8086 Instruction Set Summary

The following is a brief summary of the 8086 instruction set:

Data Transfer Instructions

`MOV`	Move byte or word to register or memory
`IN, OUT`	Input byte or word from port, output word to port
`LEA`	Load effective address
`LDS, LES`	Load pointer using data segment, extra segment
`PUSH, POP`	Push word onto stack, pop word off stack
`XCHG`	Exchange byte or word
`XLAT`	Translate byte using look-up table

Logical Instructions

`NOT`	Logical NOT of byte or word (one's complement)
`AND`	Logical AND of byte or word
`OR`	Logical OR of byte or word
`XOR`	Logical exclusive-OR of byte or word
`TEST`	Test byte or word (AND without storing)

Shift and Rotate Instructions

`SHL, SHR`	Logical shift left, right byte or word by 1 or CL
`SAL, SAR`	Arithmetic shift left, right byte or word by 1 or CL
`ROL, ROR`	Rotate left, right byte or word by 1 or CL
`RCL, RCR`	Rotate left, right through carry byte or word by 1 or CL

Arithmetic Instructions

`ADD, SUB`	Add, subtract byte or word
`ADC, SBB`	Add, subtract byte or word and carry (borrow)
`INC, DEC`	Increment, decrement byte or word
`NEG`	Negate byte or word (two's complement)
`CMP`	Compare byte or word (subtract without storing)
`MUL, DIV`	Multiply, divide byte or word (unsigned)
`IMUL, IDIV`	Integer multiply, divide byte or word (signed)
`CBW, CWD`	Convert byte to word, word to double word (useful before multiply/divide)
`AAA, AAS, AAM, AAD`	ASCII adjust for addition, subtraction, multiplication, division (ASCII codes 30-39)
`DAA, DAS`	Decimal adjust for addition, subtraction (binary coded decimal numbers)

Transfer Instructions

`JMP`	Unconditional jump
`JA (JNBE)`	Jump if above (not below or equal)
`JAE (JNB)`	Jump if above or equal (not below)
`JB (JNAE)`	Jump if below (not above or equal)
`JBE (JNA)`	Jump if below or equal (not above)
`JE (JZ)`	Jump if equal (zero)
`JG (JNLE)`	Jump if greater (not less or equal)
`JGE (JNL)`	Jump if greater or equal (not less)

8259 Programmable Interrupt Controller (PIC)

ICW1

7	6	5	4	3	2	1	0
0	0	0	1	LTIM	0	SNGL	IC4

LTIM: 0=Edge Triggering 1=Level Triggering
SNGL: 0=Cascaded PICs 1=Master Only
IC4: 0=No ICW4 1=ICW4 Required

ICW2

7	6	5	4	3	2	1	0
Off7	Off6	Off5	Off4	Off3	0	0	0

Off7..Off3: Programmable Interrupt Vector Offset

ICW3 (Master)

7	6	5	4	3	2	1	0
S7	S6	S5	S4	S3	S2	S1	S0

S7..S0: 0=IR Line is Connected to Peripheral Device or is Free
1=IR Line is Connected to Slave PIC

ICW4

7	6	5	4	3	2	1	0
0	0	0	SFNM	BUF	M/S	AEOI	µPM

SFNM: 0=No Special Fully Nested Mode 1=Special Fully Nested Mode
BUF: 0=No Buffered Mode 1=Buffered Mode
M/S: 0=Slave PIC 1=Master PIC
AEOI: 0=Manual EOI 1=Automatic EOI
µPM: 0=MCS-80/85 Mode 1=8086/88 Mode

OCW1

A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	M7	M6	M5	M4	M3	M2	M1	M0

INTERRUPT MASK
1 = MASK SET
0 = MASK RESET

8253/54 Programmable Interval Timer (PIT)

Kontrol kelimesinin formatı

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC_1	SC_0	RW_1	RW_0	M_2	M_1	M_0	BCD

