**COAL LAB 06**

**---------------------------------------------------------**

**Name: Abubaker**

**Roll no: P20-0560**

**=========================================================**

**MULTIPLICATION ALGORITHM**

The multiplication method we used to do in early classes is same as our computer does i.e. multiplying by a digit and then crossing it, multiplying by the next digit and crossing it again, and so on, totaling the intermediate values at the end. The procedure is well-known, however. We've never thought of the process as an algorithm, but we need to. To get it to the CPU, we'll need to use an algorithm. It is also used to speed up the performance of the multiplication process. It is very efficient too.

1101

0101

----------

1101

0000x

1101xx

0000xxx

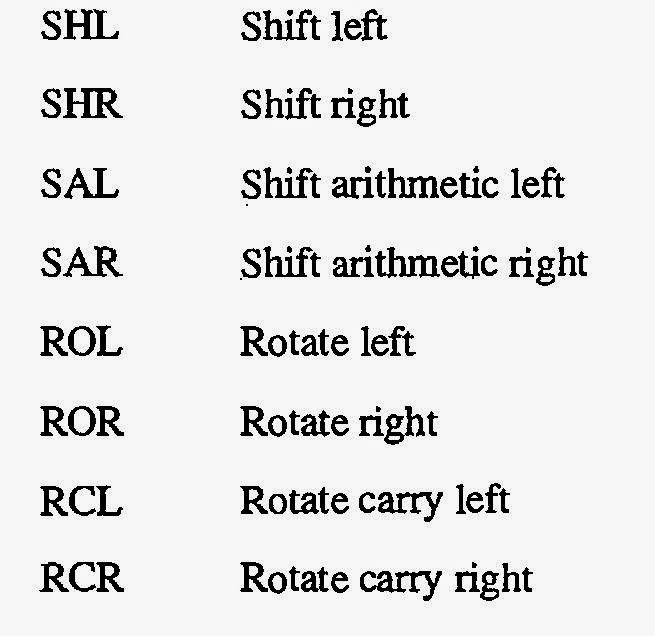
---------------

01000001 = 65

**SHIFTING AND ROTATIONS**

The shift operations allow bits to be moved to the left or right in a word. There are three types of shift operations: logical, rotate and arithmetic.

The major Shift Instructions use for bit shifting.



**SHL**

The shift logical right operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag.

**SHL and SAL**

The shift logical left operation is the exact opposite of shift logical right. In this operation the zero bit is inserted from the right and every bit moves one position to its left with the most significant bit dropping into the carry flag.

**SAR**

A Right Arithmetic Shift of one position moves each bit to the right by one. The least significant bit is discarded and the vacant MSB is filled with the value of the previous (now shifted one position to the right) MSB.

**ROR**

The right rotate instruction shifts all bits in the register or memory operand specified. The least significant bit is rotated to the carry flag, the carry flag is rotated to the most significant bit position, all other bits are shifted to the right.

**ROL**

In the operation of rotate left instruction, the most significant bit is copied to the carry flag and is inserted from the right, causing every bit to move one position to the left. It is the reverse of the rotate right instruction

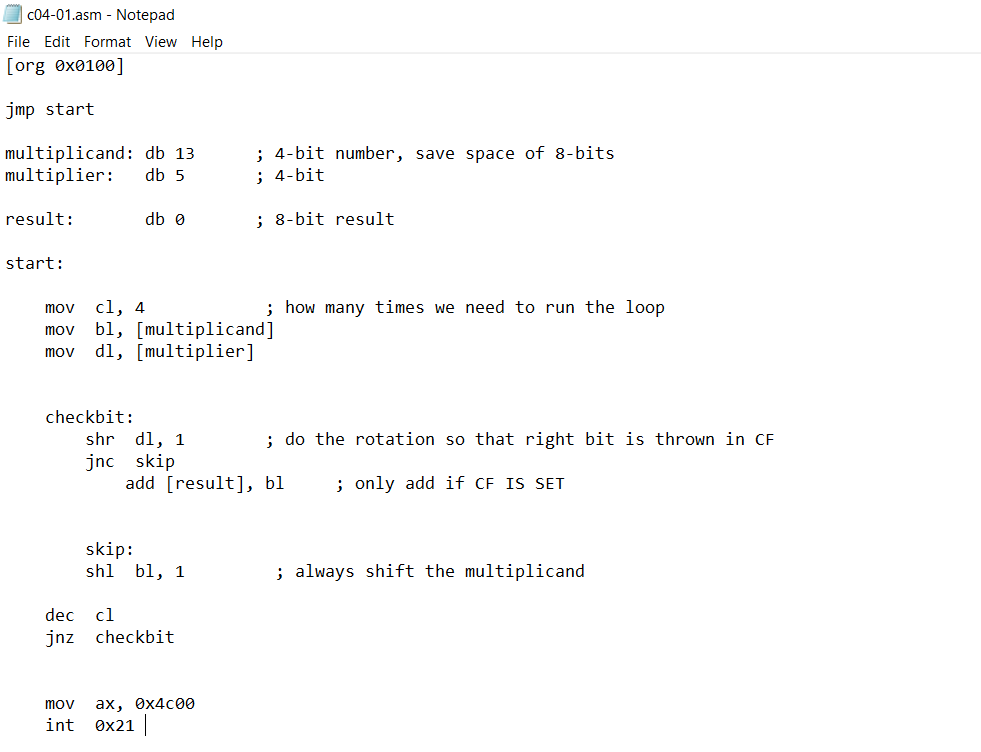
**RCR**

In the rotate through carry right instruction, the carry flag is inserted from the left, every bit moves one position to the right, and the right most bit is dropped in the carry flag. Effectively this is a nine bit or a seventeen-bit rotation instead of the eight- or sixteen-bit rotation as in the case of simple rotations

**RCL**

The exact opposite of rotate through carry right instruction is the rotate through carry left instruction. In its operation the carry flag is inserted from the right causing every bit to move one location to its left and the most significant bit occupying the carry flag.

**Code:**



**AFD Window**