DEFINING LOGIC INSTRUCTIONS

Logic operations such as AND, OR, and NOT, applied to individual bits, are the basic building blocks of digital circuits, as described. It is also useful to be able to perform logic operations is software, which is done using instructions that apply these operations to all bits of a word or byte independently and in parallel. For example, the instruction

Not dst

SHIFT AND ROTATE INSTRUCTIONS:-

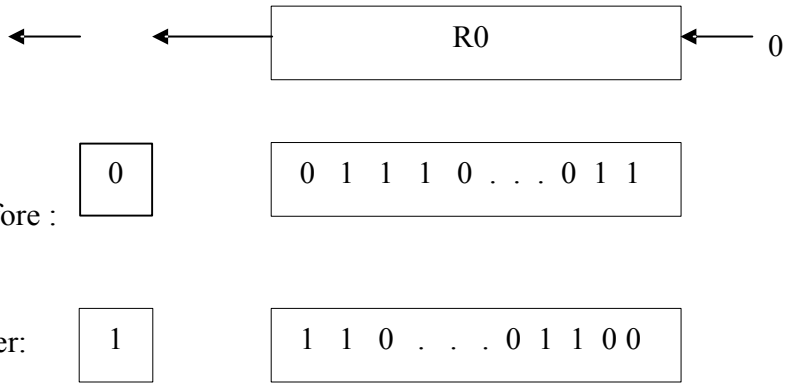
There are many applications that require the bits of an operand to be shifted right or left some specified number of bit positions. The details of how the shifts are performed depend on whether the operand is a signed number or some more general binary-coded information. For general operands, we use a logical shift. For a number, we use an arithmetic shift, which preserves the sign of the number.

Logical shifts:-

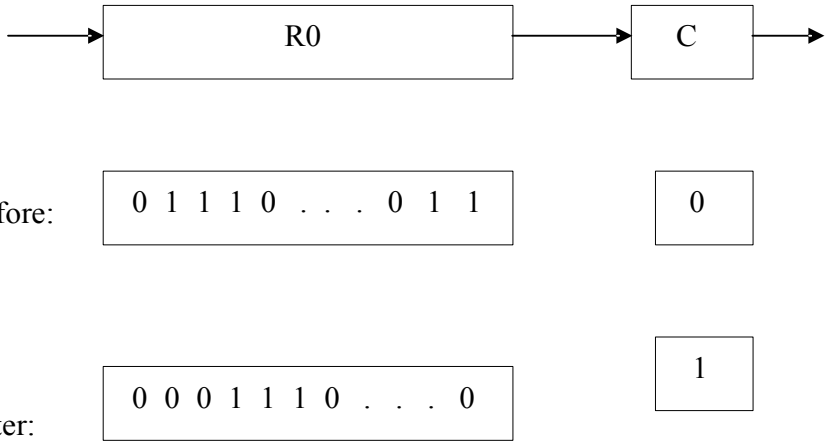
Two logical shift instructions are needed, one for shifting left (LShiftL) and another for shifting right (LShiftR). These instructions shift an operand over a number of bit positions specified in a count operand contained in the instruction. The general form of a logical left shift instruction is

