

1. **ASCII** (American Standard Code for Information Interchange) merupakan Kode Standar Amerika untuk Pertukaran Informasi atau sebuah standar internasional dalam pengkodean huruf dan simbol seperti Unicode dan Hex tetapi ASCII lebih bersifat universal.

Tabel ASCII Control Character

Karakter	Unicode (Heksadesimal)	ANSI ASCII (Desimal)	Keterangan
NUL	0	0	Null (tidak tampak)
SOH	1	1	Start of heading (tidak tampak)
STX	2	2	Start of text (tidak tampak)
ETX	3	3	End of text (tidak tampak)
EOT	4	4	End of transmission (tidak tampak)
ENQ	5	5	Enquiry (tidak tampak)
ACK	6	6	Acknowledge (tidak tampak)
BEL	7	7	Bell (tidak tampak)
BS	8	8	Menghapus satu karakter di belakang kursor (Backspace)
HT	9	9	Horizontal tabulation
LF	000A	10	Pergantian baris (Line feed)
VT	000B	11	Tabulasi vertikal
FF	000C	12	Pergantian baris (Form feed)
CR	000D	13	Pergantian baris (carriage return)
SO	000E	14	Shift out (tidak tampak)
SI	000F	15	Shift in (tidak tampak)
DLE	10	16	Data link escape (tidak tampak)
DC1	11	17	Device control 1 (tidak tampak)
DC2	12	18	Device control 2 (tidak tampak)
DC3	13	19	Device control 3 (tidak tampak)
DC4	14	20	Device control 4 (tidak tampak)
NAK	15	21	Negative acknowledge (tidak tampak)
SYN	16	22	Synchronous idle (tidak tampak)
ETB	17	23	End of transmission block (tidak tampak)
CAN	18	24	Cancel (tidak tampak)
EM	19	25	End of medium (tidak tampak)
SUB	001A	26	Substitute (tidak tampak)
ESC	001B	27	Escape (tidak tampak)
FS	001C	28	File separator
GS	001D	29	Group separator
RS	001E	30	Record separator
US	001F	31	Unit separator
DEL	007F	127	Delete

Tabel ASCII Printable Character

Karakter	Unicode (Heksadesimal)	ANSI ASCII (Desimal)	Keterangan
SP	20	32	Spasi
!	21	33	Tanda seru (exclamation)
“	22	34	Tanda kutip dua
#	23	35	Tanda pagar (kres)
\$	24	36	Tanda mata uang dolar
%	25	37	Tanda persen
&	26	38	Karakter ampersand (&)
‘	27	39	Karakter Apostrof
(28	40	Tanda kurung buka
)	29	41	Tanda kurung tutup
*	002A	42	Karakter asterisk (bintang)
+	002B	43	Tanda tambah (plus)
,	002C	44	Karakter koma
—	002D	45	Karakter hyphen (strip)
.	002E	46	Tanda titik
/	002F	47	Garis miring (slash)
0	30	48	Angka nol
1	31	49	Angka satu
2	32	50	Angka dua
3	33	51	Angka tiga
4	34	52	Angka empat
5	35	53	Angka lima
6	36	54	Angka enam
7	37	55	Angka tujuh
8	38	56	Angka delapan
9	39	57	Angka sembilan
:	003A	58	Tanda titik dua
;	003B	59	Tanda titik koma
<	003C	60	Tanda lebih kecil
=	003D	61	Tanda sama dengan
>	003E	62	Tanda lebih besar
?	003F	63	Tanda tanya
@	40	64	A keong (@)
A	41	65	Huruf latin A kapital
B	42	66	Huruf latin B kapital
C	43	67	Huruf latin C kapital

