second $\overline{\text{INTA}}$, the external interrupt system (e.g., an Intel® 8259A Programmable Interrupt Controller) places a byte on the data bus that identifies the source of the interrupt (the vector number or vector "type"). This byte is read by the CPU and then multiplied by four with the resultant value used as a pointer into the interrupt vector table. Before calling the corresponding interrupt routine, the CPU saves the machine status by pushing the current contents of the flags register onto the stack. The CPU then clears the interrupt enable and trap bits in the flags register to prevent subsequent maskable and single-step interrupts, and establishes the interrupt routine return linkage by pushing the current CS and IP register contents onto the stack before loading the new CS and IP register values from the vector table.

The four classes of interrupts are prioritized with software-initiated interrupts having the highest priority and with maskable and single-step interrupts sharing the lowest priority (see section 2.6). Since the CPU disables maskable and single-step interrupts when acknowledging any interrupt, if recognition of maskable interrupts or single-step operation is required as part of the interrupt routine, the routine first must set these bits.

The processing times for the various classes of interrupts are given in table 4-6. (These times also are included with the 8086/8088 instruction times cited in section 2.7.)

Table 4-6. Interrupt Processing Time

Interrupt Class	Processing Time
External Maskable Interrupt (INTR)	61 clocks
Non-Maskable Interrupt (NMI)	50 clocks
INT (with vector)	51 clocks
INT Type 3	52 clocks
INTO	53 clocks
Single Step	50 clocks

Note that the times shown in table 4-6 represent only the time required to process the interrupt request after it has been recognized. To determine interrupt latency (the time interval between the posting of the interrupt request and the execution of "useful" instructions within the interrupt routine), additional time must be included for the completion on an instruction being executed when the interrupt is posted (interrupts are generally processed only at instruction boundaries), for saving the contents of any additional registers prior to interrupt processing (interrupts automatically save only CS, IP and Flags) and for any wait states that may be incurred during interrupt processing.

Machine Instruction Encoding and Decoding

Writing a MOV instruction in ASM-86 in the form:

```
MOV destination,source
```

will cause the assembler to generate 1 of 28 possible forms of the MOV machine instruction. A programmer rarely needs to know the details of machine instruction formats or encoding. An exception may occur during debugging when it may be necessary to monitor instructions fetched on the bus, read unformatted memory dumps, etc. This section provides the information necessary to translate or decode an 8086 or 8088 machine instruction.

To pack instructions into memory as densely as possible, the 8086 and 8088 CPUs utilize an efficient coding technique. Machine instructions vary from one to six bytes in length. One-byte instructions, which generally operate on single registers or flags, are simple to identify. The keys to decoding longer instructions are in the first two bytes. The format of these bytes can vary, but most instructions follow the format shown in figure 4-20.

The first six bits of a multibyte instruction generally contain an opcode that identifies the basic instruction type: ADD, XOR, etc. The following bit, called the D field, generally specifies the "direction" of the operation: 1 = the REG field in the second byte identifies the destination operand, 0 = the REG field identifies the source operand. The W field distinguishes between byte and word operations: 0 = byte, 1 = word.

One of three additional single-bit fields, S, V or Z, appears in some instruction formats. S is used in conjunction with W to indicate sign extension of immediate fields in arithmetic instructions. V distinguishes between single- and variable-bit shifts and rotates. Z is used as a compare bit with the zero flag in conditional repeat and loop instructions. All single-bit field settings are summarized in table 4-7.

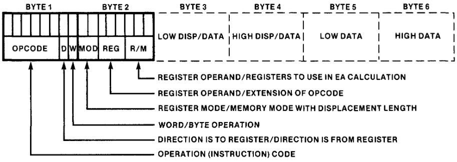
BYTE 1
BYTE 2
BYTE 3
BYTE 4
BYTE 5
BYTE 6
OPCODE
D
W
MOD
REG
R/M
LOW DISP/DATA
HIGH DISP/DATA
LOW DATA
HIGH DATA
REGISTER OPERAND/REGISTERS TO USE IN EA CALCULATION
REGISTER OPERAND/EXTENSION OF OPCODE
REGISTER MODE/MEMORY MODE WITH DISPLACEMENT LENGTH
WORD/BYTE OPERATION
DIRECTION IS TO REGISTER/DIRECTION IS FROM REGISTER
OPERATION (INSTRUCTION) CODE

Figure 4-20. Typical 8086/8088 Machine Instruction Format

Table 4-7. Single-Bit Field Encoding

Field	Value	Function
S	0	No sign extension
	1	Sign extend 8-bit immediate data to 16 bits if W=1
W	0	Instruction operates on byte data
	1	Instruction operates on word data
D	0	Instruction source is specified in REG field
	1	Instruction destination is specified in REG field
V	0	Shift/rotate count is one
	1	Shift/rotate count is specified in CL register
Z	0	Repeat/loop while zero flag is clear
	1	Repeat/loop while zero flag is set

The second byte of the instruction usually identifies the instruction's operands. The MOD (mode) field indicates whether one of the operands is in memory or whether both operands are registers (see table 4-8). The REG (register) field identifies a register that is one of the instruction operands (see table 4-9). In a number of instructions, chiefly the immediate-to-memory variety, REG is used as an extension of the opcode to identify the type of operation. The encoding of the R/M (register/memory) field (see table 4-10) depends on how the mode field is set. If MOD = 11 (register-to-register mode), then R/M identifies the second register operand. If MOD selects memory mode, then R/M indicates how the effective address of the memory operand is to be calculated. Effective address calculation is covered in detail in section 2.8.

Bytes 3 through 6 of an instruction are optional fields that usually contain the displacement value of a memory operand and/or the actual value of an immediate constant operand.

Table 4-8. MOD (Mode) Field Encoding

CODE	EXPLANATION
00	Memory Mode, no displacement follows*
01	Memory Mode, 8-bit displacement follows
10	Memory Mode, 16-bit displacement follows
11	Register Mode (no displacement)

*Except when R/M = 110, then 16-bit displacement follows

Table 4-9. REG (Register) Field Encoding

REG	W=0	W=1
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

There may be one or two displacement bytes; the language translators generate one byte whenever possible. The MOD field indicates how many displacement bytes are present. Following Intel convention, if the displacement is two bytes, the most-significant byte is stored second in the instruction. If the displacement is only a single byte, the 8086 or 8088 automatically sign-extends this quantity to 16-bits before using the information in further address calculations. Immediate values always follow any displacement values that may be present. The second byte of a two-byte immediate value is the most significant.

Table 4-12 lists the instruction encodings for all 8086/8088 instructions. This table can be used to predict the machine encoding of any ASM-86 instruction. Table 4-13 lists the 8086/8088 machine instructions in order by the binary value of their first byte. This table can be used to decode any machine instruction from its binary representation. Table 4-11 is a key to the abbreviations used in tables 4-12 and 4-13. Table 4-14 is a more compact instruction decoding guide.

