line as the CALL command. For example (in Extended XYBASIC):

```
CALL #A000, I%, S$, Y(1,2), *Z
```

Each parameter may be either a variable reference, or the character * followed by an array variable name. To find the value of a parameter, your machine language routine calls the subroutine GTPAR, which returns information about the next parameter in the CALL command's parameter list. Chapter II describes how to access GTPAR in your version of XYBASIC. You should execute a machine language CALL to location 103H in CP/M versions, location 3283H in ISIS-II versions, or location 103H in Custom I/O versions of XYBASIC.

GTPAR returns the type of the next parameter in the A register: 0 if no more parameters, 1 if integer, 2 if string, 3 if floating. The B register returns the number of bytes in the parameter's value: 2 if integer, 3 if string, 4 if floating, as explained in greater detail below. The C register returns the number of dimensions of the parameter: 0 if a simple variable or an array element, n if the parameter is * followed by the name of an n-dimensional array variable. For simple variables and array elements, HL returns the address of the parameter's first value byte. For arrays, DE returns the address of the first dimension byte and HL the address of the first value byte.

For GTPAR to be useful you need to understand how XYBASIC stores values. Integer values are represented by two bytes in two's complement representation, with the least significant byte first in keeping with 8080 convention. For example, the integer value 10 is stored as the two bytes 0AH, 00H; the integer value -10 is stored as the two bytes 0F6H, 0FFH.

String values are stored in three bytes. The first contains the length of the string. The second and third are meaningless if the length is 0 (null string); otherwise they give the location (in string space) of the first character of the string. Successive characters are stored in successive locations in string space. For example, the string value "ABCD" contains 4 characters, so it is stored as a length byte containing 4 followed by two bytes giving a location in string space. The four characters "A", "B", "C" and "D" are stored in ASCII as the byte values 41H, 42H, 43H and 44H, starting at the given location in string space.

Floating point values are stored in four bytes. The first byte is zero and the remaining bytes meaningless for a floating point zero. Otherwise the first byte contains the binary exponent of the normalized value, with a bias of 80H. Bytes 2 - 4 contain the binary mantissa of the normalized value, with an assumed binary point preceding byte 2 and an assumed 1 replacing bit 7 of byte 2. Bit 7 of byte 2 contains the sign of the value, 0 if positive and 1 if negative. For example, the floating point value 10.5 decimal is equal to 1010.1 binary, which is normalized as .10101 binary times 2 to the 4th power. Therefore 10.5 decimal is represented by the four bytes 84H, 28H, 00H, 00H. The first byte contains the biased exponent (80H + 4H = 84H). Bit 7 of byte 2 is 0, indicating that the value is positive. Bytes 2, 3 and 4 contain the binary mantissa, with the leading 1 "hidden" by the sign bit.

For an array, the first element is stored in the first location and successive elements of the array are stored by rows. For example, a floating point array declared with DIM A(2,3) is stored A(0,0), A(0,1),