x86: Práctica 3

Llamada a subprogramas

Departamento de Ingeniería en Automática, Electrónica, Arquitectura y Redes de Computadores

Universidad de Cádiz

Índice

Índice

Forma básica de un subprograma en ensamblador

```
1  ;Codigo del programa llamador
2      mov ecx, retorno          ;Direccion de retorno
3      mov eax, parametro1       ;Parametro 1 subprograma
4      jmp nombresubprograma
5  retorno:
6      ;Resto del codigo del programa llamador
7  ;Fin programa llamador
8
9  ;Inicio de subprograma
10 nombresubprograma:
11     ;Cuerpo del subprograma
12     ;El subprograma espera en eax el parametro1
13     jmp ecx     ;Vuelve a la direccion de retorno
14                 ;almacenada en ecx.
15 ;Fin subprograma
```

Índice

La pila y las instrucciones CALL y RET

La **pila** y las instrucciones **CALL** y **RET** son herramientas que la arquitectura ofrece para facilitar la llamada a subprogramas.

- La pila suele utilizarse para **pasar parámetros a subprogramas**. El registro EBP se utilizará para poder acceder a los parámetros insertados en la pila **sin sacarlos**. Se introducirá el valor de EBP en la pila al principio de los subprogramas y se igualará a **ESP** (**prólogo**). Al final se restaurará su valor original(**epílogo**).
- Las instrucciones **CALL** y **RET** serán utilizadas siempre para llamar a un subprograma y volver de un subprograma al programa llamador, respectivamente.

La pila

- Es una estructura **LIFO (Last In First Out)**
- El tamaño de cada marco de la pila depende de la máquina. Normalmente suele ser de **doble palabra** (32 bits, 4 bytes). Cuando se mete o se saca un dato de la pila se hace siempre **con el tamaño del marco de pila**.
- Al introducir un dato en la pila, esta **crece hacia las direcciones menores (puntero de pila - 4)**
- Al sacar un dato de la pila, **esta decrece hacia las direcciones mayores (puntero de pila + 4)**

