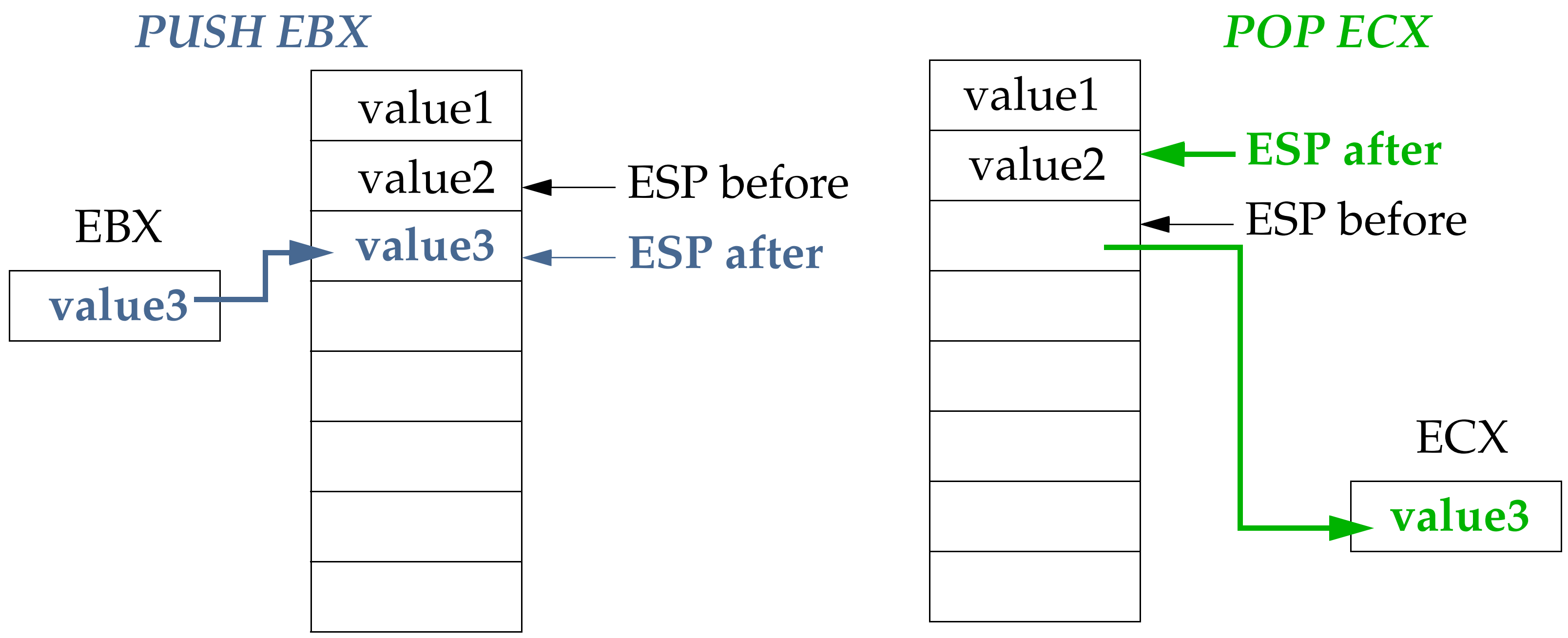
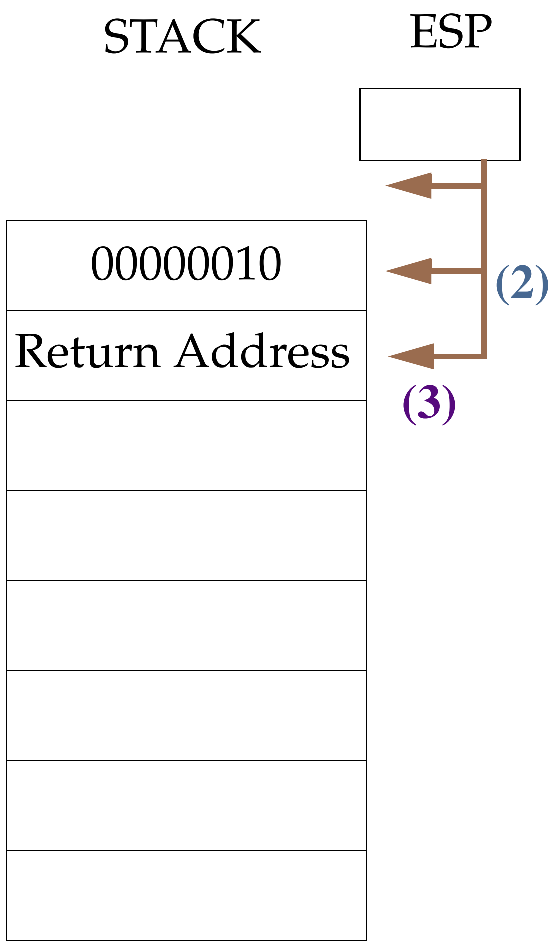
Purpose of Stack

