12. USING MICRO-EXEC

Micro-Exec is a stand-alone Maxc program used to maintain and start up the Maxc Tenex system. A few overall remarks on its structure will be helpful before describing the specific facilities available.

Micro-Exec is normally loaded by the MEXECGO Midas command file (see Section 5) from Maxc disk unit A. Micro-Exec loads into the high portion of the first 128K of memory, starting at 350000. All references to "core" in Micro-Exec commands refer to the range 20-to-347777. (The program which boots in Micro-Exec runs in 347000-347777 and saves the previous contents of the ACs in 347760-347777.)

Micro-Exec has a sophisticated command interpreter which allows editing, questioning, prompting for parameters -- what we in the biz call "Dwimified-Middle-English". Commands take the form of a sequence of words separated by periods. E.g.:

```
        initialize.disk.pack
or
        initialize.tape.
```

At any point in typing a string, you may type <escape> and the interpreter will complete as much as is unambiguous. Similarly, at any point you may type "?" and all the possible completions of that command will be listed. Once you are familiar with commands, you will find it convenient to abreviate them by entering the first letter of each word followed by a space. For example, I D P is an abbreviation for "initialize.disk.pack".

Many of the commands require parameters before the command execution begins. Typing an escape will prompt you with a brief description of the parameter (e.g., (octal number) or (pack number)). Typing escape a second time will furnish the default value of that parameter if there is one. Some commands require confirmation (with carriage return or period).

Micro-Exec is capable of running in user mode under Tenex as well as stand-alone. Micro-Exec may be accessed under Tenex by running the subsystem MEXEC, and it requires enabled "wheel" or "operator" privileges. All commands may be executed either stand-alone or in user mode except where noted otherwise.

12.1. Tenex Disk Structure

(Also see Section 13.4 "Disk Maintenance")

Finally a few words on the disk structure of the Maxc-Tenex system will be helpful for the following command descriptions. A disk configuration is defined by establishing a correspondence between (1) logical units (Tenex), (2) pack numbers, and (3) disk drives (i.e., controllers). The current disk configuration is usually the default configuration on the current Micro-Exec save area (usually area 1 on physical unit 0). Logical units are identified by digits 0 thru n-1 (for an n-pack Tenex). Packs are labeled with octal numbers starting at 100. Drives are labelled alphabetically starting with A. For example, the "print.disk.configuration" command might type:

```
Number of units: 2
----------Unit--------Pack Number
0.          B           102
1.          A           107
```

At present there are 20 "Save Areas" allocated in the disk structure, but outside the Tenex file system. Each save area consists of four track cylinders (i.e., 240 (decimal) pages). A directory of save areas is posted on the control room wall.

A mechanism has been provided for maintaining a simple file system on a single save area, for storage of small programs such as PDP-10 diagnostics. At present, save area 2 is assigned permanently for this purpose. All instances of the word "program" in command names refer to programs stored in this manner. Sometimes disk packs are moved from one drive to another, so that drive can be freed for maintenance. When this is done, the procedures for automatically starting Tenex or booting the Micro-Exec may not work and some of the issues are discussed here. First, the command files which start Micro-Exec or Tenex from Midas expect to find Micro-Exec on save area 1 of disk drive 0 and Tenex on save area 0 of disk drive 0. If the pack containing these areas is moved to some drive other than 0, these command files won't work. In this case, you must boot Micro-Exec by hand from AltIO rather than relying on the MEXECGO or TENGO Midas command files to do this for you (see the chapters on AltIO for how to do this). After starting up Micro-Exec (and setting the disk configuration, if necessary), you can do a "Go" command to start Tenex.

Secondly, the various methods for starting Tenex from Micro-Exec require that Micro-Exec know the correct disk configuration. If the configuration has changed since Micro-Exec was written on the same area, the configuration must be correctly entered into Micro-Exec ("Set.Disk.Configuration" discussed below) before starting Tenex. "Write.Micro.Exec.to.area.1" can be used to store Micro-Exec on its same area with a correct disk configuration.

