Máquina Virtual de MAPL

(Máquina Abstracta para Procesadores de Lenguajes)

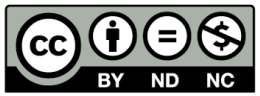
Diseño de Lenguajes de Programación

Ingeniería Informática

Universidad de Oviedo

(v2.2)

Raúl Izquierdo Castanedo



Arquitectura MAPL

Segmentos de memoria

- Segmento de datos de 1024 bytes.
- Segmento de código separado de los datos en el que cada instrucción ocupa una dirección.

Distribución del segmento de datos

- La memoria estática comienza en la dirección 0.
- La pila comienza en la última dirección del segmento de datos y crece hacia abajo (meter valores en la pila decrementa SP).

Registros

- IP (segmento de código). Dirección de la instrucción actual.
- SP (segmento de datos). Dirección de la cima de la pila.
- BP (segmento de datos). Dirección del *stack frame* (dirección de retorno y antiguo BP) de la función actual.

Tamaño de los tipos primitivo

- char = 1 byte
- int = 2 bytes
- float = 4 bytes
- address/dirección/puntero = 2 bytes

Juego de Instrucciones

Categoría	Bytes	Enteros (*)	Reales	Direcciones
Manipulación de la pila	pushb *cte*	pushi *cte*	pushf *cte*	pusha *cte*
	loadb	loadi	loadf	
	storeb	storei	storef	
	popb	popi	popf	
	dupb	dupi	dupf	
				pusha bp
Aritméticas		addi	addf	
		subi	subf	
		muli	mulf	
		divi	divf	
		mod		
Lógicas		and		
		or		
		not		
Comparación >		gti	gtf	
Comparación <		lti	ltf	
Comparación >=		gei	gef	
Comparación <=		lei	lef	
Comparación ==		eqi	eqf	
Comparación !=		nei	nef	
E/S	inb	ini	inf	
	outb	outi	outf	
Conversiones (**)		i2b		
	b2i		f2i	
		i2f		

Categoría	Instrucción
Salto	jmp *label*
	jz *label* *(**j**ump if **z**ero)*
	jnz *label* *(**j**ump if **n**o **z**ero)*
Funciones	call *label*
	ret *cte, cte, cte*
	enter *cte*
Otras	halt nop

(*) El sufijo *i* es opcional. Si no hay sufijo se asume que es una instrucción para enteros:

push, load, add, gt, eq, in, ,,,

(**) La primera letra indica el tipo del valor y la última el tipo a convertirlo:

i2f → Convertir entero a float

Instrucción *ret*. Resumen

Formato:

ret <*cte1*>, <*cte2*>, <*cte3*> (*ret* ⇒ *ret* 0, 0, 0)

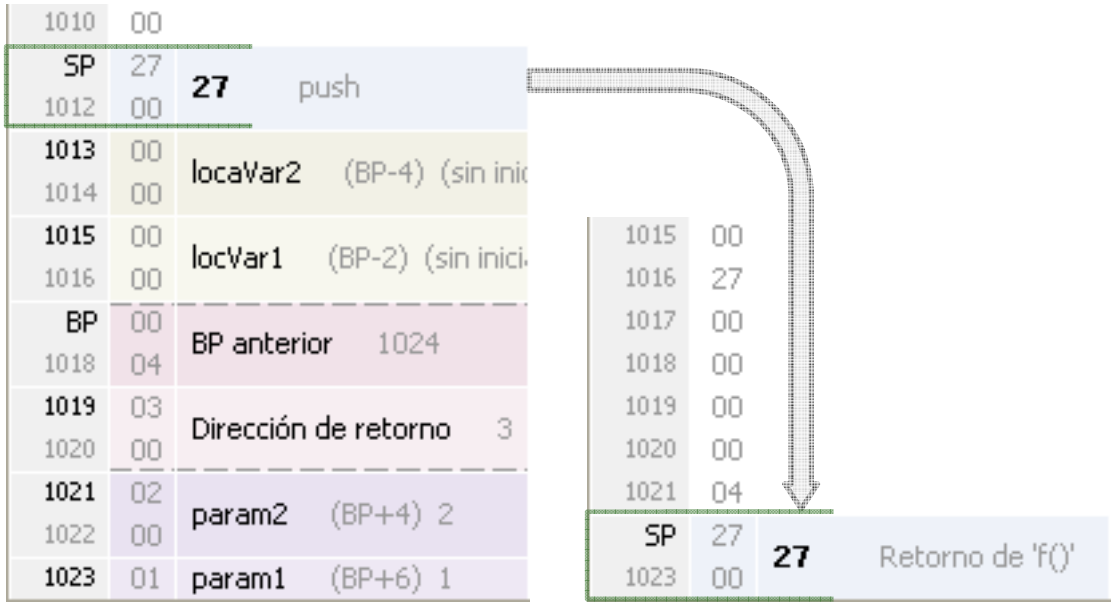
- cte1 = tamaño del valor de retorno (0 si no hay)
- cte2 = tamaño de las variables locales de la función
- cte3 = tamaño de los parámetros de la función

```
int f(byte param1, int param2)
{
   int locVar1, locVar2;

   return 27;
}
```

Código MAPL generado

```
f:  enter 4
    push 27
    ret 2, 4, 3
```



Pila antes de ret 2, 4, 3

Pila después de ret 2, 4, 3