**Reset Accumulator (8085 & 8086 microprocessor)**

* Difficulty Level : [Basic](https://www.geeksforgeeks.org/basic/)
* Last Updated : 22 May, 2018

**1. Problem –** Write the 8085 instructions which reset the accumulator.

**Solution –** There are 4 instructions to reset the accumulator in 8085. These instructions are:

Attention reader! Don’t stop learning now. Get hold of all the important CS Theory concepts for SDE interviews with the [**CS Theory Course**](https://practice.geeksforgeeks.org/courses/SDE-theory?vC=1) at a student-friendly price and become industry ready.

| S.No. | MNEMONICS | COMMENT |
| --- | --- | --- |
| 1 | MVI A, 00 | A <- 00 |
| 2 | ANI 00 | A AND 00 |
| 3 | XRA A | A XOR A |
| 4 | SUB A | A <- A – A |

**Explanation –**

1. **MVI A, 00:** instruction copies 00 to A.
2. **ANI 00:** instruction performs bit by bit AND operation of source operand (i.e. 00) to the destination operand (i.e. the accumulator A) and store the result in accumulator A.
3. **XRA A:** instruction performs XOR operation between source operand and destination operand and store the result in the accumulator. Here, source and destination operand both are same i.e. A. Therefore, the result after performing XOR operation, stored in the accumulator is 00.
4. **SUB A:** operation subtracts the contents of source operand(here, source register is A) from the contents of accumulator and store the result in the accumulator itself. Since, the source and destination operand are same. Therefore, accumulator A = 00.

**2. Problem –** Write 8086 instructions which are used to reset accumulator.

**Solution –** There are 4 instructions in 8086 to reset the accumulator. These instructions are:

| S.No. | MNEMONICS | COMMENT |
| --- | --- | --- |
| 1 | MOV AX, 0000 | AX <- 0000 |
| 2 | AND AX, 0000 | AX <- AX AND 0000 |
| 3 | XOR AX, AX | AX <- AX XOR AX |
| 4 | SUB AX, AX | AX <- AX – AX |

**Explanation –** Register AX is used.

1. **MOV AX, 0000:** copies 0000 to AX.
2. **AND AX, 0000:** operation performs bit by bit ANDs the source operand (0000) to the destination operand and store the result in AX.
3. **XOR AX, AX:** performs the XOR operation in values of source register and destination register and store the result in AX. The source and destination operands, both are same. Therefore, AX = 0.
4. **SUB AX, AX:** operation subtracts the value of source operand from the value of destination operand and store the result in AX. Here, both the operands are same .Therefore, AX = 0.