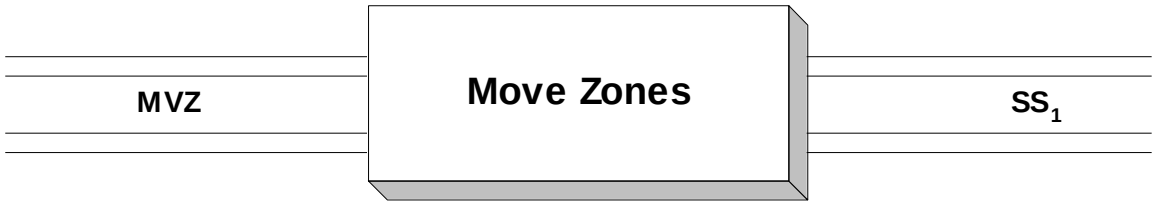


Op Code	LL_1	B_1D_1	D_1D_1	B_2D_2	D_2D_2

MVZ perfoms in a way that is analagous to **MVC**. While **MVC** works on entires bytes, **MVZ** only processes the zoned parts (leftmost 4 bits) of the bytes it references. The purpose of a move zones instruction is to move the zoned parts of a consecutive collection of bytes from one location in memory to another location. As you can see from the instruction format above, the instruction carries with it the number of bytes to be copied (LL_1), as well as the beginning addresses of the source ($B_2D_2D_2D_2$) and target ($B_1D_1D_1D_1$) fields. Notice that the instruction does not specifiy the ending addresses of either field - the instruction is no respecter of fields. **MVZ** copies the zoned parts of LL_1 + 1 consecutive bytes from the storage location designated by $B_2D_2D_2D_2$ to the storage location designated by $B_1D_1D_1D_1$.

The length (LL_1) determines the number of “half-bytes” which will be copied. The length is usually determined implicitly from the length of operand 1 but the programmer can provide an explicit length. Consider the two example MVZ’s below,

Object code Assembler code

```
                FIELDA    DS        CL8
                FIELDB    DS        CL5
                          ...
D307C008C010              MVZ     FIELDA,FIELDB     Implicit length
D302C008C010              MVZ     FIELDA(3),FIELDB  Explicit length
```

In the first example, the length implicitly defaults to 8, the length of FIELDA. In the second example, the length is explicitly 3. Notice that the assembled length (LL_1) is one less than the implicit or explicit length. This can be seen in the object code above where the assembled lengths are x’07’ and x’02’.

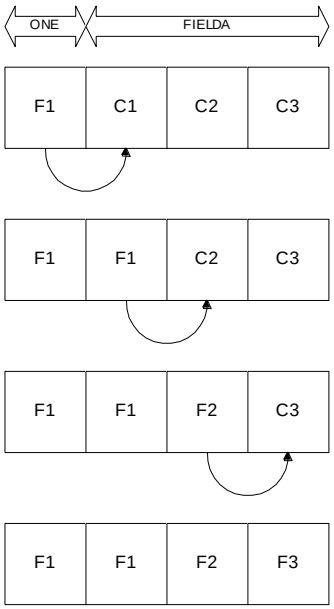
The copying operation is usually straightforward, but can be more complicated by overlapping the source and target fields. Keep in mind that the copy is made one byte at a time. Consider the following examples,

Object code Assembler code

```
                ONE       DC    C'1'         ONE = X'F1'
                FIELDA    DC    CL3'ABC'     FIELDA = X'C1C2C3'
                FIELDB    DC    XL3'123456'  FIELDB = X'123456'
...
  D302C008C00B            MVZ   FIELDA,FIELDB After FIELDA = X'113253'
  D302C00EC008            MVZ   FIELDB,FIELDA After FIELDB = X'C2C4C6'
  D302C008C007            MVZ   FIELDA,ONE    After FIELDA = X'F1F2F3'
```

In the first **MVZ** above, 3 consecutive zoned half-bytes in FIELDB are simply copied to the zoned portions of FIELDA. The half-bytes are copied, one at a time moving left to right within both operands. In the second example, 3 consecutive bytes are copied into FIELDB (implicit length = 3) from FIELDA. The third **MVZ** is complicated by the fact that the source and target fields overlap. We will examine the third move in some detail.

```
                                MVZ    FIELDA,ONE    THIS IS A 3 BYTE MOVE
```



First half-byte of source copied to first half-byte of target.

Second half-byte of source copied to second half-byte of target.

Third half-byte of source copied to third half-byte of target.

Examples

Some Unrelated MVZ's:

```
   A          DC    X'123456'
   B          DC    X'ABCDEF'
   C          DC    X'A1B2'
                    ...              Result:
              MVZ   A,B              A = X'A2C4E6'  B = X'ABCDEF'
              MVZ   A+1,B            A = X'12A4C6'  B = X'EBCDEF'
              MVZ   A+1(2),B         A = X'12A4C6'  B = X'ABCDEF'
              MVZ   B,=X'D1E2'       B = X'DBED?F'  One byte copied from
                                                    the literal pool
              MVZ   B,B+1            B = 'CBEDAF'   Left shift
              MVZ   B+1(2),B         B = 'ABADAF'   1st byte is propagated
              MVZ   C,A              C = '1132'     A = X'123456'
              MVZ   A(L'C),C         A = 'A2B456' Explicit Length
attribute
              MVZ   A(1000),B    Assembly Error - max length is 256
bytes
              MVZ   A,B(20)      Assembly Error - Op-1 determines length
```

Tips

1. Pay attention to the lengths of the fields involved in any **MVZ** statement. If the target field is longer than the source field, bytes following the source may be transferred. If the target field is shorter than the source field, bytes in the source may may be truncated.