

Chapter 5

Virtual Partitions (vPars) on Integrity Servers

Introduction

With Virtual Partitions (vPars) you can take almost any HP Integrity server and turn it into many "virtual" computers. These virtual computers each run their own instance of HP-UX and associated applications. The virtual computers are isolated from one another at the software level. Software running on one Virtual Partition does not affect software running in any other Virtual Partition. In the Virtual Partitions you can run different patch levels of HP-UX, different applications, or any software you want and not affect other partitions.

About Virtual Partitions

There are some base requirements that must be met in order to run vPars on your system. At the time of this writing, the following minimum requirements must be met for each vPar on your system:

- An nPartition on a cell-based Integrity server with a minimum of one core.

- Sufficient memory to run HP-UX and any other software that will be present in the vPar.
- A boot disk off which HP-UX can be booted. The example in this chapter uses a cell-based Integrity system with four Itanium processors in the partition on which we're working. I recommend booting off internal disks but vPars can be SAN booted. There is a special procedure to setup fibre cards, at the time of this writing, that you'll want to discuss with your HP expert before booting off SAN.
- A console for managing the system which is a virtual terminal that can move between vPars in an nPartition with *Ctrl A*.
- An HP Integrity server supported by HP-UX 11i Version 2 or later. I use Version 2 in the example in this chapter but there are no significant differences between vPars under Version 2 versus later versions at the time of this writing. The configuration and EFI-related procedures are the same.

The nPartition on the rx8640 used in the example in this chapter has several I/O cards and a DVD and tape drive for each of the three vPars configured. This is an elaborate three-vPars setup in an nPartition.

The vPars product is mature so you can confidently use it in production environments, keeping in mind that full software isolation is employed in vPars and the hardware isolation is part of the nPartitions which are an integral part of the Integrity servers on which vPars run.

Virtual Partitions Background

HP-UX Virtual Partitions (vPars) allow you to run multiple instances of HP-UX on the same HP Integrity Server. From a hardware perspective, a vPar consists of processor, memory, and I/O that is a subset of the overall hardware in the nPartition. From a software perspective, a vPar consists of the HP-UX 11i Operating Environment and all application-related software to successfully run your workload. The following figure shows a conceptual diagram of the way in which HP computer-system resources can be allocated to support multiple vPars.

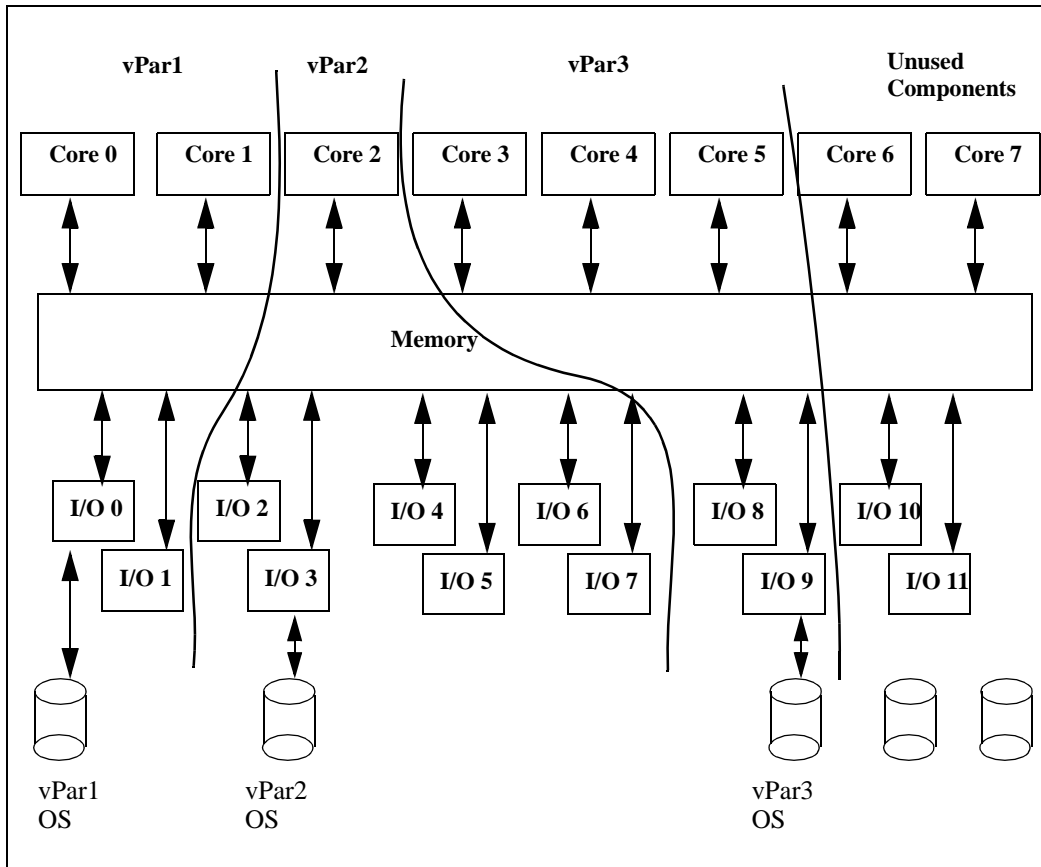


Figure 5-1 Example of a Four-Processor Node Partition Allocation with vPars

The components of which your HP server is comprised can be allocated in a variety of ways. You can see that the eight-way nPartition shown in the figure has a different number of cores, different amount of memory, and different number of I/O cards allocated to each vPar. The unused components can be added to any of the vPars or be the basis for yet another vPar. In addition, components can be moved from one vPar to another (with some restrictions described later in this chapter).

Uses of Virtual Partitions

I have worked on many vPars installations that have a variety of uses for vPars. The following is a sampling of the reasons to use vPars:

- | | |
|---|---|
| Increased System Utilization | Many servers are underutilized. With vPars, you can devote a subset of system resources to each vPar. With each vPar running its own instance of HP-UX 11i and associated applications, you get higher overall system utilization. |
| Quick Deployment | You can deploy a new environment quickly without procuring an entire new system. |
| Flexibility | Many applications have resource needs that change. With vPars, you can devote fewer system components when application needs are low and additional resources when an application needs them. An increased end-of-the-month workload, for example, can be given more system resources to complete faster. |
| Server Consolidation | Running multiple instances of HP-UX 11i and their associated applications on one HP server reduces the overall number of servers required. Web servers that had run on different servers can now run in different vPars on the same computer. |
| Application Isolation | HP vPars are fully software-isolated from one another. A software failure in one vPar does not affect other vPars. |
| Mixed Production, Test, and Development | Production and testing can take place on the same server with vPars. When testing is complete, the test vPar can become the production vPar. Similarly, development usually takes place on a separate system. With the software isolation of vPars; however; development can take place on the same system with other applications. |

These are just a sampling of the uses I've seen for vPars. Many others will emerge as vPars become widely used and systems experts implement them in more computing environments.

