C provides six operators for bit manipulation; these may only be applied to integral operands, that is, char, short, int, and long, whether signed or unsigned. & bitwise AND | bitwise inclusive OR ^ bitwise exclusive OR « left shift » right shift ~ one's complement (unary) The bitwise AND operator & is often used to mask off some set of bits, for example n = n & 0177; sets to zero all but the low-order 7 bits of n. The bitwise OR operator | is used to turn bits on: x = x | SET_ON; sets to one in x the bits that are set to one in SET_ON. The bitwise exclusive OR operator ^ sets a one in each bit position where its operands have different bits, and zero where they are the same. One must distinguish the bitwise operators & and | from the logical operators && and ||, which imply left-to-right evaluation of a truth value. For example, if x is 1 and y is 2, then x & y is zero while x && y is one. The shift operators « and » perform left and right shifts of their left operand by the number of bit positions given by the right operand, which must be non-negative. Thus x « 2 shifts the value of x by two positions, filling vacated bits with zero; this is equivalent to multiplication by 4. Right shifting an unsigned quantity always fits the vacated bits with zero. Right shifting a signed quantity will fill with bit signs ("arithmetic shift") on some machines and with 0-bits ("logical shift") on others. The unary operator ~ yields the one's complement of an integer; that is, it converts each 1-bit into a 0-bit and vice versa. For example x = x & ~077 sets the last six bits of x to zero. Note that x & ~077 is independent of word length, and is thus preferable to, for example, x & 0177700, which assumes that x is a 16-bit quantity. The portable form involves no extra cost, since ~077 is a constant expression that can be evaluated at compile time. As an illustration of some of the bit operators, consider the function getbits(x,p,n) that returns the (right adjusted) n-bit field of x that begins at position p. We assume that bit position 0 is at the right end and that n and p are sensible positive values. For example, getbits(x,4,3) returns the three bits in positions 4, 3 and 2, right-adjusted.