SC - Select counter

SC_1	SC_0	
0	0	Select counter 0
0	1	Select counter 1
1	0	Select counter 2
1	1	Illegal for 8253 Read -Back command for 8254 (See Read operations)

M - Mode

M_2	M_1	M_0	
0	0	0	Mode 0
0	0	1	Mode 1
x	1	0	Mode 2
x	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

RW - Read /Write

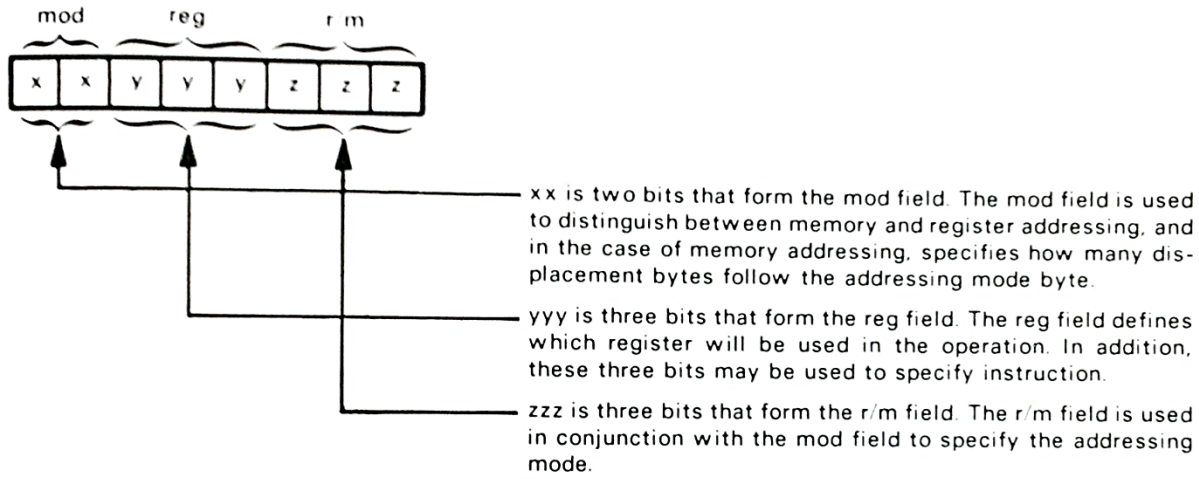
RW_1	RW_0	
0	0	Counter latch command (See Read operations)
0	1	Read / Write least significant byte only
1	0	Read / Write most significant byte only
1	1	Read / write least significant byte first, then most significant byte

BCD :

0	Binary counter 16 - bits
1	Binary coded decimal (BCD) Counter (4 Decades)

ADDRESSING MODE BYTE

The 8086 obviously offers an extensive selection of addressing modes. The next question is: how are these addressing modes implemented in the object code? The 8086 specifies most data memory addressing modes in an instruction's object code using one byte of object code, known as the addressing mode byte. The addressing mode byte may have one or two additional displacement bytes associated with it. The addressing mode byte is always the second byte of the instruction object code, unless a prefix instruction has been included prior to the initial object code. The addressing mode byte may be illustrated as follows:



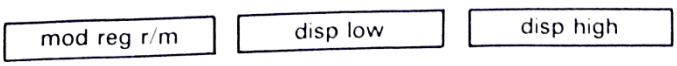
mod =

00 Memory addressing mode. r/m specifies the exact addressing option. There are no displacement bytes.

01 Memory addressing mode. r/m specifies the exact addressing option. There is one displacement byte. This displacement byte is viewed as a signed number in the range +127 to −128. When this number is used in the memory address calculation, the number is sign extended to 16 bits. In this case, the addressing mode bytes can be illustrated as follows:

where mod = 01 and disp is the 8-bit signed displacement value.

