Date: June 17, 1985

Author: Fern Bachman

Subject: Tools Overview/Notes

Document Version Number: 00:00

Some new tools provided in ROM are as follows.

1. SANE
2. Super Hi-Res graphics core routines
3. Memory manager
4. Capability module manager
5. Screen dump
6. Sound toolkit

What follows is some notes on these tools.

1. SANE (Standard Apple Numerics Environment)

The following notes were taken from the Apple Numerics Manual.

1. Approximately 11-12K of ROM required
2. Approximately 8 bytes of firmware RAM required
3. Provides extended precision IEEE standard floating point arithmetic, together with compatible elementary functions.
4. Provides identical functionality as the built-in Macintosh numerics.
5. Data types supported
    1. X - extended (80 bits - floating point)
    2. D - double (64 bits - floating point)
    3. S - single (32 bits - floating point)
    4. I - integer (16 bits - integer)
    5. C - comp (64 bits - integer)
    6. Dec - decimal record
    7. Decform - decform record
6. Arithmetic Operations / Auxiliary Routines
    1. Add
    2. Subtract
    3. Multiply
    4. Divide
    5. Square root
    6. Round to Integer
    7. Truncate to Integer
    8. Remainder
    9. LOG binary
    10. Scale binary
    11. Negate
    12. Absolute value
    13. Copy sign
    14. Next after
7. Conversion routines provided
    1. Convert binary to binary
    2. Convert binary to decimal
    3. Convert decimal to binary
8. Compare and classify operations provided
9. Environmental control provided
10. Halt control provided
11. Elementary functions provided
    1. Base-e logarithm (LNx)
    2. Base-2 logarithm (LOG2x)
    3. Base-e log1 (LN1x)
    4. Base-2 log1 (LOG21x)
    5. Base-e exponential (EXPx)
    6. Base-2 exponential (EXP2x)
    7. Base-e exp1 (EXP1x)
    8. Base-2 exp1 (EXP21x)
    9. Integer exponentiation (xPWR1)

10. General exponentiation (xPWRY)
11. Compound interest (COMPOUND)
12. Annuity factor (ANNUITY)
13. ARCTANGENT (ATANx)
14. SINE (SINx)
15. COSINE (COSx)
16. TANGENT (TANx)
17. RANDOM (RANDx)

2. Super Hi-Res Graphics Core Routines

One new feature added to Columbia is the 32K super high resolution graphics mode. It is significantly different from the standard Apple // graphics modes in that it uses 32K of linearly addressed screen memory to display. It has resolution up to 640 by 200 in 4-colors per pixel mode and 320 by 200 in 16-colors per pixel mode.

To encourage developers to write graphics software to use this feature and to make that job easier for them we are providing a selected range of core routines in Columbia's ROM. The developer will have access to these routines via the capability manager. The routines will include but not be limited to windowing, fonts, clipping, console driver, etc.

Hopefully hooks can be made available for Applesoft to allow the casual and experienced programmer to have easy access to this new graphics feature to do at least rudimentary drawing on the super hi-res screen. These hooks would bring the new graphics feature to more users faster than if only assembly language hooks are provided.

For a detailed description of the super hi-res graphics core routines see the ERS written by Steve Glass.

3. Memory Manager

The memory manager for Columbia is as the name implies a RAM memory manager. It provides programs with a way to request (allocate) memory, free (deallocate) memory, and to find out how much memory is currently available in the system. The memory manager works in memory sizes of 1 page (256 decimal) bytes. It maintains a dynamic memory allocation buffer in firmware RAM. A page in use is marked as a 1 in a bit in a byte of the allocation buffer. A free page is marked as a 0 in a bit in a byte of the allocation buffer. As pointed out in James Jatczynski's memo since Columbia has no hardware memory management, software programs must cooperate with this software implementation of memory management. This software memory management type system in not a foreign idea to our current developers. PRODOS does a subset of this type of memory management already. Compliance with Columbia's memory manager in new programs should not therefore be a problem.

Full details of the memory manager are in James Jatczynski's ERS entitled Cortland Memory Manager.

4. Capability Module Manager

The capability module manager gives Columbia's application programs a way to access ROM and RAM based tools transparently. That is the application program calls the capability manager for a particular function to be performed. The capability manager via its ROM or RAM or ROM/RAM tables directs the call to the appropriate subroutine. The application never knows whether it is executing out of RAM, ROM or both. The Capability module manager also gives us the ability to pseudo 'patch' ROM tools if errors are detected in the ROMs.

Full details of the capability module manager are in James Jatczynski's ERS entitled A Framework for Implementing ROM- and RAM Based Capability Modules in Cortland.

5. Screen Dump

Screen dump is a utility to, as you probably already guessed, to print the current screen image to a printer. This function can be invoked via the DeskTop manager, an application program, or hopefully from a sequence of key strokes. This utility will be able to dump text screens to any printer in slot 1 only. Graphics screens are a problem. We can dump any graphics screen to an Apple printer. It is the printers that do not accept an Apple Imagewriter's graphics control codes that we cannot print to. To overcome this, if it is deemed necessary to overcome this problem, we could add a printer control character setup screen to the MENU program. This screen would have to prompt the user for control codes to use and save them in BATTERYRAM for the screen dump program. We are open for suggestions on the screen dump issue.

6. Sound Toolkit Support

The sound toolkit support will be code in ROM to simplify communicating with the ENSONIC sound chip Columbia uses and possibly provide generic routines to modify the standard sine wave (which we are providing in ROM) to produce various sounds. Applesoft sound support will also be provided in this toolkit. Since many, many programs are still written in BASIC or a least concepts to be converted to assembly language are still written in BASIC this seems like a prudent thing to do.