



Charon-AXP V4.12 for Linux User's Guide

Contents

Introduction	3
About this guide	6
Charon-AXP for Linux installation	8
Running Charon-AXP for Linux	24
CHARON-AXP for Linux configuration	30
Migration to CHARON-AXP for Linux	45
CHARON-AXP for Linux virtual network	53
CHARON-AXP for Linux Licensing	55
CHARON-AXP for Linux HASP licensing	56
CHARON-AXP for Linux VE Licensing	66
CHARON-AXP for Linux utilities	71
mkdskcmd	72
mtd	75
hasp_srm_view	77
hasp_update	79
ncu	80
CHARON Guest Utilities for OpenVMS	88
CHARON-AXP for Linux configuration Details	92
General Settings	93
Core Devices	103
Console	120
Remote Management Console (RMC)	128
Placement of peripheral devices on PCI bus	132
Disks and tapes	169
KZPBA PCI SCSI adapter	170
KGPSA-CA PCI Fibre Channel adapter	179
Acer Labs 1543C IDE/ATAPI CD-ROM adapter	202
PCI I/O Bypass controller	203
Finding the target "/dev/sg" device	209
Networking	211
PBXDA PCI serial lines adapter	219
AlphaStation Sound Card (AD1848) emulation	223
PBXGA graphics card	224
Sample configuration files	229
CHARON-AXP for Linux deinstallation	230
Appendixes	231
glibc.i686 installation without Internet connection	232
How to implement time synchronisation between CHARON-AXP Host OS and Guest OS	236

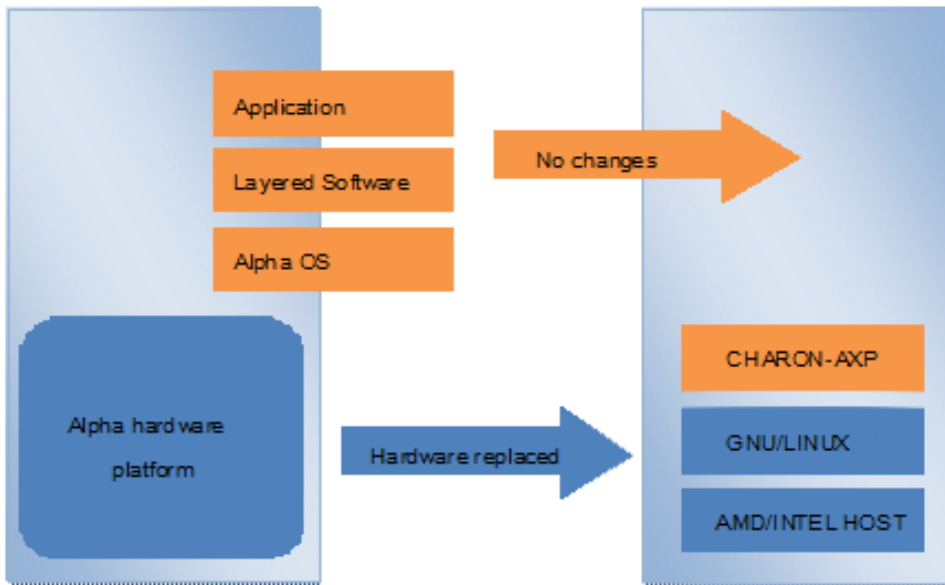
Introduction

Table of Contents

- General Description
- The principles of Alpha Hardware Virtualization
 - Virtualized hardware
 - Host platform

General Description

Alpha Hardware Virtualization allows users of Alpha (known as DIGITAL Alpha, Compaq Alpha or HP Alpha) computers to move application software and user data to a modern Intel or AMD based x64 compatible platform without having to make changes to software and data. Alpha Hardware Virtualization is a software solution that replaces Alpha hardware.



This approach is best understood when the Alpha Hardware Virtualization Software is viewed as a special interface between the old Alpha software and a new hardware platform. Basically, the Charon software presents an Alpha hardware interface to the original Alpha software, so that the existing software cannot detect a difference. This means no changes have to be made to the existing software. User programs and data can be copied to a new modern industry standard server (64-bit Intel or AMD) and continue to run for many more years.

The Alpha virtualization software is designed to replace single and multi-CPU Alpha computer systems, including:

- | | |
|--|--|
| <ul style="list-style-type: none"> • AlphaServer 400 • AlphaServer 800 • AlphaServer 1000 • AlphaServer 1000A • AlphaServer 1200 • AlphaServer 2000 • AlphaServer 2100 • AlphaServer 4000 • AlphaServer 4100 • AlphaServer DS10 • AlphaServer DS10L | <ul style="list-style-type: none"> • AlphaServer DS15 • AlphaServer DS20 • AlphaServer DS25 • AlphaServer ES40 • AlphaServer ES45 • AlphaServer GS80 • AlphaServer GS160 • AlphaServer GS320 |
|--|--|

The principles of Alpha Hardware Virtualization

Virtualized hardware

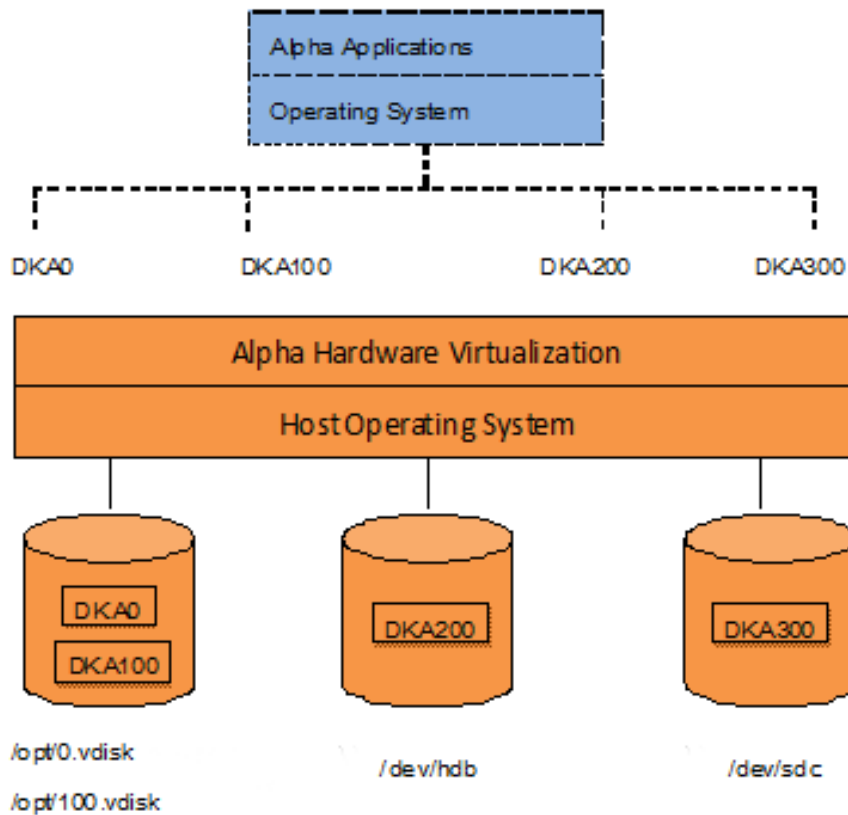
Charon-AXP virtualizes various Alpha architectures and meets or exceeds the performance level of these Alpha systems when run on the recommended hardware platform.

The following table shows which hardware boards Charon virtualizes:

Subsystem	Covered Alpha hardware
Serial Lines Controllers	On-board serial line ports COM1 and COM2, PBXDA
IDE/ATAPI CD-ROM Controller	Virtual Acer Labs 1543C
PCI Fibre Channel Controller	KGPSA-CA
PCI SCSI Controller	KZPBA
PCI Network Controllers	DE435, DE450, DE500AA, DE500BA, DE602, DE602AA
PCI Audio Controllers	PCXBJ
PCI Graphics card	PBXGA

Host platform

The Virtualization Software presents standard Alpha devices to the Alpha operating system, allowing the OS to function as though it were still running on an Alpha computer. For example, virtual disk container files in a directory or physical devices of the host Linux platform are presented by the Virtualization Software to the Alpha OS as emulated SCSI disks attached to a PCI SCSI adapter.



With the use of current storage technology, disks do not have to be physically attached to the Host platform, they can also reside on a SAN or iSCSI storage structure. A similar translation process is also valid for other emulated hardware devices.

About this guide

Table of contents

- Obtaining Documentation
- Obtaining Technical Assistance or General Product Information
 - Obtaining Technical Assistance
 - Obtaining General Product Information
- Conventions
- Definitions
- Related documents

Obtaining Documentation

The latest released version of this manual and other related documentation are available on the Stromasys support website at [Product Documentation and Knowledge Base](#).

Obtaining Technical Assistance or General Product Information

Obtaining Technical Assistance

Several support channels are available to cover the Charon virtualization products.

If you have a support contract with Stromasys, please visit <http://www.stromasys.com/support/> for up-to-date support telephone numbers and business hours. Alternatively, the support center is available via email at support@stromasys.com.

If you purchased a Charon product through a Value-Added Reseller (VAR), please contact them directly.

Obtaining General Product Information

If you require information in addition to what is available on the Stromasys [Product Documentation and Knowledge Base](#) and on the [Stromasys web site](#) you can contact the Stromasys team using <https://www.stromasys.com/contact/>, or by sending an email to info@stromasys.com.

For further information on purchases and the product best suited to your requirements, you can also contact your regional sales team by phone:

Region	Phone	Address
Americas	+1 919 239 8450	Stromasys LLC 871 Marlborough Ave, suite 100, Riverside CA 92507 USA
Europe, Middle-East and Africa	+41 22 794 1070	Avenue Louis-Casai 84 1216 Cointrin Switzerland

Conventions

Notation	Description
\$	The dollar sign in interactive examples indicates an operating system prompt for VMS. The dollar sign can also indicate non superuser prompt for UNIX / Linux.
#	The number sign represents the superuser prompt for UNIX / Linux.
>	The right angle bracket in interactive examples indicates an operating system prompt for Windows command (cmd.exe).
User input	Bold monospace type in interactive examples indicates typed user input.
<path>	Bold monospace type enclosed by angle brackets indicates command parameters and parameter values.
Output	Monospace type in interactive examples, indicates command response output.
[]	In syntax definitions, brackets indicate items that are optional.
...	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
<i>dsk0</i>	Italic monospace type, in interactive examples, indicates typed context dependent user input.

Definitions

Term	Description
Host	The system on which the emulator runs, also called the Charon server
Guest	The operating system running on a Charon instance, for example, Tru64 UNIX, OpenVMS, Solaris, MPE or HP-UX

Related documents

- [Charon-AXP V4.12 for Linux - User's Guide](#)
- [Charon-AXP V4.12 for Linux - Release Notes](#)

Charon-AXP for Linux installation

Table of contents

- Introduction
- Hardware Requirements
 - Number of CPU cores
 - Disable Hyper-Threading
 - CPU type and speed
 - Host memory
 - Disk storage
 - Ethernet adapters
- Software Requirements
- Host system preparation
- Before installation
- Distribution preparation
- Installation
- Charon-AXP home directory
- Non-privileged user account creation
- Charon License Installation
 - VE Licensing
 - HASP Licensing
 - Regular HASP USB dongle
 - Network HASP USB dongle
 - Software license
 - License validity verification
 - Troubleshooting
- Network configuration
 - Configuration with NCU utility
 - Manual Configuration
 - Choosing a network interface
 - Designation of network interface to Charon
 - Switching off the offload parameters
- Final steps
- Upgrade from previous version

Introduction

The Charon-AXP product is distributed as a self-extracting shell archive ("SHAR"). The SHAR is executed in order to unpack it. Before you can unpack the contents you must accept the EULA. The result is a set of RPM packages that are installed using the standard yum or dnf commands.

The RPM modules provide different components. Generally it is recommended to install all the RPM modules but it is possible to omit some RPM packages if they are not needed.

Charon installation consists of the following steps:

- Host system checks (hardware and software) to ensure the host platform meets the minimum Charon-AXP installation requirements
- Installation of any 3rd party material, for example, the utilities required for Charon-AXP
- Running the SHAR to unpack the RPM modules and their individual installation
- Installation of the Charon-AXP license (hardware dongle or software license)
- Configuration of the Charon-AXP host system. It assumes creating a specific user, configuring the network, etc.

Hardware Requirements

Number of CPU cores

Each Charon-AXP emulated CPU requires a corresponding physical core. So the total number of the host CPUs must exceed the number of emulated CPUs since some of the host CPUs must be dedicated to serving Charon I/O operations and host operating system needs. If several Charon instances run in parallel, the required number of CPU cores is cumulative.

The following table lists the minimum and recommended number of CPUs required for each virtual Alpha instance (note that each Charon instance is able to run on 2 CPU cores hosts, but this configuration does not support emulation of all the virtual CPUs):

Charon-AXP product	Minimum number of host CPU cores	Recommended number of host CPU cores
AlphaServer 400 - AlphaServer 4100	2	2
AlphaServer DS10/DS10L/DS15	2	2
AlphaServer DS20/DS25	4	4
AlphaServer ES40/ES45	6	8
AlphaServer GS80	10	16
AlphaServer GS160	18	32
AlphaServer GS320	34	48

When starting, the Charon-AXP software checks the available number of host CPU cores. This check is based on the maximum number of AXP CPUs that can be emulated if this number is not restricted by the "n_of_cpus" parameter. If the available number of host CPU cores is below this number, Charon-AXP will issue a warning message even if the requirements for the configured number of AXP CPUs are fulfilled. The Charon-AXP software will work despite this warning if the requirements for the configured number of AXP CPUs are fulfilled.

Disable Hyper-Threading

Hyper-threading should be switched off completely. Disable hyper-threading in the BIOS settings of the physical host or, for a VMware virtual machine, edit the virtual machine properties, select the Resources tab then select Advanced CPU. Set the Hyper-threaded Core Sharing mode to *None*. If Hyper-threading cannot be disabled, please contact Stromasys support for alternative resource requirements and instructions.

CPU type and speed

Platform emulation is a complex and CPU-intensive task. The performance of the individual cores of the host processors is the single most important factor which determines the emulated CPU performance. For the best performance, Stromasys recommends using the fastest available x86_64 processors of the latest generation. At the time of writing, the current generation of Intel server processors is "4th Gen Intel Xeon® Scalable Processors". The current generation of AMD server processors is "4th Generation AMD EPYC™ Processors".

There is a trade-off between the number of cores a processor provides versus the base (or "sustained all-core") frequency – the higher the number of cores, the lower the frequency. In order to choose the best processor, you must understand the workload you wish to emulate. If your workload can be easily distributed across a large number of CPUs ("multi-threaded"), a host processor with a higher number of cores running at a lower frequency might be appropriate. If, on the other hand, your workload consists of single-threaded tasks, probably a host processor with a low number of cores running at the highest possible frequency would be best.

Both Intel and AMD processors have the capability of increasing the frequency on one or very few cores when the other cores are idle. Intel calls this "Max Turbo Frequency" and AMD calls it "Max. Boost Clock". Because the emulator consists of many threads that all need their own host CPUs, this single/few CPU turbo/boost frequency is of no consequence. You should consider only the "Base Frequency/Clock", or, if a processor is capable, the "all-core turbo/boost speed", which is the frequency that all cores can reach when under load.

in general we recommend that the CPU frequency be 3 GHz or higher.

Host memory

The minimum host memory size:

- depends on the amount of Alpha memory to be emulated and on the number of Charon-AXP instances to be executed on one host.
- is calculated according to the following formula:

The minimum host memory = (2Gb + the amount of Alpha memory emulated) per Charon-AXP instance.

Disk storage

The total amount of disk space required for Charon-AXP can be calculated as a sum of:

- 500 MB for the Charon software
- All the disk/tape image sizes plus 500 MB for the Charon software
- The space required for the host operating system.

Keep in mind that Temporary disk storage is often needed when setting up a new emulator instance, for example for source disks backups storage, software installation kits, and others.

When virtual disks/tapes are used to represent physical disk drives / magnetic tapes, the disk/tape image files have the same size as their hardware equivalent, regardless of their degree of utilization.

Ethernet adapters

Charon-AXP networking requires dedicated host Ethernet adapters; their number must be equal to the emulated adapters to be configured in Charon-AXP. One adapter (optionally) can be left to the host for TCP/IP networking, management interface, etc.

It is also possible to use [virtual network interfaces](#), but for performance considerations, it is recommended to use physical ones only.

Software Requirements

- Red Hat Enterprise Linux (RHEL) and Oracle Linux 7.x to 9.x (64-bit)
- Rocky Linux 8.x and 9.x (64-bit)
- CentOS 7.x (64-bit)
- Hypervisors: VMware ESXi 5.5 – 8.0; Microsoft Hyper-V; KVM (require a supported Linux operating system running in the virtual machine). Note that prerequisites of additional products may limit the choice of hypervisors. For example, a VE license server VM requires VMware ESXi 6.5 or higher. Please refer to the appropriate documentation.

Host system preparation

The automatic installation of updates must be disabled. Updates to the Charon host must be done only in specific service maintenance periods established by the system administrator. Before applying new updates one must shutdown the operating system running on Charon and stop all the running Charon instances and services.

If a network-wide license (red dongle or software license) is going to be used, do the following:

- *On the license server (where the network license will reside)*: open port 1947 for both TCP and UDP
- *On the client*, if broadcast search for remote licenses is to be used, UDP traffic from port 1947 of the license server to ports 30000-65535 of the client must be permitted
 - If stricter firewall rules are required, it is possible to open the ports 30000-65535 and 1947 only for the `"/usr/sbin/hasplmd"` daemon.
- *Both on license server and client*: set default gateway

Please consult with your Linux User's Guide on details.

Before installation

1. Create a directory for the Charon-AXP distribution as shown in the following example:

```
# mkdir /charon_dist
```

2. On RHEL/CentOS 7, and RHEL 8, the "libev" package is required. If it is reported as missing during Charon installation on RHEL 7/8, check that the repository "extras" is included and enabled, if not, include and enable it. Please refer to your Linux distribution administrator's guide. The "libev" package is included in the "Base-OS" repository for RHEL/Rocky Linux 9, so there is no need to enable or install an additional repository.