12.2. Micro-Exec Command Descriptions

In the following descriptions, a (C) means confirmation is required. In commands that read from or write to disk, the number following the command indicates the error retry count used; "default" means that the retry count may be overridden by the count specified in the "set.disk.error.retry.count" command. If not specified, the retry count is 20 and is unaffected by the "set.disk.error.retry.count" command.

brief
: Disables extended typeout of disk error status bits.

check.next.tape.file <magtape unit number> (stand-alone only)
: Checksums next file on tape and types entry point on console.

clear.core (C)
: Zeros locations 20 through 347777 and 400000 through 777777.

compare.disk.packs <pack$_1$> <pack$_2$>
: Compares the contents of the specified packs and reports discrepancies (usually used after a pack-to-pack copy).

copy.pack.to.pack <pack$_1$> <pack$_2$> (C) (default 20)
: Does a bit-by-bit copy of data from first pack onto second pack.

copy.quadruple <pack$_1$> <pack$_2$> <pack$_3$> <pack$_4$> (C) (stand-alone only)
: Simultaneously copies the first pack onto the second pack and the third pack onto the fourth pack.

ddt
: Enters DDT for debugging Micro-Exec. Re-enter Micro-Exec by 20$G.

debug.disk.controller <unit$_1$> <unit$_2$> (C) (stand-alone only)
: Permits checking out the disk controller hardware without using a real disk. This requires connecting the write cable of <unit$_1$> to the read cable of <unit$_2$> and connecting a pulse generator to simulate sector pulses.

dismount.auxiliary.pack
: Removes auxiliary pack from disk structure.

dump.core.to.area <area>
: Dumps locations 20 through 347777 on <area>. Registers are saved at 347760 when Micro-Exec is booted in. This command is used to save crashes for later analysis.

dump.core.to.file <filename> (user mode only)
: Dumps locations 20 through 347777 on <filename> (default extension .SAV).

dump.program <program name> (C)
: Dumps the (small) program presently in core onto the current program save area (usually 2) with the given name (5 or fewer characters). If a program already exists by that name, it is overwritten. The program in core should conform to PDP-10 conventions: the LH of location JOBCOR (133) should contain the highest location loaded with code, and the RH of JOBSA (120) should contain the starting address.

dump.save.areas.to.auxiliary.pack (C)
: Copies all the save areas from the currently configured file system onto the auxiliary pack. The save areas are arranged as if the auxiliary pack constituted a one-pack file system; hence, the last 4*20 = 80 (decimal) tracks on the pack are overwritten.

eddt
: Enters Exec DDT if Tenex is in core. Re-enter Micro-Exec by typing 20$G.

erase.program <program name>
: Deletes the named program on the current program save area.

exercise.disk.random <pack> (default 0)

Reads pages at random on the specified pack, reporting any device-detected errors (checksum, etc.) but not actually looking at the data. The pack is not written on.

exercise.disk.specific <pack> (default 0)

Prompts for a list of disk addresses, which it then cycles through repeatedly, reading each specified page as explained above.

find.chip.for.address <address> <bit designator>

Translates the supplied memory address and bit number into a physical memory chip location. <address> is a 21-bit octal number, and <bit designator> is one of "Bit n", "Tag n", "Check n", or "Parity" (abbreviations allowed), where n is a decimal number. Data bits are numbered 0-35, tag bits 0-3, and check bits 1, 2, 4, 8, 16, and 32.

go (stand-alone only)

Executes a "test.memory.fast", then loads Tenex from area 0 and starts it at SYSGO1.

goto <address>

Starts whatever is in memory at the specified address.

initialize.area.for.programs (C)

Initializes the "directory" of the current program save area (usually 2).

initialize.disk.pack (C) (default 0)

Initializes the auxiliary pack by: (1) writing headers, using the pack number entered in the mount.auxiliary.pack command, (2) writing random data on the entire pack, and (3) checking the data and printing discrepancies. An error retry count of zero is used throughout. Pages in which errors are detected are recorded in the bad spot list on page zero of the pack.