Table 4-10. R/M (Register/Memory) Field Encoding

MOD=11			EFFECTIVE ADDRESS CALCULATION			
R/M	W=0	W=1	R/M	MOD=00	MOD=01	MOD=10
000	AL	AX	000	(BX)+(SI)	(BX)+(SI)+D8	(BX)+(SI)+D16
001	CL	CX	001	(BX)+(DI)	(BX)+(DI)+D8	(BX)+(DI)+D16
010	DL	DX	010	(BP)+(SI)	(BP)+(SI)+D8	(BP)+(SI)+D16
011	BL	BX	011	(BP)+(DI)	(BP)+(DI)+D8	(BP)+(DI)+D16
100	AH	SP	100	(SI)	(SI)+D8	(SI)+D16
101	CH	BP	101	(DI)	(DI)+D8	(DI)+D16
110	DH	SI	110	DIRECT ADDRESS	(BP)+D8	(BP)+D16
111	BH	DI	111	(BX)	(BX)+D8	(BX)+D16

Table 4-11. Key to Machine Instruction Encoding and Decoding

IDENTIFIER	EXPLANATION
MOD	Mode field; described in this chapter.
REG	Register field; described in this chapter.
R/M	Register/Memory field; described in this chapter.
SR	Segment register code: 00=ES, 01=CS, 10=SS, 11=DS.
W, S, D, V, Z	Single-bit instruction fields; described in this chapter.
DATA-8	8-bit immediate constant.
DATA-SX	8-bit immediate value that is automatically sign-extended to 16-bits before use.
DATA-LO	Low-order byte of 16-bit immediate constant.
DATA-HI	High-order byte of 16-bit immediate constant.
(DISP-LO)	Low-order byte of optional 8- or 16-bit unsigned displacement; MOD indicates if present.
(DISP-HI)	High-order byte of optional 16-bit unsigned displacement; MOD indicates if present.
IP-LO	Low-order byte of new IP value.
IP-HI	High-order byte of new IP value
CS-LO	Low-order byte of new CS value.
CS-HI	High-order byte of new CS value.
IP-INC8	8-bit signed increment to instruction pointer.
IP-INC-LO	Low-order byte of signed 16-bit instruction pointer increment.
IP-INC-HI	High-order byte of signed 16-bit instruction pointer increment.
ADDR-LO	Low-order byte of direct address (offset) of memory operand; EA not calculated.
ADDR-HI	High-order byte of direct address (offset) of memory operand; EA not calculated.
——	Bits may contain any value.
XXX	First 3 bits of ESC opcode.
YYY	Second 3 bits of ESC opcode.
REG8	8-bit general register operand.
REG16	16-bit general register operand.
MEM8	8-bit memory operand (any addressing mode).
MEM16	16-bit memory operand (any addressing mode).
IMMED8	8-bit immediate operand.
IMMED16	16-bit immediate operand.
SEGREG	Segment register operand.
DEST-STR8	Byte string addressed by DI.

Table 4-11. Key to Machine Instruction Encoding and Decoding (Cont'd.)

IDENTIFIER	EXPLANATION
SRC-STR8	Byte string addressed by SI.
DEST-STR16	Word string addressed by DI.
SRC-STR16	Word string addressed by SI.
SHORT-LABEL	Label within ±127 bytes of instruction.
NEAR-PROC	Procedure in current code segment.
FAR-PROC	Procedure in another code segment.
NEAR-LABEL	Label in current code segment but farther than −128 to +127 bytes from instruction.
FAR-LABEL	Label in another code segment.
SOURCE-TABLE	XLAT translation table addressed by BX.
OPCODE	ESC opcode operand.
SOURCE	ESC register or memory operand.

Table 4-12. 8086 Instruction Encoding

DATA TRANSFER

MOV = Move:	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
Register/memory to/from register	1 0 0 0 1 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate to register/memory	1 1 0 0 0 1 1 w	mod 0 0 0 r/m	(DISP-LO)	(DISP-HI)	data	data if w = 1
Immediate to register	1 0 1 1 w reg	data	data if w = 1			
Memory to accumulator	1 0 1 0 0 0 0 w	addr-lo	addr-hi			
Accumulator to memory	1 0 1 0 0 0 1 w	addr-lo	addr-hi			
Register/memory to segment register	1 0 0 0 1 1 1 0	mod 0 SR r/m	(DISP-LO)	(DISP-HI)		
Segment register to register/memory	1 0 0 0 1 1 0 0	mod 0 SR r/m	(DISP-LO)	(DISP-HI)		

PUSH = Push:

Register/memory	1 1 1 1 1 1 1 1	mod 1 1 0 r/m	(DISP-LO)	(DISP-HI)
Register	0 1 0 1 0 reg			
Segment register	0 0 0 reg 1 1 0			

POP = Pop:

Register/memory	1 0 0 0 1 1 1 1	mod 0 0 0 r/m	(DISP-LO)	(DISP-HI)
Register	0 1 0 1 1 reg			
Segment register	0 0 0 reg 1 1 1			

Table 4-12. 8086 Instruction Encoding (Cont'd.)

	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
DATA TRANSFER (Cont'd.)						
XCHG = Exchange:						
Register/memory with register	1 0 0 0 0 1 1 w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Register with accumulator	1 0 0 1 0 reg					
IN = Input from:						
Fixed port	1 1 1 0 0 1 0 w	DATA-8				
Variable port	1 1 1 0 1 1 0 w					
OUT = Output to:						
Fixed port	1 1 1 0 0 1 1 w	DATA-8				
Variable port	1 1 1 0 1 1 1 w					
XLAT = Translate byte to AL	1 1 0 1 0 1 1 1					
LEA = Load EA to register	1 0 0 0 1 1 0 1	mod reg r/m	(DISP-LO)	(DISP-HI)		
LDS = Load pointer to DS	1 1 0 0 0 1 0 1	mod reg r/m	(DISP-LO)	(DISP-HI)		
LES = Load pointer to ES	1 1 0 0 0 1 0 0	mod reg r/m	(DISP-LO)	(DISP-HI)		
LAHF = Load AH with flags	1 0 0 1 1 1 1 1					
SAHF = Store AH into flags	1 0 0 1 1 1 1 0					
PUSHF = Push flags	1 0 0 1 1 1 0 0					
POPF = Pop flags	1 0 0 1 1 1 0 1					
ARITHMETIC						
ADD = Add:						
Reg/memory with register to either	0 0 0 0 0 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate to register/memory	1 0 0 0 0 0 s w	mod 0 0 0 r/m	(DISP-LO)	(DISP-HI)	data	data if s: w=01
Immediate to accumulator	0 0 0 0 0 1 0 w	data	data if w=1			
ADC = Add with carry:						
Reg/memory with register to either	0 0 0 1 0 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate to register/memory	1 0 0 0 0 0 s w	mod 0 1 0 r/m	(DISP-LO)	(DISP-HI)	data	data if s: w=01
Immediate to accumulator	0 0 0 1 0 1 0 w	data	data if w=1			
INC = Increment:						
Register/memory	1 1 1 1 1 1 1 w	mod 0 0 0 r/m	(DISP-LO)	(DISP-HI)		
Register	0 1 0 0 0 reg					
AAA = ASCII adjust for add	0 0 1 1 0 1 1 1					
DAA = Decimal adjust for add	0 0 1 0 0 1 1 1					

Table 4-12. 8086 Instruction Encoding (Cont'd.)

