Inter-Office Memorandum

To	Dorado users	Date	February 19, 1983
From	Ed Taft	Location	PARC/CSL
Subject	Dorado booting operation (edition 5)	File	[Indigo]<DoradoDocs>DoradoBooting.press

XEROX

Dorado bootstrapping is a complex process, involving a number of hardware, microcode, and software components. This memo describes how to boot a Dorado and how to configure it to boot in certain ways. A companion memo, Dorado booting implementation’’, file [Indigo]<DoradoDocs>DoradoBootingImpl.press, is a comprehensive description of how it works and how the components are constructed and maintained.

1. Operation

Booting a Dorado from scratch involves two main activities: loading its control store with the correct *microcode*, and then forcing it to load and run the correct *software*. The descriptions here concentrate on the first step.

1.1 Emulators

The Dorado is able to execute programs written in a particular language (Mesa, Lisp, Smalltalk, or Alto BCPL) by virtue of running *emulator microcode* that is, microcode for emulating that language. In addition, there is some common *I/O microcode* for controlling the I/O devices (display, disk, Ethernet, etc.) This microcode is not built into the Dorado but rather must be loaded into its control store from some external source.

The Dorado’s control store is not large enough to hold the emulator microcode for all languages simultaneously. Therefore, the microcode is divided up into several independent microprograms, only one of which is used at a time.

Many microprograms include Alto emulation’’ capability: each contains the Alto emulator and I/O microcode, and additionally contains the emulator for one other language. That is, the Alto/Mesa microcode’’ consists of the Alto emulator, the I/O microcode, and the microcode for emulating the Mesa language; similarly for the Lisp microcode’’ and the Smalltalk microcode’’.

The Cedar/Mesa’’ microprogram includes only the Cedar/Mesa emulator and Pilot I/O facilities, and does not include an Alto emulator.

1.2 Microcode booting

When the Dorado is first turned on, it is in a shut down’’ state that minimizes power consumption. Pushing the boot button on the back of the keyboard initiates a bootstrap sequence that takes about one minute. (Most of this time is spent waiting for the disk to become ready.) If all goes well, at the end of this time, microcode is boot-loaded from some source (the disk or an Ethernet boot server), and further actions depend on what that microcode is.

If the Dorado is *already* on, this full bootstrap sequence can also be initiated by pressing the reset button on the front panel of the Dorado chassis (inside the cabinet, if the Dorado is cabinet-mounted). However, if the Dorado is *not* already on, i.e., is in the shut down state, pushing the reset button has no effect.

Assuming that the Dorado is already on, other forms of booting are initiated by pressing the boot button on the back of the keyboard, as detailed in the following paragraphs. There are single- and multiple-push boots. You must push the boot button for no more than 2.5 seconds; multiple pushes must be spaced no more than 1.5 seconds apart; and the bootstrap operation does not commence until 1.5 seconds after you release the button for the last time.

1-push boot. This causes the currently-running microcode to go through the software bootstrap sequence, i.e., loading the Alto OS, or the NetExec if you have the BS and Quote keys pressed down, or the physical volume boot file (Pilot or Othello) if you were running Pilot, or whatever. A 1-push boot is the nearest equivalent to pressing the boot button on an Alto. It works only if the correct emulator microcode is already loaded and the Dorado is still running normally.

2-push boot. This causes the currently-running microcode to be forcibly restarted. It is similar to a 1-push boot, but it will work even if the Dorado has halted, and it does more extensive microcode initialization (in particular, it zeroes main memory, and it reverts to the default disk partition for the current emulator.) This does require, however, that there be intact emulator microcode in the Dorado’s control store.

3-push boot. This initiates a complete microcode bootstrap sequence, similar to the automatic power-on boot. This is the normal way to switch from one emulator to another (see below). It assumes nothing about the current state of the machine.

4, 5-, or 6-push boot. Same as 3-push boot.

7-push boot. This isn’t really a boot: it causes the Dorado to shut down, a process that takes about 30 seconds. When the machine is in the shut down state, pushing the boot button (one or more times) will initiate a complete microcode bootstrap, as described at the beginning of this section.