initialize.tape <magtape unit> (C) (stand-alone only)

Rewinds tape to load point, writes a boot header, and waits at the end of the boot header.

install.new.micro.exec (C)

Transfers a new version of Micro-Exec (loaded via a previous "load.dump.from.area") to its runtime location, and starts it up.

load.dump.from.area <area>

Loads core from specified area (inverse of dump.core.to.area).

load.dump.from.file <filename> (user mode only)

Loads core from specified file (inverse of dump.core.to.file).

load.program <filename>

Loads into memory the specified program from the current program save area (usually 2).

load.save.areas.from.auxiliary.pack (C)

Loads all save areas onto the currently configured file system from the auxiliary pack, which must previously have been written by dump.save.areas.to.auxiliary.pack.

loop.on.specified.disk.page <pack> <track> <head> <sector>
: Continuously reads the specified disk page ignoring errors (useful for disk tune-ups). The bell is rung for each error detected.

mount.auxiliary.pack <unit> <pack>
: Identifies auxiliary (i.e., extra-Tenex) pack to disk configuration. Certain commands such as initialize.disk.pack are valid only for the auxiliary pack. Note that there can be at most one auxiliary pack known to Micro-Exec at a given time. In user mode, one should be careful to "dismount.auxiliary.pack" before leaving Micro-Exec to avoid confusion.

print.disk.configuration
: Prints disk configuration in the format illustrated above.

print.disk.error.count
: Prints the total number of disk errors (both recovered and unrecovered) since Micro-Exec was started.

print.program.directory
: Prints the name and size (pages) of each program in the current program save area (usually save area 2).

quit (user mode only)
: Returns control to the Tenex Exec.

read.specified.disk.page <pack> <track> <head> <sector> (default 20)
: Reads specified page into buffer at location BUF.

read.tape.to.core <magtape unit> (stand-alone only)
: Reads tape into core starting at location 20.

read.tenex.from.area <area>
: Loads Tenex from specified area and checks its "fingerprint".

read.tenex.from.file <filename> (user mode only)
: Loads Tenex from the specified file (default extension .SAV).

read.tenex.from.tape <magtape unit> (stand-alone only)
: Loads Tenex from tape and checks its "fingerprint".

reset.disk.error.count

rewind.tape <magtape unit> (stand-alone only)

run.diagnostic.program <program name>
: Loads and starts the named program from the current program save area, assuming that the program conforms to the conventions for controlling PDP-10 diagnostics. The diagnostic is run for one complete pass, and then control returns to Micro-Exec.

run.program <program name>
: Loads and starts the named program from the current program save area.

scan.disk.pack.for.errors <pack> (default 0)
: Scans pack, counting soft errors, and reporting non-recoverable errors to console.

set.disk.configuration (stand-alone only)
: Sets disk configuration; asks for parameters via the same format used in print.disk.configuration.

set.disk.error.retry.count
: Overrides the default disk error retry count in certain commands.

set.disk.timeout <decimal number> (stand-alone only)
: Sets the disk timeout interval (seconds), at the end of which Micro-Exec will report that it has timed out.

set.program.save.area <area>
: Declares the program save area to be used for subsequent program operations ("run.program", etc.) Default area is 2.

silent (C)
: Completely turns off disk error typeouts (useful for setting up scope loops for hardware debugging).

start.tenex.from.area <area> (C) (stand-alone only)
: Reads Tenex from the specified area and starts it at SYSGO1.

test.memory.fast (stand-alone only)
: Runs a quick (~30 second) memory checkout and summarizes errors found. Also builds a map of bad memory for communication to Tenex.

test.memory.slow (stand-alone only)
: Runs a thorough (~8-minute) memory test and summarizes errors found.

test.memory.verbose (stand-alone only)
: Same as "test.memory.slow", but errors are printed out individually rather than only summarized.