Command to enable the "extras" repository for RHEL 7.x:

```
# yum-config-manager --enable rhel-7-server-extras-rpms
```

WARNING

- If you plan to install Charon-VAX on the same server, both products, Charon-AXP and Charon-VAX, must have the same build number.
- If you upgrade from a previous version of Charon-AXP, please stop all running Charon virtual machines, uninstall Charon products and **reboot the Linux server** (recommended) before proceeding with the installation steps described below.

Distribution preparation

Starting with version 4.12, Charon-AXP is delivered as a self-extracting shell-archive with a file name format as follows:

```
# cp /tmp/charon-axp-<VER>-<BN>.<ZZ>.sh /charon_dist
```

where:

Item	Description
VER	Version of Charon-AXP product, for example 4.12
BN	Build Number of Charon-AXP product, for example 21009
ZZ	Charon-AXP target operating system identifier where: <ul style="list-style-type: none"> ■ ZZ = "el90" for RHEL/Rocky Linux 9 ■ ZZ = "el8" for RHEL/Rocky Linux 8 ■ ZZ = "el74" for RHEL/CentOS/Rocky Linux 7

To **unpack the archive**, perform the following steps:

- Copy the package file to some location in your filesystem, for example `/var/tmp/charon-axp-4.12-21009-el90.sh`
- Go to the directory where you wish to unpack the package, for example `/charon_dist`
- Run the archive shell script: `# sh /path/to/<archive-name>`
For example: `# sh /var/tmp/charon-axp-4.12-21009-el90.sh`
- Accept the EULA. To successfully unpack the archive, the end-user license agreement must be accepted.
- After this, the software packages making up the Charon-AXP kit will be extracted into a version-specific sub-directory of the current working directory of the user.

Example command and output:

```
# sh /var/tmp/charon-axp-4.12-21009.e190.sh
Verifying archive integrity... 100% MD5 checksums are OK. All good.
Uncompressing Charon-AXP for Linux, Version 4.12 (Build 21009) 100%
End User License Agreement for : STROMASYS SOFTWARE.

<...lines removed...>

Please confirm EULA (yes/no) > yes
EULA accepted
gpg: directory `/home/root/.gnupg' created
gpg: new configuration file `/home/root/.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/root/.gnupg/gpg.conf' are not yet active during this run
<...lines removed...>
```

Example Command:

```
# cd /charon_dist
# sh /var/tmp/charon-axp-4.12-21009-e190.sh
```

As a result, the new subdirectory "charon-axp-<VER>-<BN>.<ZZ>" will be created.

Switch to that directory:

```
# cd charon-axp-<VER>-<BN>.<ZZ>
```

Example:

```
# cd charon-axp-4.12-21009.e190
```

The distribution directory contains the following RPM files:

File name	Description
aksusbd-8.13-1.x86_64.rpm	★ HASP Run-time
charon-axp-VER-BN.ZZ.x86_64.rpm	Charon-AXP
charon-license-VER-BN.ZZ.x86_64.rpm	★ Charon Libraries
charon-mtd-VER-BN.ZZ.x86_64.rpm	MTD utility
charon-utils-VER-BN.ZZ.x86_64.rpm	Charon Utilities

★ These packages are only required if you plan to use HASP licensing. If you plan to use VE licensing, you should not install these packages.

Example:

```
# ls
aksusbd-8.13-1.x86_64.rpm
charon-axp-4.12-21009.e174.x86_64.rpm
charon-license-4.12-21009.e174.x86_64.rpm
charon-mtd-4.12-21009.e174.x86_64.rpm
charon-utils-4.12-21009.e174.x86_64.rpm
```

Installation

Issue the following command to install all the RPM files present in the directory:

```
# yum install *.rpm
```

If you plan to use VE licensing, you can use the following command to only install the packages required for that:

```
# yum install charon-{axp,mtd,utils}-*.rpm
```

Enter "y" to agree to install all the listed packages.

Example:

```
Dependencies Resolved

=====
=
Package Arch Version Repository Size
=====
=
Installing:
aksusbd x86_64 8.13-1 /aksusbd-8.13-1.x86_64 2.9 M
charon-axp x86_64 4.12-21009 /charon-axp-4.12-21009.e174.x86_64 260 M
charon-license
x86_64 4.11-20404 /charon-license-4.12-21009.e174.x86_64 2.9 M
charon-utils
x86_64 4.11-20404 /charon-utils-4.12-21009.e174.x86_64 1.8 M
charon-mtd
x86_64 4.11-20404 /charon-mtd-4.12-21009.68704.e174.x86_64 1.2 M

Transaction Summary
=====
=
Install 4 Packages

Total size: 267 M
Installed size: 267 M
Is this ok [y/d/N]:y
```

Check the installation process has completed successfully.

Example:

```

Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction (shutdown inhibited)
Installing : aksusbd-8.13-1.x86_64 1/4
Starting aksusbd (via systemctl): [ OK ]
Installing : charon-utils-4.12-21009.x86_64 2/5
Installing : charon-mtd-4.12-21009.x86_64 3/5
Installing : charon-license-4.12-21009.x86_64 4/5
Installing : charon-axp-4.12-21009.x86_64 5/5
Verifying : aksusbd-8.13-1.x86_64 1/5
Verifying : charon-license-4.12-21009.x86_64 2/5
Verifying : charon-axp-4.12-21009.x86_64 3/5
Verifying : charon-utils-4.12-21009.x86_64 4/5
Verifying : charon-mtd-4.12-21009.x86_64 4/5

Installed:
aksusbd.x86_64 0:8.13-1 charon-axp.x86_64 0:4.12-21009
charon-license.x86_64 0:4.12-21009 charon-utils.x86_64 0:4.12-21009
charon-mtd.x86_64 0:4.12-21009

Complete!


```

Re-login (as "root") to apply the PATH settings or execute the following command:

```
# . /etc/profile.d/charon.sh
```

If "PuTTY" terminal emulator is going to be used as an additional option copy the following file to your home directory:

```
# mkdir -p $HOME/.config/putty/sessions (if it does not already exist)
# cp /opt/charon/putty/sessions/CHTERM-VT100 $HOME/.config/putty/sessions
```

 Note that the "charon-utils" package has the following dependencies:

- ethtool
- bridge-utils
- net-tools
- iproute
- NetworkManager

During "charon-utils" installation using "yum", these packages, with the exception of "bridge-utils", will be installed automatically from the standard repository if some of them are absent on the host system. In order to install the "bridge-utils", you must first install the EPEL (Extra Packages for Enterprise Linux) repository. Please see the EPEL web page for how to do this:

<https://docs.fedoraproject.org/en-US/epel/>

Charon-AXP home directory

By default Charon is installed in the `"/opt/charon/"` directory. It has the following subdirectories:

Directory	Description
<code>/bin</code>	Contains all the executable files
<code>/cfg</code>	Contains the configuration files templates
<code>/doc</code>	Contains the documentation
<code>/log</code>	Contains the log files
<code>/disks</code>	Contains the disk containers
<code>/drivers</code>	Contains the Charon drivers

The most important directory at this stage is the `"/cfg"` directory since it contains template configuration files with examples of typical configuration parameters and comments. This will be described in the next chapter.

Non-privileged user account creation

Create a non-privileged user account named `"charon"` for running Charon:

```
# useradd -G disk,tape,cdrom,dialout,lock -c "Charon User" -m charon
# passwd charon
```

Any existing user can also be used to run Charon. In this case issue the following command to include this existing user into these specific groups:

```
# usermod -G disk,tape,cdrom,dialout,lock -g <user name> <user name>
```

Example:

```
# usermod -G disk,tape,cdrom,dialout,lock -g tommy tommy
```

If PuTTY is required for the non-privileged user, repeat the installation steps for CHTERM-VT100 (above) for the user.

Re-login to apply changes.

Please note: If the emulator will be configured to use a physical serial port (`"/dev/ttyNN"`), it must either be run as the `root` user or the non-privileged user must be a member of the `dialout` group. The non-privileged account created above does not allow the use of FC presentation mode (`kgpsa_generic_storage`). If you plan to use either of this feature, you should plan to run Charon-AXP as the `root` user.

Charon License Installation

Charon-AXP requires a valid product license to run the emulator. You have the choice of "VE" or "HASP" licensing. These are mutually exclusive. More information can be found below and in the [CHARON-AXP for Linux Licensing](#) section.

VE Licensing

VE licensing requires that a license server be installed and running that can serve a valid license. Stromasys recommends that you deploy a separate, dedicated system for the license server. If you need to set up a license sever, please refer to [Virtual Environment \(VE\) License Server Documentation](#). If you already have a running VE license server, see the following and if necessary the [CHARON-AXP for Linux Licensing](#) section.

For Charon-AXP/VAX, the configuration of primary and (optionally) backup license server must be specified in the emulator configuration file using a text editor.

Configuration file general format:

```
set session license_key_id = "VE://<license-server-IP-Address>[:<port>]/[<passphrase>]"
```

Description of the parameters:

- *<license-server-IP>*: the IP address of the VE license server (127.0.0.1 if the VE license server is on the same host).
- *<port>*: the TCP port on which the license is served (if not specified, the default port 8083 will be used).
- *<passphrase>*: the passphrase of the correct product section on the license (optional). The parameter may be required for the emulator in some cases to identify the correct section.

To configure a **backup license server**, add the backup license server information to the same line after the primary license server information:


```
set session license_key_id = "VE://<primary-licserv-IP-Address>[:<port>]/<passphrase>/, VE://<backup-licserv-IP-Address>[:<port>]/<passphrase>"
```

Only **one** backup server can be configured. The backup server typically provides a license limited to a certain number of runtime hours should the primary server become unavailable. If all valid licenses are lost or removed while an emulator is running, there is a grace period (configured on the license; default: 2 hours). The grace period is the time period during which the emulator continues to run after its license has been lost or removed. If there is no valid license after the grace period ends, the emulator will stop (this could cause data loss for a running guest system).

HASP Licensing

Regular HASP USB dongle

If the Charon license is located on a regular USB dongle, it must be connected to a "local" host USB port. "local" means either a USB port directly connected to the emulator-host, or one belonging to a USB-over-IP device, such that the USB port appears local to the emulator-host.

-  If the Charon host is accessed remotely (Remote Desktop for example), please note that regular HASP licenses cannot be displayed or used to start a Charon virtual machine. As a workaround it is possible to install Charon as a daemon (service). This procedure will be described later.

Network HASP USB dongle


If the Charon license is a network license (red USB dongle), it is possible either to connect it to the host USB port (to use it locally and provide it to other hosts on the local network at the same time) or to install it on a local network "license server" for remote access from this particular host.

If a remote license server is to be used:

- Copy the aksusbd-8.13-1.x86_64.rpm and charon-license-4.12-<build>.<OS identifier>.x86_64.rpm files to the server (see Installation section above), for example to "/tmp".
 - Login as "root" on the server.
 - Switch to that directory.
 - Install the copied files using "yum".
- Example:**

```
# cd /tmp
# yum install aksusbd* charon-license-*
```


- Connect the network HASP dongle to one of the server USB ports.

 The network HASP (red dongles) licenses have no restrictions with respect to remote access.

Software license


If the Charon license is a software license (SL), it is installed on the host using the following procedure:


1. Run the `hasp_srm_view` utility in the following way to get the host fingerprint file ("my_host.c2v" in this example):

```
# hasp_srm_view -fgp my_host.c2v
```

2. Send the resulting file to STROMASYS. In return STROMASYS will provide you with a ".v2c" file, for example "your_license.v2c".
3. Copy the received file to any folder on the Charon host, invoke the system default web browser and enter the URL <http://localhost:1947> to display the "**Sentinel Admin Control Center**" (**ACC**) web interface. This interface allows you to view and manage the Charon licenses.
4. In the **ACC** perform the following steps: select **Update/Attach** from the menu on the left pane then use the **Browse** button to select the received file and click on the **Apply File** button to install the license.
5. Ensure that the software license is now visible in the "**Sentinel Keys**" section of the **ACC**.

 It is also possible to use the "`hasp_update`" utility for applying ".v2c" files.

 The Software Licenses (SL) are always network licenses, they have no restrictions with respect to being displayed or accessed via a remote connection.

 A "Provisional" (demo) license does not require collecting a fingerprint. For its installation start at step 3 in the sequence above

License validity verification

To check the Charon license validity, invoke the `hasp_srm_view` utility to make sure that the Charon license is visible and is correct:

- Text of the license is displayed correctly by the `hasp_srm_view` utility, no error messages are shown.
- The content of the license looks correct. For example: license number, major and minor versions, minimum and maximum build numbers, Charon-AXP products and allowed hardware (Charon-AXP models) should be checked. More details on the license content can be found in the [CHARON-AXP Licensing chapter](#) of this Guide.

Example:

```
# hasp_srm_view


License Manager running at host: dlao.msc.
masq
License Manager IP address: 192.168.1.129

HASP Net key detected


The Physical KeyId: 1422726238
License Type: License Dongle (Network Capable)
CHARON Sentinel HASP License key section
Reading 4032 bytes


The License Number: 000.msc.sanity.tests
The License KeyId: 1422726238
The Master KeyId: 827774524
Release date: 10-MAR-2020
Release time: 15:15:15
...
```



 If multiple licenses are available, it is possible to check them using the "-all" parameter with the `hasp_srm_view` utility in the following way:

```
# hasp_srm_view -all
```


 It is also possible to display the license content for one specific key using the "-key" parameter and specifying the Key Id (see "# `hasp_srm_view -h`" for more)

 **Reminder:** If the Charon host is accessed over a remote connection, please note that regular HASP licenses cannot be displayed and used in this case. As a workaround it is possible to install Charon as a daemon (service). This procedure will be described later.

Troubleshooting

If the Charon license content cannot be displayed by the `hasp_srm_view` utility or it is incorrect, check the license is available and correctly used:

1. Invoke the system default web browser and enter the URL <http://localhost:1947> to display the "**Sentinel Admin Control Center**" (ACC) web interface.
2. Click on "**Sentinel Keys**" link to open the corresponding page.
3. Make sure that one and only one Charon HASP or SL license is present.


 To facilitate troubleshooting, Stromasys recommends to enable logging from the Sentinel Admin Control Center as described in this article: [Enabling logging in Sentinel Admin Control Center](#).

Problem	Action
No license is displayed	Make sure that all the recommendations above about remote access to the host are fulfilled (if remote access takes place), that the HASP USB key is not broken and its LED indicator is lit (meaning that it is used by the host).
Only one License key / SL is seen and its content is incorrect	Contact STROMASYS to request a new license update.
Several License keys / SLs are displayed	Remove all of them except the one provided by STROMASYS for the just installed version of Charon.


Removing licenses can be done by physical disconnection of the corresponding USB HASP keys from the Charon host and physical disconnection of the network HASP keys from all hosts on the local network (or by disabling remote access to network licenses from the Charon host - see detailed explanations below).

For license servers accessible only via non-broadcast search it is also possible to disable access to network licenses if only a local license is to be used: Click on the "**Configuration**" link to open the "**Configuration for Sentinel Manager**" page.

Uncheck the "**Allow Access to Remote Licenses**" checkbox from the "**Access to Remote License Managers**" tab then press the "**Submit**" button to apply changes.

 Starting with Charon-AXP/VAX 4.9 for Linux and Charon-AXP/VAX version 4.8 for Windows the Charon emulator products do not follow the settings in the Sentinel ACC with respect to querying remote license servers and network visibility. They perform a **broadcast search** for network licenses even if this has been disabled in the Sentinel ACC. If this behavior has to be prevented for specific reasons, the network access of the system has to be temporarily restricted or disabled, for example by blocking the relevant traffic in a firewall. Another possibility would be to block access to the network license at the license server side.

Note that such methods can negatively impact other functions of the system or, in the case of blocking access to a network license on the server, even the functions on other license clients.

 It is also possible to leave several licenses available to Charon-AXP at the same time but in this case they have to be specified in the configuration file.

Example:

```
set session license_key_id=1877752571
```

It is also possible to have one "main" and one "backup" license in case the main license becomes unavailable:

```
set session license_key_id="1877752571,354850588"
```

Charon-AXP checks its licenses from time to time starting with the main license. If it becomes unavailable, it attempts to access the backup license.

Network configuration

In most cases Charon will use a network. In this case Charon requires one or more dedicated network interfaces with any other protocols including TCP /IP removed at the host level.

Two ways of network configuration are possible:

- With the help of the "[ncu](#)" utility
- Manually

The first way is recommended. Use the manual approach only in absence of the "[ncu](#)" utility or if it impossible to use it.