If you have no keys pressed down during a power-on boot or a 3-push boot, and your Dorado’s disk has Initial microcode installed on it, and the disk is on-line, then you get the installed Initial microcode; subsequent actions depend on what that microcode is (see the next section). If no Initial microcode is present or the disk is not on-line, you get the Alto/Mesa emulator.

Regardless of what is installed on the disk, you may bootstrap a specific emulator from an Ethernet boot server by holding a key down during the boot:

C	for DoradoCedar.eb (Pilot as modified for Cedar)
L	for DoradoLisp.eb
M	for DoradoMesa.eb (Alto/Mesa)
S	for DoradoSmalltalk.eb (Smalltalk-76)
T	for DoradoTest.eb (usually Cedar microcode under development)

For correct operation, you must hold the key down for at least 5 seconds after you initiate a 3-push boot.

These keys are independent of the ones used to select the desired *software* to run (with Alto-based emulators; see below). The latter keys are the same as on the Alto; that is, to run the NetExec, you hold down the BS and Quote keys while booting. Thus, to bootstrap the Smalltalk microcode and then run the NetExec, you might have to hold down S, BS, and Quote and initiate a 3-push boot. Since selecting this key combination would require you to have a pet octopus, some more convenient (though less mnemonic) alternate keys are defined for selecting the microcode to bootstrap. (In fact, these are the *only* microcode keys that work while BS is pressed down, because the normal keys form part of the Alto boot file number in this case.)

[	for Lisp
RETURN	for Alto/Mesa (same as all-keys-up boot)
SHIFT	for Smalltalk

1.3 Software booting

When Alto Emulator based microcode is loaded and started, it will then boot-load software from disk or Ethernet according to the standard Alto conventions. That is, if you are not holding down the BS key, the Alto Operating System will be booted from the disk; if you are holding down BS, a program will be booted from the Ethernet according to what other keys you are holding down simultaneously (e.g., BS and Quote boot the NetExec).

A Dorado’s disk can have up to 5 *partitions* (complete self-contained Alto file systems), numbered 1 through 5. Ordinarily, when you boot Alto Emulator based microcode, it selects partition 5 by default. However, if you hold down a digit key in the range 1 through 5 while booting, the corresponding partition will be selected instead. Also, a 1-push boot of already-running Alto Emulator based microcode will not change partitions unless you are holding down a digit key.

When Pilot or Cedar microcode is loaded and started, it will boot-load software from the disk according to standard Pilot conventions. Briefly, the installed Pilot germ’’ file is boot-loaded, which in turn loads the installed physical volume boot file’’.

Note that Ethernet booting of *software* is possible only with Alto emulator-based microprograms, not with Pilot or Cedar. To boot Pilot software (e.g., Othello) from the Ethernet, you must first boot the NetExec using Alto/Mesa, then use the NetExec to call the MesaNetExec, then use the MesaNetExec to call the desired program.

1.4 Initial and Pilot microcode

There are two places on a Dorado’s disk where you may install microcode. One is called the *Initial* region, which is a special region of the disk reserved for microcode. The other is called the *Pilot microcode* or *soft microcode* file, which is a more-or-less ordinary Pilot file which may reside in any Pilot volume. (If you don’t have a Pilot file system on your disk then you cannot install Pilot microcode. There can actually be a Pilot microcode file in each volume; but the booting logic pays attention only to the one that has been declared to be the physical volume Pilot microcode’’.)

The Initial microcode is what is initially loaded and started when you boot microcode from the disk, as described in section 1.2. You can choose from among many different pieces of microcode to install on the Initial region of your Dorado’s disk. The most useful ones are the following.

DoradoCedar.eb, DoradoLisp.eb, DoradoMesa.eb, DoradoSmalltalk.eb. These are identical to the microcode that is booted from the Ethernet if you hold down C, L, M, or T while booting.

