Centro Asociado Palma de Mallorca

Arquitectura de Ordenadores

Tutor: Antonio Rivero Cuesta

Unidad Didáctica 2

Estructura de un Computador

Tema 8

Tipos de Instrucciones

En este capítulo se describe el juego de instrucciones en lenguaje ensamblador de un ordenador genérico.

Formatos de Instrucciones

La sintaxis de las instrucciones presenta una cierta uniformidad.

Un procesador determinado sólo entiende unos pocos formatos de instrucciones, de no ser así el diseño de la electrónica de la CPU sería mucho más complejo.

El juego de instrucciones y el formato de las mismas está íntimamente ligado al diseño del procesador.

El formato de una instrucción típica en un procesador genérico es:

FIGURA 11.1. Formato de una instrucción genérica

Los Bits de Condición

Los procesadores disponen de un conjunto de bits.

Se ponen a cero o uno dependiendo del resultado de la operación anterior.

Por ejemplo el Motorola 68000 dispone de un flag Z que se pone a cero o uno dependiendo que el resultado de la operación anterior haya sido cero o diferente de cero.

Las instrucciones de salto chequean los bits de condición para decidir si el salto se produce.

Normalmente todos los bits de condición se agrupan en un solo registro que se denomina registro de estado.

El número, nombre y significado de los diferentes bits de condición varía de unos procesadores a otros.

No obstante algunos bits son muy comunes.

Veamos los siguientes:

- Negativo (N).
- Cero (Z).
- Desbordamiento (V) Overflow.
- Acarreo (C) Carry.

Negativo (N)

Igual al bit más significativo (signo) del resultado.

Cero (Z)

1 si todos los bits del resultado son 0.

En caso contrario vale 0.

Desbordamiento (V) Overflow

En las operaciones aritméticas se pone a 1 si ocurre un complemento a 2. Se pone a 0 si no hay desbordamiento.

Cuando se produce una adición o substracción de dos números de *n* bits con signo.

V=1 indica que el resultado es mayor que $2^{n-1}-1$ o menor que 2^{n-1}.

La expresión N XOR V siempre da el signo correcto de un resultado en complemento a dos, pues V=1 indica que N está equivocado.

Acarreo (C) Carry

Durante las operaciones de adición se pone igual al bit de acarreo del bit más significativo.

Cuando se suman dos números de *n* bits, C=1 indica que el resultado es mayor que $2^{n-1}-1$.

En las operaciones de resta es lo mismo. Cuando se restan números sin signo C=1 indica resultado negativo.

Las reglas exactas de funcionamiento de los bits de condición pueden variar de forma más o menos arbitraria de unos procesadores a otros.

Por ejemplo qué se hace con los bits de condición C y V después de una operación de movimiento de información de un registro a otro o dentro de la memoria no está claro atendiendo a la definición de estos bits y hay variaciones de un procesador a otro.

De hecho algunos procesadores ni siquiera cambian estos bits.

Control del Procesador y bits de Estatus

La mayoría de los procesadores tienen un pequeño número de bits que controlan ciertos modos generales de operación del procesador.

Los siguientes son algunos de ellos:

- Interrupción habilitada.
- Seguimiento de interrupciones.
- Funciones especiales.
- Nivel de privilegio.

Interrupción Habilitada

Uno o más bits pueden controlar el funcionamiento del procesador de cara a eventos externos llamados interrupciones.

Seguimiento de Interrupciones

Uno o varios bits ponen al procesador en modo paso a paso de forma que el programador puede examinar los efectos de las instrucciones de una en una a fin de depurar errores del programa.

Funciones Especiales

Uno o varios bits pueden controlar el funcionamiento de partes especiales del procesador, tales como la memoria caché.

Nivel de Privilegio

Uno o varios bits pueden controlar el modo de funcionamiento del procesador para que algunas instrucciones y recursos se utilicen a alto nivel de privilegio.

Esto permite el desarrollo de entornos multitarea en los cuales los sistemas operativos y los programas se protegen unos de otros.

En los procesadores con nivel de privilegio las operaciones de entrada/ salida, así como las instrucciones que modifican los bits de estatus del procesador tienen privilegio alto de forma que sólo se ejecutan cuando el nivel de privilegio es alto.

Tipos de Instrucciones

A continuación presentamos una clasificación de las instrucciones según su funcionalidad.

