7.3.5 Signed Integer Division

Signed integer division is nearly identical to unsigned division, with one important difference: The dividend must be sign-extended before the division takes place. Sign extension is the term used for copying the highest bit of a number into all of the upper bits of its enclosing variable or register. To show why this is necessary, let's try leaving it out. The following code uses MOV to assign 101 to AX, which is the lower half of DX:AX:

```
.data
wordVal SWORD -101      ; FF9Bh
.code
mov  dx,0
mov  ax,wordVal         ; DX:AX = 0000FF9Bh
mov  bx,2               ; BX is the divisor
idiv bx                 ; divide DX:AX by BX (signed operation)
```

Unfortunately, the 0000FF9Bh in DX:AX equals +65,435, so the quotient produced by the division is incorrect. Instead, the correct way to set up the problem is to use the CWD instruction (convert word to doubleword), which sign-extends AX into EAX before performing the division:

```
.data
wordVal SWORD -101      ; 009Bh
.code
mov  dx,0
mov  ax,wordVal         ; DX:AX = 0000009Bh (+155)
cwd                     ; DX:AX = FFFFFF9Bh (-101)
mov  bx,2
idiv bx
```

We introduced the concept of sign extension in Chapter 4 along with the MOVSX instruction. The x86 instruction set includes several instructions for sign extension. First, we will look at these instructions, and then we will apply them to the signed integer division instruction, IDIV.