Configuration with NCU utility

Login as root and enter "ncu". The following menu will appear:

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces Dedicated to State
-----
eth0      host      connected to host
eth1      host      disconnected from host
lo        host      unmanaged by host
virbr0-nic bridge    unmanaged by bridge

=====
bridge name bridge id          STP enabled  interfaces
=====
virbr0    8000.5254004608c0  yes          virbr0-nic
=====

select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit
:> 1
```

The utility lists the available network interfaces (both physical and virtual) and indicates whether they are dedicated to the host or to Charon and whether they are currently in use by the host operating system.

"ncu" offers several options:

- Dedicate interface to Charon (option "1")
- Release interface to host (option "2")
- Create a bridge between a chosen Linux network interface and the Linux virtual bridge and create a number of virtual network interfaces ("TAP") (option "3")
- Remove the Linux virtual bridge and all the created virtual network interfaces (option "4")
- Add VLAN (option "5")
- Remove VLAN (option "6")
- Print status (option "7") - use it to display status of network interfaces and the menu shown above
- Exit (option "8")

In the example above we see 2 network interfaces, "eth0" and "eth1", that are dedicated to the host and the host uses only the interface "eth0".

Let's dedicate the interface "eth1" to Charon-AXP.

Enter "1" then "eth1":

```
Specify the interface to dedicate to CHARON:eth1
Turning off offloading for eth1.. Please wait

select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 7
```

Now the interface "eth1" is dedicated to CHARON-AXP:

```
Interfaces Dedicated to State
-----
eth0      host      connected to host
eth1      CHARON    disconnected from host
lo        host      unmanaged by host
virbr0-nic bridge    unmanaged by bridge

=====
bridge name bridge id          STP enabled  interfaces
=====
virbr0     8000.5254004608c0  yes         virbr0-nic
=====

select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:>
```

Enter "8" to return to the console prompt.

Now "eth1" can be used by Charon-AXP.

Manual Configuration

Choosing a network interface

To choose an interface to be used for Charon networking, do the following:

```
# ifconfig
eth0 Link encap:Ethernet HWaddr 00:60:52:0A:A9:1E
...
eth1 Link encap:Ethernet HWaddr 00:C0:26:60:FB:15
...
eth2 Link encap:Ethernet HWaddr 00:1A:92:E1:3F:7F
```

Choose an interface to be used by Charon, for example "*eth1*"

Designation of network interface to Charon

To designate the chosen interface to Charon open up the file `/etc/sysconfig/network-scripts/ifcfg-ethN` (where *N* is the number of the interface to be used for Charon, in this case it is "1") and make sure that all the IP-setup related parameters are removed. The file must look like this:

```
DEVICE="eth1"
HWADDR="00:06:2B:00:6A:87"
NM_CONTROLLED="no"
ONBOOT="no"
```

Switching off the offload parameters

Determine what additional parameters are currently set to "on" on the host network adapter to be used by Charon using the following command:

```
# ethtool -k <device>
```

Example:

```
# ethtool -k eth1
Offload parameters for eth1:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: off
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: off
large-receive-offload: off
```

Use "ethtool" to switch off all the offload parameters:

```
# ethtool -K <device> <parameter> off
```

Example:

```
# ethtool -k eth1
Offload parameters for eth1:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: off
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: off
large-receive-offload: off
```

For the example above let's create a temporary file containing the commands to be executed at system startup as the offload parameters must be switched off following each reboot:

```
ethtool -K eth1 rx off
ethtool -K eth1 tx off
ethtool -K eth1 sg off
ethtool -K eth1 gso off
ethtool -K eth1 gro off
```

Let's suppose the name of the file is "offload_off_eth1.txt". To execute it on system startup, execute the following command (example):

```
# cat offload_off_eth1.txt >> /etc/rc.d/rc.local
```

Final steps

- Reboot the host system
- Login as user "charon"
- Verify the offload parameters are effective

Upgrade from previous version

To upgrade an already installed Charon-AXP kit to a more recent one:

1. Ensure your license allows you to upgrade to that version. If not, please generate a C2V file and send it to STROMASYS for update. See [CHARON-AXP for Linux utilities - 'hasp_srm_view' utility](#)
2. Prepare the new kit RPM files as it is described in "[Charon-AXP for Linux installation#Before installation](#)" and "[Charon-AXP for Linux installation#Distribution preparation](#)" sections.
3. [Stop all running CHARON-AXP instances.](#)
4. Make sure that no template files (i.e. "es40.cfg.template") have been used for your specific configuration otherwise copy those files to a dedicated folder.
5. Login as "root" user.
6. Remove the old Charon-AXP version as described in the "[CHARON-AXP for Linux deinstallation](#)" chapter and reboot the Linux server (recommended).
7. Proceed with the instructions on the new kit installation as described in the "[Charon-AXP for Linux installation#Installation](#)" section.
8. Once installation is completed, it is recommended to reboot the Linux server (possible issues with licenses detection could occur).
9. Install the license for the new Charon-AXP as described in the "[Charon-AXP for Linux installation#License installation](#)" section.
10. [Start all the CHARON-AXP services](#) stopped at step #3.



If you did not reboot your Linux server at step 6, you may experience issues with 'aksusbd' service installation and then license detection.

Example:

```
Installing : aksusbd-8.13-1.x86_64 1/5
Failed to execute operation: Access denied
Failed to restart aksusbd.service: Access denied
```

To solve this problem, remove all Charon installed product and restart from step 6 above.

Running Charon-AXP for Linux

Table of Contents

- [Charon-AXP symbolic links](#)
- [Running Charon-AXP emulators](#)
 - [Running from the console](#)
 - [Options for running Charon-AXP from console](#)
 - [Running as system service \(daemon\)](#)
 - [Installation and start of Charon-AXP service](#)
 - [Stopping Charon-AXP service](#)
 - [Removing Charon-AXP service](#)

Charon-AXP symbolic links

Each model of Charon-AXP has a symbolic link defined to point to the corresponding Charon executable (see the table below). If the PATH is correctly defined, you can start a virtual machine by specifying the link followed by the configuration file. This is described further.

Link name	Emulator to run
as400	AlphaServer 400
as800	AlphaServer 800
as1000	AlphaServer 1000
as1000a	AlphaServer 1000A
as2000	AlphaServer 2000
as2100	AlphaServer 2100
as4000	AlphaServer 4000
as4100	AlphaServer 4100
ds10	AlphaServer DS10
ds10l	AlphaServer DS10L
ds15	AlphaServer DS15
ds20	AlphaServer DS20
ds25	AlphaServer DS25
es40	AlphaServer ES40
es45	AlphaServer ES45
gs80	AlphaServer GS80
gs160	AlphaServer GS160
gs320	AlphaServer GS320

Running Charon-AXP emulators

It is possible to run one or several instances of Charon-AXP at the same time if your license allows it.

For multiple instances, please use only absolute paths and unique names to all the files referenced in the configuration file of each Charon-AXP instance (log, toy clock, rom files and all the other data such as disk images [Exception: clustering] - all these objects will be explained later in this document) and check the hardware devices (the CDROM drive for example) are used by only one instance at a time (not shared).

Example:

```
...
set session log="/CharonInstances/1st_es40.log"
set rom container="/CharonInstances/1st_es40.bin"
set toy container="/CharonInstances/1st_es40.dat"

load KZPBA PKA scsi_id = 7
set PKA container[0]="/CharonInstances/1st_es40_boot_disk.vdisk"
...
```

Please refer to the next chapters for more details concerning Charon-AXP configuration details.

Running from the console

Copy the selected configuration template from the `/opt/charon/cfg/` directory to some local file and set the correct privileges for the file to be edited.

Example:

```
$ cp /opt/charon/cfg/es40.cfg.template my_es40.cfg
$ chmod 644 my_es40.cfg
```

Run the virtual machine using this configuration file:

```
$ es40 my_es40.cfg
```

Below is an example of a normal Alpha test sequence, followed by the prompt sign ("`>>>`"):

```
initializing ...

polling for units on kzpba0, slot 2, bus 0, hose 1 ...
pka0.0.0.2.1 PKA0 Q-Logic/ISP PCI SCSI HBA

... enter console

CHARON-AXP/ES40 for Linux (AlphaServer ES40 6/667), Version 4.11.20403
(C) 2009-2020 STROMASYS SA.
All rights reserved.

P00>>>
```

The next stage can be either installation of a new Alpha/VMS system using distribution media or data transfer from some existing Alpha system. These possibilities will be discussed in details in the next chapters.

If for some reason Charon-AXP refuses to start, please look for files with `.log` extension (Charon-AXP log files) located in the directory from where Charon-AXP was started, open them with an editing tool and analyze their content. In most cases those files contain very helpful information on what went wrong.

To exit from the Charon-AXP emulator use the following methods:

Configuration	How to exit
No change to the template configuration file	Enter " power off " in the Charon console
Enable " F6 " button in configuration file to trigger exit from Charon: <code>set OPA0 stop_on = F6</code>	Press the " F6 " key

Please note the following:

- Before stopping Charon-AXP, a clean shutdown of the operating system running on the virtual machine has to be performed.
- The total number of devices (both controllers and units, including disks) displayed by Charon SRM console is limited to 48. This does not affect the actual number of configured devices provided to Charon guest OS.

Options for running Charon-AXP from console

If "-h" or "--help" option is specified when running Charon-AXP from console, it displays a list of additional available options:

```
Usage:
  <program-name> [options] [<configuration-file-name>]

Command line options:
  -l,--log <file-name>           - write log to the file (overwrite),
                                until configuration is loaded;
  -la,--log-append <file-name>  - write log to the file (append),
                                until configuration is loaded;
  -f,--configuration <file-name> - read configuration from the file;
  -d,--daemon                    - run detached;
  -h,--help                      - display this text

Note that configuration file must be supplied either as a command line
operand <configuration-file-name> or using '-f' command line option followed
by name of the configuration file.

Options to use with CHARON Manager:
  -a,--alias <alias-name>       - virtual machine alias name;
  -s,--shm-name <shm-name>      - name of shared memory section
```

The last 2 options are used for running Charon-AXP in Baremetal environment,

Running as system service (daemon)

It is possible to run Charon-AXP as a daemon (service). In this case the Charon-AXP process will be detached from its parent process and from the terminal window in which it runs.

Follow the description below to install and execute Charon-AXP as a daemon:

Installation and start of Charon-AXP service

1. Copy the sample script `/opt/charon/bin/charon.service` to the `/usr/lib/systemd/system/` directory.

Example:

```
$ cp /opt/charon/bin/charon.service /usr/lib/systemd/system/es40.service
$ chmod 755 /usr/lib/systemd/system/es40.service
```

2. Edit the copied file to replace sample values of the following parameters.

Example:

```
ExecStart=/opt/charon/bin/es40 -d /my_services/es40-service.cfg
WorkingDirectory=/my_services
```

3. Create and edit the configuration file (`/my_services/es40-service.cfg` in the examples above) the way it was described before and make sure the following pre-requisites are met:

- **OPA0** must be configured as a virtual port or physical console, not as an operator console.

Example:

```
set COM1 alias=OPA0 port=10003
#set COM1 alias = OPA0 line = (console)
```

- Use only absolute paths to **log**, **toy clock**, **nvr** files and all the other data such as disk images, etc. The names of the references files must be unique.

Example:

```
...
set session log="/CharonInstances/1st_es40.log"
set rom container="/CharonInstances/1st_es40.bin"
set toy container="/CharonInstances/1st_es40.dat"

set PKA container[0]="/CharonInstances/1st_es40_boot_disk.vdisk"
...
```

- Make sure the same physical devices are not used by other Charon-AXP daemons and the OPA0 console port number is unique across the Charon server.

Once the configuration file is ready, execute the following commands (based on the examples above) to install and start Charon-AXP as a daemon:

```
# systemctl enable es40.service
# systemctl start es40.service
```

Please note:

- If you update the `/usr/lib/systemd/system/<my virtual machine>.service` file, the following command must be executed in order to take changes into account:

```
# systemctl daemon-reload
```

Important information:

Note that a certain delay may appear in finding network licenses by Sentinel Run-time on Charon-AXP host system startup. If the Charon-AXP service is starting automatically at host system startup, it may report a "License not found" error and exit.

This problem can be avoided by specifying the "license_key_lookup_retry" parameter in the following way:

```
set session license_key_lookup_retry = "N [, T]"
```

where:

- N = Number of retries looking for the license key (or keys)
- T =Time between retries in seconds. If not specified, 60 seconds is used

Example:

```
set session license_key_lookup_retry = 5
```

In this example, if the license key is not found during initial scan, Charon-AXP will do 5 more attempts waiting 60 seconds between them.

See [General Settings](#) section for more details.

Stopping Charon-AXP service

To stop a Charon-AXP daemon, use the following commands.

Example:

```
# systemctl stop es40
```

Please note that before stopping a Charon-AXP service, a clean shutdown of the operating system ("guest") running on the emulator must be performed. Stopping the emulator when the guest is still running should be the last resort when all attempts to stop the guest have failed.

Removing Charon-AXP service

To remove a Charon-AXP daemon use the following commands.

Example:

```
# systemctl disable es40.service  
# rm -f /usr/lib/systemd/system/es40.service
```

Please note that before removing a Charon-AXP service, a clean shutdown of the operating system running on the virtual machine has to be performed and the service has to be stopped.

Please refer to the next chapters for more details concerning Charon-AXP configuration details

CHARON-AXP for Linux configuration

Table of Contents

- Creation of your own configuration file using a template
- Alpha model specification
- Configuration name
- Log file parameters
 - Rotating log (default)
 - Single log
- CPU affinity
- Number of host CPUs dedicated to Charon I/O
- Setting a specific Alpha model
- Configuring the number of emulated CPUs
- Setting system serial number
- TOY and ROM containers
- Emulated memory (RAM) size
- Console
 - Mapping to system resources
 - Exit on pressing F6 key
- Improve granularity of emulated timer
- Networking
- Disk/tape subsystem
 - KZPBA PCI SCSI disk/tape controller
 - KGPSA-CA PCI FC disk controller
 - KGPSA-CA mapping to the host resources
 - KGPSA-CA pass through mode
- Auto boot
- Network boot

Creation of your own configuration file using a template

By default, all the Charon templates are located in the `/opt/charon/cfg` folder. Copy the appropriate template configuration file(s) to your home directory or to any directory intended for CHARON-AXP, name them meaningfully and set proper privileges.

Example:

```
$ cp /opt/charon/cfg/es40.cfg.template /my_charon_cfg/my_es40.cfg
$ chmod 644 /my_charon_cfg/my_es40.cfg
```

Please do not edit the original template configuration files since they can be updated or even removed on update/deinstallation of Charon-AXP

Once the file has been created you can open it in your favorite editing tool and proceed with modifications to reflect the exact features of the system you are going to emulate.

We will review all the parameters step by step issuing some recommendations and guidelines.

Note: the lines preceded by the comment sign `"#"` inside the configuration files will not be interpreted. You can use this sign to debug your configuration.

Alpha model specification

The first configuration statement is the specification of the exact Alpha hardware model to emulate.

Example:

```
set session hw_model = AlphaServer_ES40
```

You must leave this line untouched.

If you create the Charon-AXP configuration file from scratch, it must be the very first uncommented line in the configuration file.

Configuration name

The next configuration statement is the "Configuration name" option.

Example:

```
# set session configuration_name = My_ES40
```

You can optionally uncomment this line to differentiate this Charon-AXP instance from all others in a multi-instances environment. The configuration name can be any label that is meaningful. It is reported in the log file and is used to set the log file name for rotating log (see further: [Rotating log \(default\)](#)).

Log file parameters

Execution of CHARON-AXP creates one log file or a set of log files reflecting the progress of its start-up and ongoing operation - start and end time of execution, system information, license and configuration details, warnings, reports on problems that may occur, etc. In case of possible problems either with the running CHARON-AXP or the emulated system configuration (such as the absence or malfunction of certain devices), the log file(s) is the primary source to be analyzed for troubleshooting. If it becomes necessary to contact Stromasys for support, the configuration and log files, plus the license number, will be requested to begin problem resolution.

CHARON-AXP log file example (part1)

```
20200310:035210:INFO :0:000003A5:hexane.cxx(5938): session: loading built-in configuration
"AlphaServer_ES40"...
20200310:035210:INFO :0:000003A6:hexane.cxx(5959): session: ... done loading built-in configuration
"AlphaServer_ES40"
20200310:035210:INFO :0:000003AA:hexane.cxx(5988): session: loading configuration file "es40.cfg"...
20200310:035210:INFO :0:000003AB:hexane.cxx(6012): session: ... done loading configuration file "es40.cfg"
20200310:035210:INFO :0:000003F2:sesmgr.cxx(1410): session: default log file size limit is 4194304 bytes
20200310:035210:INFO :0:0000032B:hexane.cxx(2698): Start request received.
20200310:035211:INFO :0:000003AC:hexane.cxx(1424): session: process affinity is 000000000000000F, system
affinity is 000000000000000F
20200310:035211:INFO :0:000003D1:hexane.cxx(1686): session: I/O domain affinity is 0000000000000001, CPU
domain affinity is 000000000000000E
20200310:035211:INFO :0:0000024D:licenseman(1823): Checking the available license key "1918154109".
20200310:035211:INFO :0:0000024D:licenseman(1823): Found license key: "1918154109".
20200310:035211:INFO :0:0000024D:licenseman(1823): Checking product section 0.
20200310:035211:INFO :0:0000024D:licenseman(1823): License number: "000.msc.test.center.nikolaev".
20200310:035211:INFO :0:0000024D:licenseman(1823): Product License number: "CHAXP/AXP".
20200310:035211:INFO :0:0000024D:licenseman(1823): CHARON product code: "CHAXP-411xx-WI-LI".
20200310:035211:INFO :0:0000024D:licenseman(1823): Unlimited license.
20200310:035211:INFO :0:0000024D:licenseman(1823): Feature 2 check interval 1 hour(s).
20200310:035211:INFO :0:0000024D:licenseman(1823): Concurrency info:
20200310:035211:INFO :0:0000024D:licenseman(1823): There are 10 instances allowed.
20200310:035211:INFO :0:0000024D:hexane.cxx(2848): STROMASYS SA, (C) 2009-2020
20200310:035211:INFO :0:00000408:hexane.cxx(2892): CHARON-AXP (AlphaServer ES40), V 4.12 B 21009, Mar 10
2023 / 000.msc.test.center.nikolaev / 1918154109
20200310:035211:INFO :0:00000336:hexane.cxx(2924): The end user of this software has agreed to STROMASYS'
Terms and Conditions for Software License and Limited Warranty, as described at: http://www.stromasys.com/pub
/doc/30-17-033.pdf
```

CHARON-AXP log file example (part2)

```

20200310:035211:INFO :0:00000097:hexane.cxx(2999): OS Environment: Red Hat Enterprise Linux Server release
7.4 (Maipo), Linux 3.10.0-693.5.2.el7.x86_64 #1 SMP Fri Oct 13 10:46:25 EDT 2017 x86_64.
20200310:035211:INFO :0:00000098:hexane.cxx(3004): Host CPU: GenuineIntel, Family 6, Model 42, Stepping 1,
Intel Xeon E312xx (Sandy Bridge), 1 Sockets, 1 Cores per Socket, 4 Threads per Core, at ~2593 MHz, 4 cpu's
available
20200310:035211:INFO :0:00000099:hexane.cxx(3009): Host Memory: 3840Mb
20200310:035211:INFO :0:0000041F:hexane.cxx(3224): Configuration dump::
. session:
. . configuration_name = "AlphaServer_ES40"
. . log_method = "overwrite"
. . hw_model = "AlphaServer_ES40"
. . log_mode = "shared"
. . log_locale = "english"
. RAM:
. . size = "256"