ARITHMETIC (Cont'd.)

SUB = Subtract:	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
Reg/memory and register to either	0 0 1 0 1 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate from register/memory	1 0 0 0 0 0 s w	mod 1 0 1 r/m	(DISP-LO)	(DISP-HI)	data	data if s: w=01
Immediate from accumulator	0 0 1 0 1 1 0 w	data	data if w=1			

SBB = Subtract with borrow:						
Reg/memory and register to either	0 0 0 1 1 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate from register/memory	1 0 0 0 0 0 s w	mod 0 1 1 r/m	(DISP-LO)	(DISP-HI)	data	data if s: w=01
Immediate from accumulator	0 0 0 1 1 1 0 w	data	data if w=1			

DEC Decrement:				
Register/memory	1 1 1 1 1 1 1 w	mod 0 0 1 r/m	(DISP-LO)	(DISP-HI)
Register	0 1 0 0 1 reg			
NEG Change sign	1 1 1 1 0 1 1 w	mod 0 1 1 r/m	(DISP-LO)	(DISP-HI)

CMP = Compare:						
Register/memory and register	0 0 1 1 1 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate with register/memory	1 0 0 0 0 0 s w	mod 1 1 1 r/m	(DISP-LO)	(DISP-HI)	data	data if s: w=1
Immediate with accumulator	0 0 1 1 1 1 0 w	data				
AAS ASCII adjust for subtract	0 0 1 1 1 1 1 1					
DAS Decimal adjust for subtract	0 0 1 0 1 1 1 1					
MUL Multiply (unsigned)	1 1 1 1 0 1 1 w	mod 1 0 0 r/m	(DISP-LO)	(DISP-HI)		
IMUL Integer multiply (signed)	1 1 1 1 0 1 1 w	mod 1 0 1 r/m	(DISP-LO)	(DISP-HI)		
AAM ASCII adjust for multiply	1 1 0 1 0 1 0 0	0 0 0 0 1 0 1 0	(DISP-LO)	(DISP-HI)		
DIV Divide (unsigned)	1 1 1 1 0 1 1 w	mod 1 1 0 r/m	(DISP-LO)	(DISP-HI)		
IDIV Integer divide (signed)	1 1 1 1 0 1 1 w	mod 1 1 1 r/m	(DISP-LO)	(DISP-HI)		
AAD ASCII adjust for divide	1 1 0 1 0 1 0 1	0 0 0 0 1 0 1 0	(DISP-LO)	(DISP-HI)		
CBW Convert byte to word	1 0 0 1 1 0 0 0					
CWD Convert word to double word	1 0 0 1 1 0 0 1					

LOGIC

NOT Invert	1 1 1 1 0 1 1 w	mod 0 1 0 r/m	(DISP-LO)	(DISP-HI)
SHL/SAL Shift logical/arithmetic left	1 1 0 1 0 0 v w	mod 1 0 0 r/m	(DISP-LO)	(DISP-HI)
SHR Shift logical right	1 1 0 1 0 0 v w	mod 1 0 1 r/m	(DISP-LO)	(DISP-HI)
SAR Shift arithmetic right	1 1 0 1 0 0 v w	mod 1 1 1 r/m	(DISP-LO)	(DISP-HI)
ROL Rotate left	1 1 0 1 0 0 v w	mod 0 0 0 r/m	(DISP-LO)	(DISP-HI)

Table 4-12. 8086 Instruction Encoding (Cont'd.)

LOGIC (Cont'd.)	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
ROR Rotate right	1 1 0 1 0 0 v w	mod 0 0 1 r/m	(DISP-LO)	(DISP-HI)		
RCL Rotate through carry flag left	1 1 0 1 0 0 v w	mod 0 1 0 r/m	(DISP-LO)	(DISP-HI)		
RCR Rotate through carry right	1 1 0 1 0 0 v w	mod 0 1 1 r/m	(DISP-LO)	(DISP-HI)		
AND = And:						
Reg/memory with register to either	0 0 1 0 0 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate to register/memory	1 0 0 0 0 0 0 w	mod 1 0 0 r/m	(DISP-LO)	(DISP-HI)	data	data if w=1
Immediate to accumulator	0 0 1 0 0 1 0 w	data	data if w=1			
TEST = And function to flags no result:						
Register/memory and register	0 0 0 1 0 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate data and register/memory	1 1 1 1 0 1 1 w	mod 0 0 0 r/m	(DISP-LO)	(DISP-HI)	data	data if w=1
Immediate data and accumulator	1 0 1 0 1 0 0 w	data				
OR = Or:						
Reg/memory and register to either	0 0 0 0 1 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate to register/memory	1 0 0 0 0 0 0 w	mod 0 0 1 r/m	(DISP-LO)	(DISP-HI)	data	data if w=1
Immediate to accumulator	0 0 0 0 1 1 0 w	data	data if w=1			
XOR = Exclusive or:						
Reg/memory and register to either	0 0 1 1 0 0 d w	mod reg r/m	(DISP-LO)	(DISP-HI)		
Immediate to register/memory	0 0 1 1 0 1 0 w	data	(DISP-LO)	(DISP-HI)	data	data if w=1
Immediate to accumulator	0 0 1 1 0 1 0 w	data	data if w=1			
STRING MANIPULATION						
REP = Repeat	1 1 1 1 0 0 1 z					
MOVS = Move byte/word	1 0 1 0 0 1 0 w					
CMPS = Compare byte/word	1 0 1 0 0 1 1 w					
SCAS = Scan byte/word	1 0 1 0 1 1 1 w					
LODS = Load byte/wd to AL/AX	1 0 1 0 1 1 0 w					
STDS = Stor byte/wd from AL/A	1 0 1 0 1 0 1 w					

Table 4-12. 8086 Instruction Encoding (Cont'd.)

