

лаба 4 - выполнение комплекса программы

Организация подпрограмм в БЭВМ. Команды вызова подпрограммы и возврата.

Часто встречаются случаи, когда отдельные части программы должны выполнять одни и те же действия по обработке данных (например, вычисление тригонометрической функции). В подобных случаях повторяющиеся части программы выделяют в подпрограмму, а в соответствующие места программы заносят лишь команды обращения к этой подпрограмме.

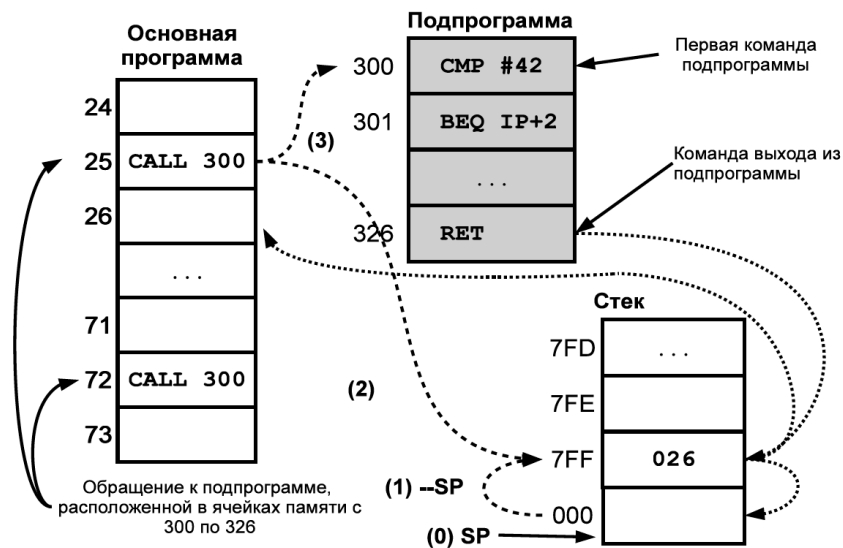


Рисунок В.6. Обращение к подпрограмме и возврат из нее

Подпрограммы осуществляют запись адресов возвратов в структуру данных, которая называется стек.

По команде

CALL 300, расположенной в ячейке 25 (рис В.6), **выполняется запись адреса возврата** $25 + 1 = 26$ (значение счетчика команд после цикла выборки команды) **на вершину стека. Адрес возврата** - это адрес следующей

команды после CALL(IP+1), и этот адрес хранится, чтобы мы потом смогли вернуться к выполнению программы по завершении подпрограммы.

Для этого сначала уменьшается указатель стека на 1, и производится запись в ячейку памяти, на которую указывает SP. Если стек пуст, как в нашем случае, то указатель будет находиться в ячейке 7FF(26 записывается в 7FF). После того как адрес возврата записан в стек, нужно передать управление подпрограмме: **записывается аргумент команды CALL (прямого абсолютного адреса 300) в IP (адрес первой команды подпрограммы, т.е IP = 300).**

Далее начинается процесс выполнения команд подпрограммы, который завершается командой

RET, расположенной в ячейке 326.

Эта команда выхода из подпрограммы предписывает ЭВМ выполнить переход к команде, расположенной по адресу, сохраненному в текущей ячейке указателя стека (7FF). Получается, будет исполняться команда с адресом 26. Аналогично выполняется команда CALL 300, расположенная в ячейке 72 (после выполнения команд подпрограммы будет выполнен переход к ячейке 73).

Аргументы и возвращаемые значения подпрограммы. Способы организации передачи аргументов и возвращаемых значений.

Как можно передавать аргументы в подпрограмму и получать результаты

1. Через аккумулятор (регистр общего назначения). Самый быстрый способ

Сколько параметров можно передать в БЭВМ?

Аккумулятор состоит из двух слов, поэтому через аккумулятор можно передать 16 значений (если трактовать как логическое значение, то есть отдельно 16 логических значений)

2. Адресуемые ячейки памяти. (Данные для подпрограммы записываются в заранее определенные ячейки памяти, а затем подпрограмма может обращаться к этим ячейкам для получения аргументов)

3. Стек. Является достаточно быстрым, т.к стек может быть кэширован

Рекурсивный вызов подпрограмм. Организация стека.

