FAR**. По умолчанию процедура имеет атрибут **NEAR**.

При выполнении команды **CALL NAME** сначала нужно занести адрес возвращения стек, а затем сделать переход на имя **NAME**.

Для **NEAR**-процедуры в стек достаточно занести лишь адрес смещения.

Следовательно, в этом случае команда **CALL NAME** эквивалентна

PUSH IP JMP NAME
---------------------

Для **FAR**-процедуры адрес возвращения является абсолютным адресом и состоит из двух слов **CS:IP**. Поэтому команда **CALL NAME** будет эквивалентна трем командам:

PUSH CS PUSH IP JMP NAME
--------------------------------

Другими словами:

1. Сначала в стек заносится адрес начала сегмента кода, который содержится в регистре **CS**
2. Затем заносится содержание регистра указателя команд **IP** - смещение.

Команда **RET** возвращает управление к точке вызова процедуры. Эквивалентна двум командам:

POP IP POP CS
------------------

Если процедура имеет атрибут **NEAR** - выполняется только первая команда. То есть, особенности выполнения команды **RET** определяются заглавием процедуры - наличием атрибута **NEAR** или **FAR**.

Пример: Пусть, первый фрагмент программы транслирован по таким адресам:

<b>0012</b>	E8 0007	CALL MY_PROC; вызов процедуры
<b>0015</b>	8B C3	MOV AX, BX; вернуться сюда из процедуры
0017	B8 4C00	MOV AX, 4c00h
001A	CD 21	INT 21h
001C		MAIN ENDP
		; описание процедуры
<b>001C</b>		MY_PROC PROC; начало процедуры
001C	B1 0A	MOV CL, 10; первая команда процедуры
001E	C3	RET; вернуться к точке вызова
<b>001F</b>		MY_PROC ENDP

По умолчанию MY\_PROC имеет атрибут **NEAR**.

Влияние процедуры на стек:

1. Перед выполнением **CALL**:

<div> <div>ss:0146 3245</div> <div>ss:0144 6791</div> <div>ss:0142 0809</div> <div>ss:0140 0501</div> <div>ss:013E 5BC9</div> </div>			<div> <div>sp 013E</div> <div>ds 5BED</div> <div>es 5BC9</div> <div>ss 5BD9</div> <div>cs 5BEF</div> <div>ip 0012</div> </div>		
SI 0000	CS 2700	IP 0012	Stack +0	26DA	
DI 0000	DS 26FE		+2	0501	
BP 0000	ES 26DA	HS 26DA	+4	0809	
SP 013E	SS 26EA	FS 26DA	+6	6791	

После выполнения **CALL**:

<div> <div>ss:0144 6791</div> <div>ss:0142 0809</div> <div>ss:0140 0501</div> <div>ss:013E 5BC9</div> <div>ss:013C 0015</div> </div>			<div> <div>sp 013C</div> <div>ds 5BED</div> <div>es 5BC9</div> <div>ss 5BD9</div> <div>cs 5BEF</div> <div>ip 001C</div> </div>		
SI 0000	CS 2700	IP 001C	Stack +0	0015	
DI 0000	DS 26FE		+2	26DA	
BP 0000	ES 26DA	HS 26DA	+4	0501	
SP 013C	SS 26EA	FS 26DA	+6	0809	

Перед выполнением **RET**:

			<div> <div>SUB SP, 2</div> <div>MOV [SP], IP</div> <div>LEA IP, MY PROC</div> </div>		
<div> <div>ss:0144 6791</div> <div>ss:0142 0809</div> <div>ss:0140 0501</div> <div>ss:013E 5BC9</div> <div>ss:013C 0015</div> </div>			<div> <div>sp 013C</div> <div>ds 5BED</div> <div>es 5BC9</div> <div>ss 5BD9</div> <div>cs 5BEF</div> <div>ip 001E</div> </div>		
SI 0000	CS 2700	IP 001E	Stack +0	0015	
DI 0000	DS 26FE		+2	26DA	
BP 0000	ES 26DA	HS 26DA	+4	0501	
SP 013C	SS 26EA	FS 26DA	+6	0809	

После выполнения **RET**:

<div> <div>ss:0146 3245</div> <div>ss:0144 6791</div> <div>ss:0142 0809</div> <div>ss:0140 0501</div> <div>ss:013E 5BC9</div> </div>			<div> <div>sp 013E</div> <div>ds 5BED</div> <div>es 5BC9</div> <div>ss 5BD9</div> <div>cs 5BEF</div> <div>ip 0015</div> </div>		
SI 0000	CS 2700	IP 0015	Stack +0	26DA	
DI 0000	DS 26FE		+2	0501	
BP 0000	ES 26DA	HS 26DA	+4	0809	
SP 013E	SS 26EA	FS 26DA	+6	6791	

MOV IP [SP]

ADD SP, 2

Продлится выполнение, начиная с команды **NEXT**:

Поскольку значение **IP** изменилось, то начнется выполнение процедуры, начиная с команды MOV CL, 10 и до команды **RET**.

Процедура может размещаться в том же сегменте, откуда осуществляются к ней обращения, или в другом сегменте. В первом случае атрибутом процедуры может быть **NEAR** и обращение также типа **NEAR**. Во втором случае атрибутом процедуры будет **FAR** и обращение также будет **FAR**. Но если процедура все-таки имеет атрибут **FAR**, то обращение к ней также должно иметь атрибут **FAR**.

Например:

```
SEGX SEGMENT
.....
SUBT PROC FAR
.....
RET
SUBT ENDP
.....
CALL FAR PTR SUBT
.....
SEGX ENDS
SEGY SEGMENT
.....
CALL FAR PTR SUBT
.....
SEGY ENDS
```

То есть, в обращении к процедуре могут быть атрибуты **FAR PTR** или **NEAR PTR**. В данном случае процедура определена перед ее вызовом, то есть, осуществляется вызов “назад”. Поэтому если не отметить **FAR PTR**, то ассемблер все равно правильно реализует и вызовы и возвращения.

В данном случае их можно оставить для ясности. Но когда процедура реализуется после вызова, то есть, осуществляется вызов “вперед”, то эти атрибуты обязательны.

Однако таких действий маловато. Для вызова процедуры из сегмента **SEGY** в нем обязательно нужно объяснить, что имя **SUBT** является внешним. То есть, включить директиву **EXTRN SUBT: FAR**.

В сегменте **SEGX** нужно объяснить, что к имени **SUBT** может быть доступ извне, то есть, нужно включить директиву **Public SUBT**.

Отметим, что имя **SUBT** находится в том же сегменте, что и директива **Public**. Поэтому в ней отмечать тип имени не нужно (транслятор разберется). В директиве **EXTRN** отметить тип нужно. По аналогии с командой **JMP** в процедуре возможны обращения:

1. прямые;
2. не прямые;
3. близкие;
4. далекие.

Не позволяют короткие **SHORT**, потому что считается, что процедура не располагается рядом с командой вызова. Поэтому, возможны такие варианты вызова:

```
CALL P; близкий переход
CALL FAR PTR Q; далекий переход
CALL Q; близкий переход
CALL NEA; близкий не прямой вызов
CALL FA; далекий не прямой вызов
LEA BX,Q
CALL [BX]; близкий вызов процедуры Q
CALL DWORD PTR [BX]; далекий не прямой вызов
Пример:
NEAR DW P
FA DD Q
P PROC
P1: ....
P ENDP
Q PROC FAR
Q1: .....
Q ENDP
```