Instrucción	**Significado**
NOP	;
END	exit(0);
MOV r_i, *entero*	r_i = *entero*;
MOV r_i, r_j	r_i = r_j;
GET r_i, BYTE [*entero*]	r_i = *(char*)*entero*;
GET r_i, WORD [*entero*]	r_i = *(int*)*entero*;
GET r_i, BYTE [r_j]	r_i = *(char*)r_j;
GET r_i, WORD [r_j]	r_i = *(int*)r_j;
PUT r_i, BYTE [*entero*]	*(char*)*entero* = r_i;
PUT r_i, WORD [*entero*]	*(int*)*entero* = r_i;
PUT r_i, BYTE [r_j]	*(char*)r_j = r_i;
PUT r_i, WORD [r_j]	*(int*)r_j = r_i;
ADD r_i, r_j, r_k	r_i = r_j + r_k;
SUB r_i, r_j, r_k	r_i = r_j - r_k;
MUL r_i, r_j, r_k	r_i = r_j * r_k;
DIV r_i, r_j, r_k	r_i = r_j / r_k;
MOD r_i, r_j, r_k	r_i = r_j % r_k;
LST r_i, r_j, r_k	r_i = (r_j < r_k);
LEQ r_i, r_j, r_k	r_i = (r_j <= r_k);
GRT r_i, r_j, r_k	r_i = (r_j > r_k);
GEQ r_i, r_j, r_k	r_i = (r_j >= r_k);
EQU r_i, r_j, r_k	r_i = (r_j == r_k);
NEQ r_i, r_j, r_k	r_i = (r_j != r_k);
JMP *etiqueta*	goto *etiqueta*;
JIF r_i, *etiqueta*	if (r_i != 0) goto *etiqueta*;
CALL *etiqueta*	$pila_{ip}$.push(IP); goto *etiqueta*;
RET	IP = $pila_{ip}$.pop( );
PUSH r_i	$pila_{datos}$.push(r_i);
POP r_i	r_i = $pila_{datos}$.pop( );
ASK WORD r_i	int x; scanf("%d", &x); r_i = x;
ASK BYTE r_i	char x; scanf("%c", &x); r_i = x;
SHOW WORD r_i	printf("%d", r_i);
SHOW BYTE r_i	printf("%c", r_i);

Instrucciones del lenguaje de bajo nivel y su significado.

La máquina virtual tiene diez registros $r_0, r_1, r_2, \ldots, r_9$ que guardan enteros de 32 bits con signo. La máquina virtual también cuenta con una memoria principal de 2^{24} bytes direccionables e indizados a partir de 0. Tanto los registros como los bytes de la memoria comienzan valiendo 0. Las instrucciones de un programa están numeradas implícitamente de arriba hacia abajo a partir de 0. El contador de programa *IP* es un registro que no se puede manipular arbitrariamente y que contiene el número de la siguiente instrucción a ejecutar (en principio, la instrucción que aparece abajo). La máquina virtual cuenta con una pila de tamaño arbitrario para contadores de programa y sólo se puede interactuar con ella mediante las instrucciones `CALL` y `RET`. A su vez, la máquina virtual cuenta con una pila de tamaño arbitrario para datos y sólo se puede interactuar con ella mediante las instrucciones `PUSH` y `POP`. Ambas pilas están fuera de la memoria principal. La ejecución del programa comienza en la primera instrucción y cualquier acción inválida (división entre cero, acceder a una dirección en memoria inválida, hacer un `RET` cuando la pila de contadores de programa está vacía, hacer un `POP` cuando la pila de datos está vacía, salirse de la lista de instrucciones del programa) resulta en comportamiento indefinido.

En el lenguaje de ensamblado, cada instrucción debe aparecer en su propia línea. Las líneas vacías se ignoran. Los comentarios de línea comienzan con `#` y no hay comentarios de bloque. Una etiqueta se declara con la notación *etiqueta:* como prefijo de una instrucción. Un identificador de etiqueta puede contener letras, dígitos y guiones bajos, excepto que no puede comenzar con dígito. Los símbolos de puntuación pueden estar rodeados de espacios en blanco. Los espacios entre nemónicos o especificadores de ancho de memoria pueden aparecer en exceso. La diferencia entre mayúsculas y minúsculas se ignora, incluso en etiquetas. Las instrucciones que toman etiquetas no pueden tomar enteros y las instrucciones que toman enteros no pueden tomar etiquetas.