Loading the Software

The "Installing HP-UX," chapter covers loading HP-UX 11i in detail. If you haven't before loaded HP-UX 11i, this chapter helps you complete the task of loading 11i on all of the disks that you will use for your vPars. The following is a bullet list of steps you need to perform on every disk that will act as a vPar boot device:

1. Install the HP-UX 11i *Operating Environment*.
2. Set system parameters at the time of first boot after loading HP-UX 11i with **set_parms**.
3. Download and install select patches on your system (at the time of this writing there are many patches required to support vPars.)
4. Install vPars software.
5. Configure vPars.
6. Install additional software.

You would typically load software in the order just shown: Install the *Operating Environment*; boot your system and use **set_parms**; load patches; install vPars software; configure vPars; and then install and configure all other software.

I cover installing Virtual Partitions software in this section. I assume that you already have HP-UX 11i installed on your system or know how to do so. If you have not yet installed HP-UX 11i, see Chapter 3, which covers installing HP-UX 11i.

Keep in mind that HP-UX must be loaded for each Virtual Partition you want to run. If, for example, you want to run two Virtual Partitions, as we do in our examples in this book, HP-UX 11i needs to be loaded for both Virtual Partitions. The procedure covered for loading HP-UX 11i needs to be performed for every Virtual Partition you want to run. HP-UX 11i can be loaded from media, such as your HP-UX 11i distribution on a DVD or from

an Ignite/UX server. You can use any method to load HP-UX 11i and the Virtual Partitions software for every Virtual Partition you want to run.

All the vPars software must be loaded on every HP-UX 11i volume that will be used on your vPars server. The loading of this software takes place for every HP-UX 11i instance that you want to run simultaneously on your vPars server.

A lot of software has been loaded as a result of loading the vPars software. The `/sbin` directory has in it the `vpar` commands we'll use in upcoming sections. The following is a long listing of the `vpar` commands in `/sbin`:

```
$ ll /sbin/vp*
-r-xr-xr-x 1 bin      bin      265616 Jul 11  2006 /sbin/vparadmin
-r-xr-xr-x 1 bin      bin      243496 Jul 11  2006 /sbin/vparboot
-r-xr-xr-x 1 bin      bin      385384 Jul 11  2006 /sbin/vparcreate
-r-xr-xr-x 1 bin      bin      160536 Jul 11  2006 /sbin/vpard
-r-xr-xr-x 1 bin      bin      96960  Jul 10  2006 /sbin/vpardump
-r-xr-xr-x 1 bin      bin      278888 Jul 11  2006 /sbin/vparefiutil
-r-xr-xr-x 1 bin      bin      210880 Jul 11  2006 /sbin/vparenv
-r-xr-xr-x 1 bin      bin      38736  Jul 10  2006 /sbin/vparextract
-r-xr-xr-x 1 bin      bin      388424 Jul 11  2006 /sbin/vparmodify
-r-xr-xr-x 1 bin      bin      22680  Jul 10  2006 /sbin/vparreloc
-r-xr-xr-x 1 bin      bin      264912 Jul 11  2006 /sbin/vparremove
-r-xr-xr-x 1 bin      bin      215424 Jul 11  2006 /sbin/vparreset
-r-xr-xr-x 1 bin      bin      360504 Jul 11  2006 /sbin/vparstatus
-r-xr-xr-x 1 bin      bin      40800  Jul 11  2006 /sbin/vparutil
-r-xr-xr-x 1 bin      bin      20304  Jul 10  2006 /sbin/vphbd
psdev02:/>
```

These are the commands that you use to create, view, modify, and work with vPars in general.

There are several files in `/stand` related to the vPars kernel. The following listing shows some of them:

```
# ll /stand/vp*
-r-xr-xr-x 1 bin      bin      58223248 Jul 10  2006 /stand/vpmon
# ll /etc/rc.config.d/vpar*
-r--r--r-- 1 bin      bin      291 Apr 14  2004 /etc/rc.config.d/vpard
-r--r--r-- 1 bin      bin      553 Apr 14  2004 /etc/rc.config.d/vparhb
-r--r--r-- 1 bin      bin      1244 Mar 12  2005 /etc/rc.config.d/vparinit
# ll /sbin/init.d/vpar*
-r-xr-xr-x 1 bin      bin      793 Apr 14  2004 /sbin/init.d/vpard
```

`vpmon` is loaded at the time of system startup and is the basis for running vPars. `vpdb` is the vPars database that contains all information related to all the vPars running on your system. This file is automatically synchronized by the vPars monitor to ensure that all vPars have the same information about all vPars on your system.

There are several startup-related files, including those shown below, which are covered in more detail in the "System Startup and Shutdown" chapter.

```
# ll /etc/rc.config.d/vpar*
-r--r--r-- 1 bin      bin      291 Apr 14 2004 /etc/rc.config.d/vpard
-r--r--r-- 1 bin      bin      553 Apr 14 2004 /etc/rc.config.d/vparhb
-r--r--r-- 1 bin      bin      1244 Mar 12 2005 /etc/rc.config.d/vparinit

# ll /sbin/init.d/vpar*
-r-xr-xr-x 1 bin      bin      793 Apr 14 2004 /sbin/init.d/vpard
```

Very important to your work related to vPars are the online man pages. The following listing shows the man pages from *vpartition*:

```
$ man vpartition
```

Command	Description
vecheck	Check for virtual partition environment.
vparadmin	Modify virtual partition flexible administrative capability related attributes.
vparboot	Boot (start) a virtual partition.
vparcreate	Create a new virtual partition.
vpardump	Manage monitor dump files.
vparefiutil	Update EFI device paths of bootable disks in the vPar database. Itanium(R)-based platforms only.
vparenv	Set vPars or nPars mode, ILM or CLM granularity in system firmware, or display current values of these settings. Itanium-based platforms only.
vparextract	Extract memory images from a running virtual partition system.
vparmodify	Modify an existing virtual partition.
vparreloc	Relocate the load address of a vmunix file, determine if a vmunix file is relocatable, or promote the scope of symbols in a relocatable vmunix file. PA-RISC platforms only.
vparremove	Remove (delete) an existing virtual partition.
vparreset	Send a hard reset (TOC) to a virtual partition.
vparresources	Description of virtual partition resources and their requirements.
vparstatus	Display virtual partition and available resources information.
vparutil	Get and set SCSI parameters for disk devices from a virtual partition. PA-RISC platforms only.

You may have to run the **catman** command to create cat files for these manual pages.

At this point, we have HP-UX 11i and the Virtual Partitions software loaded on the system.

The remainder of this chapter covers numerous vPars topics, including creating, booting, and modifying vPars.

With both HP-UX 11i and the Virtual Partitions software on our disk, we can begin the process of creating partitions. Our goal is to have a system that looks like what's shown in the following figure.