10 Memory addressing mode. r/m specifies the addressing option. There are two displacement bytes. The first displacement byte is the low-order eight bits of the displacement. The second displacement byte is the high-order eight bits of the displacement. When this number is used in the memory address calculation, the number is treated as an unsigned 16-bit number. In this case, the addressing mode bytes can be illustrated as follows:



where mod = 10, disp low is the low-order eight bits of the displacement, and disp high is the high-order eight bits of the displacement.

11 register addressing mode. r/m specifies a register. Used in conjunction with the w bit to determine if an 8- or 16-bit register is selected.

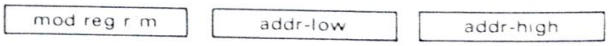
reg reg is used in conjunction with another bit, the w bit, in the selection of the register to be used in the operation. The w bit, which is part of the instruction op-code, selects whether an 8- or 16-bit operation is performed.

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

r/m r/m specifies the addressing mode in conjunction with mod, as follows:

r/m	mod - 00	mod - 01	mod - 10	mod - 11	
				w = 0	w = 1
000	BX + SI	BX + SI + DISP	BX + SI + DISP	AL	AX
001	BX + DI	BX + DI + DISP	BX + DI + DISP	CL	CX
010	BP + SI	BP + SI + DISP	BP + SI + DISP	DL	DX
011	BP + DI	BP + DI + DISP	BP + DI + DISP	BL	BX
100	SI	SI + DISP	SI + DISP	AH	SP
101	DI	DI + DISP	DI + DISP	CH	BP
110	Direct Address	BP + DISP	BP + DISP	DH	SI
111	BX	BX + DISP	BX + DISP	BH	DI

This table is self-explanatory, with the exception of Direct Address. When mod is 00 and r/m is 110, the offset address is taken directly from the two bytes that follow the addressing mode byte. This can be illustrated as follows:



MEMORY ADDRESSING TABLES

Memory addressing modes and memory addressing byte information can be combined and summarized as follows:

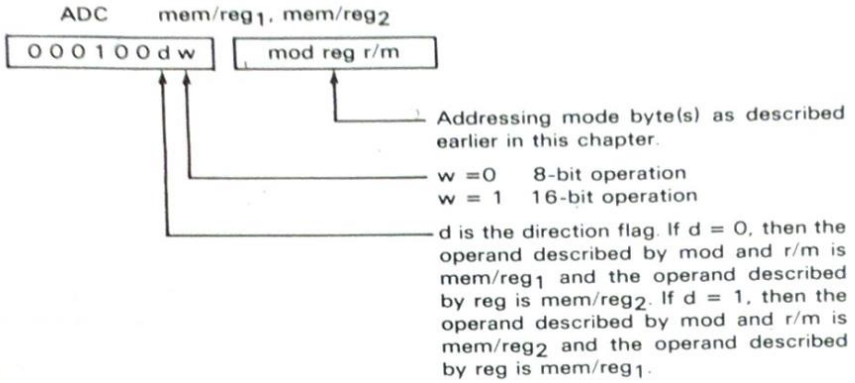
r/m =	mod = 00	mode = 01	mod = 10
000	Base Relative Indexed BX + SI	Base Relative Indexed BX + SI + DISP	Base Relative Direct Indexed BX + SI + DISP
001	Base Relative Indexed BX + DI	Base Relative Direct Indexed BX + DI + DISP	Base Relative Direct Indexed BX + DI + DISP
010	Base Relative Indexed Stack BP + SI	Base Relative Direct Indexed Stack BP + SI + DISP	Base Relative Direct Indexed Stack BP + SI + DISP
011	Base Relative Indexed Stack BP + DI	Base Relative Direct Indexed Stack BP + DI + DISP	Base Relative Direct Indexed Stack BP + DI + DISP
100	Implied SI	Direct, Indexed SI + DISP	Direct, Indexed SI + DISP
101	Implied DI	Direct, Indexed DI + DISP	Direct, Indexed DI + DISP
110	Direct Direct Address	Base Relative Direct Stack BP + DISP	Base Relative Direct Stack BP + DISP
111	Base Relative BX	Base Relative Direct BX + DISP	Base Relative Direct BX + DISP