CONTROL TRANSFER

CALL = Call:	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
Direct within segment	1 1 1 0 1 0 0 0	IP-INC-LO	IP-INC-HI			
Indirect within segment	1 1 1 1 1 1 1 1	mod 0 1 0 r/m	(DISP-LO)	(DISP-HI)		
Direct intersegment	1 0 0 1 1 0 1 0	IP-lo	IP-hi			
		CS-lo	CS-hi			
Indirect intersegment	1 1 1 1 1 1 1 1	mod 0 1 1 r/m	(DISP-LO)	(DISP-HI)		

JMP = Unconditional Jump:				
Direct within segment	1 1 1 0 1 0 0 1	IP-INC-LO	IP-INC-HI	
Direct within segment-short	1 1 1 0 1 0 1 1	IP-INC8		
Indirect within segment	1 1 1 1 1 1 1 1	mod 1 0 0 r/m	(DISP-LO)	(DISP-HI)
Direct intersegment	1 1 1 0 1 0 1 0	IP-lo	IP-hi	
		CS-lo	CS-hi	
Indirect intersegment	1 1 1 1 1 1 1 1	mod 1 0 1 r/m	(DISP-LO)	(DISP-HI)

RET = Return from CALL:			
Within segment	1 1 0 0 0 0 1 1		
Within seg adding immed to SP	1 1 0 0 0 0 1 0	data-lo	data-hi
Intersegment	1 1 0 0 1 0 1 1		
Intersegment adding immediate to SP	1 1 0 0 1 0 1 0	data-lo	data-hi
JE/JZ = Jump on equal/zero	0 1 1 1 0 1 0 0	IP-INC8	
JL/JNGE = Jump on less/not greater or equal	0 1 1 1 1 1 0 0	IP-INC8	
JLE/JNG = Jump on less or equal/not greater	0 1 1 1 1 1 1 0	IP-INC8	
JB/JNAE = Jump on below/not above or equal	0 1 1 1 0 0 1 0	IP-INC8	
JBE/JNA = Jump on below or equal/not above	0 1 1 1 0 1 1 0	IP-INC8	
JP/JPE = Jump on parity/parity even	0 1 1 1 1 0 1 0	IP-INC8	
JO = Jump on overflow	0 1 1 1 0 0 0 0	IP-INC8	
JS = Jump on sign	0 1 1 1 1 0 0 0	IP-INC8	
JNE/JNZ = Jump on not equal/not zer0	0 1 1 1 0 1 0 1	IP-INC8	
JNL/JGE = Jump on not less/greater or equal	0 1 1 1 1 1 0 1	IP-INC8	
JNLE/JG = Jump on not less or equal/greater	0 1 1 1 1 1 1 1	IP-INC8	
JNB/JAE = Jump on not below/above or equal	0 1 1 1 0 0 1 1	IP-INC8	
JNBE/JA = Jump on not below or equal/above	0 1 1 1 0 1 1 1	IP-INC8	
JNP/JPO = Jump on not par/par odd	0 1 1 1 1 0 1 1	IP-INC8	
JNO = Jump on not overflow	0 1 1 1 0 0 0 1	IP-INC8	

Table 4-12. 8086 Instruction Encoding (Cont'd.)

CONTROL TRANSFER (Cont'd.)

RET = Return from CALL:	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
JNS = Jump on not sign	0 1 1 1 1 0 0 1	IP-INC8				
LOOP = Loop CX times	1 1 1 0 0 0 1 0	IP-INC8				
LOOPZ/LOOPE = Loop while zero/equal	1 1 1 0 0 0 0 1	IP-INC8				
LOOPNZ/LOOPNE = Loop while not zero/equal	1 1 1 0 0 0 0 0	IP-INC8				
JCXZ = Jump on CX zero	1 1 1 0 0 0 1 1	IP-INC8				
INT = Interrupt:						
Type specified	1 1 0 0 1 1 0 1	DATA-8				
Type 3	1 1 0 0 1 1 0 0					
INTO = Interrupt on overflow	1 1 0 0 1 1 1 0					
IRET = Interrupt return	1 1 0 0 1 1 1 1					
PROCESSOR CONTROL						
CLC = Clear carry	1 1 1 1 1 0 0 0					
CMC = Complement carry	1 1 1 1 0 1 0 1					
STC = Set carry	1 1 1 1 1 0 0 1					
CLD = Clear direction	1 1 1 1 1 1 0 0					
STD = Set direction	1 1 1 1 1 1 0 1					
CLI = Clear interrupt	1 1 1 1 1 0 1 0					
STI = Set interrupt	1 1 1 1 1 0 1 1					
HLT = Halt	1 1 1 1 0 1 0 0					
WAIT = Wait	1 0 0 1 1 0 1 1					
ESC = Escape (to external device)	1 1 0 1 1 x x x	m o d y y y r / m	(DISP-LO)	(DISP-HI)		
LOCK = Bus lock prefix	1 1 1 1 0 0 0 0					
SEGMENT = Override prefix	0 0 1 reg 1 1 0					

Table 4-13. Machine Instruction Decoding Guide

1ST BYTE		2ND BYTE	BYTES 3, 4, 5, 6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
00	0000 0000	MOD REG R/M	(DISP-LO),(DISP-HI)	ADD	REG8/MEM8,REG8
01	0000 0001	MOD REG R/M	(DISP-LO),(DISP-HI)	ADD	REG16/MEM16,REG16
02	0000 0010	MOD REG R/M	(DISP-LO),(DISP-HI)	ADD	REG8,REG8/MEM8
03	0000 0011	MOD REG R/M	(DISP-LO),(DISP-HI)	ADD	REG16,REG16/MEM16
04	0000 0100	DATA-8		ADD	AL,IMMED8
05	0000 0101	DATA-LO	DATA-HI	ADD	AX,IMMED16
06	0000 0110			PUSH	ES
07	0000 0111			POP	ES