Se trata de instrucciones de movimiento de datos entre las diversas partes de la CPU:

- Registros.
- Memoria.
- Unidad aritmético-lógica.
- Controladores de entrada/salida.

Algunos procesadores tienen restringidos algunas de las transferencias, en otros casos se permiten todas las combinaciones (ortogonalidad).

En los movimientos de datos desde un origen a un destino se tienen instrucciones del tipo:

Todas estas instrucciones mueven el contenido de ***org*** a ***dst***.

Se utilizan prácticamente como sinónimos.

MOV	dst, org	(intel 8086)
LD	dst, org	(6809)
ST	dst, org	(6809)
MOVE	dst, org	(Motorola MC68000)

Dentro de este tipo de instrucciones, en algunos procesadores se sitúan las de poner y sacar un dato de la pila o stack. Se utiliza el símbolo SP (stack pointer) para el puntero de pila. Los nemotécnicos más utilizados son:

PUSH org Poner dato en la pila e incrementar el puntero de pila (SP).

POP dst Sacar dato de la pila y llevarlo al destino decrementando SP.

Algunos procesadores tienen una instrucción de intercambio EXCHANGE o EXG que intercambia el contenido de dos registros o bien de un registro y de una posición de memoria.

Es frecuente disponer de instrucciones que mueven bloques de información por ejemplo load múltiple (LDM) y store multiple (STM) son mnemotécnicos correspondientes a instrucciones que permiten salvar un grupo de registros en la memoria y cargar registros desde la memoria respectivamente, en una sola instrucción.

Otras instrucciones permiten mover bloques enteros de memoria de una posición a otra.

Un mnemotécnico común para esto es MOVEM.

Aritméticas

Se realizan instrucciones aritméticas, como adición, sustracción, etc.,

Tanto en coma fija como flotante (dependiendo de la potencia del procesador).

A veces hay también instrucciones especiales de comparación que se suelen utilizar junto con las de salto condicional.

La operación aritmética más básica en un ordenador es la adición.

Ésta suele tener el formato siguiente:

ADD org, dst

Cuyo significado es dst = dst+org+0.

Esta operación suma dos operandos de *n* bits con un bit de acarreo inicial que vale 0.

Simultáneamente se establecen una serie de condiciones o flags como la condición de bit de acarreo a su valor correspondiente.

También se establecen otras condiciones como la de número negativo o números sin signo.

La operación se aplica tanto a operandos codificados con complemento a dos, así como para operandos sin signo.

El resultado se guarda en el operando ***dst***.

La mayoría de los procesadores tienen un mnemotécnico específico para la substracción.

Éste suele ser:

SUB org, dst

Cuyo significado es dst = dst – org – 0.

El 0 último es el bit de préstamo que inicialmente se supone a cero.

Se utiliza la substracción con complemento a dos.

Es decir el operando org se complenta a dos y se suma a dst. Se tiene en cuenta el bit de préstamo que es el complemento del bit de acarreo.

La siguiente tabla muestra algunos ejemplos:

Operación	N Z V C	Interpretación con signo
00000100		+4
+00000010		+2
————————		
00000110	0 0 0 0	

Algunas Instrucciones de un Solo Operando

Como añadir o sustraer 1 a un dato es una operación muy frecuente por ejemplo para actualizar contadores y moverse por tablas, la mayoría de los procesadores tienen instrucciones para realizar esto.

Por ejemplo:

INC dst hace dst = dst+1

DEC dst hace dst = dst–1

Normalmente estas operaciones se aplican a registros, pero algunos procesadores también permiten aplicarlas a posiciones de memoria.

En algunos procesadores hay una diferencia sutil entre:

- ADD dst, #1
- INC dst

ADD afecta al bit de acareo.

INC no lo hace.

Otras instrucciones de un solo operando:

CLR dst hace dst = 0

COM dst hace dst = 2**b-1-dst

NEG dst hace dst = ((2**b-1-dst)+1) mod 2**b

Donde *b* es la longitud en bits del operando y 2**b es 2 elevado a la b .

mod es la operación módulo.

CLEAR pone todos los bits de dst a 0.

COM hace el complemento a 1 de los bits de dst, es decir cambia los 1 por 0 y viceversa.

NEG hace el complemento a dos, es decir complementa los bits y suma 1.

Los bits de estatus son modificados consecuentemente.

Lógicas

Se implementan las operaciones de la lógica booleana.

Operaciones de unión e intersección lógicas, negación, or exclusivo (OR, AND, NOT, XOR), comparaciones, así como desplazamientos y rotaciones de bits a izquierda y derecha.