- Memory used to pass parameters to procedures (including C function calls)
- Memory used for allocating space for local variables
- Save return address in procedure calls
- Save registers to be preserved across procedure calls



Passing Parameters to Procedures

```
        section .data
input_filename_ptr :  dd 0                      (1)
        section .text
main:
        push dword input_filename_ptr           (2)
        call GetCommandLine                     (3)
        add esp, 4
```



(1) *input_filename_ptr :*

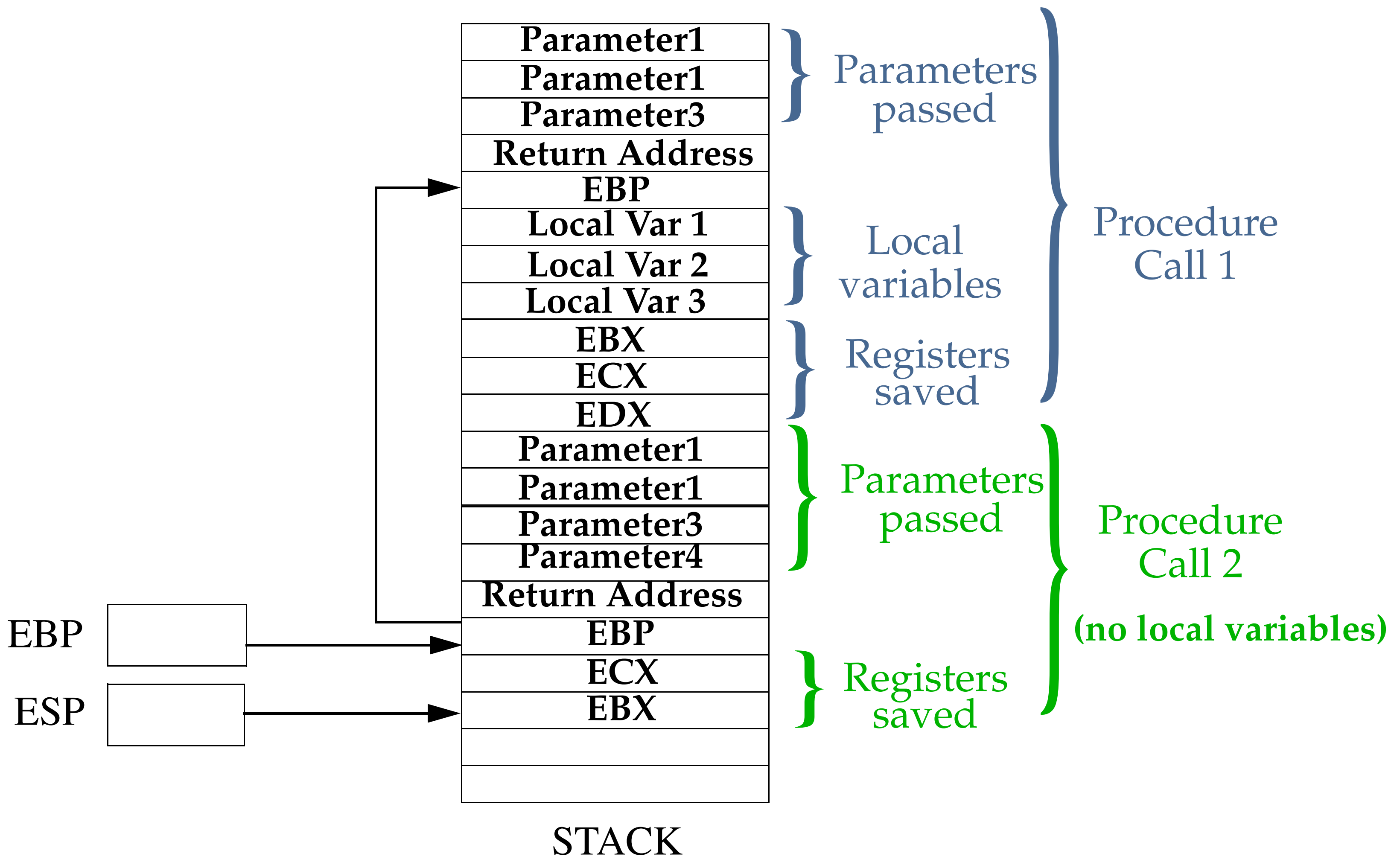