LShiftL count, dst

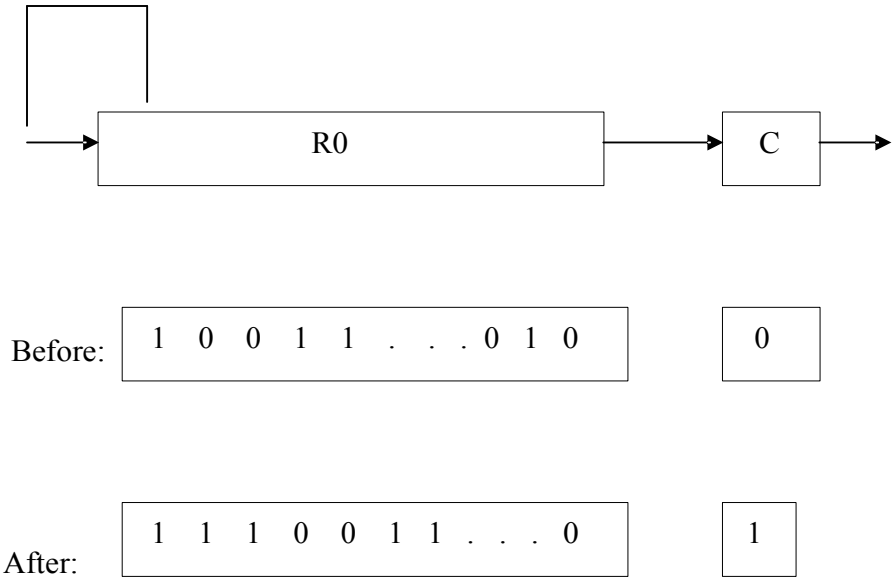
(a) Logical shift left LShiftL #2, R0



(b) Logical shift right LShiftR #2, R0



(c) Arithmetic shift right AShiftR #2, R0

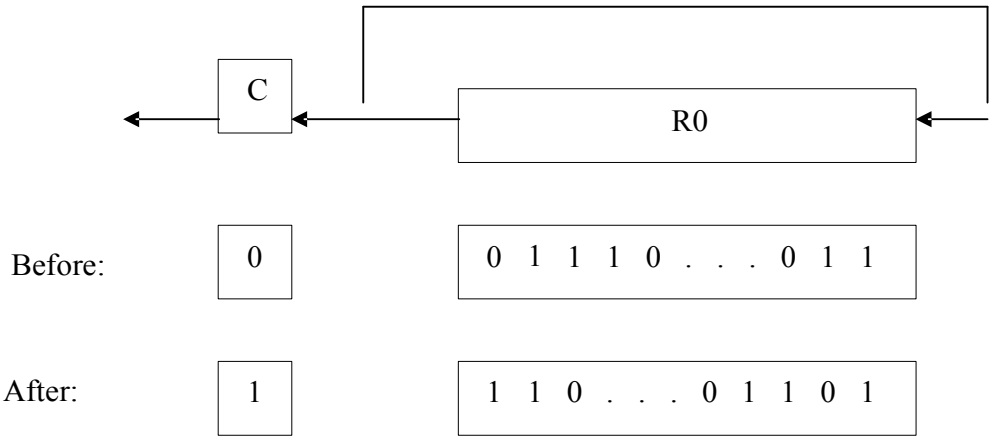


Rotate Operations:-

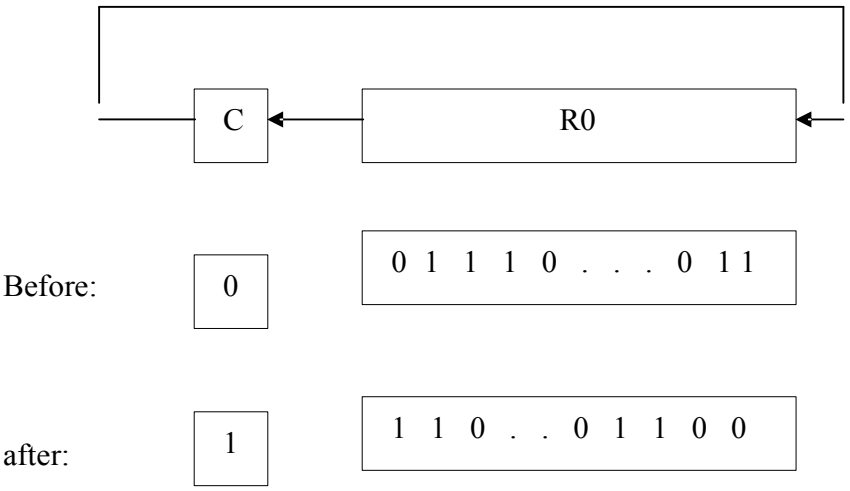
In the shift operations, the bits shifted out of the operand are lost, except for the last bit shifted out which is retained in the Carry flag C. To preserve all bits, a set of rotate instructions can be used. They move the bits that are shifted out of one end of the operand back into the other end. Two versions of both the left and right rotate instructions

are usually provided. In one version, the bits of the operand are simply rotated. In the other version, the rotation includes the C flag.