Estas operaciones tratan una palabra de datos como una cadena de bits sobre los cuales se efectúan operaciones lógicas.

Cada bit se maneja de forma independiente.

La instrucción NOT complementa los bits del operando.

Las instrucciones más utilizadas son:

- AND.
- OR.
- XOR (EOR).

Su funcionamiento es el siguiente:

AND dst, org dst = AND(dst, org)

OR dst, org dst = OR(dst, org)

XOR dst, org dst = XOR(dst,org)

Con estas instrucciones se puede manipular un bit individual de una palabra utilizando lo que se denomina una máscara.

Por ejemplo la instrucción:

AND 7FFF, dst

Deja todos los bits de dst inmutados excepto el primero que lo pone a cero.

Manejo de bits

Permiten asignar valores a los bits de un operando de forma individual, es decir uno a uno.

También permiten consultar los valores que tiene cada bit de un operando.

Veamos por ejemplo las siguientes instrucciones las cuales operan sobre un bit individual de dst.

Este bit está indicado por ***bnum***. La última instrucción BTST actúa sobre el bit de condición Z.

BCLR ***bnum***, dst dst(***bnum***) = 0

BCSET ***bnum***, dst dst(***bnum***) = 1

BCHG ***bnum***, dst dst(***bnum***) = Not(dst(***bnum***))

BTST ***bnum***, dst Z = Not(dst(***bnum***))

Desplazamientos y Rotaciones

Mueven los bits de una palabra o doble palabra una o más posiciones a la izquierda y derecha.

Todos los procesadores disponen de estas operaciones para los registros y algunos las tienen también para las posiciones de memoria.

La figura 11.10 ilustra cómo se producen los desplazamientos.

Vemos que el contenido de un bit determinado se copia en el contiguo (ya sea a izquierda o a derecha).

Los bits de los extremos son especiales y son copiados a o desde (según su caso) otros bits independientes de la palabra que está sufriendo el desplazamiento, los cuales hemos denominado E y S respectivamente.

FIGURA 11.10. Desplazamientos a derecha e izquierda.

Mnemotécnicos muy usuales para estas operaciones son:

LSL dst Desplazamiento lógico a izquierda.

LSR dst Desplazamiento lógico a derecha.

El nombre de desplazamiento lógico se reserva normalmente para aquellos desplazamientos en los que la primera posición que se desplaza o bit vacante se pone a 0.

El bit sobrante se suele guardar en el bit de condición de acarreo C.

El anterior contenido de C se pierde.

Alternativamente a los desplazamientos lógicos se tienen los desplazamientos aritméticos.

Sus mnemotécnicos más habituales son:

ASL dst Desplazamiento aritmético a izquierda.

ASR dst Desplazamiento aritmético a derecha.

Los desplazamientos aritméticos tratan el operando como con bit de signo y en complemento a dos.

Operan de tal forma que un desplazamiento a derechas es equivalente a dividir por dos el operando y un desplazamiento a izquierdas es equivalente a multiplicar el operando por dos.

Además otra diferencia entre los desplazamientos lógicos y los aritméticos se refiere a los bits de condición.

Un desplazamiento lógico suele borrar el bit de desbordamiento V (es decir lo pone a 0).

En un desplazamiento aritmético el bit V se interpreta, es decir si se ha producido un cambio de signo en el operando V se pone a 1 y en caso contrario se pone a 0.

Se pueden desplazar los bits varias posiciones a derecha o izquierda con una sola operación.

El mnemotécnico utilizado es:

LSL #n, dst Desplazamiento lógico *n* posiciones a izquierda.

LSR #n, dst Desplazamiento lógico *n* posiciones a derecha.

ASL #n, dst Desplazamiento aritmético *n* posiciones a izquierda.

ASR #n, dst Desplazamiento aritmético *n* posiciones a derecha.

Las rotaciones son iguales que los desplazamientos, pero el último bit desplazado se guarda en el bit vacante.