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
08	0000 1000	MOD REG R/M	(DISP-LO),(DISP-HI)	OR	REG8/MEM8,REG8
09	0000 1001	MOD REG R/M	(DISP-LO),(DISP-HI)	OR	REG16/MEM16,REG16
0A	0000 1010	MOD REG R/M	(DISP-LO),(DISP-HI)	OR	REG8,REG8/MEM8
0B	0000 1011	MOD REG R/M	(DISP-LO),(DISP-HI)	OR	REG16,REG16/MEM16
0C	0000 1100	DATA-8		OR	AL,IMMED8
0D	0000 1101	DATA-LO	DATA-HI	OR	AX,IMMED16
0E	0000 1110			PUSH	CS
0F	0000 1111			(not used)	
10	0001 0000	MOD REG R/M	(DISP-LO),(DISP-HI)	ADC	REG8/MEM8,REG8
11	0001 0001	MOD REG R/M	(DISP-LO),(DISP-HI)	ADC	REG16/MEM16,REG16
12	0001 0010	MOD REG R/M	(DISP-LO),(DISP-HI)	ADC	REG8,REG8/MEM8
13	0001 0011	MOD REG R/M	(DISP-LO),(DISP-HI)	ADC	REG16,REG16/MEM16
14	0001 0100	DATA-8		ADC	AL,IMMED8
15	0001 0101	DATA-LO	DATA-HI	ADC	AX,IMMED16
16	0001 0110			PUSH	SS
17	0001 0111			POP	SS
18	0001 1000	MOD REG R/M	(DISP-LO),(DISP-HI)	SBB	REG8/MEM8,REG8
19	0001 1001	MOD REG R/M	(DISP-LO),(DISP-HI)	SBB	REG16/MEM16,REG16
1A	0001 1010	MOD REG R/M	(DISP-LO),(DISP-HI)	SBB	REG8,REG8/MEM8
1B	0001 1011	MOD REG R/M	(DISP-LO),(DISP-HI)	SBB	REG16,REG16/MEM16
1C	0001 1100	DATA-8		SBB	AL,IMMED8
1D	0001 1101	DATA-LO	DATA-HI	SBB	AX,IMMED16
1E	0001 1110			PUSH	DS
1F	0001 1111			POP	DS
20	0010 0000	MOD REG R/M	(DISP-LO),(DISP-HI)	AND	REG8/MEM8,REG8
21	0010 0001	MOD REG R/M	(DISP-LO),(DISP-HI)	AND	REG16/MEM16,REG16
22	0010 0010	MOD REG R/M	(DISP-LO),(DISP-HI)	AND	REG8,REG8/MEM8
23	0010 0011	MOD REG R/M	(DISP-LO),(DISP-HI)	AND	REG16,REG16/MEM16
24	0010 0100	DATA-8		AND	AL,IMMED8
25	0010 0101	DATA-LO	DATA-HI	AND	AX,IMMED16
26	0010 0110			ES:	(segment override prefix)
27	0010 0111			DAA	
28	0010 1000	MOD REG R/M	(DISP-LO),(DISP-HI)	SUB	REG8/MEM8,REG8
29	0010 1001	MOD REG R/M	(DISP-LO),(DISP-HI)	SUB	REG16/MEM16,REG16
2A	0010 1010	MOD REG R/M	(DISP-LO),(DISP-HI)	SUB	REG8,REG8/MEM8
2B	0010 1011	MOD REG R/M	(DISP-LO,(DISP-HI)	SUB	REG16,REG16/MEM16
2C	0010 1100	DATA-8		SUB	AL,IMMED8
2D	0010 1101	DATA-LO	DATA-HI	SUB	AX,IMMED16
2E	0010 1110			CS:	(segment override prefix)
2F	0010 1111			DAS	
30	0011 0000	MOD REG R/M	(DISP-LO),(DISP-HI)	XOR	REG8/MEM8,REG8
31	0011 0001	MOD REG R/M	(DISP-LO),(DISP-HI)	XOR	REG16/MEM16,REG16
32	0011 0010	MOD REG R/M	(DISP-LO),(DISP-HI)	XOR	REG8,REG8/MEM8
33	0011 0011	MOD REG R/M	(DISP-LO),(DISP-HI)	XOR	REG16,REG16/MEM16
34	0011 0100	DATA-8		XOR	AL,IMMED8
35	0011 0101	DATA-LO	DATA-HI	XOR	AX,IMMED16
36	0011 0110			SS:	(segment override prefix)

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
37	0011 0110			AAA	
38	0011 1000	MOD REG R/M	(DISP-LO),(DISP-HI)	CMP	REG8/MEM8,REG8
39	0011 1001	MOD REG R/M	(DISP-LO),(DISP-HI)	CMP	REG16/MEM16,REG16
3A	0011 1010	MOD REG R/M	(DISP-LO),(DISP-HI)	CMP	REG8,REG8/MEM8
3B	0011 1011	MOD REG R/M	(DISP-LO),(DISP-HI)	CMP	REG16,REG16/MEM16
3C	0011 1100	DATA-8		CMP	AL,IMMED8
3D	0011 1101	DATA-LO	DATA-HI	CMP	AX,IMMED16
3E	0011 1110			DS:	(segment override prefix)
3F	0011 1111			AAS	
40	0100 0000			INC	AX
41	0100 0001			INC	CX
42	0100 0010			INC	DX
43	0100 0011			INC	BX
44	0100 0100			INC	SP
45	0100 0101			INC	BP
46	0100 0110			INC	SI
47	0100 0111			INC	DI
48	0100 1000			DEC	AX
49	0100 1001			DEC	CX
4A	0100 1010			DEC	DX
4B	0100 1011			DEC	BX
4C	0100 1100			DEC	SP
4D	0100 1101			DEC	BP
4E	0100 1110			DEC	SI
4F	0100 1111			DEC	DI
50	0101 0000			PUSH	AX
51	0101 0001			PUSH	CX
52	0101 0010			PUSH	DX
53	0101 0011			PUSH	BX
54	0101 0100			PUSH	SP
55	0101 0101			PUSH	BP
56	0101 0110			PUSH	SI
57	0101 0111			PUSH	DI
58	0101 1000			POP	AX
59	0101 1001			POP	CX
5A	0101 1010			POP	DX
5B	0101 1011			POP	BX
5C	0101 1100			POP	SP
5D	0101 1101			POP	BP
5E	0101 1110			POP	SI
5F	0101 1111			POP	DI
60	0110 0000			(not used)	
61	0110 0001			(not used)	
62	0110 0010			(not used)	
63	0110 0011			(not used)	
64	0110 0100			(not used)	
65	0110 0101			(not used)	
66	0110 0110			(not used)	
67	0110 0111			(not used)	