. ACE:
. . num_entries = "2139"
. . num_translators = "0"
. . cache_size = "1024"
. . cache_base_size = "200"
. . host_options = " --locked-size=16"
. . enabled = "true"
. . ext_compiler = "ml64.exe"
. . ext_path = ""
. . cpu_architecture = "EV67"
. . locked_size = "16"
. cpu_0:
. . locked_size = "16"
. . wtint_idle = "true"
. cpu_1:
. . locked_size = "16"
. . wtint_idle = "true"
. cpu_2:
. . locked_size = "16"
. . wtint_idle = "true"
. cpu_3:
. . locked_size = "16"
. . wtint_idle = "true"
. ROM:
. . container = "clipper.bin"
. . dsrdb[0] = "1820"
. . dsrdb[1] = "50"
. . dsrdb[4] = "50"
. . dsrdb[11] = "1050"
. . dsrdb[12] = "1050"
. . system_name = "AlphaServer ES40 6/667"
. ISA:
. . clock_period = "10000"
. TOY:
. . container = "clipper.dat"
. COM1:
. . line = "OPA0"
. . communication = "console"
. COM2:
. . line = "(void)"
. . communication = "ascii"
. OPA0:
. . trace = "disabled"
. . stop_on = "F6"
. . tx_flush_delay = "0"

```


CHARON-AXP log file example (part3)

```
. EWA:
. . adapter_mode = "auto"
. . interface = "EWA0"
. . rx_fifo_delay_on_overload = "false"
. EWA0:
. . interface = "eth1"
. . disabled_mode = "10BaseT-HD"
. . port_show_driver_statistics = "false"
. . port_enable_mac_addr_change = "true"
. . port_snd_sock_buf_size_kb = "0"
. PKA:
. . scsi_id = "7"
. . min_n_of_threads = "0"
. . container[0] = "/home/charon/Charon/test/performancecomparison-axp.vdisk"
```

The next group of parameters defines the name of the Charon-AXP log file and how Charon-AXP will use it:

```
set session log_method = append
#set session log_method = overwrite
#set session log = "AlphaServer_ES40.log"
```

Rotating log (default)

By default Charon-AXP utilizes a so-called "rotating log" mechanism. This means that a new default log file is created each time Charon starts and can switch to another log file if the size of the log file exceeds 4 MB (this behavior can be changed with the "set session log_file_size" and "set session log_rotation_period" parameters; see more details in the "[General Settings](#)" chapter of this guide).

This mode is turned on if the log_method is not specified or the "session_log" parameter is pointing to an existing directory rather than to a file. If a directory is specified, the log files will be created in that directory.

The names of the rotating log files are composed as follows:

```
configuration_name-YYYY-MM-DD-hh-mm-ss-xxxxxxxxx.log
```

If the "Configuration name" parameter described before is omitted (commented out), the log name has the following format instead:

```
hw_model-YYYY-MM-DD-hh-mm-ss-xxxxxxxxx.log
```

Note that "xxxxxxxx" is an increasing decimal number starting from "00000000". During a session (the time between start and stop of the emulator), when the log file is rotated, this number will be incremented by 1.

Please note: only an existing directory can be used as a value of the "log" parameter.

Single log

Alternatively it is possible to use a single log file: uncomment the "set session log" line and specify the desired Charon-AXP log file name. Optionally, a path can be added to the log file name. If the path is not specified, the log file is created in the directory from where the guest (emulated machine) is started.

The log file can be extended ("log_method = append") or overwritten ("log_method = overwrite") by Charon-AXP.

Below is a specification of a Charon-AXP log file located in the "/my_logs" directory which will be appended each time Charon-AXP starts:

```
set session log_method = append
set session log = "/my_logs/my_es40.log"
```

CPU affinity

This setting binds the running instance of the emulator CPUs to particular host CPUs. This should be used for soft partitioning host CPU resources or for isolating multiple Charon instances on the same host from each other. By default the emulator instance allocates as many host CPUs as possible.

“Affinity” overrides the default and allows explicit specification of which host CPUs will be used by the instance. Affinity does not reserve the CPU for exclusive use.

Example:

```
set session affinity="0, 1, 2, 3"
```

The example above directs Charon-AXP to use CPU 0,1,2 and 3.

If this parameter is omitted Charon host will allocate available CPUs automatically.

Note that the **number of the specified host CPUs must correspond to the number of the emulated CPUs** (one host CPU for one emulated CPU; this value is specific for each Alpha model) and number of CPUs needed for Charon application itself ("n_of_io_cpus").

Number of host CPUs dedicated to Charon I/O

This parameter reserves host CPUs for use by the emulator for I/O handling. By default the emulator instance reserves one third of the number host CPUs for I/O processing (rounded down, at least one).

The "n_of_io_cpus" overrides the default by specifying the number of I/O host CPUs explicitly

Example:

```
set session n_of_io_cpus=2
```

The example above directs Charon-AXP to use 2 CPUs for Charon I/O operations.

Note that the number of the specified CPUs dedicated to Charon I/O operations can NOT exceed the total number of host CPUs minus the number of Alpha emulated CPUs.

Setting a specific Alpha model

Charon-AXP allows to specify an exact model of Alpha.

For example for AlphaServer ES40 family the "es40.cfg" sample configuration file contains the following options:

```

#-----
#
# AlphaServer ES40 6/500
#
#-----

#set ace cpu_architecture = EV6
#set rom dsrdb[0] = 1816 system_name = "AlphaServer ES40 6/500"
#set rom version[1] = 1.98-4 version[2] = 1.92-5

#-----
#
# AlphaServer ES40 6/667
#
#-----

set ace cpu_architecture = EV67
set rom dsrdb[0] = 1820 system_name = "AlphaServer ES40 6/667"

```

Just uncomment the provided lines to apply a certain model (It is "AlphaServer ES40 6/667" in the example above).

The full description of the parameters, with other models that can be also configured, is available in the "[Configuration details](#)" chapter of this User's Guide.

Do not change the basic Alpha model ("DS10", "DS20", "ES40", "ES45", etc). You should always start with the appropriate configuration template file for the model you wish to emulate in `/opt/charon/cfg/`.

Configuring the number of emulated CPUs

If the number of emulated CPUs is not specified, the emulator will try to create the maximum number of CPUs as determined by the Alpha model being emulated. See the Software Product Description to determine the maximum number of CPUs possible for a particular Alpha model.

If you wish to emulate fewer CPUs than the maximum, for example because the Charon host does not contain enough CPUs to emulate the maximum number of CPUs provided by a certain Alpha model, it is possible to specify the number of emulated Alpha CPUs in the configuration:

```
set session n_of_cpus=1
```

This parameter can also be used to avoid warning messages in the log if the number of CPUs allowed by the license is less than the default number of CPUs of the emulated Alpha model. For example, the default number of CPUs for an Alpha DS20 is 2. Many DS20 installations have only 1CPU and are licensed that way. To avoid any warning messages, the use of the command above would be advised. Please note that the warning message is just that and will not cause any problems or issues.

Setting system serial number

This parameter is used to set a specific system serial number instead of the default one:

```
set rom system_serial_number = SN01234567
```

TOY and ROM containers

The next objects to be configured are TOY and ROM containers (their presence depends on the Alpha model). It is always recommended to enable them. If a container file of the given name does not exist, Charon-AXP will create it. It is recommended to specify the path for each file so that time and console parameters will be kept whatever the current directory is when starting the guest.

TOY means "Time of Year"; its container records time, date and some console parameters while Charon-AXP is not running. To enable the TOY, uncomment the following line:

```
set toy container="clipper.dat"
```

The ROM container stores an intermediate state of the Flash ROM and some console parameters. It is highly recommended to define its location:

```
set rom container="clipper.bin"
```

Emulated memory (RAM) size

The next parameter defines the amount of host memory the chosen CHARON-AXP model reserves for the emulation.

Example:

```
#set ram size=4096
set ram size=32768
```

The amount of RAM is specified in MB. It cannot exceed or be lower than certain values specific for each Alpha model. It is very important to keep the listed predefined increment between possible memory values.

The following table lists all the parameters per model:

Hardware Model	RAM size (in MB)			
	Min	Max	Default	Increment
AlphaServer 400	64	1024	512	64
AlphaServer 800	256	8192	512	256
AlphaServer 1000	256	1024	512	256
AlphaServer 1000A	256	1024	512	256
AlphaServer 1200	256	32768	512	256
AlphaServer 2000	64	2048	512	64
AlphaServer 2100	64	2048	512	64
AlphaServer 4000	64	32768	512	64
AlphaServer 4100	64	32768	512	64
AlphaServer DS10	64	32768	512	64
AlphaServer DS10L	64	32768	512	64
AlphaServer DS15	64	32768	512	64
AlphaServer DS20	64	32768	512	64
AlphaServer DS25	64	32768	512	64
AlphaServer ES40	64	32768	512	64
AlphaServer ES45	64	32768	512	64
AlphaServer GS80	256	65536	512	256
AlphaServer GS160	512	131072	512	512
AlphaServer GS320	1024	262144	1024	1024

It is possible to leave the RAM line commented out. In this case the model's default RAM amount is used.

Note that in some particular orders your license may restrict the maximum RAM amount of each Alpha model.

Console

Mapping to system resources

The next step is the specification of the Alpha console (OPA0) serial line.

Example:

```
#set OPA0 line = "/dev/ttyN"
#set OPA0 port = 10003
#set OPA0 port = 10003 application = "xterm -title OPA0 -e telnet 127.0.0.1 10003"
#set OPA0 port = 10003 application = "xterm -title OPA0 -e cterm -h 127.0.0.1:10003"
set OPA0 line = (console)
```

The goal of this configuration step is to tell Charon-AXP what host device to use as the virtual system console. The following options are available:

Option	Description
line	<p>Mapping to host serial line, both physical and virtual. Use the following mapping for different types of host serial lines:</p> <ul style="list-style-type: none"> ■ /dev/tty<N> for virtual serial lines ■ /dev/ttyS<N> for onboard serial lines ■ /dev/ttyUSB<N> for modem or usb serial lines adapters ■ (console) for mapping to the current TTY console <p>Please note: If the emulator will be configured to use a physical serial port ("/dev/ttyNN"), it must either be run as the <code>root</code> user or the non-privileged user must be a member of the <code>dialout</code> group.</p>
port	<p>Mapping to an IP port of the CHARON host. Using this mapping it is possible to connect to the Charon console and disconnect from it at any time.</p>
application	<p>Starting some application (typically another xterm terminal) with its specific options and switches to communicate to Charon using the IP port defined by the "port" parameter (see above)</p>
alias	<p>Define some meaningful name for "COM1" and "COM2". Usually it is "OPA0" for "COM1" and "TTA0" for "COM2" (see below)</p>

The default setting for OPA0 is "line = (console)".

The second serial line "COM2"/"TTA0" can be also optionally configured for all Alpha models ("TTA0" is a predefined alias for "COM2"):

```
#set TTA0 line = "/dev/ttyN"
#set TTA0 port = 10000
set TTA0 port = 10000 application = "xterm -title TTA0 -e telnet 127.0.0.1 10000"
#set TTA0 port = 10000 application = "xterm -title TTA0 -e cterm -h 127.0.0.1:10000"
```

There are a number of additional parameters for Charon-AXP serial lines configuration. Follow [this link](#) for details.

Exit on pressing F6 key

Charon-AXP can be exited by issuing the "power off" command on the SRM console. It is also possible to configure a hot key to immediately terminate the emulator from the console:

```
set OPA0 stop_on = F6
```

This line configures the "F6" key to send the emulator the signal to immediately stop itself. Unless you have no choice, you should never stop the emulator if the guest is still running.

Improve granularity of emulated timer

The next configuration option can be applied for improving granularity of emulated CHARON-AXP timer:

```
#set isa clock_period=1000
```

Do not uncomment this parameter unless there are some problems with system time or system clock intervals in guest OS.

Networking

Charon-AXP supports DE435, DE450, DE500AA, DE500BA, DE602 and DE602AA virtual network adapters.

All of them are configured in a similar way:

```
load DE500BA/dec21x4x EWA interface=EWA0
load packet_port/chnetwrk EWA0 interface="eth0"
```

```
load DE602/i8255x EIA interface=EIA0
load packet_port/chnetwrk EIA0 interface="eth0"
```

In the examples above the first line loads DE500BA/DE602 virtual adapter with a name "EWA"/"EIA" (note that "/i8255x" syntax must be used only in case of DE602 and DE602AA adapters); the following line maps it to host network interface "eth0". Note that the mapping is performed in 2 steps:

1. A mapping object "packet_port" with a name "EWA0"/"EIA0" is loaded and connected to host interface "eth0", so Charon-AXP will use this interface for its networking
2. The loaded virtual adapter "EWA"/"EIA" is connected to the "packet_port" object "EWA0"/"EIA0"

It is possible to load several DE435, DE450, DE500AA, DE500BA or DE602 controllers, for example (for DE500BA):

```
load DE500BA/dec21x4x EWA interface=EWA0
load packet_port/chnetwrk EWA0 interface="eth0"

load DE500BA/dec21x4x EWB interface=EWB0
load packet_port/chnetwrk EWB0 interface="eth1"
```

Some network adapters available in Charon-AXP are preloaded (for example, an AlphaServer DS15 contains 2 preloaded adapters EWA and EWB), so their configuration is even more simple:

```
load packet_port/chnetwrk EWA0 interface = "eth0"

load packet_port/chnetwrk EWB0 interface = "eth1"
```

Charon supports VLAN adapters. If used, proceed with their installation and configuration according to the network adapter vendor User's Guide and then use the resulting VLAN interface the same way as the regular network interface.

Please note:

The AlphaServer DS15 and DS25 contain two built-in PCI Ethernet adapters. Models and names (EI* or EW*) of them depend on configuration add-on. Choose one of the two or none, but not both. The first instantiates onboard network interfaces as EIA and EWA. While the second - EWA and EWB (enabled by default for backward compatibility)

Example:

```
#include ds25-onboard-nics.icfg
include ds25-onboard-nics-ew.icfg
```

It could be necessary to specify the path to the .icfg file if you use your own configuration file using the "include /opt/charon/cfg/ds25-onboard-nics-ew.icfg" syntax.

Follow [this link](#) for more details of CHARON-AXP network controllers configuration.

Disk/tape subsystem

The next step is configuration of the disk/tape subsystem and mapping it to system resources using the samples given in the template configuration files.

Charon-AXP supports KZPBA and KGPSA-CA adapters.

KZPBA PCI SCSI disk/tape controller

Below is the typical configuration options for KZPBA PCI SCSI disk/tape controller:

```
load KZPBA PKA scsi_id = 7

# Disks
#set PKA container[0] = "<file-name>.vdisk"
#set PKA container[100] = "/dev/sd<L>"

# Unknown SCSI device
#set PKA container[200] = "/dev/sg<N>"

# CD-ROM
#set PKA container[300] = "/dev/cdrom"
#set PKA container[300] = "/dev/cdrom1"
#set PKA container[300] = "/dev/cdrom<N>"
#set PKA container[300] = "/dev/sr0"
#set PKA container[300] = "/dev/sr<N>"

# CD-ROM image
#set PKA container[400] = "<file-name>.iso"

# Tape
#set PKA container[500] = "/dev/sg<N>"
#set PKA container[600] = "<file-name>.vtape"
```


The first line ("load KZPBA PKA") loads disk controller KZPBA with name "PKA", followed by 8 group of lines showing different ways of mapping to the host resources:

Type of mapping	Description
"<file-name>.vdisk"	Mapping to files representing physical disks of the Alpha system (disk images). These files can be created from scratch with "mkdiskcmd" utility. Data and OS disk backups are transferred from the original system via tapes or network and restored into these container files. Mapping may also include the full path, for example: "/my_disks/my_boot_disk.vdisk"
"/dev/sd<L>"	Mapping to physical disk. "L" is letter here. Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake. These disks must not be formatted by the host OS. It is also possible to use not a whole disk, but previously created partitions on it. In this case the syntax is the following: "/dev/sd<L><N>" where N is the number of partition to be used. Please note: Since "/dev/sd<L>" addressing is not persistent, it is strongly recommended to use "/dev/disk/by-id/wwn-*" syntax instead to refer the disk by its WWID - especially in the environments utilizing FC and SAN storages (see below).
"/dev/dm-<N>" "/dev/mapper /mpath<N>" "/dev/mapper /disk<N>"	Mapping to multipath disk. Be careful not to destroy all the information from the disk dedicated to Charon-AXP by mistake. These disks must not be formatted by the host OS.
"/dev/disk/by-**"	Mapping to physical disk. <ul style="list-style-type: none"> by-id (addressing by the disk ID, for example "/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4") by-label (addressing by the disk label, for example "/dev/disk/by-label/MyStorage") by-uuid (addressing by the disk UUID, for example "/dev/disk/by-uuid/0e808a2f-cdd3-4944-a245-f729ffd73882") Be careful not to destroy all the information from the disk dedicated to Charon-AXP by mistake. These disks must not be formatted by the host OS.
"/dev/sg<N>"	Direct mapping to some SCSI device, for example, a SCSI disk or tape reader. How to find proper "/dev/sg" device is explained in this section .
"/dev/sr<N>" "/dev/cdrom" "/dev /cdrom<N>"	Mapping to host CD-ROM device.
"<file-name>.iso"	Mapping to an ISO file for reading distribution CD-ROM image. Mapping may also include the full path (recommended), for example: "/my_disks/vms_distributive.iso"
"<file-name>.vtape"	Mapping to the file representing the tape (tape image). These files are created automatically. Mapping may also include a full path (recommended), for example: "/my_tapes/backup.vtape"

Additionally it is possible to specify a parameter "media_type" to assign the type of the attached media explicitly.

Example:

