Apple // Extended Debugging Monitor

Introduction:

The Extended Debugging Monitor (EDM) is a superset of the standard Apple //e monitor with which can access all of the 128K of main and auxiliary memory. It is made available on a set of modified Apple //e ROMs.

The standard ROM monitor was designed to really only support 48K of memory in an Apple //. With the addition of both the language card and now the extended 80-column card, it is now necessary to have a new monitor that can access all of the memory available in an Apple //.

In addition to providing access to the full 128K //e memory, the monitor features a "RESUME" command to resume operation in a non-standard memory environment and an improved memory location display format.

In implementing EDM, the diagnostics and cassette support have been removed and the debugging monitor extensively revised. However, as the remaining firmware remains untouched, applications which run on the revised //e ROMs should run on the EDM ROM.

Description:

The EDM is a superset of the old monitor. Commands in the previous monitor format produce very similar results. Commands like SEARCH and ASSEMBLE and the as ASCII input modes added in the //e ROM revision are still available. The address xxxx refers to memory location xxxx under the current memory mapping configuration. If the memory configuration is changed by hitting soft switches like RDCARDRAM, ALTZP, xxxx may change its meaning accordingly. This is still true in the EDM, although hitting memory softswitches from the monitor is now neither necessary nor recommended since more direct techniques for getting to the various parts of memory are provided.

The internal-ROM/slot-ROM softswitch has no effect in the EDM. Examination of the internal ROM is not permitted from the EDM since the internal ROM contains routines internal to the firmware and need not be directly accessed by the user. This should not be a major handicap.

Using the old commands results in some differences. Memory display is 16 bytes per row when the screen width allows it, and both the hexadecimal and ASCII values are shown. In ASCII display, the high

bit is ignored and unprintable characters are replaced by "." so output can be sent to printers. Because memory locations will be read twice during the display, some soft switches affected by multiple accesses may be affected differently by the new monitor.

EDM has a 65C02 disassembler and mini-assembler, but EDM itself does not use any 65C02 instructions and hence may run on machines with either an NMOS 6502 or a 65C02.

A new address syntax is used to specify all memory. Addresses are represented by b/xxxx. b, a bank number, is either 0 or 1. (For b > 1, even values are treated as 0, odd values as 1.) Bank 0 is the RAM memory on the main logic board and bank 1 is the AUX memory on the 80 column card. This convention applies for main or AUX Memory in the ranges $0000-$BFFF and SE000-$FFFF. There is a different convention for the bank switched memory and I/O areas ($C000-$DFFF).

ADDRESS SYNTAX:

```
0/nnnn = address nnnn in MAIN memory
1/nnnn = address nnnn in AUX memory
```

where nnnn is in the range $0000 - $BFFF or $E000 - $FFFF

There is no system RAM in the address range $C000 to $CFFF. ROM residing there could be accessed by $Cxxx. Taking advantage of this opportunity, a different meaning is assigned to b/Cxxx. b/Cxxx refers to the main/aux language card bank 1 address $Dxxx, while b/Dxxx refers to the main/aux language card bank 2 address $Dxxx. Using these notations we can refer to all RAM and all ROM except the internal ROM. This notation can be used in commands like EXAMINE, STORE, MOVE, VERIFY, SEARCH, DISASSEMBLE and even ASSEMBLE. The only exception is the bank is ignored in the GO command.

ADDRESS SYNTAX:

```
0/Cnnn = address SDnnn in MAIN language card bank 1
0/Dnnn = address $Dnnn in MAIN language card bank 2
1/Cnnn = address $Dnnn in AUX  language card bank 1
1/Dnnn = address $Dnnn in AUX  language card bank 2
```

When an address range is specified as in {adrs1}.{adrs2}, the second address should not have a bank number. If one is present, it is ignored. In the mini-assembler, a bank number may be specified in the beginning address, but you cannot put a bank number in the middle of an instruction. BNE 1/300 is treated as an error.