Las rotaciones también se denominan desplazamientos circulares o cíclicos

La siguiente figura ilustra el concepto:

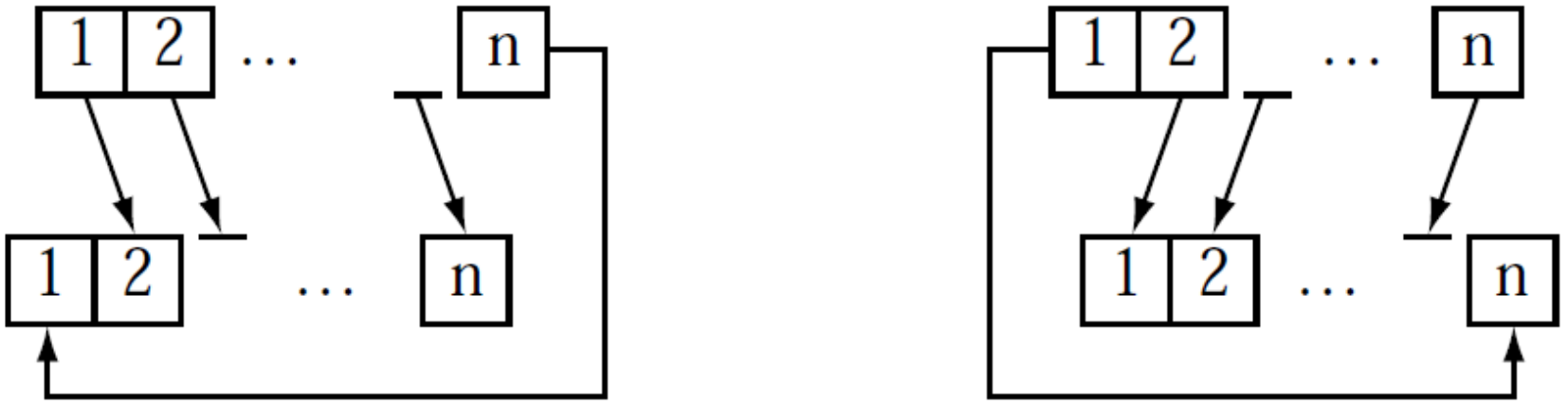

FIGURA 11.11. Rotaciones a derecha e izquierda.

Mnemotécnicos habituales para estas instrucciones son:

ROL dst Rotación a izquierda

ROR dst Rotación a derecha

Normalmente el bit que se realimenta a la posición vacante, se guarda también en el bit de acarreo C.

Muchos procesadores tienen instrucciones que permiten rotar los bits a izquierda o derecha varias posiciones.

La operación de rotar un registro o el contenido de una dirección de memoria n bits es equivalente a la de rotar dicho registro o posición de memoria *n* veces un bit.

Mnemotécnicos utilizados para estas rotaciones múltiples son:

ROL #n, dst Rotar dst n bits a izquierdas.

ROR #n, dst Rotar dst n bits a derechas.

Control de Flujo

Se trata de instrucciones que permiten realizar saltos tanto condicionales (si una condición se verifica) como incondicionales.

Se incluyen además los saltos con retorno los cuales permiten la ejecución de subrutinas.

Se incluyen desviaciones al sistema operativo.

También se pueden formar bucles de instrucciones que se ejecutan cíclicamente, ya sea un número fijo de veces o bien hasta que una condición se cumple.

La instrucción de control de flujo más simple es la de salto incondicional (jump JMP).

Su forma es:

JMP dst pone en el contador de programa la dirección dst

La instrucción de salto a subrutina (JSR), permite realizar llamadas a subrutinas.

Normalmente esta instrucción guarda la dirección de retorno de la subrutina en la pila antes de saltar a la subrutina.

La sintaxis es:

JSR	dst	pone en la pila el contador de programa conteniendo la dirección de memoria de la instrucción actual y pone en el contador de programa dst

Cuando la subrutina termina ésta ejecuta la instrucción return (RTS) que saca la dirección de retorno del tope de la pila (donde la había dejado JMP y la carga en el contador de programa.

Las instrucciones de salto condicional chequean una determinada condición y saltan si la condición es satisfecha.

Existen también instrucciones de salto condicional a posiciones próximas a la instrucción de salto.

En este caso en el formato de la instrucción sólo se requiere un byte para determinar la posición de memoria de destino.

El salto se hace relativo al contenido del contador de programa.

Estas instrucciones tienen la ventaja de que ocupan poco debido a lo corto del salto.

FIGURA 11.12. Formato de instrucción de bifurcación corta.