D	44	68	Huruf latin D kapital
E	45	69	Huruf latin E kapital
F	46	70	Huruf latin F kapital
G	47	71	Huruf latin G kapital
H	48	72	Huruf latin H kapital
I	49	73	Huruf latin I kapital
J	004A	74	Huruf latin J kapital
K	004B	75	Huruf latin K kapital
L	004C	76	Huruf latin L kapital
M	004D	77	Huruf latin M kapital
N	004E	78	Huruf latin N kapital
O	004F	79	Huruf latin O kapital
P	50	80	Huruf latin P kapital
Q	51	81	Huruf latin Q kapital
R	52	82	Huruf latin R kapital
S	53	83	Huruf latin S kapital
T	54	84	Huruf latin T kapital
U	55	85	Huruf latin U kapital
V	56	86	Huruf latin V kapital
W	57	87	Huruf latin W kapital
X	58	88	Huruf latin X kapital
Y	59	89	Huruf latin Y kapital
Z	005A	90	Huruf latin Z kapital
[005B	91	Kurung siku kiri
\	005C	92	Garis miring terbalik (backslash)
]	005D	93	Kurung sikur kanan
^	005E	94	Tanda pangkat
_	005F	95	Garis bawah (underscore)
`	60	96	Tanda petik satu
a	61	97	Huruf latin a kecil
b	62	98	Huruf latin b kecil
c	63	99	Huruf latin c kecil
d	64	100	Huruf latin d kecil
e	65	101	Huruf latin e kecil
f	66	102	Huruf latin f kecil
g	67	103	Huruf latin g kecil
h	68	104	Huruf latin h kecil
i	69	105	Huruf latin i kecil
j	006A	106	Huruf latin j kecil
k	006B	107	Huruf latin k kecil
l	006C	108	Huruf latin l kecil

m	006D	109	Huruf latin m kecil
n	006E	110	Huruf latin n kecil
o	006F	111	Huruf latin o kecil
p	70	112	Huruf latin p kecil
q	71	113	Huruf latin q kecil
r	72	114	Huruf latin r kecil
s	73	115	Huruf latin s kecil
t	74	116	Huruf latin t kecil
u	75	117	Huruf latin u kecil
v	76	118	Huruf latin v kecil
w	77	119	Huruf latin w kecil
x	78	120	Huruf latin x kecil
y	79	121	Huruf latin y kecil
z	007A	122	Huruf latin z kecil
{	007B	123	Kurung kurawal buka
	007C	124	Garis vertikal (pipa)
}	007D	125	Kurung kurawal tutup
~	007E	126	Karakter gelombang (tilde)

Tabel ASCII Extended Character

Karakter	Unicode (Heksadesimal)	ANSI ASCII (Desimal)	Keterangan
€	80	128	Dicadangkan
Û	81	129	Dicadangkan
é	82	130	Dicadangkan
Â	83	131	Dicadangkan
ä	84	132	Index
à	85	133	Next line
å	86	134	Start of selected area
ç	87	135	End of selected area
ê	88	136	Character tabulation set
ë	89	137	Character tabulation with justification
è	008A	138	Line tabulation set
ï	008B	139	Partial line down
î	008C	140	Partial line up
ì	008D	141	Reverse line feed
Ä	008E	142	Single shift two
Å	008F	143	Single shift three
É	90	144	Device control string
æ	91	145	Private use one
Æ	92	146	Private use two

ô	93	147	Set transmit state
ö	94	148	Cancel character
ò	95	149	Message waiting
Û	96	150	Start of guarded area
ù	97	151	End of guarded area
ÿ	98	152	Start of string
Ö	99	153	Dicadangkan
Ü	009A	154	Single character introducer
ø	009B	155	Control sequence introducer
£	009C	156	String terminator
Ø	009D	157	Operating system command
×	009E	158	Privacy message
<i>f</i>	009F	159	Application program command
á	00A0	160	Spasi yang bukan pemisah kata
í	00A1	161	Tanda seru terbalik
ó	00A2	162	Tanda sen (Cent)
ú	00A3	163	Tanda Poundsterling
ñ	00A4	164	Tanda mata uang (Currency)
Ñ	00A5	165	Tanda Yen
	00A6	166	Garis tegak putus-putus (broken bar)
§	00A7	167	Section sign
¨	00A8	168	Diaeresis
©	00A9	169	Tanda hak cipta (Copyright)
ª	00AA	170	Feminine ordinal indicator
«	00AB	171	Left-pointing double angle quotation mark
¬	00AC	172	Not sign
	00AD	173	Tanda strip (hyphen)
®	00AE	174	Tanda merk terdaftar
ˉ	00AF	175	Macron
°	00B0	176	Tanda derajat
±	00B1	177	Tanda kurang lebih (plus-minus)
²	00B2	178	Tanda kuadrat (pangkat dua)
³	00B3	179	Tanda kubik (pangkat tiga)
´	00B4	180	Acute accent
µ	00B5	181	Micro sign
¶	00B6	182	Pilcrow sign
·	00B7	183	Middle dot
¸	00B8	184	Spasi cedillia
¹	00B9	185	Superscript satu
º	00BA	186	Indikator ordinal maskulin
»	00BB	187	kutipan sudut ganda