The GO command in EDM docs not use a bank number in the address. The syntax is exactly the same as in the standard monitor. If your program is running in a non-standard memory configuration, you cannot simply use the GO command to resume execution. In such a case, you must set up the correct memory state first. Whenever a BRK or an interrupt occurs, EDM handles it just as the //e revision ROM's handle it, except that EDM will correctly handle a BRK in the alternate Zero Page. For an interrupt or BRK, the firmware sets the machine back to the standard memory configuration and encodes the memory state in a byte located at $64. The GO command leaves the return address on the stack, which means that an RTS instruction will cause a return to the monitor.

In order to allow the correct resumption of execution after an interrupt or BRK, EDM has added a RESUME command. The syntax is nnnn^R (^R = control-R), where nnnn is the address at which to resume execution. Note, this address does not use a bank number. The RESUME command will use the memory state value stored in location $44 to setup the correct memory state before beginning execution. You can modify this value in order to begin execution in any memory configuration. The RESUME command leaves nothing on the stack and is, therefore, a true resume operation. You can think of a BRK-RESUME combination as an interrupt that occurs at a specified point in your program that allows you to examine and modify memory during the "interrupt".

For both the GO and the RESUME commands if nnnn is not specified, the address stored at locations $3A and $3B is used.

The memory state value at location $44 is displayed as register M in the register display (using the ^E command). The interpretation of the bits in register M is shown below:

```
bit 7 = 1 if auxiliary zero-page/stack switched in
bit 6 = 1 if 80STORE and PAGE2 both on
bit 5 = 1 if auxiliary RAM switched in for reading
bit 4 = 1 if auxiliary RAM switched in for writing
bit 3 = 1 if bank-switched RAM being read
bit 2 = 1 if bank-switched $D000 page 1 switched in
bit 1 = 1 if bank-switched $D000 page 2 switched in
bit 0 = 1 if internal CX ROM switched in
```

In the standard monitor location $40-$41 is set to $45, providing a short-cut for modifying the pseudo-accumulator at location $45. EDM now sets this location to $44 so that modifications of this type can begin with register M.

If you want to run any program using the alternate zero page, and if you plan to use either interrupt or BRK, you must follow the stack pointer convention used by the firmware. Normally, the top of the stack when a BRK occurs is located at the "stack-register S" + 5. If a BRK occurs when the alternate zero-page/stack is used, the value in "stack-register S" is invalid. To find the top-of-stack of the alternate stack, look at location $101 in the alternate stack and add 1 to that value.

Since diagnostics have been removed, SOLID-APPLE CONTROL RESET will not bring up the diagnostics. Instead, this will cause a direct jump into EDM without trashing any memory. This can be very useful for debugging.

Details on difference between EDM and the //e revision ROM:

EDM uses some extra memory locations. They are located in slot 3 screen holes and slot 0 screen holes. Some of these are temporary locations already used by the firmware, so they should not conflict with any program. The slot 3 screen holes used are:

    $6FB - contains the bank number (normally used by PASCAL only)
    $7FB - contains the bank number during MOVE/VERIFY
    $67B - normally used by video firmware
    $77B - normally used by PASCAL or video firmware

The slot 0 screen holes are:

    $6F8, $778 - modified by EDM if alternative zero page is accessed.

Entry points $FECD and $FEFD for cassette I/O are now dummy routines

Entry points VERIFY ($FE36) and LT ($FE5E) no longer exist.

Entry points INSTDSP ($F8D0) and LIST ($FE5E) still disassemble 1 or 20 instructions, but unless the bank register location $6FB is set up properly, you may disassemble the wrong bank. Valid bank values are $80 (no bank), $00 (bank 0) and $01 (bank 1).

In the old monitor, commands of the format xxxx<yyyy.zzzzC would copy locations $3E-3F to $44-45. This does not seem to serve any useful purpose and destroys machine state value in $44-45, therefore EDH no longer does this.

Because EDM recognizes 65C02 opcodes, you must be careful to ignore any 65C02 instructions that may be displayed by the disassembler when the code does not actually contain these instructions.

The ID byte at $FBC0 contains $E1 to identity the EDM ROM.

EDM is identified at boot time by the display of "Apple][DB", where the "DB" stands for "Debugging".