La siguiente tabla contiene ejemplos de instrucciones de bifurcación condicional en el Motorola 68000

Tipo	Mnemo.	cc	cccc	Bifurcar si	Condición
Incondicional 1 bit	BRA	T	0000	Siempre	Verdad (T)
	BCC	CC	0100	Acarreo=0	C=0
	BCS	CS	0101	Acarreo=1	C=1
	BNE	NE	0110	No igual a 0	Z=0
	BEQ	EQ	0111	Igual a 0	Z=1
	BVC	VC	1000	No overflow	V=0
	BVS	VS	1001	Si overflow	V=1
	BPL	PL	1010	Mas	N=0
	BMI	MI	1011	Menos	N=1
Con signo					
	BGE	GE	1100	Mayor o igual 0	$(N \oplus V)=0$
	BLT	LT	1101	Menor que 0	$(N \oplus V)=1$
	BGT	GT	1110	Mayor que 0	$(N \oplus V)+Z=0$
	BLE	LE	1111	Menor o igual 0	$(N \oplus V)+Z=1$
Sin signo					
	BHI	HI	0010	Más alto	(C+Z)=0
	BLS	LS	0011	Menor o el mismo	(C+Z)=1
	BHS	HS	0100	Mayor o el mismo	C=0
	BLO	LO	0101	Menor	

Una construcción típica es:

CMP	x, y	se comparan x e y. El resultado modifica los bits de estado
BHI	LABEL	se salta si x>y tomados como sin signo

Una forma de realizar bucles es la siguiente:

LOOP		… … …	REPETIR …
	SUB	#1, CNT	cnt = cnt–1;
	BNE	LOOP	HASTA QUE cnt = 0;

Vemos que la última instrucción envía el flujo de ejecución a la etiqueta LOOP hasta que el contador CNT sea cero.

Una implementación de bucles más sencilla y que se ejecuta más rápidamente es:

LOOP		… … …	REPETIR …
	DJNZ	CNT, LOOP	cnt = cnt–1;
	BNE	LOOP	HASTA QUE cnt = 0;

La instrucción DJNZ es jump if not zero.

Esta construcción tiene la ventaja de que no altera los códigos de condición por lo que éstos quedan libres para ser utilizados por el resto de las instrucciones del bucle.

Instrucciones de control de flujo:

Mnemónico	Operandos	Descripción
JMP	cdst	Salto a cdst
JSR	cdst	Salto a subrutina en cdst
RTS		Retorno de subrutina
RTR		Sacar CCR y retornar
BRA	dir16	Bifurca a dir16
BSR	dir16	Bifurca a subrutina en dir16
Bcc	dir16	Bifurca a dir16 si cc es verdadero
DBcc	Dn, dir16	Bucle condicional
Scc	dst	Pone dst según cc
LINK	An, #disp	Enlaza subrutina
UNLK	An	Limpia el enlace de la subrutina

La instrucción BRA provoca un salto incondicional a posiciones próximas utilizando un tipo especial de direccionamiento relativo.

La instrucción BSR es similar pero realiza un salto a una subrutina por lo que primero guarda el contador de programa en la pila (para que luego se pueda retornar al punto de salto) y después pone en el contador de programa la dirección de salto.

Como en una instrucción de bifurcación corta, el operando es de longitud byte, éste especifica una dirección dentro del rango –128 a –2 o bien +2 a +126 bytes de la palabra siguiendo la instrucción.

En las instrucciones de salto largo se especifica una palabra de 16 bits por lo que el rango es –32768 a +32768 bytes desde la posición del salto.

Control del Procesador y Misceláneas

Se incluyen instrucciones especiales que son capaces de parar el procesador, instrucciones de espera, consultas y manipulación en general de registros de estado.

Algunas de estas operaciones son:

NOP

Esta instrucción no hace nada.

A veces se utiliza para insertar un retraso en el programa, quizás para realizar alguna espera, por ejemplo para entrada/salida.

TAS

Esta operación significa test and set.

Lee un operando de un byte y pone los bits de condición de acuerdo con su valor.

A continuación almacena el operando con su bit más significativo a 1.

Esta instrucción es muy importante en los sistemas multiprocesador en los que varios procesadores diferentes trabajan al unísono ejecutando en paralelo partes diferentes de un programa y requiriendo mecanismos de sincronización entre procesadores.

TRAP

Es como una llamada a una subrutina pero con algunas diferencias importantes.

Sirve para que el programa de usuario entre en modo supervisor y se ejecuten subrutinas del sistema operativo en mod privilegiado.

STOP

Esta instrucción detiene el procesador.

RESET

Esta instrucción inicializa el procesador.