Notes for 8086 emulation core instruction decoding

General opcode format:

Instruction encoding byte ordering:

1. Prefix code(s) i.e. repetition, seg override, or lock (optional)
2. Opcode byte (required)
3. Mod-Reg-R/M addressing mode byte (optional)
4. Low disp/addr/data (optional)
5. High disp/addr/data (optional)
6. Low data (optional)
7. High data (optional)

Prefix codes:

Repetition:
- $F3 - REP/REPE/REPZ
- $F2 - REPNE/REPNZ

Segment override:
- $2E - CS
- $3H - DS (default for non-BP indexed memory addressing)
- $26 - ES
- $36 - SS (default for BP indexed memory addressing)

Other:
- $F0 - Lock

Segment prefix opcodes can force an instruction to use the specified segment base. If no segment prefix is present, the default segment DS is implied. However, if the BP register is used, the default will be SS rather than DS. A segment prefix will override that condition as well.

Addressing mode byte: (aka Mod-Reg-R/M byte)

Mod-Reg-R/M byte is present if either of the following references are used:

- General or Segment Register operand (G or S)
- Memory address

Layout of the byte:

```
Bit # | 7   6 | 5   4   3 | 2   1   0 |
      |  Mode |  (S)Reg   | R/M field |
```

Meaning of mode field values:

0 - Use R/M table 1 for operand with no displacement bytes, unless:

* If R/M table value is 6, then memory address is directly supplied by the 16-bit displacement value.

1 - Use R/M table 2 plus 8-bit displacement byte for operand, sign-extended to 16-bits.

2 - Use R/M table 2 plus 16-bit displacement word for operand.

3 - Operand is a second register. Use REG table.

Meaning of register field values:

```
    8-bit ops | 16-bit ops
0 -    AL     |     AX
1 -    CL     |     CX
2 -    DL     |     DX
3 -    BL     |     BX
4 -    AH     |     SP
5 -    CH     |     BP
6 -    DH     |     SI
7 -    BH     |     DI
```

Or, if an instruction implies using a segment register:

```
0 - ES
1 - CS
2 - SS
3 - DS
```

R/M table 1: (Mode = 0)

```
0 - SEG:[BX+SI]    2 - SEG:[BP+SI]    4 - SEG:[SI]    6 - Direct addressing*
1 - SEG:[BX+DI]      3 - SEG:[BP+DI]    5 - SEG:[DI]    7 - SEG:[BX]
* - Note: Direct addressing means SEG:[disp] where disp is 16-bit unsigned
```

R/M table 2: (Mode = 1 or Mode = 2; add displacement value to these)

```
0 - SEG:[BX+SI]    2 - SEG:[BP+SI]    4 - SEG:[SI]    6 - SEG:[BP]
1 - SEG:[BX+DI]      3 - SEG:[BP+DI]    5 - SEG:[DI]    7 - SEG:[BX]
```

IMPORTANT NOTE When referencing R/M table 2, the displacement byte is treated as a **sign-extended** 8-bit signed value when addr. mode = 1. If addr. mode = 2, then there is instead a two-byte unsigned displacement word.