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
68	0110 1000			(not used)	
69	0110 1001			(not used)	
6A	0110 1010			(not used)	
6B	0110 1011			(not used)	
6C	0110 1100			(not used)	
6D	0110 1101			(not used)	
6E	0110 1110			(not used)	
6F	0110 1111			(not used)	
70	0111 0000	IP-INC8		JO	SHORT-LABEL
71	0111 0001	IP-INC8		JNO	SHORT-LABEL
72	0111 0010	IP-INC8		JB/JNAE/ JC	SHORT-LABEL
73	0111 0011	IP-INC8		JNB/JAE/ JNC	SHORT-LABEL
74	0111 0100	IP-INC8		JE/JZ	SHORT-LABEL
75	0111 0101	IP-INC8		JNE/JNZ	SHORT-LABEL
76	0111 0110	IP-INC8		JBE/JNA	SHORT-LABEL
77	0111 0111	IP-INC8		JNBE/JA	SHORT-LABEL
78	0111 1000	IP-INC8		JS	SHORT-LABEL
79	0111 1001	IP-INC8		JNS	SHORT-LABEL
7A	0111 1010	IP-INC8		JP/JPE	SHORT-LABEL
7B	0111 1011	IP-INC8		JNP/JPO	SHORT-LABEL
7C	0111 1100	IP-INC8		JL/JNGE	SHORT-LABEL
7D	0111 1101	IP-INC8		JNL/JGE	SHORT-LABEL
7E	0111 1110	IP-INC8		JLE/JNG	SHORT-LABEL
7F	0111 1111	IP-INC8		JNLE/JG	SHORT-LABEL
80	1000 0000	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-8	ADD	REG8/MEM8,IMMED8
80	1000 0000	MOD 001 R/M	(DISP-LO),(DISP-HI), DATA-8	OR	REG8/MEM8,IMMED8
80	1000 0000	MOD 010 R/M	(DISP-LO),(DISP-HI), DATA-8	ADC	REG8/MEM8,IMMED8
80	1000 0000	MOD 011 R/M	(DISP-LO),(DISP-HI), DATA-8	SBB	REG8/MEM8,IMMED8
80	1000 0000	MOD 100 R/M	(DISP-LO),(DISP-HI), DATA-8	AND	REG8/MEM8,IMMED8
80	1000 0000	MOD 101 R/M	(DISP-LO),(DISP-HI), DATA-8	SUB	REG8/MEM8,IMMED8
80	1000 0000	MOD 110 R/M	(DISP-LO),(DISP-HI), DATA-8	XOR	REG8/MEM8,IMMED8
80	1000 0000	MOD 111 R/M	(DISP-LO),(DISP-HI), DATA-8	CMP	REG8/MEM8,IMMED8
81	1000 0001	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	ADD	REG16/MEM16,IMMED16
81	1000 0001	MOD 001 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	OR	REG16/MEM16,IMMED16
81	1000 0001	MOD 010 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	ADC	REG16/MEM16,IMMED16
81	1000 0001	MOD 011 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	SBB	REG16/MEM16,IMMED16

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
81	1000 0001	MOD 100 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	AND	REG16/MEM16,IMMED16
81	1000 0001	MOD 101 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	SUB	REG16/MEM16,IMMED16
81	1000 0001	MOD 110 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	XOR	REG16/MEM16,IMMED16
81	1000 0001	MOD 111 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	CMP	REG16/MEM16,IMMED16
82	1000 0010	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-8	ADD	REG8/MEM8,IMMED8
82	1000 0010	MOD 001 R/M		(not used)	
82	1000 0010	MOD 010 R/M	(DISP-LO),(DISP-HI), DATA-8	ADC	REG8/MEM8,IMMED8
82	1000 0010	MOD 011 R/M	(DISP-LO),(DISP-HI), DATA-8	SBB	REG8/MEM8,IMMED8
82	1000 0010	MOD 100 R/M		(not used)	
82	1000 0010	MOD 101 R/M	(DISP-LO),(DISP-HI), DATA-8	SUB	REG8/MEM8,IMMED8
82	1000 0010	MOD 110 R/M		(not used)	
82	1000 0010	MOD 111 R/M	(DISP-LO),(DISP-HI), DATA-8	CMP	REG8/MEM8,IMMED8
83	1000 0011	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-SX	ADD	REG16/MEM16, IMMED8
83	1000 0011	MOD 001 R/M		(not used)	
83	1000 0011	MOD 010 R/M	(DISP-LO), (DISP-HI), DATA-SX	ADC	REG16/MEM16,IMMED8
83	1000 0011	MOD 011 R/M	(DISP-LO),(DISP-HI), DATA-SX	SBB	REG16/MEM16,IMMED8
83	1000 0011	MOD 100 R/M		(not used)	
83	1000 0011	MOD 101 R/M	(DISP-LO),(DISP-HI), DATA-SX	SUB	REG16/MEM16,IMMED8
83	1000 0011	MOD 110 R/M		(not used)	
83	1000 0011	MOD 111 R/M	(DISP-LO),(DISP-HI), DATA-SX	CMP	REG16/MEM16,IMMED8
84	1000 0100	MOD REG R/M	(DISP-LO),(DISP-HI)	TEST	REG8/MEM8,REG8
85	1000 0101	MOD REG R/M	(DISP-LO),(DISP-HI)	TEST	REG16/MEM16,REG16
86	1000 0110	MOD REG R/M	(DISP-LO),(DISP-HI)	XCHG	REG8,REG8/MEM8
87	1000 0111	MOD REG R/M	(DISP-LO),(DISP-HI)	XCHG	REG16,REG16/MEM16
88	1000 1000	MOD REG R/M	(DISP-LO),(DISP-HI)	MOV	REG8/MEM8,REG8
89	1000 1001	MOD REG R/M	(DISP-LO),(DISP-HI)	MOV	REG16/MEM16/REG16
8A	1000 1010	MOD REG R/M	(DISP-LO),(DISP-HI)	MOV	REG8,REG8/MEM8
8B	1000 1011	MOD REG R/M	(DISP-LO),(DISP-HI)	MOV	REG16,REG16/MEM16
8C	1000 1100	MOD 0SR R/M	(DISP-LO),(DISP-HI)	MOV	REG16/MEM16,SEGREG
8C	1000 1100	MOD 1-- R/M		(not used)	
8D	1000 1101	MOD REG R/M	(DISP-LO),(DISP-HI)	LEA	REG16,MEM16
8E	1000 1110	MOD 0SR R/M	(DISP-LO),(DISP-HI)	MOV	SEGREG,REG16/MEM16
8E	1000 1110	MOD 1-- R/M		(not used)	
8F	1000 1111	MOD 000 R/M	(DISP-LO),(DISP-HI)	POP	REG16/MEM16
8F	1000 1111	MOD 001 R/M		(not used)	
8F	1000 1111	MOD 010 R/M		(not used)	