¼	00BC	188	Fraksi seperempat
½	00BD	189	Fraksi satu setengah
¾	00BE	190	Fraksi tiga perempat
¿	00BF	191	Tanda tanya terbalik
À	00C0	192	Huruf latin a dengan kuburan
Á	00C1	193	Huruf latin a dengan akut
Â	00C2	194	Huruf latin a dengan sirkumfleksa
Ã	00C3	195	Huruf latin a dengan tilde
Ä	00C4	196	Huruf latin a dengan tanda titik dua di atas huruf suara
Å	00C5	197	Huruf latin a dengan cincin di atas
Æ	00C6	198	Huruf latin ae
Ç	00C7	199	Huruf latin c dengan cedilla
È	00C8	200	Huruf latin e dengan kuburan
É	00C9	201	Huruf latin e dengan akut
Ê	00CA	202	Huruf latin e dengan sirkumfleksa
Ë	00CB	203	Huruf latin e dengan tanda titik dua di atas huruf suara
Ì	00CC	204	Huruf latin i dengan kuburan
Í	00CD	205	Huruf latin i dengan akut
Î	00CE	206	Huruf latin i dengan sirkumfleksa
Ï	00CF	207	Huruf latin saya dengan tanda titik dua di atas huruf suara
Ð	00D0	208	Huruf latin eth
Ñ	00D1	209	Huruf latin n dengan tilde
Ò	00D2	210	Huruf latin o dengan kuburan
Ó	00D3	211	Huruf latin o dengan akut
Ô	00D4	212	Huruf latin o dengan sirkumfleksa
Õ	00D5	213	Huruf latin o dengan tilde
Ö	00D6	214	Huruf latin o dengan tanda titik dua di atas huruf suara
×	00D7	215	Tanda perkalian
Ø	00D8	216	Huruf latin o dengan garis miring
Ù	00D9	217	Huruf latin u dengan kuburan
Ú	00DA	218	Huruf latin u dengan akut
Û	00DB	219	Huruf latin u dengan sirkumfleksa
Ü	00DC	220	Huruf latin u dengan tanda titik dua di atas huruf suara
Ý	00DD	221	Huruf latin y dengan akut
Þ	00DE	222	Huruf latin thorn
ß	00DF	223	Huruf kecil s tajam latin –ess zed
à	0	224	Huruf latin kecil dengan kuburan
á	0	225	Huruf latin kecil dengan akut
â	0	226	Huruf latin kecil dengan sirkumfleksa
ã	0	227	Huruf latin kecil dengan tilde
ä	0	228	Huruf latin kecil dengan tanda titik dua di atas huruf suara