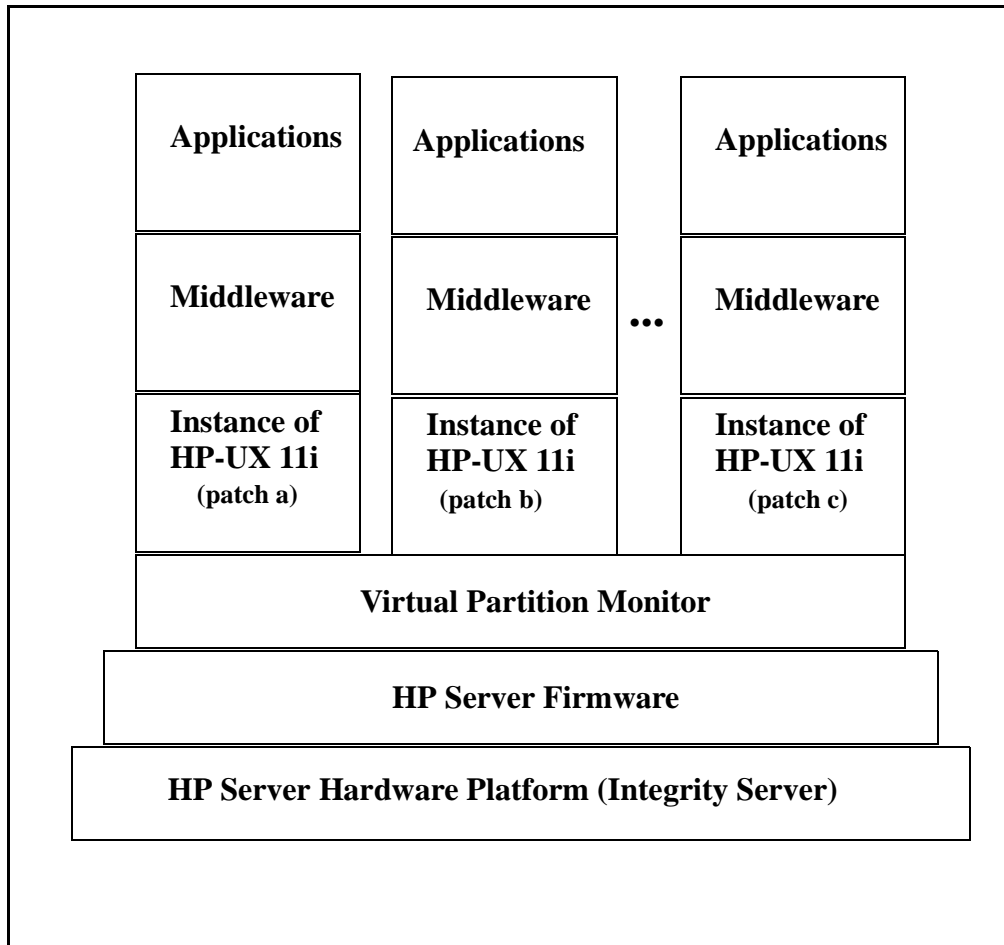


Figure 5-2 Virtual Partitions Software Stack

There are many components in this figure. We already have many of the components in this diagram on our system. Starting from the bottom, we

have the hardware, firmware, Virtual Partition Monitor, and HP-UX 11i installed on two different disks. There are two HP-UX 11i instances shown in the left-most two stacks of the figure. These are the operating systems that we have already loaded.

The two HP-UX 11i instances can't run simultaneously on a system because we have not yet created our Virtual Partitions. Without Virtual Partitions created, we can boot HP-UX off of one *or* the other of these disks, but we can't run both. Let's now create our Virtual Partitions so that we can have two instances of HP-UX 11i running simultaneously. After Virtual Partitions have been created, you can proceed to load the middleware and applications shown on top of the previous figure.

Virtual Partitions Command Summary

There are several commands used to create and work with Virtual Partitions. A table and a tear-out card in my vPars book, *HP-UX Virtual Partitions*, provide an overview of many commonly used Virtual Partitions related commands. The following table is an abbreviated version of the command summary and there is also the man page summary of commands from the **man vpartition** command:

Table 5-1 Virtual Partition Commands

Command	Description
<pre>Shell> fs0: EFI Shell prompt fs0:vaprconfig file system commands HPUX> boot vpmom HPUX prompt MON> vaprconfig vparmonitor</pre>	<p>Select file system from EFI shell:</p> <pre>Shell> fs0:</pre> <p>Run vparconfig command from file system prompt:</p> <pre>fs0:\> vparconfig vparconfig supports the following options: vparconfig vparconfig reboot vPars vparconfig reboot nPars fs0:\efi> vparconfig reboot vPars fs0:\efi> fs0:\efi> hpux</pre> <p>Boot <i>vpmom</i> from HPUX> prompt:</p> <pre>HPUX> boot vpmom</pre> <p>vPars monitor prompt from which you can load vPars with</p> <pre>MON> vparload -p psdev01</pre> <p>Many other commands can be issued from <i>MON</i>. Type help or ? to list. (Commands include: scan, vparinfo, ls, log, getauto, lfls, cbuf, cat.)</p>
<p>vparload</p> <p>Load Virtual Partitions from <i>MON</i>> prompt only.</p>	<p>To boot a Virtual Partition from <i>MON</i>>:</p> <pre>MON> vparload -p vPar_name</pre>
<p>vparboot</p> <p>Boot a Virtual Partition from the command line only.</p>	<p>To boot a Virtual Partition from the command line:</p> <pre># vparboot -p vPar_name</pre>
<p>vparcreate</p> <p>Create a Virtual Partition.</p>	<p>To create a Virtual Partition with three processors (<i>num</i>) total, two bound (<i>min</i>), 2048MB RAM, all components on 0/0, boot disk at 0/0/1/1.2.0, with a kernel of <i>/stand/vmunix</i>, autoboot on, and console at 0/0/4/0:</p> <pre># vparcreate -p vPar_name -a cpu::3 -a cpu:::2:4 -a mem::2048 -a io:0/0 -a io:0/0/1/1.2.0:boot -b /stand/vmunix -B auto</pre>

Command	Description
vparmodify Modify the attributes of a Virtual Partition.	To add processor at path <i>IO9</i> (adds this proc to those already assigned): # vparmodify -p vPar_name -a cpu:109
vparremove Delete a Virtual Partition.	To delete a Virtual Partition in the currently running database: # vparremove -p vPar_name
vparreset Reset a Virtual Partition.	To reset a Virtual Partition without TOC (t), hard (h), bypassing display of PIM data (q), or forcing (f): # vparreset -p vPar_name
vparresources(5) man page Provides description of Virtual Partitions and their resources.	This is a manual page that describes Virtual Partition resources in general and how resources are specified in other commands, such as vparmodify .
vparstatus Display the status of Virtual Partitions.	To display the status of a Virtual Partition in verbose mode: # vparstatus -v -p vPar_name
vpartition man page Display information about the Virtual Partition Command Line Interface.	Provides the following brief description of Virtual Partitions commands: vparboot Boot (start) a virtual partition. vparcreate Create a new virtual partition. vparmodify Modify an existing virtual partition. vparremove Remove (delete) an existing virtual partition. vparreset Simulate a TOC or hard reset to a virtual partition. vparstatus Display virtual partition and available resources information.
Specify CPU Resources by:	Number of bound and unbound CPUs: <i>cpu::num</i> CPU hardware path(s): <i>cpu:path</i> Minimum and maximum number: <i>cpu::[min][:[max]]</i>
Specify Memory by:	Size <i>mem::size</i> Base and range: <i>mem::base:range</i> combination of both above.
Specify I/O:	Use path: <i>io:path[:attr1[,attr2[...]]]</i> (see man page vparresources for details).

