

## Shift Operations

In a shift operation, all bits of the operand are moved one or more places to the left or right, subject to the variations described below. The 68000 is particularly well endowed with shift operations.

All shifts can be categorized as logical, arithmetic, or circular. In a logical shift, a 0 enters at the input of the shifter and the bit shifted out is clocked into the carry flip-flop. An arithmetic shift left is identical to a logical shift left, but an arithmetic shift right causes the most significant bit, the sign bit, to be propagated right. This action preserves the correct sign of a two's complement value. For example, if the bytes 00101010 and 10101010 are shifted one place to the right (arithmetically), the results are 00010101 and 11010101, respectively. In a circular shift, the bit shifted out is moved to the position of the bit shifted in. No bit is lost during a circular shift. In the figure note that an arithmetic shift left and a logical shift left operation are virtually identical. In each case, all the bits are shifted one place left. The bit shifted out enters the carry bit and extend bit of the CCR and a 0 enters the vacated position (the least significant bit). There is, however, one small difference between an ASL and an LSL. Since an arithmetic left shift multiplies a number by 2, it is possible for the most significant bit of the value being shifted to change sign and therefore generate an arithmetic overflow. The V bit of the CCR is set if this event occurs during an ASL. Since logical operations are applied to strings of bits, an LSL instruction clears the V bit (since arithmetic overflow is meaningless). The 68000 has eight shift operations in its instruction set, illustrated in the figure the symbol C denotes the carry bit of the condition code register; X means the extend bit of the CCR.

Arithmetic shifts update *all* bits of the CCR. The N and Z bits are set or cleared as we would expect. The V bit is set if the most significant bit of the operand is changed at any time during the shift operation. The C and X bits are set according to the last bit shifted out of the operand. However, if the shift count is zero, C is cleared and X is unaffected. Logical shifts and rotates clear the V bit.

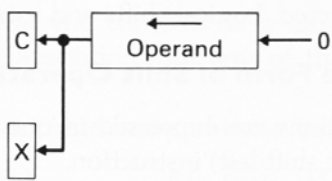
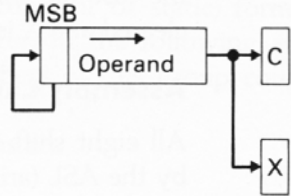
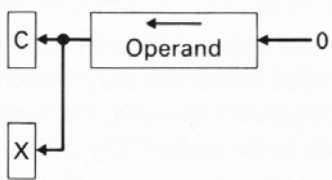
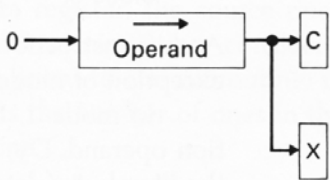
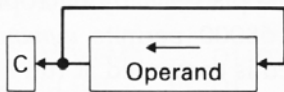
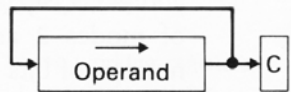
## Assembly Language Form of Shift Operations

All eight shift instructions are expressed in one of three ways. These are illustrated by the ASL (arithmetic shift left) instruction.

Mode 1.	ASL Dx,Dy	Shift Dy by Dx bits
Mode 2.	ASL #<data>,Dy	Shift Dy by # data bits
Mode 3.	ASL <ea>	Shift the contents of ea by one place

A shift instruction can be applied to a byte, word, or longword operand, with the exception of mode 3 shifts, which act only on words. In mode 1, the "source" operand, Dx, specifies how many places the destination operand, Dy, needs to be shifted. Dy may be shifted by 1 to 32 bits. In mode 2, the literal, +<data>, specifies

how many places  $D_y$  needs to be shifted; this must be in the range 1 to 8. In mode 3, the memory location specified by the effective address,  $\langle ea \rangle$ , is shifted one place. Many microprocessors permit only the *static* shifts of modes 2 and 3. The 68000 permits *dynamic* shifts (i.e., mode 1) because the number of bits to be shifted is computed at run-time.

Class	Left shift	Right shift
ASL, arithmetic shift left ASR, arithmetic shift right		
LSL, logical shift left LSR, logical shift right		
ROL, rotate left ROR, rotate right  Note X bit not affected		
ROXL, rotate left through extend ROXR, rotate right through extend	