00000010 | Pointer to the filename

(2) Push the address of the pointer to the filename

(3) Return address pushed to the stack.
Address of the add instruction.

Call Frames

One call frame created per procedure call



Setting up Call Frames

```
GetCommandLine:
        Enter 0                        (1)
        Push_Regs ebx, ecx, edx        (2)
```

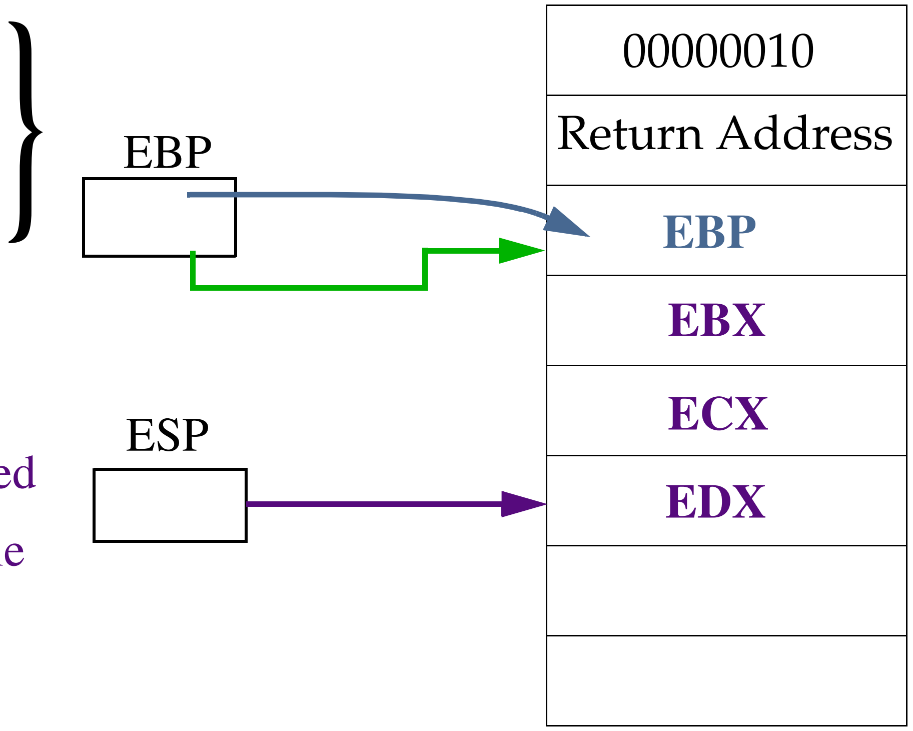
```
%macro  Enter 1
    push ebp
    mov ebp, esp
    sub esp, %1
%endmacro
```

(1) Push EBP

Move ESP into EBP
i.e. EBP points to the pushed EBP

Allocate space for local variables
(none in this example)