```
set PKA media_type[600]="RX23"
```

Numbers in the square brackets represent SCSI addresses and LUNs associated with each container of the KZPBA controller. They have the following structure:

[XXYY], where

Parameter	Range	Description
XX	0...15	Stands for SCSI ID of each connected unit. Note that KZPBA itself has some ID associated with it. By default it is 7, but it can be changed in the following way: <code>load KZPBA PKA scsi_id = 0</code> In this example an instance "PKA" of KZPBA controller is assigned with SCSI ID 0.
YY	00...07	Stands for LUN.

It is possible to load several KZPBA controllers: DKB, DKC, etc. by configuring specific placement for them on the PCI bus. It is discussed in details in the "[Configuration details](#)" chapter of this Guide.

Some Alpha systems emulated by CHARON-AXP have already had one or two KZPBA controllers preloaded. If the system has only one preloaded controller, the template configuration file usually provides some sample line on how to add another one, for example:

```
load KZPBA PKA bus=pci_1 device=1 function=0 irq_bus=isa irq=24
```

Follow [this link](#) for details of KZPBA controllers configuration.

KGPSA-CA PCI FC disk controller

Optionally it is possible to configure KGPSA-CA FC disk controller.

It can be configured in 2 modes:

- [CHARON-AXP for Linux configuration#KGPSA-CA mapping to the host resources](#)
- [CHARON-AXP for Linux configuration#KGPSA-CA pass through mode](#)

Below is an example of KGPSA-CA controller loading:

```
load KGPSA FGA
```

Optionally another KGPSA-CA adapter can be loaded similar way:

```
load KGPSA FGB
```

Follow [this link](#) for details of KGPSA-CA controllers configuration.

KGPSA-CA mapping to the host resources

Below is the typical configuration options for KGPSA-CA PCI FC disk controller, mapped to the host resources ("L" is letter here):

```
load KGPSA FGA
#set FGA container[0] = "<file-name>.vdisk"
#set FGA container[100]="/dev/sd<L>"
```

The first line ("load KGPSA FGA") loads disk controller KGPSA with name "FGA", followed by 2 groups of lines showing different ways of mapping to the host resources:

Type of mapping	Description
"<file-name>.vdisk"	<p>Mapping to the file representing a physical disk of the Alpha system (disk image). These files can be created from scratch with "mkdskcmd" utility. Data and OS disk backups are transferred from the original system via tapes or network and restored into these container files. Mapping may also include the full path (recommended), for example: "/my_disks/my_boot_disk.vdisk"</p>
"/dev/sd<L>"	<p>Mapping to physical disk. "L" is letter here. Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake.</p> <p>These disks must not be formatted by the host OS.</p> <p>It is also possible to use not a whole disk, but previously created partitions on it. In this case the syntax is the following: "/dev/sd<L><N>" where N is the number of partition to be used.</p> <p>Please note:</p> <p>Since "/dev/sd<L>" addressing is not persistent, so it is strongly recommended to use "/dev/disk/by-id/wwn-*" syntax instead to refer the disk by its WWID - especially in the environments utilizing FC and SAN storages (see below).</p>
"/dev/dm-<N>" "/dev/mapper/mpath<N>" "/dev/mapper/disk<N>"	<p>Mapping to multipath disk. Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake.</p> <p>These disks must not be formatted by the host OS.</p>
"/dev/disk/by-*"	<p>Mapping to physical disk.</p> <ul style="list-style-type: none"> • by-id (addressing by the disk ID, for example "/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4") • by-label (addressing by the disk label, for example "/dev/disk/by-label/MyStorage") • by-uuid (addressing by the disk UUID, for example "/dev/disk/by-uuid/0e808a2f-cdd3-4944-a245-f729ffd73882") <p>Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake.</p> <p>These disks must not be formatted by the host OS.</p>

Numbers in the square brackets represent KGPSA-CA units. They can be in the range 0..32766, but no more than 255 units can be configured on a single controller.

KGPSA-CA pass through mode

It is also possible to use the emulated KGPSA-CA in "pass through" mode to address a physical EMULEX LightPulse PCI/PCI-X/PCIe FC adapter plugged into the host's PCI/PCI-X/PCIe slot.

The sample configuration file provides a template for this type of mapping:

```
#set FGA host_bus_location = "/dev/kgpsaX"
#set FGB host_bus_location = "/dev/kgpsaY"
```

Follow [this link](#) for detailed description of building and installation of an EMULEX LightPulse PCI/PCI-X/PCIe FC adapter driver.

Auto boot

Charon-AXP can be configured to automatically boot an operating system at start up by specifying the default boot device and setting the 'auto_action' parameter to 'restart' from the console.

Example: dka0 is defined as the default boot device

```
P00>>>set bootdef_dev dka0
P00>>>set auto_action restart
```

Network boot

Charon-AXP can be configured to boot on network if the legacy operating system allows it via MOP. MOPv3 (DECnet) and MOPv4 (LANCP) are supported.

Usage/example:

```
P00>>>boot [-flags ...] [-file ...] ewa0
```

Both EW and EI adapters are supported for MOP network boot.

Migration to CHARON-AXP for Linux

Table of Contents

- Introduction
- Collecting information about the source Alpha system
- Creation of Charon-AXP configuration file
- Making disk images
- Installation of Alpha operating system
- Making remote backups
- Restore backups to Charon-AXP disks
- Alternative ways of data transfer

Introduction

This section describes how to migrate your Alpha system to Charon-AXP. We will use a sample AlphaServer ES40 system running OpenVMS to demonstrate the migration procedure. This chapter also contains some instructions for Tru64 UNIX, but it mainly concentrates on how to migrate OpenVMS hosts. The process is similar for all Charon-AXP models.

Collecting information about the source Alpha system

The first step is to determine the exact configuration of your Alpha hardware in order to create the Charon-AXP configuration file that will accurately mimic the hardware.

Turn on your source Alpha system. At the ">>>" prompt, issue a "show device" command:


```
>>>show device

sys0.0.0.0.0      SYS0      System ROOT Device
ewa0.0.0.1.1      EWA0      F8-D1-11-00-67-E6
pka0.0.0.2.1      PKA0      Q-Logic/ISP PCI SCSI HBA
pga0.0.0.3.1      PGA0      WWN 1000-0000-0248-C550
pqa0.0.0.15.0     PQA0      ALi 1553C Integrated IDE Controller
pqb0.0.1.15.0     PQB0      ALi 1553C Integrated IDE Controller
dqa0.0.0.15.0     DQA0      TSSTcorpCDDVDW SH-222BB
dka0.0.0.2.1      DKA0      DEC RZ28 (C)DEC
dka100.1.0.2.1    DKA100    DEC RZ22 (C)DEC
dka200.2.0.2.1    DKA200    DEC RZ23 (C)DEC
mka600.6.0.2.1    MKA600    Virtual SCSI Tape

>>>
```

To get more detailed information, boot OpenVMS and issue a "show device /full" command:

```
$ show device /full
Disk PFCAXP$DKA0:, device type RZ28, is online, mounted, file-oriented device,
shareable, available to cluster, error logging is enabled.
...
Disk PFCAXP$DKA100:, device type RZ22, is online, file-oriented device,
shareable, available to cluster, error logging is enabled.
...
Disk PFCAXP$DKA200:, device type RZ23, is online, file-oriented device,
shareable, available to cluster, error logging is enabled.
...
Disk PFCAXP$DQA0:, device type TSSTcorpCDDVDW SH-222BB, is online, file-
oriented
device, shareable, available to cluster, error logging is enabled.
...
Disk $1$DGA0: (PFCAXP), device type RZ24, is online, file-oriented device,
shareable, available to cluster, error logging is enabled.
...
Magtape PFCAXP$MKA600:, device type Virtual SCSI Tape, is online, file-oriented
device, available to cluster, error logging is enabled, device supports
fastskip (per_io).
...
Terminal OPA0:, device type VT102, is online, record-oriented device, carriage
control.
...
Device EWA0:, device type DE500, is online, network device, device is a
template
only.
...
Device FGA0:, device type KGPSA Fibre Channel, is online, shareable, error
logging is enabled.
...
Device PGA0:, device type SCSI FCP, is online, error logging is enabled.
...
Device PKA0:, device type Qlogic ISP1020 SCSI port, is online, error logging is
enabled.
...
Device $1$GGA32767:, device type Generic SCSI device, is online, shareable.
$
```

 In case of Tru64 UNIX V5 running on the host system it is recommended to use the following commands to get information on the host configuration:

Command	Description
# /sbin/hwmgr view devices	Get detailed information about the host hardware configuration
# /sbin/hwmgr show scsi	Get specific information about the host SCSI controllers and attached disks
# /sbin/hwmgr view hierarchy	Get information about the host controllers

Please reference to the Tru64 UNIX User's Guide for more details.

The source Alpha peripheral configuration in this example is:

Controller	Devices on controller	Description
KZPBA	-DKA0 (RZ28) -DKA100 (RZ22) -DKA200 (RZ23) -MKA600 (tape)	SCSI disk/tape controller
KGPSA-CA	-DGA0 (RZ24)	FC disk controller
OPA0		System console
Acer Labs 1543C IDE/ATAPI CD-ROM adapter	-DQA0	IDE CD-ROM controller
EWA0		Network interface, MAC address: "F8-D1-11-00-67-E6"

Now collect some general information about the AlphaServer ES40 system:

```
>>>show cpu /full

System: PFCAXP, AlphaServer ES40 6/667

SMP execlret = 3 : Enabled : Streamlined.
Config tree = None
Primary CPU = 0
HWRPB CPUs = 4
Page Size = 8192
Revision Code =
Serial Number = SN01234567
Default CPU Capabilities:
System: QUORUM RUN
Default Process Capabilities:
System: QUORUM RUN

....

>>>
```

```
>>>show mem

System Memory Resources on 12-MAR-2020 09:29:16.42


Physical Memory Usage (pages): Total   Free  In Use  Modified
Main Memory                (512.00MB) 65536 56496   8610 430

...

>>>
```

So the collected information about the AlphaServer ES40 system is:

Component	Value
System Type	AlphaServer ES40 6/667
Serial Number	SN01234567
Number of CPUs	4
System memory	512 Mb

 In some particular situations it is also important to know the exact placement of all the peripheral devices on Alpha PCI bus. To get that information issue a "show config" command at ">>>" prompt of Alpha console, for example:

```
>>>show config
...
PCI Bus

Bus 00 Slot 03: DECchip 21142 Network Controller
ewa0.0.0.3.0 00-00-F8-03-9A-6D

Bus 00 Slot 07: Cypress PCI Peripheral Controller

Bus 00 Slot 07: Function 1: PCI IDE
Bus 00 Slot 07: Function 2: PCI IDE

Bus 00 Slot 07: PCI USB

Bus 00 Slot 08: DECchip 21052 PCI to PCI Bridge

Bus 01 Slot 08: ISP1040 Scsi Controller
pka0.7.0.1008.0 SCSI Bus ID 7
dka0.0.0.1008.0 RZ2DD-KS
dka400.4.0.1008.0 RRD45

>>>
```

The "show config" command collects the following information of placement of peripheral devices on PCI bus:

- Bus number
- Slot number
- Function number

To find out the exact types of controllers and other useful information refer to the source Alpha system documentation.

Creation of Charon-AXP configuration file

Using the above information, the following configuration can be created:

```
#
# AlphaServer model: AlphaServer ES40 6/667
#

set session hw_model = AlphaServer_ES40
set ace cpu_architecture = EV67
set rom dsrdb[0] = 1820 system_name = "AlphaServer ES40 6/667"

...

#
# Override default System Serial Number, set it to "SN01234567"
#

set rom system_serial_number = SN01234567

#
# Specify RAM size: 512 Mb
#

set ram size=512

#
# Map OPA0 console to the xtem from which CHARON-AXP runs
#

set COM1 alias=OPA0 line=(console)

#
# Connect the emulator's DQA0 to the host's ATAPI CD/DVD-ROM drive.
#

set ide container="/dev/cdrom"

#
# Load optional DE500BA PCI Ethernet Adapter (EWA0) and map it to the "eth1" host network interface
#

load DE500BA/dec21x4x EWA interface=EWA0
load packet_port/chnetwrk EWA0 interface="eth1"

#
# Load DEC-KZPBA SCSI controller and map it to 3 disk containers and 1 tape container
#

load KZPBA PKA scsi_id = 7

set PKA container[0] = "/my_disks/bootable.vdisk"
set PKA container[100] = "/my_disks/RZ22.vdisk"
set PKA container[200] = "/my_disks/RZ23.vdisk"

set PKA container[600] = "/my_tapes/my_tape.vtape"

#
# Load DEC-KGPSA-CA PCI FC adapter and map it to a disk container
#

load KGPSA FGA

set FGA container[0] = "/my_disks/RZ24.vdisk"

...
```

Making disk images

In our example, possible mappings of the [KZPBA SCSI controller](#) include disk and tape images. Tape images do not have to be manually created whereas you have to provision disk images, as described below.

The example below creates disk images of the original physical type. You should avoid the temptation to create identically-sized virtual disks, and regard the migration as an opportunity to create larger virtual disks than the original disks, if needed. Of course, if you increase the size of the disks, you will have to migrate a *logical* backup ("VMS Backup") of the filesystem, as opposed to a physical backup of the original disk.

Create special directories for storing disk and tape images, as needed. These directories are referenced in the sample configuration file above.

```
$ mkdir /my_disks
$ mkdir /my_tapes
```

Next, create the disk images using the "mkdiskcmd" utility:

```
$ mkdiskcmd -d rz24 -o /my_disks/rz24.vdisk
Please wait...
100% done
Success.
$ mkdiskcmd -d rz23 -o /my_disks/rz23.vdisk
Please wait...
100% done
Success.
$ mkdiskcmd -d rz22 -o /my_disks/rz22.vdisk
Please wait...
100% done
Success.
$ mkdiskcmd -d rz28 -o /my_disks/bootable.vdisk
Please wait...100% done
Success.
```

Installation of Alpha operating system

The next step is to transfer the data from the source Alpha system to Charon-AXP. The easiest way to do this is via backups over the network. For this operation we need a bootable, network-enabled operating system on a Charon-AXP disk image or physical disk.

The example configures the Charon-AXP AlphaServer ES40 system for installation of OpenVMS from a distribution CD-ROM (usually it is "/dev/cdrom" if the host has only one CD-ROM drive):


```
#
# DEC-KZPBA SCSI controller is mapped to 5 disk containers; one of them (DKA300) - for migration purposes;
# another one (DKA400) - for installation of fresh OpenVMS system from distributive
#

load KZPBA PKA scsi_id = 7

set PKA container[0] = "C:\my_disks\bootable.vdisk"
set PKA container[100] = "C:\my_disks\RZ22.vdisk"
set PKA container[200] = "C:\my_disks\RZ23.vdisk"
set PKA container[300] = "C:\my_disks\migration.vdisk"
set PKA container[400] = "C:\my_disks\fresh_openvms.vdisk"

#
# CD-ROM for OpenVMS installation (DQA0)
#

set ide container="/dev/cdrom"
```

 DKA300 will be the disk where all the source disks will be copied so its size needs to be large enough to store all the disk backup images.

Create an empty disk image for installation of OpenVMS and another one for storing backups from the source Alpha system as it is shown in the section above.

Run Charon-AXP and boot from the CDROM named "dqa0" ("migration.cfg" is the configuration file we use in this example):

```
$ es40.exe migration.cfg

initializing ...

polling for units on kzpba0, slot 2, bus 0, hose 1 ...
pka0.0.0.2.1 PKA0 Q-Logic/ISP PCI SCSI HBA

... enter console

CHARON-AXP/ES40 for Linux (AlphaServer ES40 6/667), Version 4.11.20403
(C) 2009-2020 STROMASYS SA.
All rights reserved.

P00>>>boot dqa0
```

Install Alpha/VMS including DECnet on "dka400". The DECnet address must belong to the same area as the source Alpha system.

Login to the newly installed OpenVMS system and initialize the disk intended for backups storage. Let's assume it's prompt is "newvms\$".

Note: The qualifier /NOHIGH is used here since highwater marking is no generally required for these operations as the disk will be overwritten by the savesets. This will speed up the process significantly.