ADC mem/reg$_1$, mem/reg$_2$

Add Data With Carry From:
- **Register to Register**
- **Register to Memory**
- **Memory to Register**

Add the contents of the register or memory location specified by mem/reg$_2$ and the Carry status to the contents of the register or memory location specified by mem/reg$_1$. An 8- or 16-bit operation may be specified. Either mem/reg$_1$ or mem/reg$_2$ may be a memory operand, but one of the operands must be a register operand.

The encoding for this instruction is:

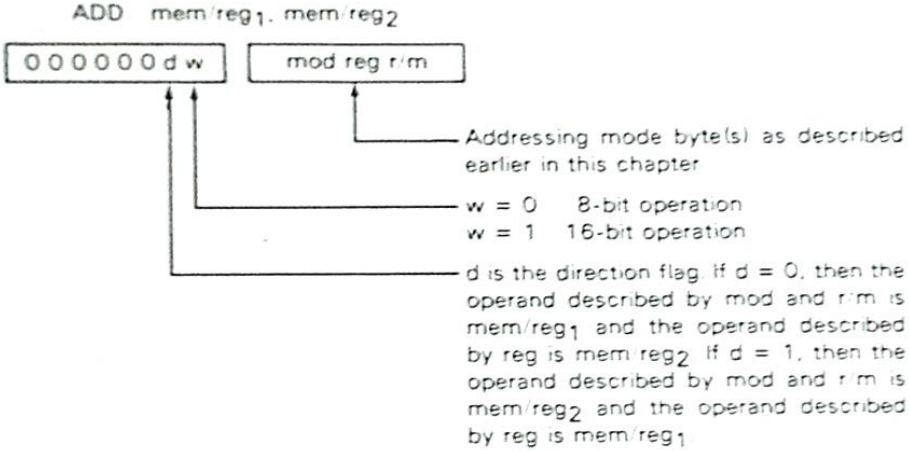


ADD mem/reg$_1$, mem/reg$_2$

Add:
1. **Register to Register**
2. **Register to Memory**
3. **Memory to Register**

Add the contents of the register or memory location specified by mem/reg$_2$ to the contents of the register or memory location specified by mem/reg$_1$. An 8- or 16-bit operation may be specified. Either mem/reg$_1$ or mem/reg$_2$ may be a memory operand, but one of the operands must be a register operand.

The encoding for this instruction is:

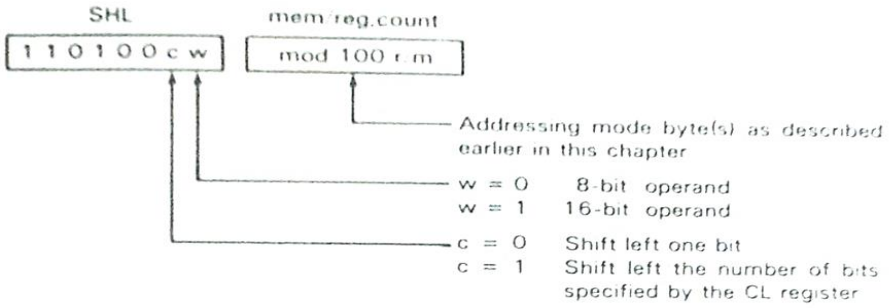


SHL mem/reg,count
SAL mem/reg,count

Shift Register or Memory Location Left

Shift the contents of the specified register or memory location left by the specified number of bits. The number of bits to shift, represented by the variable count, is either one or the number contained in the CL register. This is a logical left shift.