test.memory.write.fast (stand-alone only)
: Undoes the effect of test.memory.write.slow.

test.memory.write.slow (stand-alone only)
: Sets a mode flag that causes subsequent runs of the memory test to write data into memory using a more conservative (but slower) method than usual. Try this command if the memory test crashes due to "fatal error from wrong place in code".

verbose
: Reinstates full disk error typeout.

verify.disk.test.pattern (default 0)
: Reads the auxiliary pack and verifies the data written by a previous "initialize.disk.pack" or "write.disk.test.pattern", reporting any discrepancies (up to a maximum of 5 per page).

write.core.to.tape <magtape unit> (stand-alone only)
: Writes core from 20 to 347777 onto tape.

write.disk.test.pattern (default 0)
: Writes random data on all pages of the auxiliary pack, then rereads and verifies it, reporting any discrepancies (up to a maximum of 5 per page).

write.micro.exec.to.area <area> (C)
: Writes a bootable (via NVIO "B" and "S" commands) copy of the currently running Micro-Exec to the specified area. (Ordinarily Micro-Exec is kept on save area 1.)

write.micro.exec.to.tape <magtape unit> (stand-alone only)

write.specified.disk.page <pack> <track> <head> <sector> (C) (default 20)
: Writes page from memory location BUF to specified disk page.

write.tenex.to.area <area> (C)
: After checking "fingerprint", writes Tenex core image to specified area.

write.tenex.to.file <filename> (user mode only)
: After checking "fingerprint", writes Tenex core image to specified file (default extension .SAV).

write.tenex.to.tape <magtape unit> (C) (stand-alone only)
: After checking "fingerprint", writes Tenex core image to specified tape.

A more compact summary of Micro-Exec commands is given in the following table:

12.3. Micro-Exec Command Summary

Disk Configuration Commands

set.disk.configuration	print.disk.configuration
mount.auxiliary.pack <unit><pack>	dismount.auxiliary.pack

Disk Pack Copy and Compare Commands

initialize.disk.pack	compare.disk.packs <pk1><pk2>
copy.pack.to.pack <pk1><pk2>	copy.quadruple <pk1><pk2><pk3><pk4>
scan.disk.pack.for.errors <pack>	

Save Area and Other Commands

dump.save.areas.to.auxiliary.pack	load.save.areas.from.auxiliary.pack
dump.core.to.area <area>	load.dump.from.area <area>
write.micro.exec.to.area <area>	install.new.microexec

goto <address> clear.core
ddt eddt
read.tenex.from.area <area> write.tenex.to.area <area>
start.tenex.from.area <area> go

Program Save Area Commands

set.program.save.area <area> initialize.area.for.programs
dump.program <program name> erase.program <program name>
print.program.directory run.diagnostic.program <program name>
run.program <program name>

Disk Test Commands

reset.disk.error.count set.disk.error.retry.count
print.disk.error.count set.disk.timeout <decimal number>
write.disk.test.pattern verify.disk.test.pattern
exercise.disk.random <pack> exercise.disk.specific <pack>
brief verbose
debug.disk.controller <unit1><unit2> silent
read.specified.disk.page <pack><trk><hd><sec>
write.specified.disk.page <pack><trk><hd><sec>
loop.on.specified.disk.page <pack><trk><hd><sec>

Memory Test Commands

find.chip.for.address <address><bit designator>
test.memory.fast test.memory.slow
test.memory.write.fast test.memory.write.slow
test.memory.verbose

Tape Unit Commands

initialize.tape <unit> rewind.tape <unit>
read.tape.to.core <unit> write.core.to.tape <unit>
read.tenex.from.tape <unit> write.tenex.to.tape <unit>
write.micro.exec.to.tape <unit> check.next.tape.file <unit>

User Mode Commands

dump.core.to.file <filename> load.dump.from.file <filename>
read.tenex.from.file <filename> write.tenex.to.file <filename>
load.program <filename> quit