```
newvms$ INIT/NOHIGH DKA300: SCRATCH
newvms$ MOUNT/SYSTEM/NOASSIST DKA300: SCRATCH
```

Making remote backups

Now we are ready to create disk backups from the source Alpha system to Charon-AXP.

Boot the Charon-AXP virtual machine and make sure that the source Alpha system is reachable via DECnet.

Login to the source Alpha system, stop all the batch queues, kick off the users, stop all applications and close databases if any. The commands listed in the SYS\$MANAGER:SYSHUTDOWN.COM file may be helpful. The goal is to close as many files as possible. The system disk will have several files opened (pagefile, swapfile, etc.), this is a normal situation.

 The use of the "SHOW DEVICE /FILES" command would be of help to know files opened on a disk

Let's assume the Charon-AXP system is node 1.400 in this example. Issue then the following commands from the source Alpha whose prompt is set to "source\$":

```
source$ BACKUP/IMAGE/IGNORE=INTERLOCK DKA0: 1.400"username password": :DKA300:[000000]DKA0.BCK/SAVE
source$ BACKUP/IMAGE/IGNORE=INTERLOCK DKA100: 1.400"username password": :DKA300:[000000]DKA100.BCK/SAVE
source$ BACKUP/IMAGE/IGNORE=INTERLOCK DKA200: 1.400"username password": :DKA300:[000000]DKA200.BCK/SAVE
```

When the backup procedure will be completed, the disk "DKA300" of the Charon-AXP virtual machine will contain 3 savesets: "DKA0.BCK", "DKA100.BCK" and "DKA200.BCK"


Restore backups to Charon-AXP disks

Next, restore the new savesets to their corresponding virtual disks. Login to Charon-AXP and issue this sequence of commands to restore all the savesets created in the previous step:

```
newvms$ MOUNT/FOREIGN DKA0:
newvms$ BACKUP/IMAGE DKA300:[000000]DKA0.BCK/SAVE DKA0:
newvms$ DISMOUNT DKA0:
newvms$ MOUNT/FOREIGN DKA100:
newvms$ BACKUP/IMAGE DKA300:[000000]DKA100.BCK/SAVE DKA100:
newvms$ DISMOUNT DKA100:
newvms$ MOUNT/FOREIGN DKA200:
newvms$ BACKUP/IMAGE DKA300:[000000]DKA200.BCK/SAVE DKA200:
newvms$ DISMOUNT DKA200:
```

If you are going to have the Charon-AXP and the original physical Alpha on the network at the same time, you must change the network identity of one (usually the Charon-AXP).

The easiest way is to boot the Charon-AXP virtualized system on the restored system disk with the network disabled and to configure new addresses, as needed.

 The NIC can be disabled with a "disabled" statement in the Charon configuration file.

Then Enable the network and reboot.

Alternative ways of data transfer

Some alternative methods of data transfer are also possible. For example:

- Connect a SCSI tape drive to the Charon-AXP host via a PCI card
 - Map the tape drive in the Charon-AXP configuration file
 - a. Restore the source Alpha system backups from tape to disk images via OpenVMS running on Charon-AXP.
 - b. Boot from standalone backups and restore the content to Charon-AXP virtual disks.
 - Dump the source Alpha system backups to tape images with the "mtd" utility and:
 - a. Boot from the freshly installed OpenVMS system and restore the tape images to Charon-AXP virtual disks.
 - b. Boot from standalone backups and restore the content to Charon-AXP virtual disks.
- Create a network cluster between the source Alpha system and Charon-AXP (it is possible to use the source system as a boot server) then perform backups from one disk to another:

```
$ BACKUP/IMAGE/IGNORE=INTERLOCK REAL$DKA0: DKA0:
```

CHARON-AXP for Linux virtual network

Table of Contents

- General description
- Using "ncu" utility to establish Charon virtual network
- Usage of the virtual interface in Charon-AXP configuration

General description

It is strongly recommended to use only physical network adapters for Charon-AXP networking to gain maximum performances. In situations where the host has only one network adapter, you can use Linux virtual network Interfaces ("TUN/TAP") and map individual Charon-AXP instances to their own virtual interfaces. This can be done using the `ncu` utility.

It is also possible to perform the operations manually. Refer to your Operating System Network Administration guide for details.

Please note:

On Red Hat Enterprise Linux 7 and CentOS 7, the following packages are needed:

- `bridge-utils` - optional
- `tunctl` - optional, need only if command 'ip tuntap' not worked
- `ethtool` - mandatory
- `vconfig` - optional, if VLAN is needed

On Red Hat Enterprise Linux 8 and 9 (and derivatives), the following package is needed:

- `ethtool` - mandatory

Using "ncu" utility to establish Charon virtual network

Login as root and start the `ncu` utility:

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version
1.7

Interfaces Dedicated to State
-----
eth0 host connected to host
eth1 host disconnected from host
lo host unmanaged from host
=====
bridge name bridge id STP enabled interfaces
===== VLAN
=====
select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 3
```

Enter "3" to create a bridge between the host physical network adapter and the Linux virtual network interfaces (TAP) and specify the physical network interface ("eth1" in our example) and the number of virtual network interfaces to be created (2 in our example):

```
Specify the interface to be used for BRIDGE:eth1
How many tap should be created:2
Forming the bridge: ..1..2..3..4..5.. addif tap0 .. addif tap1 ..7..8 done!
Formed bridge br0_eth1 attached over eth1...

select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 7
```

Now enter "7" to see the created virtual interfaces:

```
Interfaces Dedicated to State
-----
eth0 host connected to host
eth1 bridge disconnected from bridge
lo host unmanaged from host
tap0 CHARON connected to host
tap1 bridge connected to bridge
=====
bridge name      bridge id          STP enabled  interfaces
br0_eth1         8000.22314588acac  no           eth1
                                                         tap0
                                                         tap1
===== VLAN
=====
select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 8
```

In the example above we see 2 virtual network interfaces, "tap0" and "tap1", connected to the created bridge. The physical network interface "eth1" is used for the bridge to the virtual network interfaces.

The interfaces "tap0" and "tap1" are ready to be used in Charon configurations, they do not need to be additionally dedicated to Charon.

Enter "8" to quit the "ncu" utility.

Usage of the virtual interface in Charon-AXP configuration

Once the "*tap<N>*" interfaces have been created, the load command maps those interfaces to Charon-AXP:

```
...
load tap_port/chnetwrk XQA0 interface="tap<N>"
...
```

CHARON-AXP for Linux Licensing

Charon emulators require a product license to run. There are two licensing options that are described in the following sections:

- Sentinel HASP licensing
- Virtual Environment (VE) licensing

CHARON-AXP for Linux HASP licensing

Table of Contents

- General description
- Parameters defined by CHARON-AXP license
- CHARON-AXP licensing models
 - Regular Sentinel HASP keys
 - Network Sentinel HASP keys
 - Software licenses
- Multiple licenses configuration
- License installation
 - Installation of Regular and Network license keys
 - Installation and update of CHARON-AXP Software or Hardware License
- License management
 - Sentinel Admin Control Center
 - General Description
 - Disable remote keys access
 - Accessing Sentinel Admin Control Center from remote hosts
 - License management utilities
- Removing CHARON-AXP software licenses
- License deinstallation
- Special "backup" license keys
- Emulator Behavior

General description

The CHARON-AXP products are protected by licenses, issued by STROMASYS for each customer individually. The CHARON-AXP license defines a set of Alpha emulators allowed to run.

The license is implemented in the form of a hardware dongle (a Sentinel HASP key) or a software license bound to the hardware. Please be careful with your license key. In case of loss or damage, CHARON-AXP will not run or start unless the license key is replaced. For extra protection, STROMASYS recommends the use of a backup license key (purchased separately) that can replace the main license key for a restricted period of time. It is possible to specify the backup license in the CHARON-AXP configuration file to prevent CHARON-AXP from stopping in case its main license is no longer accessible.

The CHARON-AXP license is read upon the start of each instance of CHARON-AXP and at a specified interval (defined by the license content) during the emulated system execution. If CHARON-AXP detects the absence (or malfunction) of the license key / software license, CHARON will try to use a backup license (if specified in the configuration file). If the license is not available / not specified, CHARON displays a warning message in the log file requesting license key reconnection or software license reactivation. If the license is not reconnected within 12 hours, CHARON-AXP exits. For more, see the *Emulator Behavior* chapter further down on this page.

Note that if the time-restricted license is used and it expires, CHARON-AXP tries to find its replacement automatically and, if found, CHARON-AXP proceeds using the replacement license.

Please note:

- The CHARON-AXP software license is not distributed in case of Proof-of-Concept and evaluation installations. Only hardware dongles are used in this case.
- Hardware dongles, apart from HL-MAX dongles, are equipped with a **battery** and a clock, which makes them independent of the host clock. The battery is not rechargeable. However, the dongle can use the power provided by the host system while it is plugged in. By doing this, the depletion of the battery can be slowed down. Check the dongle at regular intervals if it is not permanently connected to a system. If the battery becomes completely depleted, the dongle will be permanently unusable and must be replaced. See also: [How long does the license USB dongle battery last upon a full charge.](#)

Update of the CHARON-AXP license can be performed on the fly without stopping CHARON-AXP. At the next license check, CHARON-AXP will use the updated license normally.

The following sections list all the main parameters of the CHARON-AXP licensing mechanism.

Parameters defined by CHARON-AXP license

The following table represents all the parameters defined by CHARON-AXP license:

General	Products relevant	Optional
<ul style="list-style-type: none"> Physical key ID License Number End user name Master key ID License release date and time Update Number Purchasing Company name. In most cases the company to which the key was issued originally 	<ul style="list-style-type: none"> Commercial product name Commercial product code Commercial product version and range of build numbers suitable for running Range of CHARON-AXP virtual models available for running Type of host CPU required Host operating system required Number of virtual CPUs enabled for virtual SMP systems Minimum number of host CPU cores required Minimum host memory required Maximum memory emulated. If not present the value defaults to the maximum memory possible for the particular virtual system. Note that the maximum memory may not be available to the virtual system if the host computer has insufficient physical memory. Maximum number of CHARON-AXP instances that can be run concurrently Whether or not CHAPI (CHARON-AXP API) can be used with this product Product and Field Test expiration dates (if any) Product and Field Test executions counter (if any) Maximum number of hosts that may run CHARON-AXP concurrently (in the case of a networking license) Level of support (if any), end date of any support contract, the "First Line" Service Provider Frequency of CHARON-AXP license checking during CHARON-AXP execution 	<ul style="list-style-type: none"> Parameter that reduces the maximum speed of the program Parameter that prohibits use of Advanced CPU Emulation. If not present the Advanced CPU Emulation is enabled

CHARON-AXP licensing models

CHARON-AXP licensing models are divided in 3 groups:

Regular Sentinel HASP keys

This is most common way of CHARON-AXP licensing, the CHARON-AXP license is embedded in a Sentinel HASP dongle. This license is available only on the host where the dongle is physically installed.

The CHARON-AXP installation procedure takes care of the Sentinel HASP run-time (driver) installation. Once the CHARON-AXP product has been installed, it is possible to plug-in the regular license key and proceed with CHARON-AXP usage without additional configuration steps.

The number of CHARON-AXP instances allowed to run on a particular host may be restricted by the license content (see above).

Network Sentinel HASP keys

The Network Sentinel HASP key (red dongle) can be shared between several hosts running CHARON-AXP including the host on which the network license is installed.

If CHARON-AXP is installed on the host where the network key is connected, no additional steps are required. The Sentinel driver is activated as part of the CHARON-AXP installation. If the host does not have CHARON-AXP installed, the host can still distribute the connected network license to CHARON-AXP instances running on other hosts. In this case the Sentinel driver must be installed on the host manually.

The Sentinel run-time driver is distributed as a separate RPM package in the CHARON-AXP kit. Please see the "[CHARON-AXP for Linux HASP licensing#License installation](#)" section of this chapter for details.

Once the Sentinel run-time driver is installed and the network license is connected, CHARON-AXP can be started on any appropriate host on the LAN network segment. In the current CHARON-AXP/VAX versions, a network license controls the maximum overall number of active instances, which can be distributed across client host systems according to the preference of the customer.

Software licenses

The CHARON-AXP Software License is a "virtual" key with exactly the same functionality as the hardware dongle.

The CHARON-AXP Software license does not require any hardware but it requires the installation of the Sentinel run-time environment.

Please note:

Software licenses are best suited for stable environments, because their correct function depends on certain characteristics of the host system. Changing any of these characteristics will invalidate the license.

- If the CHARON host runs on real hardware, the software licenses are by default **tightly bound to the hardware** for which they were issued. If major hardware characteristics of the system are changed, the license will be disabled.
- If the CHARON host runs in a **virtual environment** (e.g. VMware), the software licenses are normally bound to the virtual machine ID and a set of additional characteristics of the virtual machine. If any of these parameters are changed, the license will be disabled.

For a more detailed description of the restrictions, please refer to the [Software Licensing restrictions](#) article or contact your Stromasys representative.

Software licenses are always network-wide on Linux so they behave the same way as Network HASP keys.

Multiple licenses configuration

For any type of licensing, CHARON-AXP can use **only one valid ("active") license (of given vendor code) at a time**.

The "hasp_srm_view" utility displays the "active" license by default and is able to display all available licenses with the "-all" parameter. It is also possible to check some specific license by its number using the "-key" parameter.

The utility provides the license number and ID / IP address of the host where the active license is installed.

The general recommendation is to avoid usage of multiple keys in one network segment. Use only one locally installed license per host or one network license per local network segment containing several CHARON-AXP hosts.

When needed, it is possible to use a special parameter in the CHARON-AXP configuration file to specify exactly which license must be used by each particular instance of CHARON-AXP:

Parameter	license_key_id
Type	Text string
Value	<p>A set of Sentinel Key IDs that specifies the license keys to be used by CHARON. It is also possible to use a keyword "any" to force CHARON to look for a suitable license in all available keys if the license is not found in the specified keys.</p> <p>Example:</p> <pre>set session license_key_id = "1877752571,354850588,any"</pre> <p>Based on the presence of this parameter in the configuration file, CHARON behaves as follows:</p> <ol style="list-style-type: none"> No keys are specified (the parameter is absent) CHARON performs an unqualified search for any suitable key in unspecified order. If no key is found, CHARON exits. One or many keys are specified CHARON performs a qualified search for a regular license key in the specified order. If it is not found, CHARON exits (if the keyword "any" is not set). <p>If the keyword "any" is specified then if no valid license has been found in the keys with specified ID's all other available keys are examined for valid license as well.</p> <p>The order in which keys are specified is very important. If a valid license was found in the key which ID was not the first one specified in configuration file, then available keys are periodically re-scanned and if the key with the ID earlier in the list than the current one is found CHARON tries to find a valid license there and in case of success switches to that key.</p>

License installation

Installation of Regular and Network license keys

Installation of CHARON-AXP regular and network licenses consists of:

1. Installation of the Sentinel run-time environment on the CHARON-AXP host (regular and network keys) or on the host that will distribute CHARON-AXP licenses over a local network segment (network key only). The Sentinel software (the "aksusbd" RPM package) is installed automatically by CHARON-AXP for Linux.
2. Physical connection of the HASP license dongle to the CHARON-AXP host or to the host distributing the CHARON-AXP license over the local network segment.

When manual installation of Sentinel run-time is required (in the case of the network license server that does not have CHARON-AXP installed), open the CHARON-AXP kit folder and proceed the following way:

```
# rpm--nodeps-ihvaksusbd-7.63-1.i386.rpm charon-license-4.10-20200.e174.x86_64.rpm
```

In case of network-wide license (red dongle) do the following:

- *On the server side (where the network license will reside):* open port 1947 for both TCP and UDP
- *On the client side,* if broadcast search for remote licenses is to be used, UDP traffic from port 1947 of the license server to ports 30000-65535 of the client must be permitted.
- *Both on server and client sides:* setup default gateway

Please consult with your Linux User's Guide on details.

If stricter firewall rules are required, it is possible to open the ports 30000-65535 and 1947 only for the "/usr/sbin/hasplmd" daemon.

Some additional packages may be needed in certain cases, for example "glibc.i686"

Replacement of currently installed Sentinel run-time

Replacement of currently installed Sentinel Run-time may be needed in case of:

- Upgrade to a newer version of CHARON-AXP
- Installation of a specific CHARON-AXP license Run-time provided by STROMASYS

Run-time replacement is a two step process:

- Remove the current run-time (and the package "charon-license-<...>.rpm" containing the run-time customization) with the command

```
# rpm --nodeps -e aksusbd charon-license-<...>
```

- Change to the directory where the new run-time RPM resides (along with the corresponding "charon-license-<...>.rpm" customization package) and issue the command:

```
# rpm --nodeps -ihv aksusbd<...>.rpm charon-license-<...>.rpm
```

Installation and update of CHARON-AXP Software or Hardware License

CHARON-AXP Software Licenses (SL) can be installed / updated according to the procedure described below.

- Install CHARON-AXP together with Sentinel run-time (Sentinel run-time is an essential part of CHARON-AXP for Linux distribution)
- Reboot the host system
- Connect the HASP dongle to the host system (in case of update of a license in a dongle)
- Collect the CHARON-AXP host fingerprint file (".c2v") - in case of first installation of a Software License:

```
# hasp_srm_view -fgp my_host.c2v
```

or collect the ".c2v" file in case a Software License is already installed or the connected HL/HASP dongle needs updating:

```
# hasp_srm_view -c2v current_license.c2v
```

- Send the ".c2v" file ("my_host.c2v" / "current_license.c2v" in the examples above) to STROMASYS
- Receive a ".v2c" file in return and put it somewhere on the CHARON-AXP host.
- Start any web browser on this system and go to <http://localhost:1947> to access the "Sentinel HASP Admin Control Center" (ACC) or configure ACC for remote access (see the details below).
- In ACC, under the Options menu, select Update/Attach, "Browse" for the "*.v2c" file and then "Apply File".
- Ensure that the license appears in the "Sentinel Keys" menu.

Alternatively it is also possible to use "[hasp_update](#)" command line utility for applying the ".v2c" file.

The content of the installed software license is not shown by the Sentinel HASP Admin Control Center. To see it please run the "[hasp_srm_view](#)" utility from the local console or configure remote access according to the instructions given in the "[hasp_srm_view](#)" utility section.

In case of network-wide software license do the following:

- *On the server side (where the network license will reside):* open port 1947 for both TCP and UDP
- *On the client side,* if broadcast search for remote licenses is to be used, UDP traffic from port 1947 of the license server to ports 30000-65535 of the client must be permitted.
- *Both on server and client sides:* setup default gateway

Please consult with your Linux User's Guide on details.

If stricter firewall rules are required, it is possible to open the ports 30000-65535 and 1947 only for the "/usr/sbin/hasplmd" daemon.

License management

CHARON-AXP license management is performed by the Sentinel Admin Control Center and specific utilities. They are described in the sub-sections below.

Sentinel Admin Control Center

General Description

The Sentinel Admin Control Center (ACC) is the web-interface to the Sentinel run-time environment. It allows viewing/managing available keys, enabling and disabling them, controlling usage of remote keys etc.

To access the ACC, start any web browser and open the <http://localhost:1947> page.

The Sentinel Admin Control Center is not able to display CHARON-AXP licenses - to view key contents, use the "hasp_srm_view" utility.

To access the Sentinel Admin Control Center start any web browser, open the <http://localhost:1947> page. The web interface of the Sentinel Admin Control Center will appear.

The screenshot below gives an example:

The screenshot shows the Sentinel Admin Control Center interface with the Gemalto logo and the title "Sentinel Admin Control Center". The main content area displays "Sentinel Keys Available on redhat7.localdomain". A table lists four keys with columns for #, Location, Vendor, Key ID, Key Type, Configuration, Version, Sessions, and Actions. The Actions column contains buttons for Products, Features, Sessions, Blink on, and C2V.

#	Location	Vendor	Key ID	Key Type	Configuration	Version	Sessions	Actions
1	Local	68704 (68704)	445532399	HASP HL Time	-	3.25	-	Products, Features, Sessions, Blink on, C2V
2	Local	68704 (68704)	527889790	HASP HL Time	-	3.25	-	Products, Features, Sessions, Blink on, C2V
3	Local	68704 (68704)	1202236799	HASP HL NetTime 10	-	3.25	-	Products, Features, Sessions, Blink on, C2V
4	Local	68704 (68704)	362831868	HASP HL Time	-	3.25	-	Products, Features, Sessions, Blink on, C2V

This example demonstrates that 4 license keys are available:

1. A network key ("HASP-HL NetTime") on the host "XEON4WAYW7"
2. A network key installed locally
3. An HASP-HL installed locally
4. A network-wide software license on the host "RH64"

The Sentinel Admin Control Center reports that there is one opened session on key #4. The other keys are not being used at the moment.