The encoding for this instruction is:

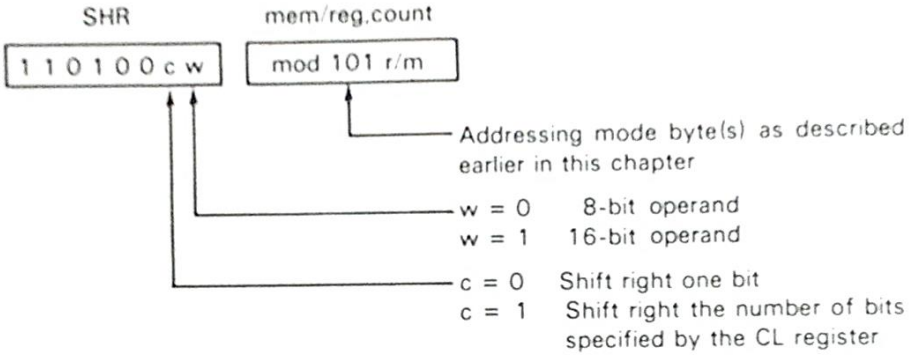


SHR mem/reg,count

Shift Register or Memory Location Right

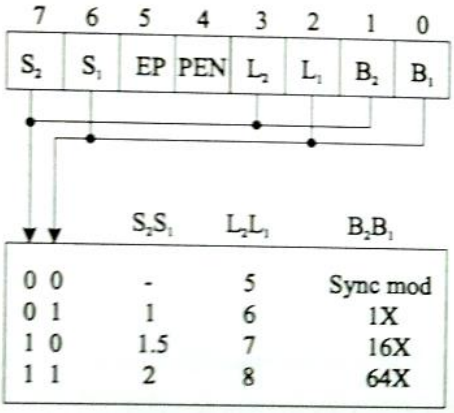
Shift the contents of the specified register or memory location right by the specified number of bits. The number of bits to shift, represented by the variable count, is either one or the number contained in the CL register. The bit shifted into the high-order bit is a zero. This is a logical right shift.

The encoding for this instruction is:



8251 USART (Universal Synchronous Asynchronous Receiver Transmitter)

Mod emir kelimesinin biçimi



Bit adı	Görevi
B_2B_1	: Baud hızı
L_2L_1	: Karakter uzunluğu
S_2S_1	: Bitiş bitleri
PEN	: Eşitlik biti yetkisi
1	= Yetkili
0	= Yetkisiz
EP	: Eşitlik denetimi
1	= Çift
0	= Tek

8251 komut kelimesinin biçimi

7	6	5	4	3	2	1	0
EH	IR	RTS	ER	SBRK	RxE	DTR	TxEN

Bit adı	Görevi
EH	: Avlanma moduna gir (1=sync karakterlerini araştırır).
IR	: İç reset (1=8251A cihazı başlatımlama moduna sokar).
RTS	: Modeme gönderme isteği (1=$\overline{\text{RTS}}$ çıkışını sıfırlar).
ER	: Eşitlik, işgal, ve çerçeveleme hatalarını sıfırlar.
SBRK	: Kesme karakteri gönderme (1=TxD çıkışını alçak yapar, 0=Normal çalışma).
RxE	: Alıcıyı yetkilendirir (1=Yetkili, 0=Yetkisiz).
DTR	: Veri terminali hazır (1=$\overline{\text{DTR}}$ çıkışını sıfırlar).
TxEN	: Göndericiyi yetkilendirir (1=Yetkili, 0=Yetkisiz).

Durum kelimesinin biçimi

7	6	5	4	3	2	1	0
DSR	SYNDET	FE	OE	PE	TxE	RxRDY	TxRDY

Bit adı	Görevi
DSR	: Veri seti hazır. $\overline{\text{DSR}}$ girişinin değerini alır.
SYNDET	: SYNC karakter algılaması (senkron çalışma).
FE	: Çerçeveleme hatası.
OE	: İşgal hatası.
PE	: Eşitlik hatası.
TxE	: Gönderici boş.
RxRDY	: Alıcı boş.
TxRDY	: Gönderici hazır.