å	0	229	Huruf latin kecil dengan cincin di atas
æ	0	230	Huruf latin ae kecil
ç	0	231	Huruf kecil latin c dengan cedilla
è	0	232	Huruf latin e kecil dengan kuburan
é	0	233	Huruf latin e kecil dengan akut
ê	00EA	234	Huruf latin e kecil dengan sirkumfleksa
ë	00EB	235	Huruf latin e kecil dengan tanda titik dua di atas huruf suara
ì	00EC	236	Huruf kecil latin i dengan kuburan
í	00ED	237	Huruf kecil latin i dengan akut
î	00EE	238	Huruf kecil latin i dengan sirkumfleksa
ï	00EF	239	Huruf kecil latin i dengan tanda titik dua di atas huruf suara
ð	00F0	240	Huruf latin eth kecil
ñ	00F1	241	Huruf latin n kecil dengan tilde
ò	00F2	242	Huruf kecil latin o dengan kuburan
ó	00F3	243	Huruf kecil latin o akut
ô	00F4	244	Huruf kecil latin o dengan sirkumfleksa
õ	00F5	245	Huruf kecil latin o dengan tilde
ö	00F6	246	Huruf kecil latin o dengan tanda titik dua di atas huruf suara
÷	00F7	247	Tanda bagi
ø	00F8	248	Huruf kecil latin o dengan garis miring
ù	00F9	249	Huruf kecil latin u dengan kuburan
ú	00FA	250	Huruf kecil latin u dengan akut
û	00FB	251	Huruf kecil latin u dengan sirkumfleksa
ü	00FC	252	Huruf kecil latin u dengan tanda titik dua di atas huruf suara
ý	00FD	253	Huruf Latin y kecil dengan akut
þ	00FE	254	Huruf Latin duri kecil
ÿ	00FF	255	Huruf Latin y kecil dengan tanda titik dua di atas huruf suara

2. Bahasa Assembly untuk x86

Terbagi menjadi 3 bagian utama yaitu :

1. Komentar

Komentar diawali dengan tanda titik koma (;).

; ini adalah komentar

2. Label

Label diakhiri dengan tanda titik dua (:).

Contoh: main: ,loop: ,proses: ,keluar:

3. Assembler directives

Directives adalah perintah yang ditujukan kepada assembler ketika sedang menerjemahkan program kita ke bahasa mesin.

Directive dimulai dengan tanda titik. **.model** : memberitahu assembler berapa memori yang akan dipakai oleh program kita.

Ada model tiny, model small, model compact, model medium, model large, dan model huge.

.data : memberitahu assembler bahwa bagian di bawah ini adalah data program.

.code : memberitahu assembler bahwa bagian di bawah ini adalah instruksi program.

.stack : memberitahu assembler bahwa program kita memiliki stack.

Program EXE harus punya stack. Kira-kira yang penting itu dulu.

Semua directive yang dikenal assembler adalah: .186 .286 .286c .286p .287 .386 .386c .386p .387 .486 .486p .8086 .8087

.alpha .break .code .const .continue .cref .data .data? .dosseg .else .elseif .endif .endw .err .err1 .err2 .errb

.errdef .errdif .errdifi .erre .erridn .erridni .errnb .errndef .errnz .exit .fardata .fardata? .if .lall .lfcond .list .listall .listif .listmacro

.listmacroall .model .no87 .nocref .nolist .nolistif .nolistmacro .radix .repeat .sall .seq .sfcond .stack

.startup .tfcond .type .until .untilcxz .while .xall .xcref .xlist.

Definisi data

DB : define bytes. Membentuk data byte demi byte. Data bisa data numerik maupun teks.

catatan: untuk membentuk data string, pada akhir string harus diakhiri tanda dolar (\$).

sintaks: {label} DB {data} contoh: teks1 db "Hello world \$" **DW** : define words.

Membentuk data word demi word (1 word = 2 byte).

sintaks: {label} DW {data} contoh: kucing dw ?, ?, ? ;mendefinisikan tiga slot 16-bit yang isinya don't care

(disimbolkan dengan tanda tanya)

DD : define double words. Membentuk data doubleword demi doubleword (4 byte).

sintaks: {label} DD {data} **EQU** : equals. Membentuk konstanta. sintaks: {label} EQU {data}

contoh: sepuluh EQU 10

Ada assembly yang melibatkan bilangan pecahan (floating point), bilangan bulat (integer), DF (define far words),

DQ (define quad words), dan DT (define ten bytes).