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
8F	1000 1111	MOD 011 R/M		(not used)	
8F	1000 1111	MOD 100 R/M		(not used)	
8F	1000 1111	MOD 101 R/M		(not used)	
8F	1000 1111	MOD 110 R/M		(not used)	
8F	1000 1111	MOD 111 R/M		(not used)	
90	1001 0000			NOP	(exchange AX,AX)
91	1001 0001			XCHG	AX,CX
92	1001 0010			XCHG	AX,DX
93	1001 0011			XCHG	AX,BX
94	1001 0100			XCHG	AX,SP
95	1001 0101			XCHG	AX,BP
96	1001 0110			XCHG	AX,SI
97	1001 0111			XCHG	AX,DI
98	1001 1000			CBW	
99	1001 1001			CWD	
9A	1001 1010	DISP-LO	DISP-HI,SEG-LO, SEG-HI	CALL	FAR_PROC
9B	1001 1011			WAIT	
9C	1001 1100			PUSHF	
9D	1001 1101			POPF	
9E	1001 1110			SAHF	
9F	1001 1111			LAHF	
A0	1010 0000	ADDR-LO	ADDR-HI	MOV	AL,MEM8
A1	1010 0001	ADDR-LO	ADDR-HI	MOV	AX,MEM16
A2	1010 0010	ADDR-LO	ADDR-HI	MOV	MEM8,AL
A3	1010 0011	ADDR-LO	ADDR-HI	MOV	MEM16,AL
A4	1010 0100			MOVS	DEST-STR8,SRC-STR8
A5	1010 0101			MOVS	DEST-STR16,SRC-STR16
A6	1010 0110			CMPS	DEST-STR8,SRC-STR8
A7	1010 0111			CMPS	DEST-STR16,SRC-STR16
A8	1010 1000	DATA-8		TEST	AL,IMMED8
A9	1010 1001	DATA-LO	DATA-HI	TEST	AX,IMMED16
AA	1010 1010			STOS	DEST-STR8
AB	1010 1011			STOS	DEST-STR16
AC	1010 1100			LODS	SRC-STR8
AD	1010 1101			LODS	SRC-STR16
AE	1010 1110			SCAS	DEST-STR8
AF	1010 1111			SCAS	DEST-STR16
B0	1011 0000	DATA-8		MOV	AL,IMMED8
B1	1011 0001	DATA-8		MOV	CL,IMMED8
B2	1011 0010	DATA-8		MOV	DL,IMMED8
B3	1011 1011	DATA-8		MOV	BL,IMMED8
B4	1011 0100	DATA-8		MOV	AH,IMMED8
B5	1011 0101	DATA-8		MOV	CH,IMMED8
B6	1011 0110	DATA-8		MOV	DH,IMMED8
B7	1011 0111	DATA-8		MOV	BH,IMMED8
B8	1011 1000	DATA-LO	DATA-HI	MOV	AX,IMMED16
B9	1011 1001	DATA-LO	DATA-HI	MOV	CX,IMMED16
BA	1011 1010	DATA-LO	DATA-HI	MOV	DX,IMMED16
BB	1011 1011	DATA-LO	DATA-HI	MOV	BX,IMMED16

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
BC	1011 1100	DATA-LO	DATA-HI	MOV	SP,IMMED16
BD	1011 1101	DATA-LO	DATA-HI	MOV	BP,IMMED16
BE	1011 1110	DATA-LO	DATA-HI	MOV	SI,IMMED16
BF	1011 1111	DATA-LO	DATA-HI	MOV	DI,IMMED16
C0	1100 0000			(not used)	
C1	1100 0001			(not used)	
C2	1100 0010	DATA-LO	DATA-HI	RET	IMMED16 (intraseg)
C3	1100 0011			RET	(intrasegment)
C4	1100 0100	MOD REG R/M	(DISP-LO),(DISP-HI)	LES	REG16,MEM16
C5	1100 0101	MOD REG R/M	(DISP-LO),(DISP-HI)	LDS	REG16,MEM16
C6	1100 0110	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-8	MOV	MEM8,IMMED8
C6	1100 0110	MOD 001 R/M		(not used)	
C6	1100 0110	MOD 010 R/M		(not used)	
C6	1100 0110	MOD 011 R/M		(not used)	
C6	1100 0110	MOD 100 R/M		(not used)	
C6	1100 0110	MOD 101 R/M		(not used)	
C6	1100 0110	MOD 110 R/M		(not used)	
C6	1100 0110	MOD 111 R/M		(not used)	
C7	1100 0111	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	MOV	MEM16,IMMED16
C7	1100 0111	MOD 001 R/M		(not used)	
C7	1100 0111	MOD 010 R/M		(not used)	
C7	1100 0111	MOD 011 R/M		(not used)	
C7	1100 0111	MOD 100 R/M		(not used)	
C7	1100 0111	MOD 101 R/M		(not used)	
C7	1100 0111	MOD 110 R/M		(not used)	
C7	1100 0111	MOD 111 R/M		(not used	
C8	1100 1000			(not used)	
C9	1100 1001			(not used)	
CA	1100 1010	DATA-LO	DATA-HI	RET	IMMED16 (intersegment)
CB	1100 1011			RET	(intersegment)
CC	1100 1100			INT	3
CD	1100 1101	DATA-8		INT	IMMED8
CE	1100 1110			INTO	
CF	1100 1111			IRET	
D0	1101 0000	MOD 000 R/M	(DISP-LO),(DISP-HI)	ROL	REG8/MEM8,1
D0	1101 0000	MOD 001 R/M	(DISP-LO),(DISP-HI)	ROR	REG8/MEM8,1
D0	1101 0000	MOD 010 R/M	(DISP-LO),(DISP-HI)	RCL	REG8/MEM8,1
D0	1101 0000	MOD 011 R/M	(DISP-LO),(DISP-HI)	RCR	REG8/MEM8,1
D0	1101 0000	MOD 100 R/M	(DISP-LO),(DISP-HI)	SAL/SHL	REG8/MEM8,1
D0	1101 0000	MOD 101 R/M	(DISP-LO),(DISP-HI)	SHR	REG8/MEM8,1
D0	1101 0000	MOD 110 R/M		(not used)	
D0	1101 0000	MOD 111 R/M	(DISP-LO),(DISP-HI)	SAR	REG8/MEM8,1
D1	1101 0001	MOD 000 R/M	(DISP-LO),(DISP-HI)	ROL	REG16/MEM16,1
D1	1101 0001	MOD 001 R/M	(DISP-LO),(DISP-HI)	ROR	REG16/MEM16,1
D1	1101 0001	MOD 010 R/M	(DISP-LO),(DISP-HI)	RCL	REG16/MEM16,1
D1	1101 0001	MOD 011 R/M	(DISP-LO),(DISP-HI)	RCR	REG16/MEM16,1
D1	1101 0001	MOD 100 R/M	(DISP-LO),(DISP-HI)	SAL/SHL	REG16/MEM16,1

