SECTION 1

ARMADILLO BASIC SPECIFICATION

1.1 General Features

Armadillo Basic will execute user programs from CPU RAM, with program sizes up to 63K bytes. Each array will be to 64K bytes (8K reals, 32K integers, 16K strings). The stack will be limited to 4K bytes. Otherwise the only memory restriction is the total available ram.

Extended Basic will be a true extension of the standard interpreter, rather than a separate interpreter that just happens to share a few routines. Designing it this way should allow us to have a smaller (or more powerful) extended basic, and one that is easier to maintain than the current package.

Execution of basic programs from Grom will be supported, but access to subprograms in GPL in command modules will not be supported. PRK, Statistics, S-F Administrative, etc. **WILL**NOT**WORK** because of such statements as CALL A(...).

The following subprograms will be added to console Basic, to make up for the fact that command module subprograms will not be supported.

* CALL SAY,SPGET (as in speech editor)

* CALL INIT,LOAD,LINK,PEEK,POKEV,PEEKV,CHARPAT (as in editor/assembler)

Notice that basic programs that use the PRK subprograms or the Carlisle Phillips "cheater" package will not run.

1.2 Added Features

1.2.1 SCREEN ACCESS. A new screen handler will be used. It will support faster scrolling, windowing, and access to all VDP modes. Alternate 80-column video may be provided if a definition of it can be supplied in time.

* CALL SCREEN(background-color) (current usage)
* CALL SCREEN(background, foreground)
* CALL SCREEN(background, foreground, mode)
* CALL SCREEN(background, foreground, mode, left-margin, right-margin, top-margin, bottom-margin)

Modes will include pattern, multicolor, text, and bit-map. The default is pattern mode with left and right margins of 2, top and bottom margins of 0.

CALL LINE(y1,x1,y2,x2,color) will draw a line segment in bitmap or multicolor mode. If the color is omitted, black is assumed.

CALL DOT(y,x,color) will draw a dot of the given color in bitmap or multicolor mode. If the color is omitted, black is assumed.

1.2.2 Sprites. Sprites will be accessible in all modes except text. They will be accessed as in Extended Basic, except that 32 sprites will be available.

1.2.3 Integers. Basic will implement an integer data type, for speed and space efficiency. The default numeric type is still real.

1.2.4 Multiple-statement lines. Standard Basic will include multiple-statement lines and the corresponding enhancements to the IF-THEN-ELSE statement.

1.2.5 Program chaining. Standard basic will implement RUN as a statement, for chaining. RUN will be enhanced to accept an arbitrary string expression as an argument, instead of just a constant.

1.2.6 Accept and Display. Standard Basic will include a simplified form of accept and display, and a function TERM to return the function code that ended the input (proceed, enter, up, down, etc.). Only the AT and SIZE clauses will be supported.

1.2.7 Loading and saving programs. While Basic is running, only 24K bytes of ram can be mapped into the logical address space. Thus memory image saves and loads (i.e. PROGRAM files) will be limited to 24K. (This assumes that all of the DSRs are written so that they can use CPU ram buffers.) Programs that are larger than 10K will be saved in the same sequential file format that Extended Basic currently uses for programs too large to fit in VDP ram. (INTERNAL, VARIABLE 254). Problem: cassette cannot currently handle variable files or records longer than 192. Basic will also be able to load DISPLAY, VARIABLE 80 text files of the type produced by LIST and by the text editors. This type of loading will crunch each line as it is input. Loading will be slower than the standard binary loads, but this will permit loading foreign programs (from RS232) as well as using text editors on Basic programs.

1.2.8 Assembly Language support. The default condition is that basic will use all of available ram for program and data. Thus the user must allocate some ram before assembly language can be loaded or accessed. This will be done in the CALL INIT statement.

CALL INIT will allocate 8K bytes of ram for the user, and will load utility routines into the first 2 or 3 K of it. CALL INIT(X) will allocate X K bytes of ram for the user. The minimum is 4, and the maximum is 64 (or as much as remains after allowing for the current basic program & data and DSR allocations). A value of 3 or less will be treated as CALL INIT(0), which will return all allocated ram to Basic.

The utilities will include NUMREF, NUMASG, STRREF, STRASG, LOADER, DSRLNK, GPLLNK, XMLLNK, etc. as in the editor/assembler and Extended Basic. Upon a CALL LINK, the BLWP vector table will be mapped in at >2000 (compatible with extended basic) and all other allocated ram at >A000. Note that the loader can directly load only 24K of ram, even though 64K may be allocated. Absolute origin code which loads on the 4A may not load properly on the Armadillo. (e.g. the SAVE utility in the editor/assembler or the text-to-speech tables in the disk text-to-speech package.

1.3 Speed enhancements

The following functions will be faster than 99/4A basic, due primarily to translation of GPL to machine code, and extra scratchpad ram.

* screen access and scrolling.

* formatting numbers for print. (convert-number-to-string)

* listing and editing programs.

* intrinsic functions. (SIN, EXP, etc.)

* string handling.

* random number generation.

1.4 Extended Basic features

Extended basic will still have the following features not found in standard basic:

* User-written subprograms with parameters and local variables.

* Program merging.

* Functions PI, MAX, MIN, RPT

* Logical operators AND, OR, XOR, NOT

* Block statements DO, LOOP, UNTIL, WHILE, EXIT. (From new ANSI standard. New for Armadillo version.)

* Full ACCEPT and DISPLAY statements.

* Formatted output via PRINT USING and DISPLAY USING.

* Tail remarks

* Error, warning, break handling under user control.