Appendix A:
8086 hex opcode listing in numerical order, including mnemonics.

```
00      ADD             Eb      Gb
01      ADD             Ev      Gv
02      ADD             Gb      Eb
03      ADD             Gv      Ev
04      ADD             AL      Ib
05      ADD             eAX     Iv
06      PUSH    ES
07      POP             ES
08      OR              Eb      Gb
09      OR              Ev      Gv
0A      OR              Gb      Eb
0B      OR              Gv      Ev
0C      OR              AL      Ib
0D      OR              eAX     Iv
0E      PUSH    CS
0F      --
10      ADC             Eb      Gb
11      ADC             Ev      Gv
12      ADC             Gb      Eb
13      ADC             Gv      Ev
14      ADC             AL      Ib
15      ADC             eAX     Iv
16      PUSH    SS
17      POP             SS
18      SBB             Eb      Gb
19      SBB             Ev      Gv
1A      SBB             Gb      Eb
1B      SBB             Gv      Ev
1C      SBB             AL      Ib
1D      SBB             eAX     Iv
1E      PUSH    DS
1F      POP             DS
20      AND             Eb      Gb
21      AND             Ev      Gv
22      AND             Gb      Eb
23      AND             Gv      Ev
24      AND             AL      Ib
25      AND             eAX     Iv
26      ES:
27      DAA
28      SUB             Eb      Gb
29      SUB             Ev      Gv
2A      SUB             Gb      Eb
2B      SUB             Gv      Ev
2C      SUB             AL      Ib
2D      SUB             eAX     Iv
2E      CS:
2F      DAS
30      XOR             Eb      Gb
31      XOR             Ev      Gv
32      XOR             Gb      Eb
33      XOR             Gv      Ev
34      XOR             AL      Ib
35      XOR             eAX     Iv
36      SS:
37      AAA
38      CMP             Eb      Gb
39      CMP             Ev      Gv
3A      CMP             Gb      Eb
3B      CMP             Gv      Ev
3C      CMP             AL      Ib
3D      CMP             eAX     Iv
3E      DS:
3F      AAS
40      INC             eAX
41      INC             eCX
42      INC             eDX
```

```
43      INC             eBX
44      INC             eSP
45      INC             eBP
46      INC             eSI
47      INC             eDI
48      DEC             eAX
49      DEC             eCX
4A      DEC             eDX
4B      DEC             eBX
4C      DEC             eSP
4D      DEC             eBP
4E      DEC             eSI
4F      DEC             eDI
50      PUSH    eAX
51      PUSH    eCX
52      PUSH    eDX
53      PUSH    eBX
54      PUSH    eSP
55      PUSH    eBP
56      PUSH    eSI
57      PUSH    eDI
58      POP             eAX
59      POP             eCX
5A      POP             eDX
5B      POP             eBX
5C      POP             eSP
5D      POP             eBP
5E      POP             eSI
5F      POP             eDI
60      --
61      --
62      --
63      --
64      --
65      --
66      --
67      --
68      --
69      --
6A      --
6B      --
6C      --
6D      --
6E      --
6F      --
70      JO              Jb
71      JNO             Jb
72      JB              Jb
73      JNB             Jb
74      JZ              Jb
75      JNZ             Jb
76      JBE             Jb
77      JA              Jb
78      JS              Jb
79      JNS             Jb
7A      JPE             Jb
7B      JPO             Jb
7C      JL              Jb
7D      JGE             Jb
7E      JLE             Jb
7F      JG              Jb
80      GRP1    Eb      Ib
81      GRP1    Ev      Iv
82      GRP1    Eb      Ib
83      GRP1    Ev      Ib
84      TEST    Gb      Eb
85      TEST    Gv      Ev
86      XCHG    Gb      Eb
87      XCHG    Gv      Ev
88      MOV             Eb      Gb
89      MOV             Ev      Gv
8A      MOV             Gb      Eb
```

```
8B      MOV             Gv      Ev
8C      MOV             Ew      Sw
8D      LEA             Gv      M
8E      MOV             Sw      Ew
8F      POP             Ev
90      NOP
91      XCHG    eCX eAX
92      XCHG    eDX eAX
93      XCHG    eBX eAX
94      XCHG    eSP eAX
95      XCHG    eBP eAX
96      XCHG    eSI eAX
97      XCHG    eDI eAX
98      CBW
99      CWD
9A      CALL    Ap
9B      WAIT
9C      PUSHF
9D      POPF
9E      SAHF
9F      LAHF
A0      MOV             AL      Ob
A1      MOV             eAX     Ov
A2      MOV             Ob      AL
A3      MOV             Ov      eAX
A4      MOVSB
A5      MOVSW
A6      CMPSB
A7      CMPSW
A8      TEST    AL      Ib
A9      TEST    eAX     Iv
AA      STOSB
AB      STOSW
AC      LODSB
AD      LODSW
AE      SCASB
AF      SCASW
B0      MOV             AL      Ib
B1      MOV             CL      Ib
B2      MOV             DL      Ib
B3      MOV             BL      Ib
B4      MOV             AH      Ib
B5      MOV             CH      Ib
B6      MOV             DH      Ib
B7      MOV             BH      Ib
B8      MOV             eAX     Iv
B9      MOV             eCX     Iv
BA      MOV             eDX     Iv
BB      MOV             eBX     Iv
BC      MOV             eSP     Iv
BD      MOV             eBP     Iv
BE      MOV             eSI     Iv
BF      MOV             eDI     Iv
C0      --
C1      --
C2      RET             Iw
C3      RET
C4      LES             Gv      Mp
C5      LDS             Gv      Mp
C6      MOV             Eb      Ib
C7      MOV             Ev      Iv
C8      --
C9      --
CA      RETF    Iw
CB      RETF
CC      INT             3
CD      INT             Ib
CE      INTO
CF      IRET
D0      GRP2    Eb      1
D1      GRP2    Ev      1
D2      GRP2    Eb      CL
```