Command	Description
To add resources use: (This adds component relative to what already exists if running vparmodify .)	<p><i>-a cpu:path</i></p> <p><i>-a cpu::num</i> (can be done with vPar running)</p> <p><i>[-a cpu::num] [-a cpu:::[min]:[max]] [-a cpu:path]</i> (::: is vparcreate only)</p> <p><i>-a io:path[:attr1[,attr2[...]]]</i></p> <p><i>-a mem::size</i></p> <p><i>-a mem:::base:range</i></p>
To delete resources use (This deletes component relative to what already exists if running vparmodify .)	<p><i>-d cpu:path</i></p> <p><i>-d cpu::num</i> (can be done with vPar running)</p> <p><i>-d io:path[:attr1[,attr2[...]]]</i></p> <p><i>-d mem::size</i></p> <p><i>-d mem:::base:range</i></p>
To modify resources use: (This modifies to absolute number rather than relative.)	<p><i>-m cpu::num</i> (can be done with vPar running)</p> <p><i>-m cpu:::[min]:[max]]</i></p> <p><i>-m io:path[:attr1[,attr2[...]]]</i></p> <p><i>-m mem::size</i></p>
vPars mkboot Options: <i>-a</i> <i>-b</i> <i>-c</i> <i>-i</i>	<p>Creates and autoexecute file AUTO.</p> <p>Boot programs are installed.</p> <p>Checks for available space.</p> <p>Copies the included LIF file.</p> <p>To create nan AUTO file: mkboot -a "/stand/vpmon -a " /dev/rdisk/c0t0d0</p>
vPars States: <i>load</i> <i>boot</i> <i>up</i> <i>shut</i> <i>down</i> <i>crash</i> <i>hung</i>	<p>The kernel image of a Virtual Partition is being loaded into memory. This is done by the Virtual Partition monitor.</p> <p>The Virtual Partition is in the process of booting. The kernel image has been successfully loaded by the Virtual Partition monitor.</p> <p>The Virtual Partition has been successfully booted and is running.</p> <p>The Virtual Partition is in the process of shutting down.</p> <p>The Virtual Partition is not running and is down.</p> <p>The Virtual Partition has experienced a panic and is crashing.</p> <p>The Virtual Partition is not responding and is hung.</p>

```

$ ll /sbin/vp*
-r-xr-xr-x 1 bin      bin      265616 Jul 11  2006 /sbin/vparadmin
-r-xr-xr-x 1 bin      bin      243496 Jul 11  2006 /sbin/vparboot
-r-xr-xr-x 1 bin      bin      385384 Jul 11  2006 /sbin/vparcreate
-r-xr-xr-x 1 bin      bin      160536 Jul 11  2006 /sbin/vpard
-r-xr-xr-x 1 bin      bin      96960  Jul 10  2006 /sbin/vpardump
-r-xr-xr-x 1 bin      bin      278888 Jul 11  2006 /sbin/vparefiutil
-r-xr-xr-x 1 bin      bin      210880 Jul 11  2006 /sbin/vparenv
-r-xr-xr-x 1 bin      bin      38736  Jul 10  2006 /sbin/vparextract

```

```

-r-xr-xr-x 1 bin      bin      388424 Jul 11  2006 /sbin/vparmodify
-r-xr-xr-x 1 bin      bin      22680  Jul 10  2006 /sbin/vparreloc
-r-xr-xr-x 1 bin      bin      264912 Jul 11  2006 /sbin/vparremove
-r-xr-xr-x 1 bin      bin      215424 Jul 11  2006 /sbin/vparreset
-r-xr-xr-x 1 bin      bin      360504 Jul 11  2006 /sbin/vparstatus
-r-xr-xr-x 1 bin      bin      40800  Jul 11  2006 /sbin/vparutil
-r-xr-xr-x 1 bin      bin      20304  Jul 10  2006 /sbin/vphbd
psdev02: />

```

We use some of the commands shown in this table in the upcoming section on creating virtual partitions.

Let's now move on to creating virtual partitions.

Steps to Create Virtual Partitions

This section covers the steps to create Virtual Partitions. I performed these steps while working with some of my clients on their vPars environments. This list should serve as a framework for working with vPars. You may chose not to perform some of the steps and to add others. It is only a framework for getting vPars working on your system.

In our upcoming examples to create our Virtual Partitions, we execute the steps shown in the following listing

1. Connect to the MP on the partitionable server and select the partition to be configured with vPars.
2. Load HP-UX onto one of the disks in the nPartition. Although this is not HP terminology I call this the *master disk* because after HP-UX is loaded on it you can obtain information about all the components in the nPartition using **ioscan** and other commands.
3. Load vPars software onto the *master disk*.
4. Obtain information about all the components in the nPartition, that will be divided among several vPars, using **ioscan** and other commands.
5. List the components of which each vPar will be comprised including I/O cards, cores, memory, disk paths and device files, vPar names, networking information, and so on.
6. Create command files for all the vPars and run the command file for the vPar1 to create it on the *master disk*. Change *vparenv* to boot in vpars mode.
7. Boot the first vPar created on the *master disk* to see that it loads properly. This includes changing the boot mode in EFI to *vPars*.

8. Create a bootable Ignite image of the *master disk* that has on it the vPar1 using **make_tape_recovery -A -I -v** and restore it to the second disk for vPar2.
9. Create a tape boot device in EFI if it is not already present using **reconnect -r** and *Add a Boot Device*
10. Select the tape as the boot device in EFI and load the ignite image from it onto the second vPars disk.
11. Boot the vPar1 in *vPars* mode. Run the vPar create command file for vPar0 and vPar2 on vPar1 (*master disk*) and run **vparboot** from the vPar1 to see vPar2 boot. The tape of the *master disk* can then be used to load additional vPars on the nPartition using the same process.
12. Perform follow-up procedures such as running **mkboot** as shown in the following example with "-a" that will autoboot all vPars.

```
mkboot -a "/stand/vpmon -a " /dev/rdisk/c0t0d0
```

The following section covers these 12 steps in more detail.

Steps to Create Virtual Partitions

The following process of loading vPars was done by producing a perfect image on one vPar, creating an Ignite tape, and loading it on multiple vPars. An Ignite server would be a better solution but it is not always available and this technique, although it takes longer, will work.

1. Connect to the MP of the partitionable server using the serial connection or MP LAN. The following listing shows the four partitions on the server on which I'll be confiding *Partition 3*:

```
Partitions available:
#   Name
---  ----
0)  Partition 0 - HCM Database RAC Server 1
1)  Partition 1 - HCM Application Server 1
2)  Partition 2 - HCM Web Server 1
3)  Partition 3 - Three Virtual Partitions
Q)  Quit

Please select partition number:
```

2. Load HP-UX onto one of the disks in the nPartition. Although this is not HP terminology I call this the *master disk* because after HP-UX is loaded on it you can

obtain information about all the components in the nPartition using **ioscan** and other commands.