La pila

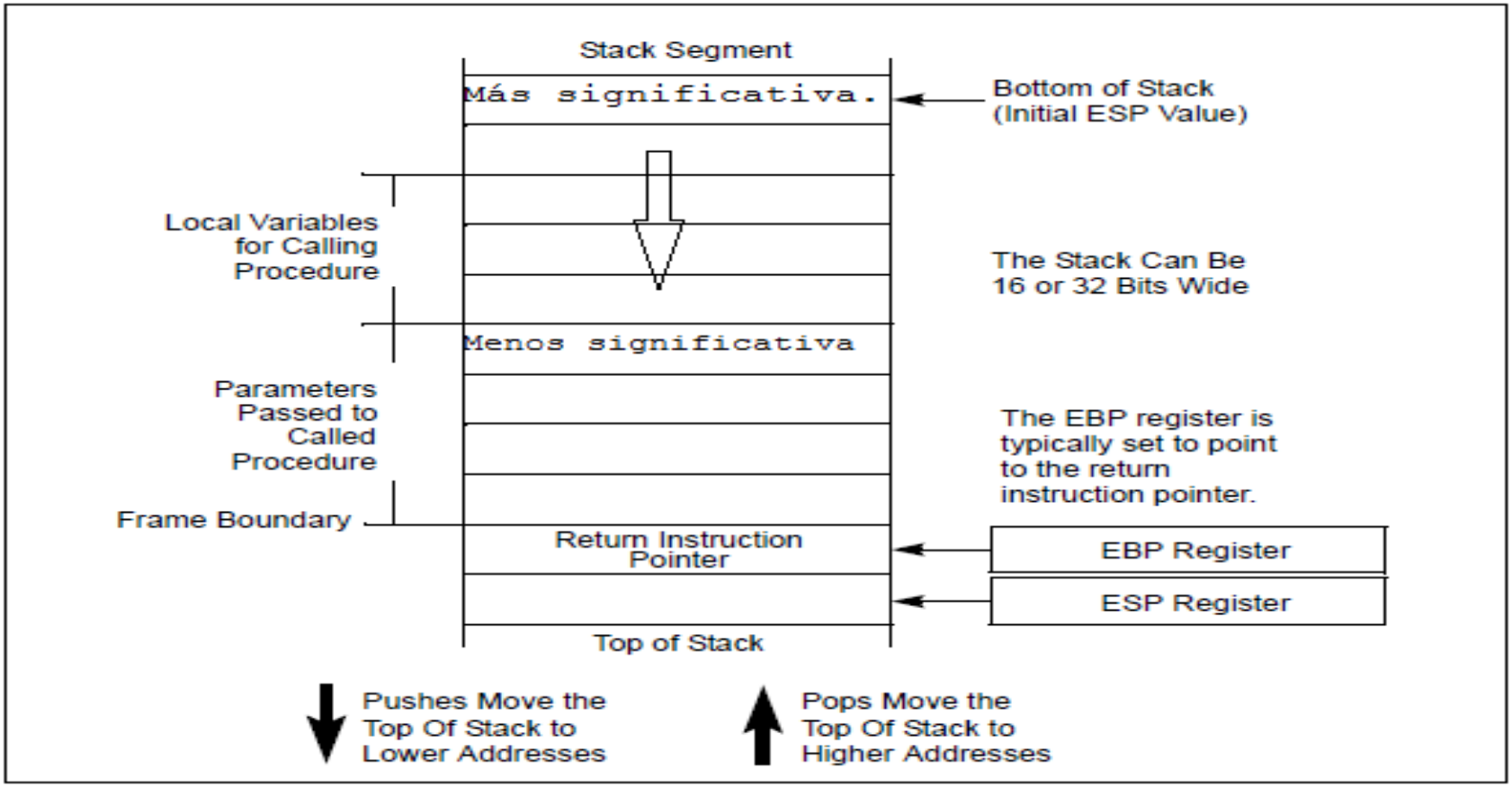

Figura: Funcionamiento de la pila. (Imagen del manual de Intel)

Las instrucciones CALL y RET

- La instrucción **CALL** seguida de un desplazamiento (normalmente representado por una **etiqueta** que apunta al inicio de un subprograma), se usa para llamar a un subprograma.
- **CALL** almacena la dirección de retorno en la pila y realiza el salto al inicio del subprograma.
- **RET** saca de la cima de la pila la dirección de retorno y **salta a dicha dirección**. La dirección de retorno debe estar **en la cima de la pila** para que funcione correctamente. Opcionalmente, puede añadirse a la instrucción un operando inmediato, que representa el número de bytes que sacará de la pila **tras haber sacado la dirección de retorno de la cima de la pila**.

Índice

Convenciones de llamada

Una convención de llamada establece un contrato entre un programa llamador y un subprograma llamado. Incluye la siguiente información:

- Cómo deben **pasarse los parámetros**.
- Cómo debe **pasarse el valor devuelto**.
- Cómo debe quedar el ambiente básico de ejecución al finalizar la invocación (**especialmente la pila y los registros**).

Índice

Convención de llamada cdecl (C declaration)

- Los parámetros se pasan al subprograma **introduciéndolos en la pila en el orden inverso a la signatura del subprograma**. El resultado devuelto se pasa en el registro **EAX**.
- Después de la llamada, el fragmento llamador **debe eliminar de la pila los parámetros que insertó**. De esta forma, el número de parámetros puede ser **variable**.
- Los registros **EAX**, **ECX** y **EDX** pueden modificarse en el subprograma, pero los demás **deben preservar su contenido**. Para conservar el contenido de un registro se almacena en la pila al principio del subprograma y se restaura al final.