Perpindahan data

MOV : move. Memindahkan suatu nilai dari register ke memori, memori ke register, atau register ke register.

sintaks: MOV {tujuan}, {sumber}

contoh:

mov AX, 4C00h ;mengisi register AX dengan 4C00(hex).

mov BX, AX ;menyalin isi AX ke BX. mov CL, [BX] ;mengisi register CL dengan data di memori yang alamatnya ditunjuk BX.

mov CL, [BX] + 2 ;mengisi CL dengan data di memori yang alamatnya ditunjuk BX lalu geser maju 2 byte.

mov [BX], AX ;menyimpan nilai AX pada tempat di memori yang ditunjuk BX. mov [BX] - 1, 00101110b

;menyimpan 00101110(bin) pada alamat yang ditunjuk BX lalu geser mundur 1 byte.

LEA : load effective address. Mengisi suatu register dengan alamat offset sebuah data.

sintaks: LEA {register}, {sumber} contoh: lea DX, teks1 **XCHG** : exchange. Menukar dua buah register langsung.

sintaks: XCHG {register 1}, {register 2} Kedua register harus punya ukuran yang sama.

Bila sama-sama 8 bit (misalnya AH dengan BL) atau sama-sama 16 bit (misalnya CX dan DX),

maka pertukaran bisa dilakukan. Sebenarnya masih banyak perintah perpindahan data, misalnya IN, OUT, LODS, LODSB, LODSW, MOVS, MOVSB, MOVSW, LDS, LES, LAHF, SAHF, dan XLAT.

Operasi logika

AND : melakukan bitwise and. sintaks: AND {register}, {angka} AND {register 1}, {register 2} hasil disimpan di register 1.

contoh: mov AL, 00001011b mov AH, 11001000b and AL, AH ;sekarang AL berisi 00001000(bin),

sedangkan AH tidak berubah.

OR : melakukan bitwise or. sintaks: OR {register}, {angka} OR {register 1}, {register 2} hasil disimpan di register 1.

NOT : melakukan bitwise not (*one's complement*) sintaks: NOT {register} hasil disimpan di register itu sendiri.

XOR : melakukan bitwise eksklusif or. sintaks: XOR {register}, {angka} XOR {register 1}, {register 2} hasil disimpan di register 1. Tips: sebuah register yang di-XOR-kan dengan dirinya sendiri akan menjadi berisi nol.

SHL : shift left. Menggeser bit ke kiri. Bit paling kanan diisi nol. sintaks: SHL {register}, {banyaknya}

SHR : shift right. Menggeser bit ke kanan. Bit paling kiri diisi nol. sintaks: SHR {register}, {banyaknya}

ROL : rotate left. Memutar bit ke kiri. Bit paling kiri jadi paling kanan kali ini. sintaks: ROL {register},

{banyaknya} Bila banyaknya rotasi tidak disebutkan, maka nilai yang ada di CL akan digunakan sebagai banyaknya rotasi.

ROR : rotate right. Memutar bit ke kanan. Bit paling kanan jadi paling kiri. sintaks: ROR {register},

{banyaknya} Bila banyaknya rotasi tidak disebutkan, maka nilai yang ada di CL akan digunakan sebagai banyaknya rotasi.

Ada lagi : RCL dan RCR.

Operasi matematika

ADD : add. Menjumlahkan dua buah register.

sintaks: ADD {tujuan}, {sumber} operasi yang terjadi: $\text{tujuan} = \text{tujuan} + \text{sumber}$.

carry (bila ada) disimpan di CF.

ADC : add with carry. Menjumlahkan dua register dan carry flag (CF).

sintaks: ADC {tujuan}, {sumber} operasi yang terjadi: $\text{tujuan} = \text{tujuan} + \text{sumber} + \text{CF}$.

carry (bila ada lagi) disimpan lagi di CF.

INC : increment. Menjumlah isi sebuah register dengan 1.

Bedanya dengan ADD, perintah INC hanya memakan 1 byte memori sedangkan ADD pakai 3 byte.

sintaks: INC {register}

SUB : subtract. Mengurangkan dua buah register.

sintaks: SUB {tujuan}, {sumber} operasi yang terjadi: $\text{tujuan} = \text{tujuan} - \text{sumber}$.

borrow (bila terjadi) menyebabkan CF bernilai 1.

SBB : subtract with borrow. Mengurangkan dua register dan carry flag (CF).

sintaks: SBB {tujuan}, {sumber} operasi yang terjadi: $\text{tujuan} = \text{tujuan} - \text{sumber} - \text{CF}$.

borrow (bila terjadi lagi) menyebabkan CF dan SF (sign flag) bernilai 1.

