In Memory Format of MTX BASIC Variables

List of Variable Names

System variable VARNAM (0xFA7B) points to the bottom of a list of variable names.

The names in this list are mangled as follows: Each item in the list starts with the second character in the name, and continues to the last character. This is then followed by a byte which encodes both the first letter and the type of the variable. This byte always has bit 7 set (indicating the end of the item), bits 5&6 indicate the variable type, and bits 0-4 give the 5 lsb of the initial letter. The values of this byte are:

0xE0 – 0xFA A numeric array

0xC0 – 0xDA A numeric value

0xA0 – 0xBA A string array

0x80 – 0x9A A string value

The list is terminated by a 0xFF byte.

The position of a variable name in this list is called (by the ROM source listing), the UVI of the variable (the first name has a UVI of one (not zero)).

To find the UVI of a variable, construct a name string, mangled as above, set DE to the start of this string and then call to 0x30DC in the BASIC ROM (This is a call to an unnamed location within the SEARCH (0x30D5) routine. Returns with the UVI in BC and carry flag set if found, otherwise with carry flag clear.

List of Variable Values

The list of variable names is followed by the list of variable values. System variable VALBOT (0xFA7D) points to the start of the list, and system variable CALCBOT (0xFA7F) points just beyond the end of the list.

The address of a value item in this list is given by:

addr = CALBOT – 5 * UVI

This is calculated by routine UVItoSRA (0x34A0) in the BASIC ROM. Call with UVI in BC, returns with address in HL.

Each of the items in this list is five bytes long, and the list is in reverse order of the names list.

The contents of the item varies according to the variable type identified above.

Numeric Array

Bytes 0-2: 24 bit virtual address of array data

Bytes 3&4: 16 bit word giving the number of dimensions.

Numeric Variable

The value of the variable is stored as a five byte floating point value.

String Array

An array of strings is actually represented as an array of characters with one extra dimension (the length of the strings)

Bytes 0-2: 24 bit virtual address of string array data

Bytes 3&4: 16 bit word giving the number of dimensions plus 1.

String Variable

Bytes 0-2: 24 bit virtual address of string data

Bytes 3&4: 16 bit word giving the length of the string..

Array Data

This includes all strings, which are treated as arrays of characters.

Virtual Addresses

MTX Basic uses 24 bit virtual addresses to describe the location of array data (and program code?) in memory. This means that MTX BASIC is not limited to 48kB. To translate a virtual address (va) to a physical address the following algorithm is used:

ram_page = va >> 15

addr = (va & 0x7FFF) + offset

where offset is 0x4000 for most pages, but 0x8000 for the last page. This means that the last page of virtual memory can extend into high memory at 0xC000 and above.

Routines VAtoLRA (0x357C) and (LRAtoVA (0x3593) in the BASIC ROM translate between physical and virtual addresses.

Numeric Array

The data for a numeric array starts with a list of the sizes of each dimension, stored as 16 bit words.

This is followed by a list of the floating point values, five bytes each, stored in the order of last dimension varying most rapidly (C style, not Fortran style).

String Array

The data for a string array starts with a list of 16 bit words that give the sizes of each dimension. This is followed by a 16 bit word giving the length of the strings in the array (the final dimension of the character array).

This is followed by the characters in the array, one byte each stored in the order of last dimension (the position in the string) varying most rapidly.

String variable

The data for a string starts with a 16 bit word (presumably the length of the string, repeating the value given in the variable values list) followed by the bytes giving the text of the string.