Loading HP-UX is covered in an earlier chapter. The filesystem layout of the operating system after the load is shown below:

```
# hostname
npar3
# bdf
Filesystem          kbytes    used    avail  %used  Mounted on
/dev/vg00/lvol13    589824    303736  283880   52%    /
/dev/vg00/lvol11    311296    231504   79272   74%    /stand
/dev/vg00/lvol18    5832704   2337880 3467560   40%    /var
/dev/vg00/lvol17    6750208   2557400 4160152   38%    /usr
/dev/vg00/lvol4     212992     8616   202912    4%    /tmp
/dev/vg00/lvol6    6963200   3777584 3160744   54%    /opt
/dev/vg00/lvol5     32768     8464   24120    26%    /home
#
```

3. Load vPars software onto the *master disk*. This can be done from a variety of sources. The **swlist** after vPars is loaded is shown in the following listing:

```
# swlist -l fileset | grep -i virtualp
# VirtualPartition          A.04.03.03    vPars Functionality
VirtualPartition.VPAR-ENG-A-MAN  A.04.03.03    vPars Manpages
VirtualPartition.VPAR-KRN      A.04.03.03    vPars Kernel Files
VirtualPartition.VPAR-MON2     A.04.03.03    Virtual Partition Monitor
VirtualPartition.VPAR-RUN      A.04.03.03    vPars User Space Commands
#

# ll /stand/vp*
-r-xr-xr-x  1 bin      bin          58223248 Jul 10  2006 /stand/vpmon
# ll /etc/rc.config.d/vpar*
-r--r--r--  1 bin      bin          291 Apr 14  2004 /etc/rc.config.d/vpard
-r--r--r--  1 bin      bin          553 Apr 14  2004 /etc/rc.config.d/vparhb
-r--r--r--  1 bin      bin          1244 Mar 12  2005 /etc/rc.config.d/vparinit
# ll /sbin/init.d/vpar*
-r-xr-xr-x  1 bin      bin          793 Apr 14  2004 /sbin/init.d/vpard
```

4. Obtain information about all the components in the nPartition that will be divided among several vPars using **ioscan** and other commands using various **ioscan** commands including the following:

```
ioscan -funC disk
ioscan -funC tape
ioscan -funC lan
ioscan -f          ; to get processor numbers, you may want to specify the cores
                   ; in a processor to be in the same vPar
```

tape:

```

# ioscan -funC tape
Class I H/W Path Driver S/W State H/W Type Description
=====
tape 0 3/0/1/1/0/4/1.0.0 stape CLAIMED DEVICE HP C7438
A
    /dev/rmt/0m /dev/rmt/c5t0d0BESTn
    /dev/rmt/0mb /dev/rmt/c5t0d0BESTnb
    /dev/rmt/0mn /dev/rmt/c5t0d0DDS
    /dev/rmt/0mnb /dev/rmt/c5t0d0DDSB
    /dev/rmt/c5t0d0BEST /dev/rmt/c5t0d0DDSn
    /dev/rmt/c5t0d0BESTb /dev/rmt/c5t0d0DDSnb
tape 1 3/0/2/1/0/4/0.0.0 stape CLAIMED DEVICE HP C7438
A
    /dev/rmt/1m /dev/rmt/c6t0d0BESTn
    /dev/rmt/1mb /dev/rmt/c6t0d0BESTnb
    /dev/rmt/1mn /dev/rmt/c6t0d0DDS
    /dev/rmt/1mnb /dev/rmt/c6t0d0DDSB
    /dev/rmt/c6t0d0BEST /dev/rmt/c6t0d0DDSn
    /dev/rmt/c6t0d0BESTb /dev/rmt/c6t0d0DDSnb
tape 3 3/0/10/1/0/4/0.0.0 stape CLAIMED DEVICE HP C7438
A
#

disk:
-----

# ioscan -funC disk
Class I H/W Path Driver S/W State H/W Type Description
=====
disk 1 3/0/0/2/0.6.0 sdisk CLAIMED DEVICE HP 73.4GST373454L
C
    /dev/dsk/c0t6d0 /dev/rdisk/c0t6d0
    /dev/dsk/c0t6d0s1 /dev/rdisk/c0t6d0s1
    /dev/dsk/c0t6d0s2 /dev/rdisk/c0t6d0s2
    /dev/dsk/c0t6d0s3 /dev/rdisk/c0t6d0s3
disk -5170A 0 3/0/0/2/1.2.0 sdisk CLAIMED DEVICE Optiarc DVD RW AD
C
    /dev/dsk/clt2d0 /dev/rdisk/clt2d0
disk 2 3/0/0/3/0.6.0 sdisk CLAIMED DEVICE HP 73.4GST373454L
C
    /dev/dsk/c2t6d0 /dev/rdisk/c2t6d0
disk 454LC 3 3/0/2/1/0/4/1.8.0 sdisk CLAIMED DEVICE HP 73.4GST373
    /dev/dsk/c7t8d0 /dev/rdisk/c7t8d0
disk 4 3/0/8/1/0/4/0.2.0 sdisk CLAIMED DEVICE _NEC DVD_R
W ND-3550A
    /dev/dsk/c8t2d0 /dev/rdisk/c8t2d0
disk 454LC 5 3/0/8/1/0/4/1.8.0 sdisk CLAIMED DEVICE HP 73.4GST373
    /dev/dsk/c9t8d0 /dev/rdisk/c9t8d0
disk 454LC 7 3/0/10/1/0/4/1.8.0 sdisk CLAIMED DEVICE HP 73.4GST373
    /dev/dsk/c11t8d0 /dev/rdisk/c11t8d0
disk 9 3/0/12/1/0/4/0.2.0 sdisk CLAIMED DEVICE _NEC DVD_R
W ND-3550A
    /dev/dsk/c12t2d0 /dev/rdisk/c12t2d0
disk 454LC 8 3/0/12/1/0/4/1.8.0 sdisk CLAIMED DEVICE HP 73.4GST373
    /dev/dsk/c13t8d0 /dev/rdisk/c13t8d0
#

dvd:
-----

# ioscan -funC disk | grep -i dvd
disk 0 3/0/0/2/1.2.0 sdisk CLAIMED DEVICE Optiarc DVD RW AD
-5170A
disk 4 3/0/8/1/0/4/0.2.0 sdisk CLAIMED DEVICE _NEC DVD_R
W ND-3550A
disk 9 3/0/12/1/0/4/0.2.0 sdisk CLAIMED DEVICE _NEC DVD_R
W ND-3550A
#

```

This is an elaborate nPartition with three tape drives, three DVDs, and six disk drives. Three vPars will be loaded on it with a tape and DVD for each and a root disk and mirror on each vPar.