(2) Push the registers that are to be saved
EBX, ECX and EDX in this example



UMBC
AN HONORS UNIVERSITY IN MARYLAND

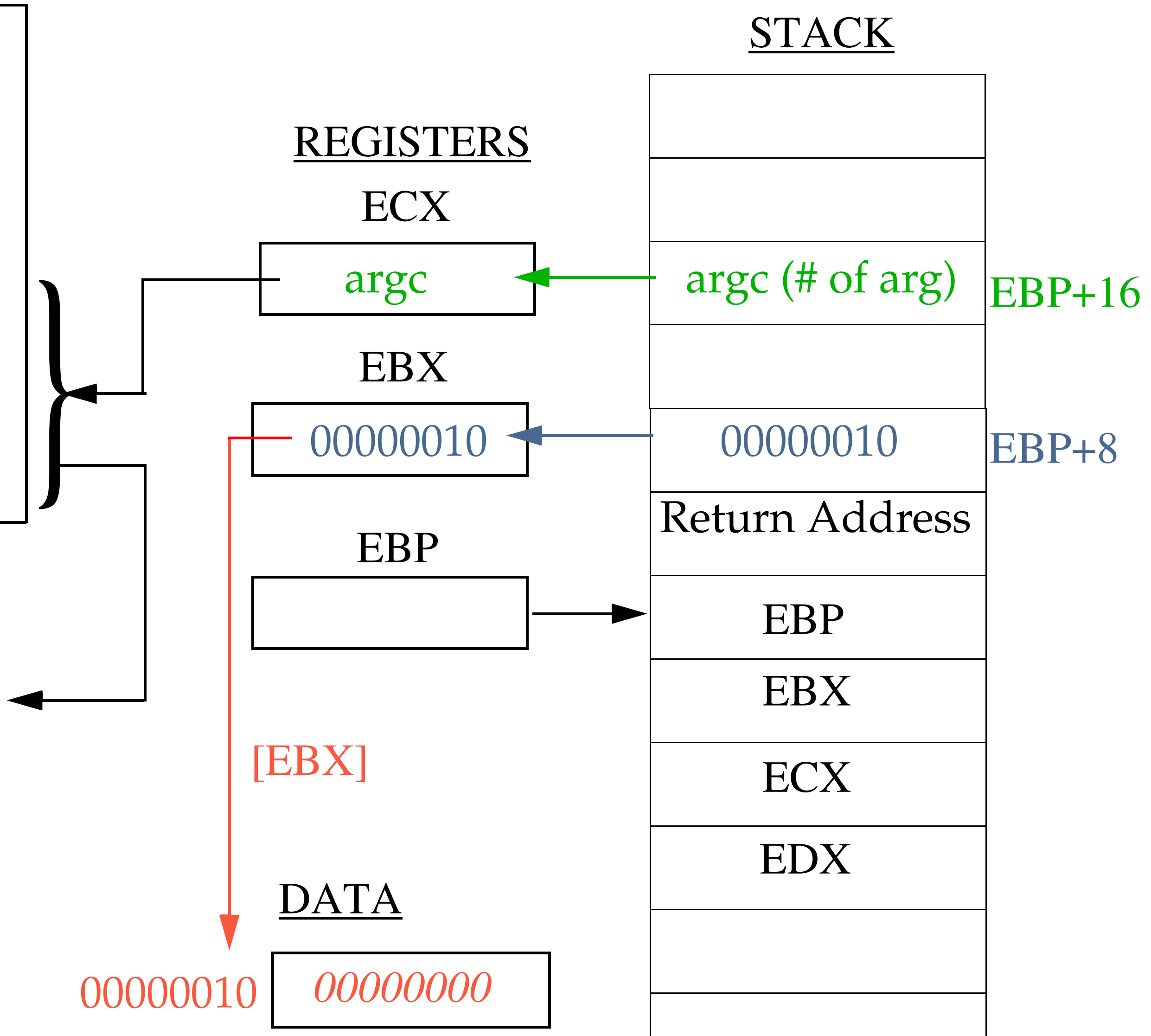
Reading Arguments

```
mov ebx, [ebp + 8]
mov [ebx], dword 0

mov ecx, [ebp + 16]
cmp ecx, 2
    if ne
        jmp gcl_done
    endif
```