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
D1	1101 0001	MOD 101 R/M	(DISP-LO),(DISP-HI)	SHR	REG16/MEM16,1
D1	1101 0001	MOD 110 R/M		(not used)	
D1	1101 0001	MOD 111 R/M	(DISP-LO),(DISP-HI)	SAR	REG16/MEM16,1
D2	1101 0010	MOD 000 R/M	(DISP-LO),(DISP-HI)	ROL	REG8/MEM8,CL
D2	1101 0010	MOD 001 R/M	(DISP-LO),(DISP-HI)	ROR	REG8/MEM8,CL
D2	1101 0010	MOD 010 R/M	(DISP-LO),(DISP-HI)	RCL	REG8/MEM8,CL
D2	1101 0010	MOD 011 R/M	(DISP-LO),(DISP-HI)	RCR	REG8/MEM8,CL
D2	1101 0010	MOD 100 R/M	(DISP-LO),(DISP-HI)	SAL/SHL	REG8/MEM8,CL
D2	1101 0010	MOD 101 R/M	(DISP-LO),(DISP-HI)	SHR	REG8/MEM8,CL
D2	1101 0010	MOD 110 R/M		(not used)	
D2	1101 0010	MOD 111 R/M	(DISP-LO),(DISP-HI)	SAR	REG8/MEM8,CL
D3	1101 0011	MOD 000 R/M	(DISP-LO),(DISP-HI)	ROL	REG16/MEM16,CL
D3	1101 0011	MOD 001 R/M	(DISP-LO),(DISP-HI)	ROR	REG16/MEM16,CL
D3	1101 0011	MOD 010 R/M	(DISP-LO),(DISP-HI)	RCL	REG16/MEM16,CL
D3	1101 0011	MOD 011 R/M	(DISP-LO),(DISP-HI)	RCR	REG16/MEM16,CL
D3	1101 0011	MOD 100 R/M	(DISP-LO),(DISP-HI)	SAL/SHL	REG16/MEM16,CL
D3	1101 0011	MOD 101 R/M	(DISP-LO),(DISP-HI)	SHR	REG16/MEM16,CL
D3	1101 0011	MOD 110 R/M		(not used)	
D3	1101 0011	MOD 111 R/M	(DISP-LO),(DISP-HI)	SAR	REG16/MEM16,CL
D4	1101 0100	00001010		AAM	
D5	1101 0101	00001010		AAD	
D6	1101 0110			(not used)	
D7	1101 0111			XLAT	SOURCE-TABLE
D8	1101 1000	MOD 000 R/M			
	1XXX	MOD YYY R/M	(DISP-LO), (DISP-HI)	ESC	OPCODE,SOURCE
DF	1101 1111	MOD 111 R/M			
E0	1110 0000	IP-INC-8		LOOPNE/ LOOPNZ	SHORT-LABEL
E1	1110 0001	IP-INC-8		LOOPE/ LOOPZ	SHORT-LABEL
E2	1110 0010	IP-INC-8		LOOP	SHORT-LABEL
E3	1110 0011	IP-INC-8		JCXZ	SHORT-LABEL
E4	1110 0100	DATA-8		IN	AL,IMMED8
E5	1110 0101	DATA-8		IN	AX,IMMED8
E6	1110 0110	DATA-8		OUT	AL,IMMED8
E7	1110 0111	DATA-8		OUT	AX,IMMED8
E8	1110 1000	IP-INC-LO	IP-INC-HI	CALL	NEAR-PROC
E9	1110 1001	IP-INC-LO	IP-INC-HI	JMP	NEAR-LABEL
EA	1110 1010	IP-LO	IP-HI,CS-LO,CS-HI	JMP	FAR-LABEL
EB	1110 1011	IP-INC8		JMP	SHORT-LABEL
EC	1110 1100			IN	AL,DX
ED	1110 1101			IN	AX,DX
EE	1110 1110			OUT	AL,DX
EF	1110 1111			OUT	AX,DX
F0	1111 0000			LOCK	(prefix)
F1	1111 0001			(not used)	
F2	1111 0010			REPNE/REPNZ	
F3	1111 0011			REP/REPE/REPZ	
F4	1111 0100			HLT	
F5	1111 0101			CMC	

Table 4-13. Machine Instruction Decoding Guide (Cont'd.)

1ST BYTE		2ND BYTE	BYTES 3,4,5,6	ASM-86 INSTRUCTION FORMAT	
HEX	BINARY				
F6	1111 0110	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-8	TEST	REG8/MEM8,IMMED8
F6	1111 0110	MOD 001 R/M		(not used)	
F6	1111 0110	MOD 010 R/M	(DISP-LO),(DISP-HI)	NOT	REG8/MEM8
F6	1111 0110	MOD 011 R/M	(DISP-LO),(DISP-HI)	NEG	REG8/MEM8
F6	1111 0110	MOD 100 R/M	(DISP-LO),(DISP-HI)	MUL	REG8/MEM8
F6	1111 0110	MOD 101 R/M	(DISP-LO),(DISP-HI)	IMUL	REG8/MEM8
F6	1111 0110	MOD 110 R/M	(DISP-LO),(DISP-HI)	DIV	REG8/MEM8
F6	1111 0110	MOD 111 R/M	(DISP-LO),(DISP-HI)	IDIV	REG8/MEM8
F7	1111 0111	MOD 000 R/M	(DISP-LO),(DISP-HI), DATA-LO,DATA-HI	TEST	REG16/MEM16,IMMED16
F7	1111 0111	MOD 001 R/M		(not used)	
F7	1111 0111	MOD 010 R/M	(DISP-LO),(DISP-HI)	NOT	REG16/MEM16
F7	1111 0111	MOD 011 R/M	(DISP-LO),(DISP-HI)	NEG	REG16/MEM16
F7	1111 0111	MOD 100 R/M	(DISP-LO),(DISP-HI)	MUL	REG16/MEM16
F7	1111 0111	MOD 101 R/M	(DISP-LO),(DISP-HI)	IMUL	REG16/MEM16
F7	1111 0111	MOD 110 R/M	(DISP-LO),(DISP-HI)	DIV	REG16/MEM16
F7	1111 0111	MOD 111 R/M	(DISP-LO),(DISP-HI)	IDIV	REG16/MEM16
F8	1111 1000			CLC	
F9	1111 1001			STC	
FA	1111 1010			CLI	
FB	1111 1011			STI	
FC	1111 1100			CLD	
FD	1111 1101			STD	
FE	1111 1110	MOD 000 R/M	(DISP-LO),(DISP-HI)	INC	REG8/MEM8
FE	1111 1110	MOD 001 R/M	(DISP-LO),(DISP-HI)	DEC	REG8/MEM8
FE	1111 1110	MOD 010 R/M		(not used)	
FE	1111 1110	MOD 011 R/M		(not used)	
FE	1111 1110	MOD 100 R/M		(not used)	
FE	1111 1110	MOD 101 R/M		(not used)	
FE	1111 1110	MOD 110 R/M		(not used)	
FE	1111 1110	MOD 111 R/M		(not used)	
FF	1111 1111	MOD 000 R/M	(DISP-LO),(DISP-HI)	INC	MEM16
FF	1111 1111	MOD 001 R/M	(DISP-LO),(DISP-HI)	DEC	MEM16
FF	1111 1111	MOD 010 R/M	(DISP-LO),(DISP-HI)	CALL	REG16/MEM16 (intra)
FF	1111 1111	MOD 011 R/M	(DISP-LO),(DISP-HI)	CALL	MEM16 (intersegment)
FF	1111 1111	MOD 100 R/M	(DISP-LO),(DISP-HI)	JMP	REG16/MEM16 (intra)
FF	1111 1111	MOD 101 R/M	(DISP-LO),(DISP-HI)	JMP	MEM16 (intersegment)
FF	1111 1111	MOD 110 R/M	(DISP-LO),(DISP-HI)	PUSH	MEM16
FF	1111 1111	MOD 111 R/M		(not used)	