D3	GRP2	Ev	CL	
D4	AAM		I0	
D5	AAD		I0	
D6	--			
D7	XLAT			
D8	--			
D9	--			
DA	--			
DB	--			
DC	--			
DD	--			
DE	--			
DF	--			
E0	LOOPNZ	Jb		
E1	LOOPZ	Jb		
E2	LOOP	Jb		
E3	JCXZ	Jb		
E4	IN		AL	Ib
E5	IN		eAX	Ib
E6	OUT		Ib	AL
E7	OUT		Ib	eAX
E8	CALL	Jv		
E9	JMP		Jv	
EA	JMP		Ap	
EB	JMP		Jb	
EC	IN		AL	DX
ED	IN		eAX	DX
EE	OUT		DX	AL
EF	OUT		DX	eAX
F0	LOCK			
F1	--			
F2	REPNZ			
F3	REPZ			
F4	HLT			
F5	CMC			
F6	GRP3a	Eb		
F7	GRP3b	Ev		
F8	CLC			
F9	STC			
FA	CLI			
FB	STI			
FC	CLD			
FD	STD			
FE	GRP4	Eb		
FF	GRP5	Ev		

Appendix B: Mnemonics for "GRP" opcodes

```
GRP1/0  ADD
GRP1/1  OR
GRP1/2  ADC
GRP1/3  SBB
GRP1/4  AND
GRP1/5  SUB
GRP1/6  XOR
GRP1/7  CMP
GRP2/0  ROL
GRP2/1  ROR
GRP2/2  RCL
GRP2/3  RCR
GRP2/4  SHL
GRP2/5  SHR
GRP2/6  --
GRP2/7  SAR
GRP3a/0TEST     Eb      Ib
GRP3a/1--
GRP3a/2NOT
GRP3a/3NEG
GRP3a/4MUL
GRP3a/5IMUL
GRP3a/6DIV
GRP3a/7IDIV
GRP3b/0TEST     Ev      Iv
GRP3b/1--
GRP3b/2NOT
GRP3b/3NEG
GRP3b/4MUL
GRP3b/5IMUL
GRP3b/6DIV
GRP3b/7IDIV
GRP4/0  INC
GRP4/1  DEC
GRP4/2  --
GRP4/3  --
GRP4/4  --
GRP4/5  --
GRP4/6  --
GRP4/7  --
GRP5/0  INC
GRP5/1  DEC
GRP5/2  CALL
GRP5/3  CALL    Mp
GRP5/4  JMP
GRP5/5  JMP             Mp
GRP5/6  PUSH
GRP5/7  --
```

Appendix C: Argument Addressing Codes

A = Direct address. The instruction has no ModR/M byte; the address of the operand is encoded in the instruction. Applicable, e.g., to far JMP (opcode EA).

E = A ModR/M byte follows the opcode and specifies the operand. The operand is either a general-purpose register or a memory address. If it is a memory address, the address is computed from a segment register and any of the following values: a base register, an index register, a displacement.

G = The reg field of the ModR/M byte selects a general register.

I = Immediate data. The operand value is encoded in subsequent bytes of the instruction.

J = The instruction contains a relative offset to be added to the address of the subsequent instruction. Applicable, e.g., to short JMP (opcode EB), or LOOP.

M = The ModR/M byte may refer only to memory. Applicable, e.g., to LES and LDS.

O = The instruction has no ModR/M byte; the offset of the operand is encoded as a WORD in the instruction. Applicable, e.g., to certain MOVs (opcodes A0 through A3).

S = The reg field of the ModR/M byte selects a segment register.

Appendix D: Argument Operand Codes:

0 = Byte argument. Unusual in that arguments of this type are suppressed in ASM output when they have the default value of 10 (0xA). Applicable, e.g., to AAM and AAD.

b = Byte argument.

p = 32-bit segment:offset pointer.

w = Word argument.

v = Word argument. (The 'v' code has a more complex meaning in later x86 opcode maps, from which this was derived, but here it's just a synonym for the 'w' code.)