- List the components of which each vPar will be comprised including I/O cards, cores, memory, disk paths and device files, vPar names, networking information, and so on. Here is an example of listing the data for three vPars:

```
vpar0:
3/0/0/0/1/0      1000Base-T builit-in LAN   Core I/O 3 near top
3/0/0/2/0.6.0   internal disk              disk bottom slot SEU front
3/0/0/2/1.x.0   internal DVD (2) DDS (3)  media bottom slot SEU front
3/0/0/3/0.6.0   internal disk              disk bottom slot SEU front
3/0/1/1         PCI-X 2.0 DDS              slot 8
3/0/14/1        PCI-X 2.0 FC               slot 4

vpar1:
3/0/8/1         root disk/DVD/LAN         slot 1
3/0/10/1        mirror disk/DVD/LAN       slot 2

vpar2:
3/0/2/1         mirror disk/DDS           slot 7
3/0/6/1         PCI-X 2.0 FC              slot 5
3/0/12/1        PCI-X 2.0 root/DVD/LAN    slot 3
```

- Create command files for all the vPars and run the command file for the vPar on the *master disk*:

```
vparcreate -p pstst02 -a io:3/0/0/2/0/6/0:BOOT -a cpu::4 -a mem::8192 -a io:3.0.0 -a
io:3.0.1 -a 3.0.4 -a io:3.0.14 -g ilm:1024:y

vparcreate -p psdev02 -a io:3/0/8/1/0/4/1.8.0:BOOT -a cpu::2 -a mem::4096 -a io:
3.0.8 -a io:3.0.10

vparcreate -p pstrn02 -a io:3/0/12/1/0/4/1.8.0:BOOT -a cpu::2 -a mem::4096 -a io
:3.0.2 -a io:3.0.6 -a io:3.0.12

"vparcreate0" 1 line, 121 characters
# /tmp/vparcreatel
Authorizing allocation of CPUs. Please wait...
vparcreate: Note: Database is created with the following information:
      CLM granularity is 128 MBytes
      ILM granularity is 1024 MBytes
All the subsequent virtual partition's CLM or ILM memory specifications must be an in-
tegral multiple of the corresponding granularity.
#
```

Now I change the next boot to vPars and issue reboot. This will bring up the nPartition in vPars mode.

```
# vparenv
vparenv: The next boot mode setting is "nPars".
vparenv: The ILM granule size setting is 1024.
vparenv: The CLM granule size setting is 128.
vparenv: Note: Any changes in the above settings will become effective
only after the next system reboot.
# vparenv -m vPars
```

```
# shutdown -r now
```

Note that I created the vPars with interleaved memory (ILM) of size 1024. You can also specify cell local memory (CLM) with vPars. My customers have found that ILM of 1024 works best in their environments and in only special circumstances would I use CLM.

7. Boot vPar2 (or whatever vPar has been loaded) on the *master disk* to see that it loads properly. In the following example the vPar is on *fs2*. You would normally view *fs0:*, *fs1:*, and so on. This includes changing the boot mode in EFI to *vPars* if you did not do this in step 6 with **vparsenv**.

```
Shell> fs2:
fs2:\> vparconfig

vparconfig supports the following options:
  vparconfig
  vparconfig reboot vPars
  vparconfig reboot nPars

fs2:\efi> vparconfig reboot vPars
fs2:\efi>
fs2:\efi> hpux

HPUX> boot vpmom

MON> vparload -p psdev01

Ignite mirrored tape:

File System Tab, Add/Remove Disk, select None for mirrored disk
```

8. Create a bootable Ignite image of the *master disk* that has on it the vPar1 using **make_tape_recovery -A -I -v** and restore it to the second disk for vPar2. You can do this in either nPars or vPars mode. If you go into nPars mode you'll have to reset the boot process:

```
# mt -f /dev/rmt/0m status
Drive: HP C7438A
Format:
Status: [0]
File: 0
Block: 0
#

# ioscan -funC tape
Class   I  H/W Path          Driver  S/W State  H/W Type  Description
=====
tape    2  3/0/10/1/0/4/0.0  stape   CLAIMED    DEVICE     HP         C7438A
         /dev/rmt/2m          /dev/rmt/c10t0d0BESTn
         /dev/rmt/2mb          /dev/rmt/c10t0d0BESTnb
         /dev/rmt/2mn          /dev/rmt/c10t0d0DDS
         /dev/rmt/2mnb        /dev/rmt/c10t0d0DDSB
         /dev/rmt/c10t0d0BEST  /dev/rmt/c10t0d0DDSn
         /dev/rmt/c10t0d0BESTb /dev/rmt/c10t0d0DDSnb
```

```
#
# /opt/ignite/bin/make_tape_recovery -A -I -v -a /dev/rmt/2mn
```

If you're installing from an existing Ignite tape on which mirroring was deployed be sure to unselect the mirrored disk in the Ignite screen *File System*, *Add/Remove Disk*, and *None* for the mirrored disk.

9. Create a new EFI entry for the tape device.

```
Shell> reconnect -r
IA64_EFI [Acpi(HWP0002,PNP0A03,300)/Pci(2|0)/Scsi(Pun6,Lun0)/HD(
IA64_EFI [Acpi(HWP0002,PNP0A03,300)/Pci(2|0)/Scsi(Pun6,Lun0)/HD(
Removable Media Boot [Acpi(HWP0002,PNP0A03,300)/Pci(2|1)/Scsi(Pu
Removable Media Boot [Acpi(HWP0002,PNP0A03,308)/Pci(1|0)/Pci(4|0
Removable Media Boot [Acpi(HWP0002,PNP0A03,30C)/Pci(1|0)/Pci(4|0
Load File [EFI Shell [Built-in]]
Load File [Acpi(HWP0002,PNP0A03,300)/Pci(1|0)/Mac(0018FE28C4B1)]
Load File [Acpi(HWP0002,PNP0A03,301)/Pci(1|0)/Pci(4|1)/Scsi(Pun0
Load File [Acpi(HWP0002,PNP0A03,301)/Pci(1|0)/Pci(6|0)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,301)/Pci(1|0)/Pci(6|1)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,302)/Pci(1|0)/Pci(4|0)/Scsi(Pun0
Load File [Acpi(HWP0002,PNP0A03,302)/Pci(1|0)/Pci(6|0)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,302)/Pci(1|0)/Pci(6|1)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,304)/Pci(1|0)/Mac(001A4B064E14)]
Load File [Acpi(HWP0002,PNP0A03,304)/Pci(1|1)/Mac(001A4B064E15)]
Load File [Acpi(HWP0002,PNP0A03,308)/Pci(1|0)/Pci(6|0)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,308)/Pci(1|0)/Pci(6|1)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,30A)/Pci(1|0)/Pci(4|0)/Scsi(Pun0
Load File [Acpi(HWP0002,PNP0A03,30A)/Pci(1|0)/Pci(6|0)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,30A)/Pci(1|0)/Pci(6|1)/Mac(0019B
Load File [Acpi(HWP0002,PNP0A03,30C)/Pci(1|0)/Pci(6|0)/Mac(0019B
Exit
```