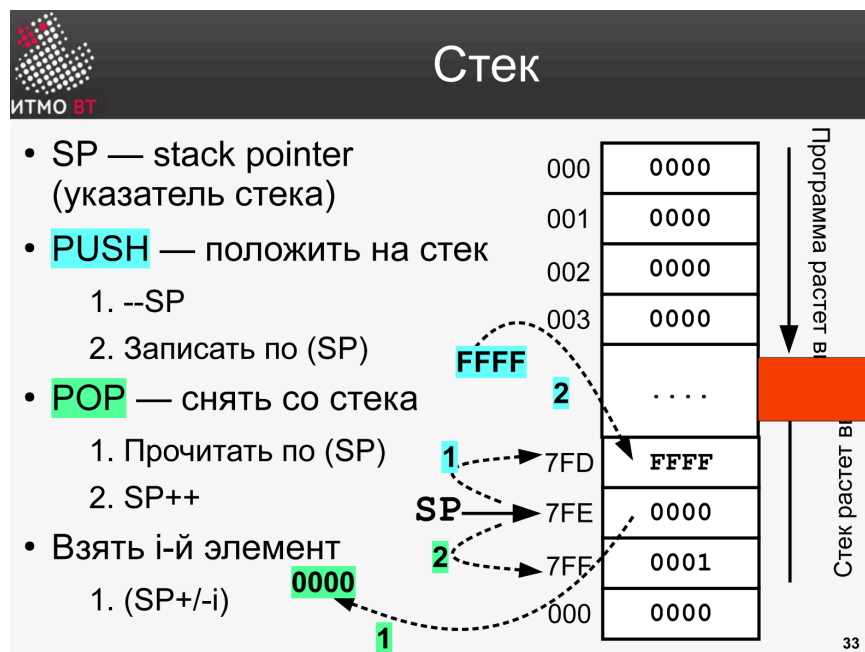
Стек - структура данных, представляющая собой список элементов, который работает по принципу(LIFO, «первым вошел — последним вышел»).

В базовой ЭВМ стек организован при помощи специального регистра — **указателя стека (SP)**, который изначально сброшен в значение 0, и так как SP 11-ти разрядный, то **первый элемент стека имеет адрес 7FF**, следующий 7FE, т.е. растет от старших адресов к младшим.

Есть 2 команды, которые работают со стеком

| 0C00 | PUSH | - | - | - | - | AC → - (SP)

| 0800 | POP | * | * | 0 | - | (SP) + → AC




PUSH - команда, с помощью которой можно положить на стек значение из AC. Она сначала уменьшает текущее значение указателя стека на 1 ($SP = SP - 1$), а потом по адресу, который получился в SP записывает значение AC

POP - команда снятия со стека, действует наоборот. Сначала по адресу, который содержится в SP, читает значение из памяти, а потом увеличивает значение SP на 1 ($SP = SP + 1$)

Взять i -ый элемент (pick) можно при помощи команды LD и адресации со смещением относительно SP

В стеке могут размещаться не только адрес возврата из подпрограммы, но и передаваемые подпрограмме параметры и возвращаемый результат. Перед вызовом подпрограммы по команде CALL в стеке размещается заданное количество параметров, которые помещаются в стек командами PUSH. После возврата из подпрограммы стек должен быть приведен в исходное состояние, из него должны быть прочитаны результаты подпрограммы, а указатель стека должен быть возвращен в состояние до первой команды PUSH. В БЭВМ для этого предназначена команда POP.

Доступ к переданным в стек параметрам и формирование результатов в БЭВМ из подпрограммы производится при помощи **специального режима адресации — со смещением относительно SP**. При этом загрузка самого верхнего параметра (который мы положили самым последним перед вызовом CALL) будет иметь смещение 1. Смещение 0 указывает на адрес возврата из подпрограммы, который в момент входа в подпрограмму лежит на вершине стека.



Рекурсивность сумма чисел от 1 до N

• Рекурсивность — способность подпрограммы вызывать саму себя.

Стек


7F9	...
7FA	0036
7FB	0001
7FC	0036
7FD	0002
7FE	0013
7FF	0003
000	0000

Основная программа →

Подпрограмма →

Адрес	Содержимое	
	Код	Мнемоника
010	AF03	LD #3
011	0C00	PUSH
012	D030	CALL 0x30
013	0800	POP
014	E016	ST 0x16
015	0100	HLT

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
030	AC01	LD &1	Аргумент в AC == 1 ?
031	7F01	CMP #1	
032	F006	BEQ IP+6	На выход!
033	0740	DEC	Уменьшить
034	0C00	PUSH	Аргумент на 1
035	DEFA	CALL IP-6	Вызвать себя
036	0800	POP	Сумма в AC
037	4C01	ADD &1	Добавить себя
038	EC01	ST &1	Сохранить Сумму
039	0A00	RET	



Описание команд CALL и RET: наименование, назначение, тип команды и вид адресации. Количество и название машинных циклов, потактовое выполнение команды, количество обращений к памяти

DXXX	CALL M	-	-	-	-	SP - 1 → SP, IP → (SP), M → IP
------	--------	---	---	---	---	--------------------------------

0A00	RET	-	-	-	-	(SP) + → IP
------	-----	---	---	---	---	-------------



Цикл исполнения CALL

CALL: DR после м.ц. OF содержит адрес перехода

- DR → BR ; Адрес перехода записать в BR
- IP → DR ; Подготовить адрес возврата для записи в стек
- BR → IP ; Переход на подпрограмму
- ~0 + SP → SP, AR ; Уменьшить стек на 1
- DR → MEM(AR) ; Записать адрес возврата
- GOTO INT ; Завершение цикла



Цикл исполнения RET

- SP → AR ; Вершину стека поместить в AR
- MEM(AR) → DR ; Прочитать адрес возврата
- DR → IP ; Вернуться из подпрограммы
- SP + 1 → SP ; Увеличить стек на 1
- GOTO INT ; Завершение цикла

В этих 2 циклах по одному обращению к памяти

CALL является адресной командой, RET - безадресная команда