DIVX$ - Return Result & Remainder from Division

DIVX$ returns the result and the remainder from a division.

1. Invocation

To obtain the result and remainder from a division:

```
CALL DIVX$ USING di
```

where *di* is a control block of the following format:

```
01      DI
  02    DIVERS  PIC 9(4) COMP       * VERSION NUMBER, MUST BE 1
        VALUE   1
  02    DIINTE  PIC 9(2) COMP       * DIGITS IN INTEGER PART, MUST BE 7
        VALUE   7
  02    DIFRAC  PIC 9(2) COMP       * DIGITS IN FRACTIONAL PART, MUST BE 2
        VALUE   2
  02    DIROUND PIC 9(2) COMP       * 0 = NO ROUNDING, 1 = ROUNDING
  02    DIDPTR  PIC PTR             * POINTER TO PIC 9(7,2) C DIVIDEND FIELD
  02    DIVPTR  PIC PTR             * POINTER TO PIC 9(7,2) C DIVISOR FIELD
  02    DIQPTR  PIC PTR             * POINTER TO PIC 9(7,2) C QUOTIENT FIELD
  02    DIRPTR  PIC PTR             * POINTER TO PIC 9(7,2) C REMAINDER FIELD
```

2. STOP Codes and Exception Conditions

The following STOP codes may be generated by DIVX$:

STOP code	Description
20002	Invalid block version. DIVERS does not contain 1.
20003	Invalid number of digits in integer part. DIINTE does not contain 7.
20004	Invalid number of digits in fractional part. DIFRAC does not contain 2.

The following EXIT codes may be returned by DIVX$:

EXIT code	$$COND	Description
20001	1	The DIVISION statement suffered an OVERFLOW condition.

3. Programming Notes

DIVX$ is only available with GSM SP-41, or later.

Note the two input parameters (the dividend and divisor fields) are passed via pointers; as are the two output parameters (the result and remainder fields). This is unlike DIVY$ where the 4 parameters are passed directly to the routine via the USING clause of the CALL instruction.

4. Examples

[EXAMPLES REQUIRED]

5. Copy-Books

None.

6. See Also

DIV$	Return Result from Simple Integer Division
DIVY$	Return Result and Remainder from Simple Integer Division
REM$	Return Remainder from Simple Integer Division