Using the Sentinel Admin Control Center it is possible to check the available keys, verify the hosts on which they reside, verify the opened sessions, etc. For a more detailed description of the Sentinel Admin Control Center, please refer to its "Help" section.

Disable remote keys access

A helpful feature of the Sentinel Admin Control Center is the ability to disable access to remote keys. If the network key is installed locally, access to the key from remote hosts can be disabled. The following examples demonstrate how this can be done.

To disable access to remote keys, switch to the "Access to Remote License managers" tab, uncheck the "Allow Access to Remote Licenses" checkbox and press the "Submit" button to apply this setting:

The screenshot shows the 'Configuration for Sentinel License Manager on ceres.stromasys.com' page. The 'Access to Remote License Managers' tab is active. The 'Allow Access to Remote Licenses' checkbox is unchecked and circled in red. Other options include 'Broadcast Search for Remote Licenses' (checked), 'Aggressive Search for Remote Licenses' (unchecked), and 'Remote License Search Parameters'.

To disable access to the locally installed license key from remote hosts, switch to the "Access from Remote Clients" tab, uncheck the "Allow Access from Remote Clients" checkbox and press the "Submit" button to apply this setting:

The screenshot shows the 'Configuration for Sentinel License Manager on ceres.stromasys.com' page with the 'Access from Remote Clients' tab selected. A red message states: 'Currently, a network-enabled Sentinel protection key is not connected to this License Manager.' The 'Allow Access from Remote Clients' checkbox is unchecked and circled in red. Below the checkbox is a text area for 'Access Restrictions'. At the bottom, there are 'Submit', 'Cancel', and 'Set Defaults' buttons.

Accessing Sentinel Admin Control Center from remote hosts

By default, the Sentinel Admin Control Center forbids accessing its web interface from remote machines.

To allow access, configure the ACC for remote management:

The screenshot shows the 'Configuration for Sentinel License Manager on Win7-Main' page. The 'Basic Settings' tab is active. The 'Machine Name' is 'Win7-Main'. The 'Allow Remote Access to ACC and Admin API' checkbox is checked and circled in red. Other settings include 'Display Refresh Time' (3 seconds), 'Table Rows per Page' (20), and 'Idle Timeout of Session' (720 minutes). There are also options for 'Write an Access Log File' and 'Include Local/Remote Requests'.

In this cannot be done using the WEB interface, edit the "hasplm.ini" file:

```
# vi /etc/hasplm/hasplm.ini
```

If the file does not exist, please refer to this article: [How-to enable remote connection to Sentinel Admin Control Center without GUI](#)

Allow remote access by changing the "ACCremote" parameter from "0" to "1", make sure the parameter "bind_local_only" is set to 0 (the value 1 means localhost-only) then restart the Sentinel Admin Control Center run-time:

```
# systemctl restart aksusbd
```

(or for RHEL 6.x: # `service aksusbd restart`)

If the CHARON-AXP host firewall is blocking remote access to the Sentinel Admin Control Center, please configure the firewall to open the port 1947 (TCP protocol). Refer to the Linux documentation for details on how to configure the firewall. It is also possible to use SSH port forwarding with the following command (replace "CHARON_MACHINE" by the real CHARON-VAX host name):

```
# ssh -L8080:CHARON_MACHINE:1947 root@CHARON_MACHINE
```

This will expose the Sentinel Admin Control Center on port 8080 to any computer, and it will believe commands are coming from the local host.

License management utilities

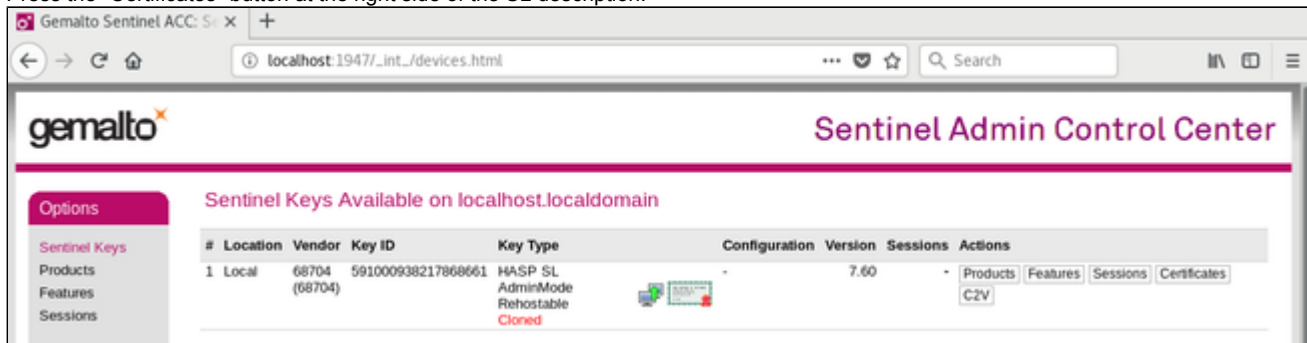
CHARON-AXP for Linux provides a specific utility for license management - "hasp_srm_view". This utility is used to display the license(-s) content, to collect the key(s) status information and host fingerprint (C2V) files.

Applying updates (".v2c" files) is typically done using the Sentinel Admin Control Center (see above) but alternatively it is also possible to use the specific "hasp_update" utility. Please refer to the [Utilities](#) section of this Guide for more details.

Removing CHARON-AXP software licenses

The following procedure must be applied to remove software license:

1. Using your web browser, open the <http://localhost:1947> page to access the "Sentinel HASP Admin Control Center" (ACC).
2. In the "Sentinel HASP Admin Control Center" (ACC), locate the target "Sentinel SL AdminMode" license.
3. Press the "Certificates" button at the right side of the SL description:



4. Note the name of the corresponding certificate and path to the certificates base in the "Certificates" section.
5. Remove the target certificate file from the specified directory, in most cases: "/var/hasplm/installed/68704/".
6. Restart the aksusbd service (# `systemctl restart aksusbd` or # `service aksusbd restart`) or reboot the CHARON host.
7. Start the "Sentinel HASP Admin Control Center" (ACC) again to ensure that the SL has been removed.

License deinstallation

To completely remove a CHARON-AXP license from a host, it is enough to remove the Sentinel run-time daemon (and the package "charon-license-<...>.rpm" containing the run-time customization) using the following command:

```
# rpm --nodeps -e aksusbd charon-license-<...>
```

Then just physically disconnect the license key (in the case of protection by dongles).

Special "backup" license keys

Backup keys are provided by STROMASYS along with standard license dongles. It is strongly recommended to order a backup key to recover immediately from damage or loss of the main license key. Backup keys use a counter (integer) value hardcoded inside the key. This integer value is a number of hours CHARON-AXP is allowed to run. Each time CHARON-AXP checks the license (every hour), the value is decreased (by 1 hour). Please note that backup keys have restricted functionality:

- CHARON run time is typically limited to 720 hours (30 days). This should be more than enough time to get a replacement key from STROMASYS.
- A backup license may be valid only until a certain date. Please check with STROMASYS management.

Emulator Behavior

Charon products **check the availability of a valid license** under several conditions:

1. At startup:

- If no valid license is found, an error message will be written to the emulator log file and the emulator will not start.
- In some emulator products it is possible to configure the number of retries and the waiting time between them by adding parameters to the emulator configuration file. Please refer to chapter [General Settings/license_key_lookup_retry](#) the details.

2. At regular intervals during the runtime of the emulator (the default license check period of 1 hour can be changed by Stromasys using the appropriate license parameters):

- If the previously used valid license has been removed, has disappeared, is defect, or has become invalid, the emulator will report the loss of the license in the log file and continue operation for a limited amount of time as described below.
- If there is another valid license, for example a backup license defined in the configuration file, it will be used.
- Charon allows for a grace period of 12 hours during which the software checks for the presence of a valid license every 10 minutes until a valid license is found. If no valid license is found after the grace period has expired, the emulator will stop.
- If a time-restricted license is used and it expires, the Charon instance tries to find its replacement automatically and, if found, proceeds using the replacement license

CHARON-AXP for Linux VE Licensing

The Charon emulator kit itself does not include VE license management tools. The following actions **must be performed on the VE license server**. However, the VE license server can be on the same system as the Charon emulator.

Please note:

- If you have not installed the license server yet (and for any more in-depth information), please refer to the VE license server user's guide (see [Licensing Documentation](#)).
- The information below shows the **command-line tools** for license management. Starting with version 1.1.16 of the VE license server, these activities can also be performed using a **web-based management GUI**. Please refer to the appropriate VE license server user's guide (see chapter *VE License Server Web-based Management GUI* in the VE license server documentation under [Licensing Documentation](#)).
- For Charon-AXP/VAX, VE license support is only available for the Linux-based products.

This section describes some important aspects of a VE licensing environment:

- [VE License Server Certificates Overview](#)
- [Firewall Considerations](#)
- [Creating a C2V File on a VE License Server](#)
 - [Running esxi_bind before First C2V Creation on VMware](#)
 - [Creating a VE C2V File and Sending it to Stromasys](#)
- [Installing a VE V2C File on a VE License Server](#)
- [Viewing the License on a VE License Server](#)
- [Charon-AXP/VAX Emulator License Configuration](#)
- [Updating an Existing License](#)

VE License Server Certificates Overview

Custom certificate support for VE licensing is planned for Charon-AXP version 4.12 or later.

This section provides a short overview of the certificates used by the VE license server and Charon-AXP/VAX. Please refer to the VE license server documentation for details.

The VE license server uses certificates for different purposes:

- License server operation: encrypted communication between license server and license clients (emulators).
 - New certificate support in the VE license server started with version 2.1.3. Changed certificate names starting with VE license server 2.2.2.
 - New certificate support for Charon-AXP/VAX is planned for version 4.12 or later (only new certificate names will be applicable).
- Web-based management GUI: encrypted (HTTPS) communication between the integrated license server web server and web browsers. Starting with VE license server version 2.1.4, the name of the certificate and its management changed. Please refer to the VE license server documentation.

Important information:

- General VE license server configuration:
 - The VE license server will – by default – use the old certificates. Therefore, compatibility with existing Charon clients will be maintained during an upgrade of the license server.
 - If the new certificates (using pre-defined names) are present in `/opt/license-server/certs`, these will be used and clients will have to use matching certificates. Please refer to the VE license server documentation for information how to activate the new certificates and, if desired, create custom certificates.
- Checking if the new certificates are enabled in a Charon-AXP/VAX installation:
 - Certificate location: `/opt/charon/bin/certs`
 - Sample certificate names: `ca.crt.sample`, `charon.crt.sample`, and `charon.key.sample`
 - If the directory contains the above files without the `.sample` suffix (e.g., `ca.crt`, `charon.crt`, `charon.key`), the new certificates have been enabled. On the license server, the sample files (for root CA and license server) are in `/opt/license_server/certs`. Please see the *VE License Server guide* in [License Documentation](#) for more information.
- **Make sure you understand the implications and possible side-effects before changing the certificate configuration.** Incorrect configurations can lead to the loss of license access and interruptions in operation.

Firewall Considerations

If the VE license server is not installed on the same system as the emulator, any intermediate firewall must allow at least the port on which the license is served. Optionally, the firewalls must allow the port on which the web-based GUI is available. These ports are **configurable** on the VE license server. The default values are the following:

- Default port on which licenses are served by the VE license server: TCP 8083.
- Default port on which the web-based GUI runs: TCP 8084.

Creating a C2V File on a VE License Server

Running esxi_bind before First C2V Creation on VMware

Only required if the license server is to be run on VMware!

The **esxi_bind** command sets up the necessary communication connection between the VE license server and the ESXi host / the vCenter Server.

It must be run on the license server (and the backup license server, if applicable):

- **once** before the first license is requested, **and**
- **again** should the user credentials, the password, or the address data for the access to the ESXi host / the vCenter Server change. Please make sure that the password of the selected user account does not automatically expire after a certain time period. This would cause disruptions in the license server operation and make it impossible for clients to receive their license.

Perform the following steps:

1. Use **ssh** to log in on the license server instance (assuming that username/password login is possible for an on-premises VMware installation).


```
# ssh <user>@<license-server-ip>
```

 where
 - a. *<user>* is the user for interactive login associated with your license server system
 - b. *<license-server-ip>* is the ip address of your license server system
2. Become the privileged user on the license server and run the **esxi_bind** program.
 - a. Become the root user: `# sudo -i`
 - b. Run the **esxi_bind** program:


```
# /opt/license-server/esxi_bind -a <address> -u <username> -p <password>
```

 where
 - i. *<address>* is the IP address of the ESXi host or vCenter Server
 - ii. *<username>* is a user on the ESXi host or vCenter Server (**see notes below**).
 - iii. *<password>* is the password of the user
3. If the command is successful, it will create the file `/opt/license-server/config.ini` containing the connection data (the password is encrypted).

Important notes regarding the user on the ESXi host or the vCenter Server:

1. The **username on the vCenter Server** can take different forms:
 - Simple username
esxi_bind parameter example: `-u myusername`
 - Username includes a domain name in one of the following two formats:
 - `<domain>\<username>`
esxi_bind parameter example (quotes are mandatory): `-u 'mydomain\myusername'`
 - `<username>@<domain>`
esxi_bind parameter example: `-u myusername@mydomain`
2. The user must have at least the following **global permissions** (i.e. the permissions cannot be limited to a specific VM):
 - Datastore > Allocate Space
 - VirtualMachine > Config > AddNewDisk
 - VirtualMachine > Config > RemoveDisk

Please note: if username and/or password contain Unix shell meta-characters, these characters must be escaped (enclose the string in single quotes, or add a backslash character in front of the meta-character).

Creating a VE C2V File and Sending it to Stromasys

The fingerprint is collected on the license server using the **c2v** utility.

Perform the following steps to collect the fingerprint on the license server and (if applicable) the backup license server:

1. Use **ssh** to log in on the license server instance.


```
# ssh -i ~/ssh/<mykey> <user>@<license-server-ip>
```

 where
 - a. *<mykey>* is the private key of the key-pair you associated with your cloud instance (for an on-premises VMware installation where login with username/password is allowed, it is not needed)
 - b. *<user>* is the user for interactive login associated with your license server instance (e.g., *opc* on OCI, *centos* for a CentOS instance on AWS, or the custom user on your VMware virtual machine or physical server; for an instance installed from a prepackaged Charon VE marketplace image, use *centos*)
 - c. *<license-server-ip>* is the ip address of your license server system
2. Become the privileged user and run the **c2v** program.
 - a. Become the root user: **# sudo -i**
 - b. Run the **c2v** program: **# /opt/license-server/c2v --filename <my-file>.c2v --platform <my-platform>**
 where
 - i. *<my-file>.c2v* is the path and name under which you want to store the fingerprint. The file type is C2V (customer-to-vendor)
 - ii. *<my-platform>* indicates the platform on which the license server runs (possible values: **physical, aws, oci, gcp, azure, ibm, nutanix, or esxi**).
3. Copy the resulting C2V file to your local system (unless you can send email from the license server system).
4. Send the C2V file to the Stromasys orders department (email address will be provided by Stromasys).

Installing a VE V2C File on a VE License Server

In response to the C2V file, Stromasys will send you a V2C file. This file contains the license data and is installed on the license server using the **v2c** utility.

Perform the following steps to install the license on the license server:

- Copy the V2C file to the license server (e.g., with SFTP).
- Use **ssh** to log in on the license server instance.


```
# ssh -i ~/.ssh/<mykey> <user>@<license-server-ip>
```

 where
 - <mykey>* is the private key of the key-pair you associated with your license server instance (for an on-premises VMware installation where login with username/password is allowed, it is not needed)
 - <user>* is the user for interactive login associated with your license server instance (e.g., *opc* on OCI, *centos* for a CentOS instance on AWS, or the custom user on your VMware virtual machine or your physical server; for an instance installed from a prepackaged Charon VE marketplace image, use *centos*)
 - <license-server-ip>* is the ip address of your license server system
- Become the privileged user and run the **v2c** program.
 - Become the root user: **# sudo -i**
 - Run the v2c program: **# /opt/license-server/v2c -f <my-file>.v2c**
 where *<my-file>.v2c* is the path and name under which you want to store the fingerprint. The file type is V2C (vendor-to-customer).

After the installation of the V2C file, the license server will be restarted.

Viewing the License on a VE License Server

The license data can be viewed via the web-based GUI of the VE license server (see [Licensing Documentation](#)). It can also be viewed with the **license_viewer** program using the following steps:

- Use **ssh** to log in on the license server instance.


```
# ssh -i ~/.ssh/<mykey> <user>@<license-server-ip>
```

 where
 - <mykey>* is the private key of the key-pair you associated with your license server instance (for an on-premises VMware installation where login with username/password is allowed, it is not needed)
 - <user>* is the user for interactive login associated with your license server instance (e.g., *opc* on OCI, *centos* for a CentOS instance on AWS, or the custom user on your VMware virtual machine or your physical server; for an instance installed from a prepackaged Charon VE marketplace image, use *centos*)
 - <license-server-ip>* is the ip address of your license server system
- Become the privileged user and run the **license_viewer** program.
 - Become the root user: **# sudo -i**
 - Run the license_viewer program: **# /opt/license-server/license_viewer**

Charon-AXP/VAX Emulator License Configuration

The license server must be added to the Charon emulator configuration.

Relevant parameter:

```
set session license_key_id = "VE://<license-server-IP>[:<port>]/<passphrase>/"
```

Where the following parameters are used:

- <license-server-IP>*: the IP address of the VE license server (127.0.0.1 if the VE license server is on the same host).
- <port>*: the TCP port on which the license is served (if not specified, the default port 8083 will be used).
- <passphrase>*: the passphrase of the correct product section on the license (optional). The parameter may be required for the emulator in some cases to identify the correct section.