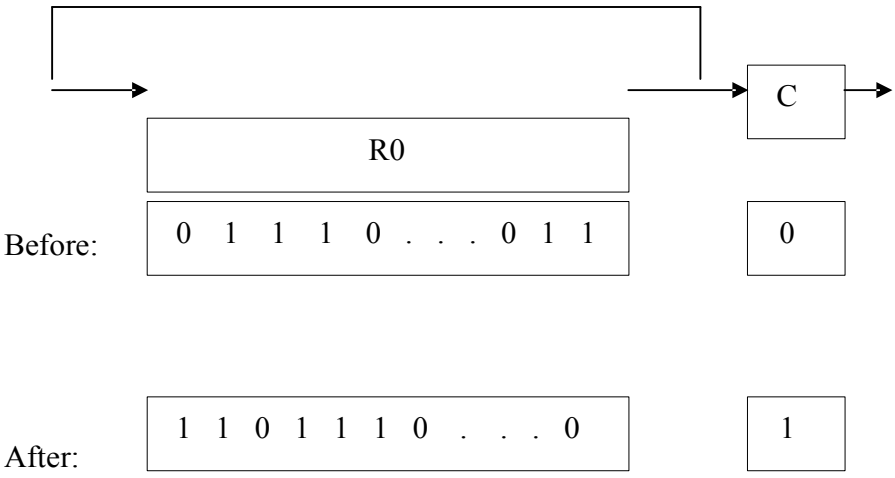
(a) Rotate left without carry RotateL #2, R0



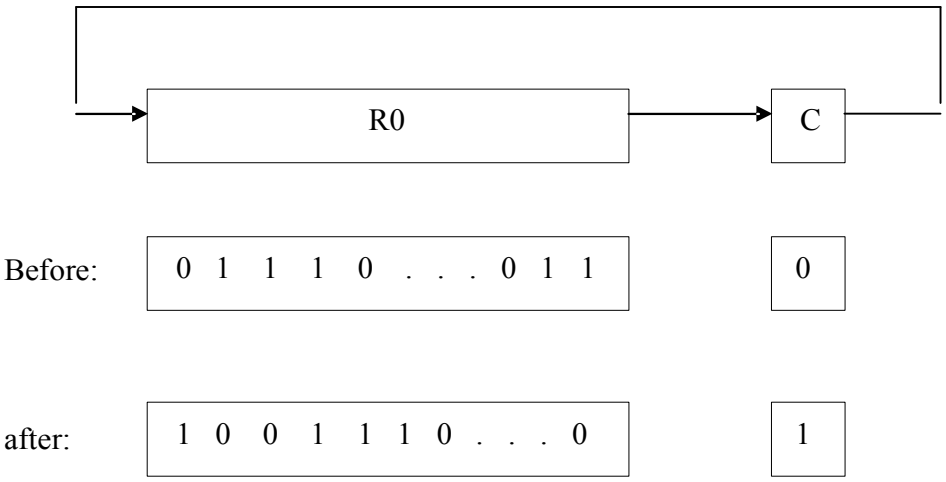
(b) Rotate left with carry RotateLC #2, R0



(c) Rotate right without carry RotateR #2, R0



(d) Rotate right with carry RotateRC #2, R0



Source : http://elearningatria.files.wordpress.com/2013/10/cse-iv-computer-organization-10cs46-notes.pdf