DEC : decrement. Mengurang isi sebuah register dengan 1.

Jika SUB memakai 3 byte memori, DEC hanya memakai 1 byte. sintaks: DEC {register}

MUL : multiply. Mengalikan register dengan AX atau AH.

sintaks: MUL {sumber} Bila register sumber adalah 8 bit,

maka isi register itu dikali dengan isi AL, kemudian disimpan di AX.

Bila register sumber adalah 16 bit, maka isi register itu dikali dengan isi AX,

kemudian hasilnya disimpan di DX:AX. Maksudnya, DX berisi high order byte-nya, AX berisi low order byte-nya.

IMUL : signed multiply. Sama dengan MUL,

hanya saja IMUL menganggap bit-bit yang ada di register sumber sudah dalam bentuk *two's complement*.

sintaks: IMUL {sumber}

DIV : divide. Membagi AX atau DX:AX dengan sebuah register.

sintaks: DIV {sumber} Bila register sumber adalah 8 bit (misalnya: BL), maka operasi yang terjadi: -AX dibagi BL,

-hasil bagi disimpan di AL, -sisa bagi disimpan di AH.

Bila register sumber adalah 16 bit (misalnya: CX), maka operasi yang terjadi: -DX:AX dibagi CX, -hasil bagi disimpan di AX, -sisa bagi disimpan di DX.

IDIV : signed divide. Sama dengan DIV, hanya saja IDIV menganggap bit-bit yang ada di register sumber sudah dalam bentuk *two's complement*.

sintaks: IDIV {sumber}

NEG : negate. Membuat isi register menjadi negatif (*two's complement*).

Bila mau *one's complement*, gunakan perintah NOT. sintaks: NEG {register} hasil disimpan di register itu sendiri.

Pengulangan

LOOP : loop. Mengulang sebuah proses. Pertama register CX dikurangi satu.

Bila CX sama dengan nol, maka looping berhenti. Bila tidak nol, maka lompat ke label tujuan.

sintaks: LOOP {label tujuan} Tips: isi CX dengan nol untuk mendapat jumlah pengulangan terbanyak.

Karena nol dikurang satu sama dengan -1, atau dalam notasi *two's complement* menjadi FFFF(hex) yang sama dengan 65535(dec).

LOOPE : loop while equal. Melakukan pengulangan selama $CX \neq 0$ dan $ZF = 1$. CX tetap dikurangi 1 sebelum diperiksa.

sintaks: LOOP {label tujuan}

LOOPZ : loop while zero. Identik dengan LOOPE.

LOOPNE : loop while not equal.

Melakukan pengulangan selama $CX \neq 0$ dan $ZF = 0$. CX tetap dikurangi 1 sebelum diperiksa.

sintaks: LOOPNE {label tujuan}

LOOPNZ : loop while not zero. Identik dengan LOOPNE.

REP : repeat. Mengulang perintah sebanyak CX kali. sintaks: REP {perintah assembly}

contoh:

mov CX, 05 rep inc BX ;register BX ditambah 1 sebanyak 5x.

REPE : repeat while equal. Mengulang perintah sebanyak CX kali, tetapi pengulangan segera dihentikan bila didapati $ZF = 1$.

sintaks: REPE {perintah assembly}

REPZ : repeat while zero. Identik dengan REPE.

REPNE : repeat while not equal. Mengulang perintah sebanyak CX kali, tetapi pengulangan segera dihentikan bila didapati $ZF = 0$.

sintaks: REPNE {perintah assembly}

REPNZ : repeat while not zero. Identik dengan REPNE.

Perbandingan

CMP : compare. Membandingkan dua buah operand. Hasilnya mempengaruhi sejumlah flag register.

sintaks: CMP {operand 1}, {operand 2}. Operand ini bisa register dengan register, register dengan isi memori, atau register dengan angka.

CMP tidak bisa membandingkan isi memori dengan isi memori.