This *reconnect -r* command at the EFI shell prompt shows all the components that EFI can see on the nPartition. To use the process of elimination to determine the tape device I use the following logic. The tape drive is a *Load File* and Scsi device that does not have a MAC address associated with it so I select the first device:

```
Load File [Acpi(HWP0002,PNP0A03,301)/Pci(1|0)/Pci(4|1)/Scsi(Pun0
```

With the tape drive having been identified, I hope, I go to the *Boot Option Maintenance Menu* and *Add a Boot Device*. I selected the tape device, gave it a name, and saved it as unicode. It now appears as a boot device in EFI as shown in the following listing:

```
EFI Boot Manager ver 1.10 [14.61] Please select a boot option

HP-UX Primary Boot: 3/0/0/2/0.6.0
Internal DVD (Lower) Cabinet 0 [Acpi(HWP0002,PNP0A03,100)/Pci(2|
SYS LAN 0 Cabinet 0 [Acpi(HWP0002,PNP0A03,0)/Pci(1|0)/Mac(0018FE
SYS LAN 1 Cabinet 0 [Acpi(HWP0002,PNP0A03,100)/Pci(1|0)/Mac(0018
EFI Shell [Built-in]
Internal DVD (Upper) Cabinet 0 [Acpi(HWP0002,PNP0A03,0)/Pci(2|1)
```


Under the *Root Disk* entry you can select the target disk device from the six disks in the nPartition:

```

Model                               Size (MB) Path                                x  x  x
x x x lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqkx  x
x x Enx x HP_73.4GST373454LC 70007 3/0/0/3/0.6.0 ^x  x
x x x x HP_73.4GST373454LC 70007 3/0/0/2/0.6.0 x  x
x x [ x x HP_73.4GST373454LC 70007 3/0/2/1/0/4/1.8.0 x  x
x x x x HP_73.4GST373454LC 70007 3/0/8/1/0/4/1.8.0 x  x
x x Fix x HP_73.4GST373454LC 70007 3/0/10/1/0/4/1.8.0 x  x
x x x x HP_73.4GST373454LC 70007 3/0/12/1/0/4/1.8.0 x  x
x x [ x x x
x x x x

```

3/0/0/3/0.6.0 is selected as the target disk for the first of the three vPars. This process will be completed for the other two vPars as well. You have to know the target disk paths for the vPars you're loading in order to use this process. The following are some common changes you might make in this process:

- *Basic* - change the target disk
- *File System* - change the root swap size
- *System* - change the IP address hostname

After all the changes are complete you can hit *Go* (hit "g" to go if you can't see it on the screen.)

Select "g" to go and proceed with the installation.

If the ignite load was an image that did not include vPars software you'll have to go back to all the ignite images and load vPars software. You'll know if vPars is loaded by issuing the following command:

```

pstrn01: />swlist -l fileset | grep -i virtualp
# VirtualPartition A.04.03.03 HP-UX Virtual Partitions Functionality
VirtualPartition.VPAR-ENG-A-MAN A.04.03.03 Virtual Partition Manpages
VirtualPartition.VPAR-KRN A.04.03.03 Virtual Partition Kernel Files
VirtualPartition.VPAR-MON2 A.04.03.03 Virtual Partition Monitor
VirtualPartition.VPAR-RUN A.04.03.03 Virtual Partition User Space
Commands
pstrn01: />

```

You'll have to go back to all the disks with the ignite images, however, in EFI only the last disk on which an ignite image was loaded will be displayed. This means you'll have to go to run the **reconnect -r** again at the *SHELL>* prompt (which rescans all the devices), and then *Boot Option Maintenance Menu* and *Boot From A File* select the disks on which the other images appear to display those disks as boot devices in EFI. All the devices

will appear in *Boot From A File* after the **reconnect -r** and you can identify (easier said than done) and select the desired disk from which to boot:

```
Shell> reconnect -r

Boot Option Maintenance Menu>

Boot From A File>

IA64_EFI [Acpi (HWP0002, PNP0A03, 300) /Pci (3|0) /Scsi (Pun6, Lun0) /HD ( ; vPar1 disk
IA64_EFI [Acpi (HWP0002, PNP0A03, 300) /Pci (3|0) /Scsi (Pun6, Lun0) /HD ( ; 1 mirror
IA64_EFI [Acpi (HWP0002, PNP0A03, 308) /Pci (1|0) /Pci (4|1) /Scsi (Pun8, ; vPar2 disk
IA64_EFI [Acpi (HWP0002, PNP0A03, 308) /Pci (1|0) /Pci (4|1) /Scsi (Pun8, ; 2 mirror
IA64_EFI [Acpi (HWP0002, PNP0A03, 30C) /Pci (1|0) /Pci (4|1) /Scsi (Pun8, ; vPar3 disk
IA64_EFI [Acpi (HWP0002, PNP0A03, 30C) /Pci (1|0) /Pci (4|1) /Scsi (Pun8, ; 3 mirror
Removable Media Boot [Acpi (HWP0002, PNP0A03, 300) /Pci (2|1) /Scsi (Pu
Removable Media Boot [Acpi (HWP0002, PNP0A03, 308) /Pci (1|0) /Pci (4|0
Removable Media Boot [Acpi (HWP0002, PNP0A03, 30C) /Pci (1|0) /Pci (4|0
Load File [EFI Shell [Built-in]]
  Load File [Acpi (HWP0002, PNP0A03, 300) /Pci (1|0) /Mac (0018FE28C4B1)]
Load File [Acpi (HWP0002, PNP0A03, 301) /Pci (1|0) /Pci (4|1) /Scsi (Pun0 ; tape
Load File [Acpi (HWP0002, PNP0A03, 301) /Pci (1|0) /Pci (6|0) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 301) /Pci (1|0) /Pci (6|1) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 302) /Pci (1|0) /Pci (4|0) /Scsi (Pun0
Load File [Acpi (HWP0002, PNP0A03, 302) /Pci (1|0) /Pci (6|0) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 302) /Pci (1|0) /Pci (6|1) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 304) /Pci (1|0) /Mac (001A4B064E14)]
Load File [Acpi (HWP0002, PNP0A03, 304) /Pci (1|1) /Mac (001A4B064E15)]
Load File [Acpi (HWP0002, PNP0A03, 308) /Pci (1|0) /Pci (6|0) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 308) /Pci (1|0) /Pci (6|1) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 30A) /Pci (1|0) /Pci (4|0) /Scsi (Pun0
Load File [Acpi (HWP0002, PNP0A03, 30A) /Pci (1|0) /Pci (6|0) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 30A) /Pci (1|0) /Pci (6|1) /Mac (0019B
Load File [Acpi (HWP0002, PNP0A03, 30C) /Pci (1|0) /Pci (6|0) /Mac (0019B
```

After the boot disk is selected select **HPUX** directory:

```
Select file or change to new directory:

04/07/07 12:26p <DIR>      4,096 .
04/07/07 12:26p <DIR>      0 ..
04/07/07 12:26p <DIR>      4,096 HPUX
04/07/07 12:26p <DIR>      4,096 Intel_Firmware
04/07/07 12:26p <DIR>      4,096 DIAG
04/07/07 12:26p <DIR>      4,096 HP
04/07/07 12:26p <DIR>      4,096 TOOLS
Exit
```

Next select **HPUX.EFI** and boot from it:

```
Select file or change to new directory:

04/07/07 12:26p <DIR>      4,096 .
04/07/07 12:26p <DIR>      4,096 ..
04/07/07 12:26p          644,703 HPUX.EFI
04/07/07 12:26p          24,576 NBP.EFI
Exit
```

Now you're booting the operating system on the selected disk.

