Op Code	L_1L_2	B_1D_1	D_1D_1	B_2D_2	D_2D_2

SP is a SS_2 instruction which is used to subtract packed decimal fields. Operand 1 is a field in storage which should contain a packed decimal number. The resulting sum develops in this field. The contents of Operand 2, another packed decimal field in storage, is subtracted from the contents of Operand 1 to produce the result which is stored in Operand 1. The operands are limited to a maximum length of 16 bytes and may have different sizes since this is a SS_2 instruction.

A decimal overflow (condition code = 3) will occur if the generated result loses a significant digit when it is placed in the target field. A decimal overflow may or may not cause a program interruption (abend). This depends on the setting of a bit in the PSW (See **SPM**). Otherwise, the condition code is set to indicate whether the result was zero (condition code = 0), negative (condition code = 1), or positive (condition code = 2). You can test these conditions with **BZ** or **BNZ**, **BM** or **BNM**, and **BP** or **BNP**.

Consider the following **SP** example.

```
        SP      APK,BPK
        BM      ANEGATIV
```

Assume the variables are initially in the following states,

```
APK     DC      PL4'34'   = X'0000034C'
BPK     DC      PL3'22'   = X'00022C'
```

After the SP instruction has executed, the variables have the following values.

```
APK  = X'0000012C'
BPK  = X'00022C'
```

The contents of BPK was subtracted from APK and the result stored in APK. BPK was unaffected by the subtract operation. The branch is not taken since the condition code is positive.

On the other hand, consider the following example,

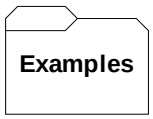
```
        SP      APK,BPK
       ...
APK     DC      PL2'999'   = X'999C'
BPK     DC      PL2'-5'     = X'005D'
```

After the SP instruction has executed, the variables have the following values.

```
APK  = X'004C'
```

```
BPK  = X'005D'
```

Notice that an overflow occurred in APK with the loss of a significant digit. APK was too small to hold the difference that developed as a result of the **SP**.



Some unrelated SP's:

```
A        DC      P'12345'      = X'12345C'
B        DC      P'-32'        = X'032D'
C        DC      Z'11'         = X'F1C1'
         ...
                              Results:
         SP    A,=P'20'    A = X'12325C'   C.C. = HIGH
         SP    B,=P'20'    B = X'052D'     C.C. = LOW
         SP    B,=P'-40'   B = X'0008C'    C.C. = HIGH
         SP    A,=P'1'     A = X'12344C'   C.C. = HIGH
         SP    A,B         A = X'12377C'   C.C. = HIGH
         SP    B,B         B = X'000C'     A FIELD CAN BE SUBTRACTED
                                           FROM ITSELF
                                           C.C. = EQUAL
         SP    B,A         B = X'377D'    AN OVERFLOW OCCURS
         SP    A,C                        DATA EXCEPTION - C NOT PACKED
```

Tips

1. You must be familiar with your data. The best way to prevent overflows is to plan the size of your fields based on the data at hand. There is a rule of thumb that you can follow for subtractions: If you are subtracting two packed fields with m and n bytes, then the difference might be as large as max(m, n) + 1 bytes. You may need to construct a work area to handle the maximum values. For instance,

```
FIELDA   DS    PL3
FIELDB   DS    PL5
WORK     DS    PL6
```

In planning to subtract FIELDB from FIELDA, we construct a work field called "WORK". The following code completes the task.

```
ZAP    WORK,FIELDA
SP     WORK,FIELDB
```