Exactly 2 arguments required

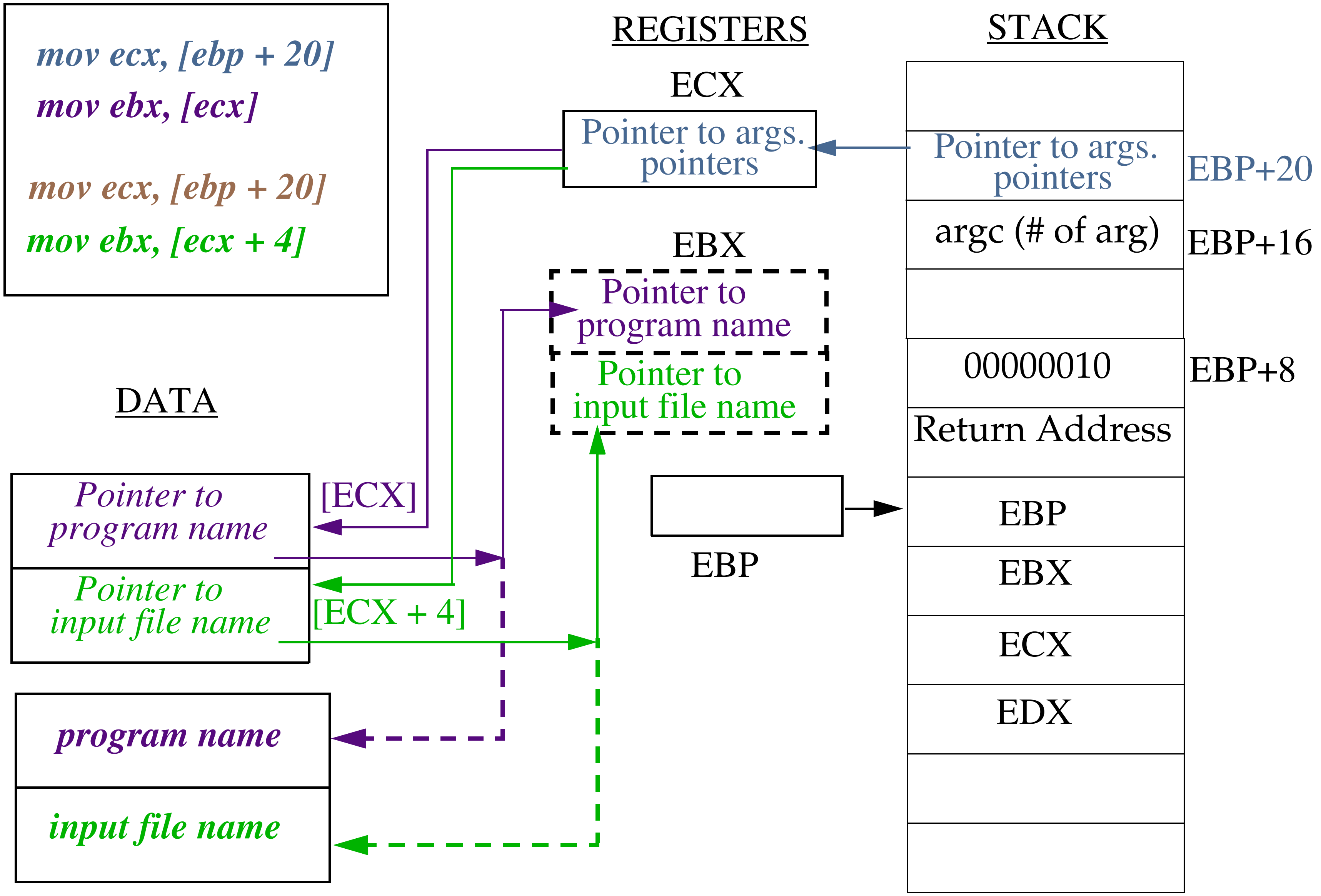
Program name and input file name

ELSE ERROR!!!