As painful as this process is you need to know the devices on your partition that are listed when you issue **reconnect -r** so you can work with them. Once you know what devices the entries refer to you can create boot options linked to them if you like.

11. Boot the vPar1 in *vPars* mode. Run the vPar create command file for vPar0 and vPar2 on vPar1 (*master disk*) and run **vparboot** from the vPar1 to see vPar2 boot.

```
# vparstatus -v
[Virtual Partition Details]
Name:          pstst02
State:         Up
Attributes:    Dynamic,Autoboot,Nosearch
Kernel Path:  /stand/vmunix
Boot Opts:

[CPU Details]
Min/Max:      1/8
User assigned [Path]:
Boot processor [Path]: 3.121
Monitor assigned [Path]: 3.122
                                     3.123
                                     3.124

Non-cell-specific:
  User assigned [Count]:      0
  Monitor assigned [Count]:   4
Cell-specific [Count]:      Cell ID/Count
                             <none>

[IO Details]
  3.0.0.2.0.6.0 BOOT
  3.0.0
  3.0.1
  3.0.14

[Memory Details]
ILM, user-assigned [Base /Range]:
                               (bytes) (MB)
ILM, monitor-assigned [Base /Range]: 0x100000000/8192
                               (bytes) (MB)
ILM Total (MB):      8192
ILM Granularity (MB): 1024

CLM, user-assigned [CellID Base /Range]:
                               (bytes) (MB)
CLM, monitor-assigned [CellID Base /Range]:
                               (bytes) (MB)
CLM (CellID MB):

CLM Granularity (MB): 128

[Virtual Partition Details]
Name:          psdev02
State:         Up
Attributes:    Dynamic,Autoboot,Nosearch
Kernel Path:  /stand/vmunix
Boot Opts:

[CPU Details]
Min/Max:      1/8
User assigned [Path]:
Boot processor [Path]: 3.120
Monitor assigned [Path]: 3.125

Non-cell-specific:
```

```

    User assigned [Count]:      0
    Monitor assigned [Count]:  2
    Cell-specific [Count]: Cell ID/Count
                           <none>

[IO Details]
  3.0.8.1.0.4.1.8.0.0.0.0 BOOT
  3.0.8
  3.0.10

[Memory Details]
ILM, user-assigned [Base /Range]:
                    (bytes) (MB)
ILM, monitor-assigned [Base /Range]: 0x300000000/3072
                    (bytes) (MB)   0x4040000000/1024
ILM Total (MB): 4096

ILM Granularity (MB): 1024

CLM, user-assigned [CellID Base /Range]:
                    (bytes) (MB)
CLM, monitor-assigned [CellID Base /Range]:
                    (bytes) (MB)
CLM (CellID MB):

CLM Granularity (MB): 128

[Virtual Partition Details]
Name:      pstrn02
State:     Up
Attributes: Dynamic, Autoboot, Nosearch
Kernel Path: /stand/vmunix
Boot Opts:

[CPU Details]
Min/Max: 1/8
User assigned [Path]:
Boot processor [Path]: 3.126
Monitor assigned [Path]: 3.127

Non-cell-specific:
  User assigned [Count]:      0
  Monitor assigned [Count]:  2
  Cell-specific [Count]: Cell ID/Count
                           <none>

[IO Details]
  3.0.12.1.0.4.1.8.0.0.0.0 BOOT
  3.0.2
  3.0.12
  3.0.6

[Memory Details]
ILM, user-assigned [Base /Range]:
                    (bytes) (MB)
ILM, monitor-assigned [Base /Range]: 0x200000000/511
                    (bytes) (MB)   0x3c0000000/960
                                         0x4080000000/1024
                                         0x40c0000000/1022
ILM Total (MB): 3517

ILM Granularity (MB): 1024

CLM, user-assigned [CellID Base /Range]:
                    (bytes) (MB)
CLM, monitor-assigned [CellID Base /Range]:
                    (bytes) (MB)
CLM (CellID MB):

CLM Granularity (MB): 128

#

```

With all three vPars up-and-running you can issue Ctrl A to toggle between vPars as shown in the following listing:

```
# hostname
psdev02
#           ; issue Ctrl A to toggle to the next vPar pstrn02
[pstrn02]

# hostname
pstrn
#           ; issue Ctrl A to toggle to the next vPar pstst02
[pstst02]

# hostname
pstst02
#
```

You can't see the Ctrl A commands I issued to toggle between the three vPars so I commented it into the listing.

12. Follow-up work and miscellaneous comments.

There may be times when you want to remove the vPars database and start the process over. Please be careful with this command, however, I use this occasionally when creating vpars.

```
rm /stand/vpdb
```

The **lvlnboot** command was used extensively throughout this procedure to get information on the boot device as shown in the following listing:

```
# lvlnboot -v
Boot Definitions for Volume Group /dev/vg00:
Physical Volumes belonging in Root Volume Group:
    /dev/dsk/c0t6d0s2 (3/0/0/2/0.6.0) -- Boot Disk
Boot: lv011   on:      /dev/dsk/c0t6d0s2
Root: lv013   on:      /dev/dsk/c0t6d0s2
Swap: lv012   on:      /dev/dsk/c0t6d0s2
Dump: lv012   on:      /dev/dsk/c0t6d0s2, 0

# hostname
pstst02
#

# hostname
psdev02
# lvlnboot -v
Boot Definitions for Volume Group /dev/vg00:
Physical Volumes belonging in Root Volume Group:
    /dev/dsk/c9t8d0s2 (3/0/8/1/0/4/1.8.0) -- Boot Disk
Boot: lv011   on:      /dev/dsk/c9t8d0s2
Root: lv013   on:      /dev/dsk/c9t8d0s2
Swap: lv012   on:      /dev/dsk/c9t8d0s2
Dump: lv012   on:      /dev/dsk/c9t8d0s2, 0

#
```

```
# lvinboot -v
Boot Definitions for Volume Group /dev/vg00:
Physical Volumes belonging in Root Volume Group:
    /dev/dsk/c13t8d0s2 (3/0/12/1/0/4/1.8.0) -- Boot Disk
Boot: lv011    on:      /dev/dsk/c13t8d0s2
Root: lv013    on:      /dev/dsk/c13t8d0s2
Swap: lv012    on:      /dev/dsk/c13t8d0s2
Dump: lv012    on:      /dev/dsk/c13t8d0s2, 0

#
```

There are other follow-up tasks as well including setting up the boot mode as shown below:

```
mkboot -a "/stand/vpmon -a " /dev/rdisk/c0t0d0
```

The "-a" after the **vpmon** will autoboot all vPars.