Lompat-lompat

JMP: jump. Lompat tanpa syarat. Lompat begitu saja. sintaks: JMP {label tujuan}

Lompat bersyarat sintaksnya sama dengan JMP, yaitu perintah jump diikuti label tujuan.

PERINTAH	HARTI	SYARAT	KASUS	KETERANGAN ("OP" = OPERAND)	MENGIKUTI CMP?
JA	jump if above				
JNBE	jump if not below or equal	$CF = 0 \wedge$ $ZF = 0$	unsigned	lompat bila $op\ 1 > op\ 2$	ya
JB	jump if below				
JNAE	jump if not above or equal	$CF = 1 \wedge$ $ZF = 0$	unsigned	lompat bila $op\ 1 < op\ 2$	ya
JAE	jump if above or equal				
JNB	jump if not below	$CF = 0 \vee$ $ZF = 1$	unsigned	lompat bila $op\ 1 \geq op\ 2$	ya
JBE	jump if below or equal				
JNA	jump if not above	$CF = 1 \vee$ $ZF = 1$	unsigned	lompat bila $op\ 1 \leq op\ 2$	ya
JG	jump if greater				
JNLE	jump if not less or equal	$OF = 0 \wedge$ $ZF = 0$	signed	lompat bila $op\ 1 > op\ 2$	ya
JGE	jump if greater or equal				
JNL	jump if not less than	$OF = 0 \vee$ $ZF = 1$	signed	lompat bila $op\ 1 \geq op\ 2$	ya
JL	jump if less than		signed	lompat bila $op\ 1 < op\ 2$	ya

JNGE	jump if not greater or equal	$OF = 1 \wedge ZF = 0$			
JLE	jump if less or equal				
JNG	jump if not greater	$OF = 1 \vee ZF = 1$	signed	lompat bila $op\ 1 \leq op\ 2$	ya
JE	jump if equal	$ZF = 1$	keduanya	lompat bila $op\ 1 = op\ 2$	ya
JZ	jump if zero	$ZF = 1$	keduanya	lompat bila $op\ 1 = op\ 2$	ya
JNE	jump if not equal	$ZF = 0$	keduanya	lompat bila $op\ 1 \neq op\ 2$	ya
JNZ	jump if not zero	$ZF = 0$	keduanya	lompat bila $op\ 1 \neq op\ 2$	ya
JC	jump if carry	$CF = 1$	N/A	1	lompat bila carry flag = 1 tidak
JNC	jump if not carry	$CF = 0$	N/A	0	lompat bila carry flag = 0 tidak
JP	jump on parity			1	lompat bila parity flag = 1
JPE	jump on parity even	$PF = 1$	N/A	genap	lompat bila bilangan genap tidak selalu
JNP	jump on not parity			0	lompat bila parity flag = 0
JPO	jump on parity odd	$PF = 0$	N/A	ganjil	lompat bila bilangan ganjil tidak selalu
JO	jump if overflow	$OF = 1$	N/A	flag = 1	lompat bila overflow flag = 1 tidak

JNO	jump if not overflow	OF = 0	N/A	lompat bila overflow flag = 0	tidak
JS	jump if sign	SF = 1	N/A	lompat bila bilangan negatif	tidak
JCXZ	jump if CX is zero	CX = 0000	N/A	lompat bila CX berisi nol	tidak

Operasi stack

PUSH : push. Menambahkan sesuatu ke stack.

Sesuatu ini harus register berukuran 16 bit (pada 386+ harus 32 bit), tidak boleh angka, tidak boleh alamat memori.

Maka Anda tidak bisa mem-push register 8-bit seperti AH, AL, BH, BL, dan kawan-kawannya.

sintaks: push {register 16-bit sumber}

contoh: push DX push AX Setelah operasi push, register SP (stack pointer) otomatis dikurangi 2 (karena datanya 2 byte).

Makanya, “top” dari stack seakan-akan “tumbuh turun”.

POP : pop. Mengambil sesuatu dari stack.

Sesuatu ini akan disimpan di register tujuan dan harus 16-bit. Maka Anda tidak bisa mem-pop menuju AH, AL, dkk.

sintaks: POP {register 16-bit tujuan}

contoh: POP BX Setelah operasi pop, register SP otomatis ditambah 2 (karena 2 byte), sehingga “top” dari stack “naik” lagi.