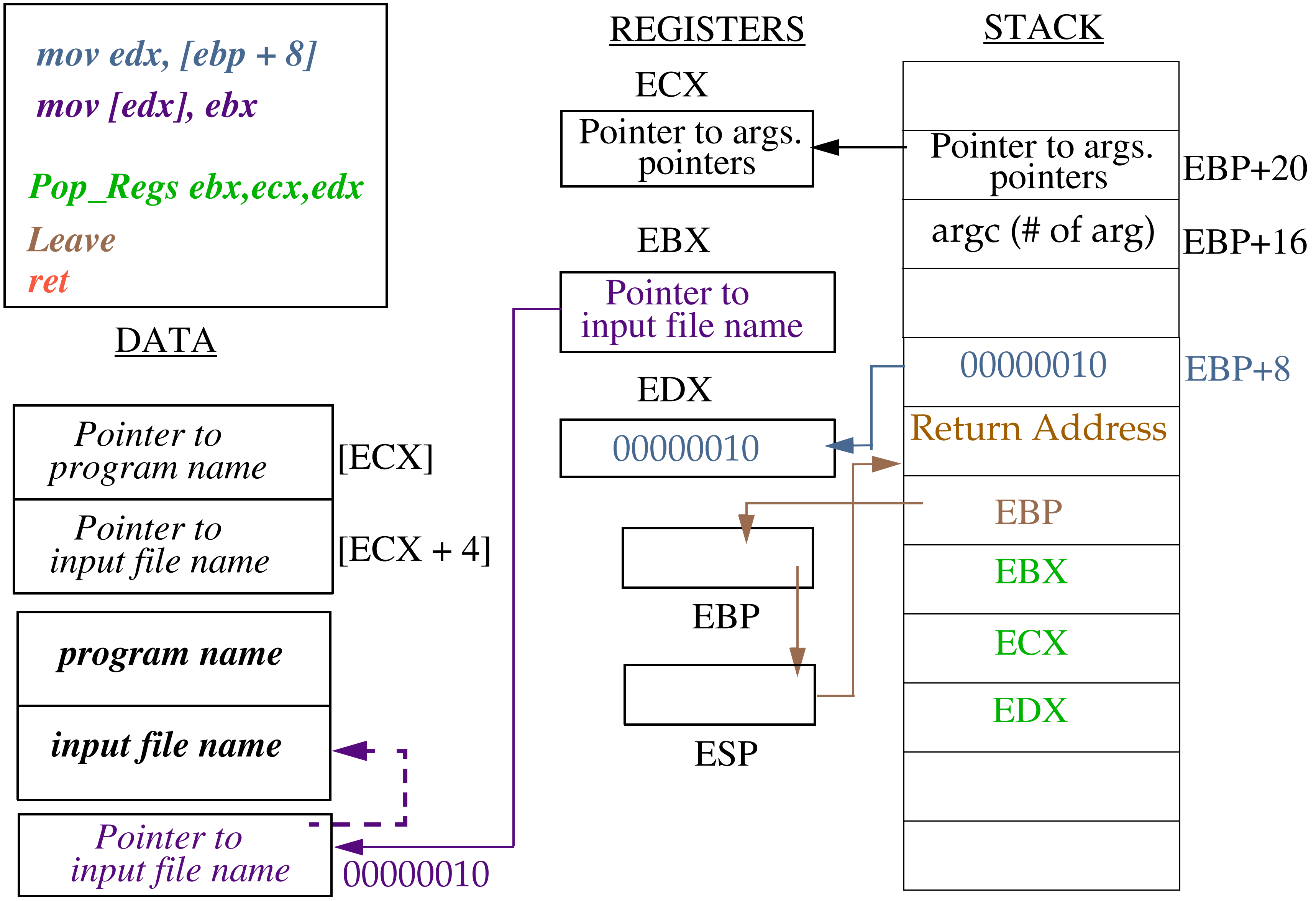
Reading Arguments

```
mov ecx, [ebp + 20]
mov ebx, [ecx]

mov ecx, [ebp + 20]
mov ebx, [ecx + 4]
```



Get argument and Return

```
mov edx, [ebp + 8]
mov [edx], ebx

Pop_Regs ebx,ecx,edx
Leave
ret
```



Procedure Calls (Steps Recap)

Caller: Before Call

- Save registers that are needed (for C functions save EAX, ECX, EDX)
- Push arguments, last first
- CALL the function

Callee:

- Save caller's EBP and set up callee stack frame (ENTER macro)
- Allocate space for local variables and temporary storage
- Save registers as needed (C functions save EBX, ESI, EDI)
- Perform the task
- Store return value in EAX
- Restore registers (C functions restore EBX, ESI, EDI)
- Restore caller's stack frame (LEAVE macro)
- Return

Caller: After Return

- POP arguments, get return value in EAX, restore registers (for C EAX, ECX, EDX)