Ejemplo de la convención de llamada de C

```
1     ;Codigo del programa llamador
2     pusha          ;Guarda los registros de
3                    ;proposito general en la pila
4     push param2    ;Los parametros se introducen
5     push param1    ;en orden inverso al que se pasan
6     call subpr
7     add esp,8      ;Sumamos a esp 4 por cada parametro
8     popa           ;Restaura el contenido de los
9                    ;registros de proposito general.
10    ;Resto del codigo del programa llamador
```

Ejemplo de la convención de llamada de C

```
1  ;Inicio de subprograma
2  subpr:
3      push ebp        ;Estas dos instrucciones son
4      mov ebp, esp    ;el prologo del subprograma
5      ;Cuerpo del subprograma
6      ;El subprograma utiliza los parametros
7      ;de la pila sin sacarlos de la misma
8      pop ebp         ;Estas dos instrucciones son
9      ret             ;el epilogo del subprograma
10     ;La direccion de retorno debe estar en la
11     ;cima de la pila
12 ;Fin del subprograma
```

Índice

Convención de llamada stdcall

- Los parámetros **se pasan a los subprogramas insertándolos en la pila en el orden inverso a la signatura del subprograma**. El resultado devuelto se pasa en el registro **EAX**.
- Antes de finalizar, **el subprograma debe eliminar de la pila los parámetros que le pasaron**. El número de estos debe ser **fijo**.
- Los registros **EAX**, **ECX** y **EDX** pueden modificarse en el subprograma, pero los demás **deben preservar su contenido**. Para conservar el contenido de un registro se almacena en la pila al principio del subprograma y se restaura al final.

Ejemplo de la convención de llamada stdcall

```
1       ;Codigo del programa llamador
2       pusha           ;Guarda los registros de
3                       ;proposito general en la pila
4       push param2     ;Los parametros se introducen
5       push param1     ;en orden inverso al que se pasan
6       call subpr
7       popa            ;Restaura el contenido de los
8                       ;registros de proposito general.
9       ;Resto del codigo del programa llamador
```

Ejemplo de la convención de llamada de stdcall

```
1  ;Inicio de subprograma
2  subpr:
3      push ebp        ;Estas dos instrucciones son
4      mov ebp, esp    ;el prologo del subprograma
5      ;Cuerpo del subprograma
6      ;El subprograma utiliza los parametros
7      ;de la pila sin sacarlos de la misma
8      pop ebp         ;Estas dos instrucciones son
9      ret 8           ;el epilogo del subprograma
10     ;La direccion de retorno debe estar en la
11     ;cima de la pila. Ademas debe ponerse como
12     ;operando inmediato el numero de bytes de
13     ;los parmaetros para que sean eliminados
14     ;antes de volver
15 ;Fin del subprograma
```

Índice

Llamadas al sistema

A diferencia de otros sistemas, **en Windows no se conocen los códigos de llamadas al sistema**, por lo que deben hacerse mediante llamadas a la **API de Win32** o a la **API de C**.

- Para llamar a las funciones, puede utilizarse la pseudoinstrucción **invoke** *nomfuncion*, *param1*, *param2*,... (teniendo en cuenta que cada parámetro separado por coma se introduce como **DWORD**)
- Es equivalente a realizar:
  push *param2*
  push *param1*
  call *nomfuncion*

Índice

La API de C

- Las funciones de la **API de C** utilizan la convención **cdecl**.
- Tras llamar a una función, el llamador debe eliminar de la pila los parámetros introducidos, **sumando a ESP tantos bytes como ocupen los parámetros**.
- Los registros **EAX**, **EDX** y **ECX** podrían ser modificados por la función llamada. Es conveniente guardarlos en la pila antes de llamar a la función, para restaurarlos después.
- Para utilizar las funciones, se debe enlazar al proyecto la librería **msvcrt.dll**.

Índice

La API de Win32

- Todas las funciones de esta API utilizan la convención de llamada **stdcall** (excepto la función **wsprintf**, que utiliza **cdecl**.)
- Al llamar funciones que utilizan **stdcall**, el llamador **no tiene que eliminar** los parámetros introducidos en la pila.
- Los registros **EAX**, **EDX** y **ECX** pueden ser modificados por la función. Es conveniente guardarlos en la pila antes de llamar a la función, para restaurarlos después.
- Para utilizar la mayoría de las funciones, se debe enlazar al proyecto la librería **kernel32.dll**