x86 Addressing Modes

The addressing mode determines, for an instruction that accesses a memory location, how the address for the memory location is specified.

To summarize, the following diagram (from http://en.wikipedia.org/wiki/X86#Addressing_modes) shows the possible ways an address could be specified:

\[
\begin{align*}
\left[ \begin{array}{c}
EAX \\
EBX \\
ECX \\
EDX \\
ESP \\
EBP \\
ESI \\
EDI \\
\end{array} \right] 
= \left[ \begin{array}{c}
EAX \\
EBX \\
ECX \\
EDX \\
EBP \\
ESI \\
EDI \\
\end{array} \right] 
+ \left( \begin{array}{c}
1 \\
2 \\
4 \\
8 \\
\end{array} \right) \times \text{[displacement]} 
\end{align*}
\]

Each square bracket in the above diagram indicates an optional part of the address specification. These parts (from left to right) are: A register used as a base address, a register used as an index, a width (or scale) value to multiply the register by, and a displacement (aka offset) which is an integer. The address is computed as the sum of: the base register, the index times the width, and the displacement.

The Intel & AT&T syntax for various addressing modes, depending on which parts of the above diagram are used, is shown in the table below from http://simon.baymoo.org/universe/tools/symset/symset.txt (slightly modified):

<table>
<thead>
<tr>
<th>Mode</th>
<th>Intel</th>
<th>AT&amp;T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absolute</td>
<td>MOV EAX, [0100]</td>
<td>movl 0x0100, %eax</td>
</tr>
<tr>
<td>Register</td>
<td>MOV EAX, [ESI]</td>
<td>movl (%esi), %eax</td>
</tr>
<tr>
<td>Reg + Off</td>
<td>MOV EAX, [EBP-8]</td>
<td>movl -8(%ebp), %eax</td>
</tr>
<tr>
<td>R*W + Off</td>
<td>MOV EAX, [EBX*4 + 0100]</td>
<td>movl 0x100(%ebx,4), %eax</td>
</tr>
<tr>
<td>B + R*W + O</td>
<td>MOV EAX, [EDX + EBX*4 + 8]</td>
<td>movl 0x8(%edx,%ebx,4), %eax</td>
</tr>
</tbody>
</table>

Note that, given the definition of a label \( x \) in the .data section of an assembly program, using \( x \) to indicate the memory location, as in

\texttt{mov eax, }x \quad \#\text{Intel}

or

\texttt{mov }x,\%eax \quad \#\text{AT&T}

is just absolute addressing (i.e. using just a displacement), where the assembler essentially replaces the name \( x \) with the address corresponding to \( x \).