Tip: karena register segmen tidak bisa diisi langsung nilainya, Anda bisa menggunakan stack sebagai perantaranya.

Contoh kodenya: mov AX, seg teks1 push AX pop DS

PUSHF : push flags. Mem-push **semua** isi register flag ke dalam stack.

Biasa dipakai untuk *membackup* data di register flag sebelum operasi matematika. Sintaks: PUSHF ;(saja).

POPF : pop flags. Lawan dari pushf. Sintaks: POPF ;(saja).

POPA : pop all general-purpose registers.

Adalah ringkasan dari sejumlah perintah dengan urutan:

pop DI pop SI pop BP pop SP pop BX pop DX pop CX pop AX

Urutan sudah ditetapkan seperti itu.

sintaks: POPA ;(saja). Jauh lebih cepat mengetikkan POPA daripada mengetik POP-POP-POP yang banyak itu.

PUSHA : push all general-purpose registers. Lawan dari POPA,

dimana PUSHA adalah singkatan dari sejumlah perintah dengan urutan yang sudah ditetapkan:

push AX push CX push DX push BX push SP push BP push SI push DI

Operasi pada register flag

CLC : clear carry flag. Menjadikan CF = 0. Sintaks: CLC ;(saja).

STC : set carry flag. Menjadikan CF = 1. Sintaks: STC ;(saja).

CMC : complement carry flag. Melakukan operasi NOT pada CF. Yang tadinya 0 menjadi 1, dan sebaliknya.

CLD : clear direction flag. Menjadikan DF = 0. Sintaks: CLD ;(saja).

STD : set direction flag. Menjadikan DF = 1.

CLI : clear interrupt flag. Menjadikan IF = 0, sehingga interrupt ke CPU akan di-disable.

Biasanya perintah CLI diberikan sebelum menjalankan sebuah proses penting yang riskan gagal bila diganggu.

STI : set interrupt flag. Menjadikan IF = 1.

Perintah lainnya

ORG : origin. Mengatur awal dari program (bagian static data).

Analoginya seperti mengatur dimana letak titik (0, 0) pada koordinat Cartesius.

sintaks: ORG {alamat awal}

Pada program COM (program yang berekstensi .com), harus ditulis “ORG 100h” untuk mengatur alamat mulai dari program pada 0100(hex),

karena dari alamat 0000(hex) sampai 00FF(hex) sudah dipesan oleh sistem operasi (DOS).

INT : interrupt. Menginterupsi prosesor.

Prosesor akan:

1. Membakup data registernya saat itu,
2. Menghentikan apa yang sedang dikerjakannya,
3. Melompat ke bagian interrupt-handler (entah dimana kita tidak tahu, sudah ditentukan BIOS dan DOS),
4. Melakukan interupsi,
5. Mengembalikan data registernya,
6. Meneruskan pekerjaan yang tadi ditunda.

sintaks: INT {nomor interupsi}

IRET : interrupt-handler return.

Kita bisa membuat interrupt-handler sendiri dengan berbagai cara.

Perintah IRET adalah perintah yang menandakan bahwa interrupt-handler kita selesai, dan prosesor boleh melanjutkan pekerjaan yang tadi tertunda.

CALL : call procedure. Memanggil sebuah prosedur.

sintaks: CALL {label nama prosedur}

RET : return. Tanda selesai prosedur.

Setiap prosedur harus memiliki RET di ujungnya.

sintaks: RET ;(saja)

HLT : halt. Membuat prosesor menjadi tidak aktif.

Sintaks: HLT ;(saja). **NOP** : no operation.

Perintah ini memakan 1 byte di memori tetapi tidak menyuruh prosesor melakukan apa-apa selama 3 clock prosesor.

Berikut contoh potongan program untuk melakukan *delay* selama 0,1 detik pada prosesor Intel 80386 yang berkecepatan 16 MHz.

mov ECX, 533333334d ;ini adalah bilangan desimal idle: nop loop idle