DoradoInitialDisk.eb. This microcode simply invokes the installed Pilot microcode file.

DoradoInitialCedar.eb, DoradoInitialLisp.eb, DoradoInitialMesa.eb, DoradoInitialSmalltalk.eb. Each of these is essentially a combination of DoradoInitialDisk.eb and a copy of a particular emulator. If booted with the P key depressed, it invokes the installed Pilot microcode file; otherwise it starts the emulator microcode that was part of the Initial file itself.

DoradoInitialEtherCedar.eb, DoradoInitialEtherLisp.eb, DoradoInitialEtherMesa.eb, DoradoInitialEtherSmalltalk.eb. Each of these is essentially a combination of DoradoInitialDisk.eb and an Ethernet microcode boot loader configured to boot a particular emulator. If booted with the P key depressed, it invokes the installed Pilot microcode file; otherwise it boots the appropriate microcode from an Ethernet boot server.

If you normally use only standard, released versions of emulators, you are advised to install only one of the last group of microcode files on the Initial region of your disk. That way, you are guaranteed always to get the most up-to-date version of microcode from a boot server, and you need not install new versions on your disk every time the microcode's maintainer fixes a bug or makes an upward-compatible improvement.

On the other hand, if you are running non-standard or pre-release versions of software that require incompatible microcode, you will find it convenient to install that microcode on your disk. Disk booting of emulator microcode should also prove useful during transitions between incompatible versions of microcode.

All microcode is kept on [Indigo]<Dorado>, and is copied to [Indigo]<Cedar>Top> during each Cedar release. Both Initial and Pilot microcode are installed using the Othello utility program. Note that even though Othello is a Pilot program, it can be used to install *any* Initial microcode, even if you don't have a Pilot file system on your disk; of course, you do have to have a Pilot file system in order to install Pilot microcode.

The following is a sample dialogue for installing Initial and Pilot microcode:

Get into the Alto NetExec somehow.

```
> MesaNetExecCR
```

Wait a few seconds for Ether booting.

```
> Switches: nCR          do this only if you don't have a Pilot file system
> OthelloDoradoCR
```

Wait a few seconds for Ether booting.

```
Othello 7.0 of 17-Jun-82 17:34:02
Please login...
User: xxxxxx  Password: xxxxxxCR...GV...OK
> Open IndigoCR
Indigo Parc IFS 1.36L, File Server of May 17, 1982
> Directory Cedar>TopCR   or Directory Dorado
> Initial Microcode Fetch
Drive name: RD0CR
File name: DoradoInitialMesa.ebCR
Are you sure? Yes
Fetching...done
> Pilot Microcode Fetch
Logical volume name: OthelloCR
Pilot microcode file name: DoradoCedar.ebCR
Fetching...installing...done
Shall I use this for the physical volume?  Yes
>
```

1.5 Recommended microcode configurations

There are many possible ways of configuring the microcode on a Dorado's disk. The following configurations are suggested, though others are reasonable also.

For a Dorado running standard versions of Alto/Mesa, Lisp, or Smalltalk-76, and never running Pilot/Cedar:

Initial microcode: DoradoInitialEtherMesa.eb, DoradoInitialEtherLisp.eb, or DoradoInitialEtherSmalltalk.eb (Smalltalk-80 users should install DoradoInitialEtherMesa.eb, as the Smalltalk-80 system loads its own microcode)

Pilot microcode: none

For a Dorado running non-standard versions of Alto/Mesa, Lisp, or Smalltalk-76, and never running Pilot/Cedar:

Initial microcode: DoradoMesa.eb, DoradoLisp.eb, or DoradoSmalltalk.eb

Pilot microcode: none

For a Dorado running both an Alto-based emulator and Pilot/Cedar, and which is to boot the Alto-based emulator by default:

Initial microcode: DoradoInitialEtherMesa.eb, DoradoInitialEtherLisp.eb, or DoradoInitialEtherSmalltalk.eb

Pilot microcode: DoradoCedar.eb (invoked by booting with P’’)

For a Dorado running Pilot/Cedar, and which is to boot Pilot/Cedar by default:

Initial microcode: DoradoInitialDisk.eb

Pilot microcode: DoradoCedar.eb

Regardless of what microcode is installed, booting with one of C’’, L’’, M’’, S’’, or T’’ (or the alternate RETURN, SHIFT, or]’’) invokes the microcode designated by the depressed key; and this microcode always comes from the Ethernet.

