Convención de llamada linux x86_64

Parámetros y valores de retorno 64 bits

- **Enteros y punteros**: RDI, RSI, RDX, RCX, R8, R9
- **Flotantes**: XMM0, … , XMM7
- **Retorno**: RAX, XMM0
- **Temporales**: RAX, R10, R11, XMM8, …, XMM15, st2, …, st7, k0, …, k7
- **long doubles (temporales)**: st0, st1

No volatiles: RBX, RBP, R12, R13, R14, R15

Las funcioens llamadas si quieren modificar registros no volatiles tienen la obligación (por convención) de restaurarlos al terminar.

Los parametros que entran por registros se pasan de izquierda a derecha. Los que no alcanzan a entrar, se pasan por stack de derecha a izquierda (viendolo desde la declaración de la función).

Para llamadas a funciones de C, se necesita la pila alineada a 16 bytes (en 32 bits también)

Parámetros y valores de retorno 32 bits

- Todos los parámetros se pasan por pila (de derecha a izquierda)
- **Retorno**: EAX
- **No volatiles**: EBX, EBP, ESI, EDI

Modos de acceso a memoria

- [inmediato]
- [registro]
- [registro + registro*escala] siendo escala 1, 2, 4 u 8
- [registro + inmediato]
- [reg + reg*escala + inm]

Registros de Propósito General

Intel 64

63 … 0
rax
rbx
rcx
rdx
rsi
rdi
rbp
rsp
r8
r9
r10
r11
r12
r13
r14
r15

Nombres para acceder a los bits del registro en las posiciones				
63-0 (64 bits)	31-0 (32 bits)	15-0 (16 bits)	15-8 (8 bits)	7-0 (8 bits)
rax	eax	ax	ah	al
rbx	ebx	bx	bh	bl
rcx	ecx	cx	ch	cl
rdx	edx	dx	dh	dl
rsi	esi	si		sil
rdi	edi	di		dil
rbp	ebp	bp		bpl
rsp	esp	sp		spl
r8	r8d	r8w		r8b
r9	r9d	r9w		r9b
r10	r10d	r10w		r10b
r11	r11d	r11w		r11b
r12	r12d	r12w		r12b
r13	r13d	r13w		r13b
r14	r14d	r14w		r14b
r15	r15d	r15w		r15b

Alineación de structs

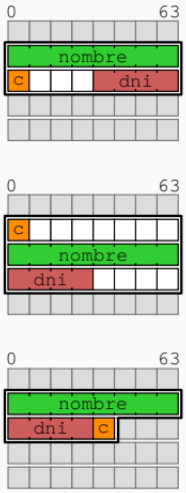
- Cada variable debe estar alineada a una pocisión multiplo de su tamaño.
- El tamaño de la estructura debe estar alineado al tamaño del atributo más grande

- En ambos casos se agrega padding para rellenar (se puede sacar con **___attribute___((*__packed__*))**)

```
struct alumno {
 char* nombre;     → 8     ⇒ 0
 char comision;    → 1     ⇒ 8
 int dni;          → 4     ⇒ 12
};                         ⇒ 16

struct alumno2 {
 char comision;    → 1     ⇒ 0
 char* nombre;     → 8     ⇒ 8
 int dni;          → 4     ⇒ 16
};                         ⇒ 24

struct alumno3 {
 char* nombre;     → 8     ⇒ 0
 int dni;          → 4     ⇒ 8
 char comision;    → 1     ⇒ 12
} __attribute__((packed));  ⇒ 13
```



Interacción con C

- Las funciones exportadas se deben declarar en la sección .text con ***global func***
- Las funciones de C llamadas desde ASM se deben declararen .text con ***extern func***

Secciones del código

- **.data**: variables globales inicializadas (DB: define byte, DW: word, DD: double word, DQ: quad word)
- **.rodata**: constantes globales inicializadas (DB, DW, DD, DQ)
- **.bss**: variables globales no inicializadas (RESB, RESW, RESD, RESQ) (reserve)
- **.text**: codigo

Dentro de *.text* la etiqueta _start sería el equivalente a la función *main*

Para ensamblar un mismo valor repetido: *"etiqueta"* **times** *"numero"* DB/BW/DD/DQ *"hexa/entero/binario/octal"*

Instrucciones más comunes

En general las instrucciones son registro-registro; registro-memoria; registro-inmediato; memoria-registro; memoria-inmediato

- ADD
- SUB
- INC
- DEC
- OR
- AND
- NOT
- XOR
- POP
- PUSH
- CALL
- RET
- MOV
- SHL
- SHR
- JE
- JGE
- JZ
- JMP

- CMP
- DIV
- MUL
- POP
- PUSH
- CALL
- RET
- MOV
- SHL
- SHR