1.6 Microcode self-loading

The Dorado can also load its own microcode under program control, using a special instruction (LoadRam) described in section 2. There exists a program called LoadMB that runs on the Dorado (in Alto emulation mode) and loads microcode from a file on the Dorado’s disk. The file must be in MB (micro binary) format, as produced by MicroD, and the microcode contained within that file must conform to certain conventions in order to be soft-bootable in this manner.

For example, the Mesa microcode is contained in a file called Mesa.MB. To load and start that microcode, type:

```
>LoadMB Mesa.mb
```

This loads the Dorado’s control store with the contents of Mesa.mb and then starts it at a special entry point that causes it to reset I/O devices but does not disturb emulator execution or the contents of main memory.

This means of loading microcode isn’t very interesting for bootstrapping the standard emulators, since the 3-push boot is easier and doesn’t require the new microcode to be on your Dorado’s disk. But it is useful for loading non-standard or experimental emulators. (The LoadMB program has some other features, described in section 3. The program is stored as [Indigo]<Dorado>LoadMB.run.)

1.7 Loading microcode using Midas

Each cluster of Dorados has an umbilical Alto that is able to control a connected Dorado for purposes such as hardware checkout and microcode debugging. The Alto program that performs these functions is called Midas.

Most Dorado users will have no occasion to use Midas. But if the normal microcode bootstrap mechanisms don’t seem to be working, you can use Midas to load your desired emulator microcode.

If Midas is not already running, boot the Alto and type:

```
>Midas
```

Midas first displays a menu of the serial numbers of all the Dorados that are connected to this Alto. Select your Dorado by positioning the cursor over your Dorado’s serial number and clicking RED (left or top mouse button). This step is skipped automatically if there is only one Dorado connected to the Alto.

If the selected Dorado is running, Midas displays a status of Running’’ and puts up a menu (near the bottom of the screen) consisting of ABORT, DTACH, and EXIT items only. To halt the Dorado, select ABORT.

Now Midas displays a large menu near the bottom of the screen. Select RUN-PROGRAM, and the menu changes to a list of microprograms that you can run; these include CEDAR, LISP, MESA, and SMALLTALK. Select one of those. The screen then goes blank while Midas loads the Dorado’s control store, a process that takes about a minute. After that, Midas starts the Dorado as if you had initiated a 2-push boot.

Important: *Do not leave Midas running on the Alto after you have loaded your microcode*, because multiple-push boots from the Dorado’s keyboard don’t work if Midas is controlling the Dorado. Therefore, exit Midas by selecting the EXIT menu item.

The versions of emulator microcode stored on the Midas Alto’s disk might not always be up-to-date. If you are in doubt, update the microcode files by issuing the Executive command:

```
>@UpdateEmulators
```

which runs FTP to retrieve the current versions of all emulator microcode.

1.8 Machine status

There is a green light on the front panel of the Dorado that indicates the machine’s present state, as follows:

1 flash. Bootstrap sequence in progress (power-on or 3-push boot).

2 flashes. Bootstrap sequence failed.

3 flashes. Transient power-supply problem voltage or current was out-of-spec at least once since the last boot.

4 flashes. Present power-supply problem.

5 flashes. Dorado is in shut-down state (never booted, or shut down by 7-push boot).

6 flashes. Dorado has overheated and shut itself down automatically.

7 flashes. Midas has taken over control of the Dorado.

Continuously on. Bootstrap sequence is believed to have completed successfully, and there have been no occurrences of the conditions indicated by 3 through 7 flashes.

Continuously off. Unknown but probably serious malfunction, or AC power turned off.