To configure a **backup license server**, add the backup license server information to the same line after the primary license server information:

```
set session license_key_id = "VE://<primary-licserv-IP>[:<port>]/<passphrase>/, VE://<backup-licserv-IP>[:<port>]/<passphrase>/"
```

Only **one** backup server can be configured. The backup server typically provides a license limited to a certain number of runtime hours should the primary server become unavailable. If all valid licenses are lost or removed while an emulator is running, there is a grace period (configured on the license; default: 2 hours). The grace period is the time period during which the emulator continues to run after its license has been lost or removed. If there is no valid license after the grace period ends, the emulator will stop (this could cause data loss for a running guest system).

Updating an Existing License

If you need to update an **existing license**, for example because the time limit on the license has expired or to upgrade to a new product versions, perform the following tasks:

1. Generate the C2V file for the existing license. This Customer-to-Vendor (C2V) file contains the license characteristics necessary for creating the license update.
2. Send the C2V file to Stromasys. Stromasys will use the data to create the necessary license update. You will receive a V2C file (the Vendor-to-Customer file).
3. Apply the license data from the V2C file(s) on the license server. This will install and activate the update for your license.

CHARON-AXP for Linux utilities

CHARON-AXP provides the following set of utilities:

Utility	Description
mkdskcmd	Used to create CHARON virtual disk containers of custom or standard types. This utility also may be used to transfer virtual disks of one type to virtual disks of another type.
mtd	Used to create CHARON tape images from physical tapes and to write tape images back to physical tapes.
hasp_srm_view	Used to display the CHARON license contents, to collect the host system fingerprint and to collect license data required for license updates.
hasp_update	Sentinel standard utility. Used with Charon to install and update licenses.
ncu	Used to dedicate a host interface to CHARON-AXP, to release it back to the host and to manage CHARON virtual interfaces (TAPs).
CHARON Guest Utilities for OpenVMS	Used to manage virtual tapes and CHARON performance.

All these utilities (except for [CHARON Guest Utilities for OpenVMS](#)) are invoked from the Linux console command line.

mkdiskcmd

Table of Contents

- Description
- Creating disk images
- Resizing disk images

Description

The "mkdiskcmd" utility:

- Creates empty disk images of a given standard disk type or a custom disk size
- Transfers existing disk images of one type to disk images of another type.

Creating disk images

The first step is to obtain the name of the disk that needs to be created:


```
$ mkdiskcmd --list
```

This command results in a list of all supported disk types.

Choose the desired disk (for example "RZ22"), then use the "mkdiskcmd" command to create the virtual disk image as shown below:

```
$ mkdiskcmd --disk rz22 --output rz22.vdisk
```

A disk container "rz22.vdisk" will be created in the current directory.

-  A file "rz22.avdisk" will also be created. This file helps CHARON accurately recognize a specific disk image type. It is recommended to put the ".avdisk" file in the same directory as the created disk image.

It is also possible to create custom disk images using "--blcount" (blocks count) and "--blsize" (blocks size) switches.

To get all the available parameters please use the "--help"switch:

```

mkdisk for CHARON utility v. 1.16
Copyright (c) 2009-2019 STROMASYS. All rights reserved.

Usage:
  mkdiskcmd [Options]

Options:
  -h, --help           - display help screen

  -o, --output <file> - specify output file name

  -d, --disk <name>   - specify the disk name from Disk table

  -z, --blsize <value> - specify the block size in bytes (custom disk image)

  -c, --blcount <value> - specify number of the blocks (custom disk image)

  -a, --avtable <file> - specify AVDISK table file

  -r, --resize <file> [<disk-name>]
                        - resize the disk image
  <file>                - file name of the disk image to be resized
  <disk-name>           - name of the disk from the Disk table

  <file> will either have the specified number of blocks added to the
  end or be truncated at the new smaller size.

  To specify a custom disk size, use the following parameters:
  --resize <file> --blsize <value> --blcount <value>

  -s, --shrink        - mandatory parameter when resizing to smaller disk

  -l, --list          - to display AVDISK table

  -q, --quiet         - run in quiet mode

Return value:
  0                - Success
  Non zero        - Failure

Examples:
  mkdiskcmd -h
  mkdiskcmd -l
  mkdiskcmd -a /opt/charon/bin/mkdisk.vtable -o rk07.vdisk -d rk07
  mkdiskcmd -o custom.vdisk -z 512 -c 16384
  mkdiskcmd -r rz22.vdisk rz25 -a /opt/charon/bin/mkdisk.vtable
  mkdiskcmd -r rz22.vdisk rz25 -a /opt/charon/bin/mkdisk.vtable -z 512 -c 32768

```

The "--avtable" parameter is used to work with an alternative disk specification database (or to point to the standard database ("mkdisk.vtable") if it is in a location other than the current directory).

The "--blcount" (blocks count) and "--blsize" (blocks size) switches are used to create custom disk images.


Resizing disk images

The "mkdiskcmd" utility is able to resize disk images of one type to a disk image of another type.

This operation is needed, for example, to obtain more free space on a disk image that already contains data.

Notes:

- It is not possible to add more free space dynamically. The virtual machine must be stopped before performing this operation.
- Resizing a disk image requires the operating system running on the Charon virtual machine to be able to handle Dynamic Volume Expansion. Please refer to the documentation of your operating system version. If this is not supported, please create a new virtual disk then backup and restore the existing data.

 If a source disk image is larger than the target disk image, the extra data is lost. If the source disk image is smaller, it will be extended and padded with null bytes ('\0').

An example of the syntax follows:

```
$ mkdiskcmd --resize <source disk file name> <source disk parameters> [--shrink]
```

where:

- <source disk file name> - a file name of the disk image to be transferred
- <source disk parameters> - the name of the disk from the list provided by the "mkdiskcmd --list" command execution or the disk geometry specification (see below).
- --shrink or -s - used in the case where the target disk is transferred to a smaller disk.

Example:


```
$ mkdiskcmd --resize /etc/rz22.vdisk rz25
```

It is also possible to specify the disk parameters manually with "--blcount / -c" (blocks count) and "--blsize / -z" (blocks size) switches:

```
$ mkdiskcmd --resize <source disk file name> -blsize <number> -blcount <number>
```

Example:

```
$ mkdiskcmd -r /etc/custom.vdisk -z 512 -c 262134
```

 There is a certain delay between the moment when the utility reports that a disk image has been transferred and its actual availability to CHARON. This delay can reach to several minutes in case of very big disks transfers. It happens because the host operating systems needs some time for actual allocation of the enlarged file on HDD.

mtd

Table of Contents

- Description
- Tape container to physical tape transfer
- Tape container formats transfer

Description

The "mtd" utility is used to:

- Create a CHARON tape image from a physical tape
- Write a tape image to a physical tape.

Usage is the following:

```
$ mtd [options] <tape device name> <tape container name>
```

Parameters:

```
MTD - CHARON Magnetic Tape Dump & Restore utility, Version 2.7 (Build 20403)
Copyright (C) 2009-2020 STROMASYS SA. All rights reserved.
```

```
Usage: mtd [options] <tape-drive-name> <file-name> - dump tape content to file
mtd -//- <file-name> <tape-drive-name> - restore dump to tape
mtd -//- <file-name> <file-name> - convert formats
```

Usage for diagnostic purposes:

```
mtd -//- <tape-drive-name> - examine tape content
mtd -//- <file-name> - examine tape dump and check integrity
```

```
<tape-drive-name> - tape drive
<file-name> - name of tape container file (.mtd or .vtape)
```

```
Options:  -l <file-name> - log file name (.log)
          -n - do not rewind tape
          -r <number> - number of attempts to retry failing tape reads
          -i - ignore failing tape reads (implies -r 0)
          -p - disable progress reporting
          -v - enable verbose trace of data transfer (implies -p)
          -s - write tape image in SMA format
          -g - gather statistics and print upon completion
          -a - do not print logo
```

Example:

```
$ mtd -l tapel.txt -r 10 /dev/st5 /charon/tapes/tapel.vtape
```

Tape container to physical tape transfer

Use the following syntax to write the content of a tape container to a physical tape:

```
$ mtd <tape container name> <tape device name>
```

Example:

```
$ mtd /charon/tapes/tape1.vtape /dev/st5
```

Tape container formats transfer

Use the following syntax to transfer the CHARON-SMA tape container format to the CHARON-AXP/VAX/PDP one:

```
$ mtd <SMA tape container name> <AXP/VAX/PDP tape container name>
```

Example:

```
$ mtd /charon/tapes/sma_tape.vtape /charon/tapes/axp_tape.vtape
```

Use the following syntax to transfer the CHARON-AXP/VAX/PDP tape container format to the CHARON-SMA one:

```
$ mtd -s <AXP/VAX/PDP tape container name> <SMA tape container name>
```

Example:

```
$ mtd -s /charon/tapes/axp_tape.vtape /charon/tapes/sma_tape.vtape
```

hasp_srm_view

Table of Contents

- [Description](#)
- [Remote collection of status information](#)

Description

The "hasp_srm_view" utility displays the Charon HASP licenses content. Do not use it if you are using VE licensing.

Run the utility with one of the following parameters to see the license(-s) details:

- "-l" (or without parameters) - CHARON HASP default license details
- "-all" - all available CHARON HASP licenses details
- "-key <key number>" - specific CHARON HASP license (defined by its "key number") details

The "hasp_srm_view" utility is used to:

- Display the HASP licenses details. It is possible to view all available license or some specific one.
- Collecting HASP license status information
- Collecting HASP host fingerprint information

Run the utility without any options to display the HASP license details.

```
# hasp_srm_view -help

CHARON Sentinel HASP utility
Copyright (c) 2009-2019 STROMASYS. All rights reserved.

Options:
  -? or -h or -help    - to see help screen
  -l                    - to see CHARON license details (for default key)
  -all                  - to see CHARON license details (for all available keys)
  -key <key number>    - to see CHARON license details (for specific key)
  -c2v <C2V file>      - to collect the key status information (C2V file)
  -c2v <C2V file> -key <key number> - to collect C2V file for specific local key
  -fgp <C2V file>      - to collect the host fingerprint information (C2V file)
```

The specific type of Charon HASP license defines what switches may be used in each case.

Collecting the "c2v" file can be done only from the Charon host console (the console of the system on which the emulator runs).

Remote collection of status information

For remote collection of HASP status information it is recommended to use "ssh" as shown in the following examples:

```
# ssh root@CHARON_HOST /opt/charon/bin/hasp_srm_view -c2v /opt/charon/bin/my_hasp_key.c2v
# ssh root@CHARON_HOST /opt/charon/bin/hasp_srm_view -fgp /opt/charon/bin/my_host_fingerprint.c2v
```

To see the license text on the console:

```
# ssh root@localhost /opt/charon/bin/hasp_srm_view
```

To collect HASP license text to an output file on host server:

```
# ssh root@localhost /opt/charon/bin/hasp_srm_view > /opt/charon/bin/hasp_srm_view.txt
```

The "hasp_srm_view" utility always reports the ID and IP address of the host(s) where active HASP licenses are found.

hasp_update

Table of Contents

- Description
- Usage

Description

The "hasp_update" is a Sentinel standard utility for HASP license management included in the Charon kit.

To invoke the "hasp_update" utility login as "root" and use the following syntax:

```
# hasp_update <option> <filename>
```

where:

Parameter	Value	Description
<option>	u	Updates a Sentinel protection key / attaches a detached license
	i	Retrieves Sentinel protection key information
	d	Detaches a license from a Sentinel Software License (SL) key
	r	Rehost a license from a Sentinel Software License (SL) key
	h	Display help
<filename>		Path to the V2C/H2R file when used with the 'u' option
		Optional path to the C2V file when used with the 'i' option
		Uses "stdout" if file name is not specified

Example:

```
# hasp_update u license_update.v2c
```

Usage

We recommend to use this tool only for "Update a Sentinel protection key / attach a detached license" function ("u" option). For the rest use "hasp_srm_view" utility.


i Dongle (hardware) license updates always include two files, one with the format "<name>_fmt.v2c" and one with "<name>.c2v". Install the "<name>_fmt.v2c" (format) file first then the "<name>.v2c" file.

ncu

Table of Contents

- Note
- Description
- Dedication of a host physical interface to CHARON
- Release of a host physical interface back to host
- Creation of a virtual network
- Removal of a virtual network
- Adding VLAN interface
- Removing VLAN interface


Note

 The ncu utility depends on the NetworkManager service and cannot be used if the NetworkManager service is not installed and running. If you do not wish to enable the NetworkManager service, please see section "[Manual configuration of CHARON networking](#)" of the [Installation chapter](#) of this Guide for instructions on configuring the network manually.

Description

The "ncu" ("Network Control Utility") is used to dedicate a host interface to CHARON, to release it back to the host and to manage CHARON virtual interfaces (TAPs).

The utility allocates chosen network interfaces (both physical and virtual) and configures the offload parameters.

 **On Red Hat Enterprise Linux 6 & 7 and CentOS 7**
Package "vconfig" must be installed to enable NCU VLAN configuration functionality. Otherwise NCU will display the status "disabled, because vlan control package is not found"

Dedication of a host physical interface to CHARON

Login as root and enter "ncu". The following menu will appear:

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces Dedicated to State
-----
eth0      host      connected to host
eth1      host      connected from host
lo        host      unmanaged from host

=====
bridge name      bridge id      STP enabled    interfaces
===== VLAN =====
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit
```



```
:> 8
```

The utility lists available network interfaces (both physical and virtual) and indicates whether they are dedicated to the host or to CHARON and whether they are currently in use by host operating system.

"ncu" offers several options:

1. Dedicate interface to CHARON
2. Release interface to host
3. Create a bridge between a chosen physical network interface and the Linux virtual network and create a number of virtual network interfaces
4. Remove the Linux virtual network and all the created virtual network interfaces
5. Add VLAN interface
6. Remove VLAN interface
7. Print status - use it to display status of network interfaces and the menu shown above
8. Exit

In the example above we see 2 network interfaces - "eth0" and "eth1", both of them are dedicated to host, but host uses only the interface "eth0".

Let's dedicate the interface "eth1" to CHARON. Enter "1", type "eth1" and press Enter:

```
Specify the interface to dedicate to CHARON:eth1
Turning off offloading for eth1.. Please wait

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 7
```

Now the interface "eth1" is dedicated to CHARON:

```
Interfaces Dedicated to State
-----
eth0      host      connected to host
eth1      CHARON    disconnected from host
lo        host      unmanaged from host

=====
bridge name      bridge id      STP enabled
interfaces
===== VLAN =====
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit
```

Enter "8" to return to console prompt.

Now "eth1" can be used by CHARON.

Release of a host physical interface back to host

Login as root and enter "ncu". The following menu will appear:

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces Dedicated to State
-----
eth0      host      connected to host
eth1      CHARON    disconnected from host
lo        host      unmanaged from host

=====
bridge name      bridge id      STP enabled
interfaces
===== VLAN =====
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 2
```

Let's say that we want to return the interface "eth1" (currently dedicated to CHARON) back to host. To do that enter "2" then "eth1":

```
Specify the interface to release to HOST:eth1
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/3)

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 8
```

Enter "8" to quit the "ncu" utility.

The interface "eth1" is released back to host system now.

Creation of a virtual network

Login as root and enter "ncu":

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces Dedicated to State
-----
eth0      host      connected to host
eth1      host      connected to host
lo        host      unmanaged from host

=====
bridge name      bridge id      STP enabled      interfaces
===== VLAN =====
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 3
```

Enter "3" to create a bridge between the host physical network adapter and the LINUX virtual network interfaces (TAP) and specify the physical network interface ("eth1" in our example) and the number of virtual network interfaces to be created (2 in our example):

```
Specify the interface to be used for BRIDGE:eth1
How many tap should be created:2
Forming the bridge: ..1..2..3..4..5.. addif tap0 .. addif tap1 ..7..8 done!
Formed bridge br0_eth1 attached over eth1...

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 7
```

Now enter "7" to see the created virtual interfaces:

```

Interfaces      Dedicated to      State
-----
eth0            host              connected to host
eth1            bridge            connected to bridge
lo              host              unmanaged from host
tap0            CHARON            connected to host
tap1            bridge            connected to bridge
=====
bridge name      bridge id          STP enabled
interfaces
br0_eth1         8000.768e1ea091d9  no              eth1
                                                         tap0
                                                         tap1
=====
=====  VLAN  =====
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 8

```

In the example above we see 2 virtual network Interfaces "tap0" and "tap1" connected to the created bridge. The physical network interface "eth1" is used for the bridge to the virtual network interfaces.

The interfaces "tap0" and "tap1" are ready to be used in CHARON configurations - they do not need to be additionally dedicated to CHARON.

Enter "8" to quit "ncu" utility.

Removal of a virtual network

Login a root. Start "ncu" utility:

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces    Dedicated to    State
-----
eth0          host            connected to host
eth1          bridge         connected to bridge
lo            host            unmanaged from host
tap0          CHARON         connected to host
tap1          bridge         connected to bridge
=====
bridge name    bridge id      STP enabled
interfaces
br0_eth1      8000.768e1ea091d9  no            eth1
                                                    tap0
                                                    tap1
=====
=====  VLAN  =====
=====

select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 4
```

Enter "4" then enter the interface name that is a bridge to the Linux virtual network on this host ("eth1" in our example):

```
Specify the phys interface used for BRIDGE:eth1
Cleanup bridge br0_eth1 with ip over eth1...
Removing the bridge: ..1..2 delif eth1
delif tap0
delif tap1
..5..6..7..8 done!

select action:
1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 8
```

Enter "8" to quit "ncu" utility.

Adding VLAN interface

If VLAN is going to be used for CHARON (See: [More information on VLAN](#)) proceed with the following instruction:

Login a root. Start "ncu" utility:

```
# ncu

CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces    Dedicated to    State
-----
eth0          host            connected to host
eth1          host            connected to host
lo            host            unmanaged from host

=====
bridge name    bridge id      STP enabled    interfaces
===== VLAN =====
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 5
```

Enter "5" then enter:

1. The physical interface name to be used for creating VLAN
2. The ID of the VLAN device
3. IP address of the VLAN device. Skip this step if no IP is required
4. Network mask of the VLAN device. Enter for no network mask.

```
Specify the phys interface used for VLAN:eth1
Specify the id of VLAN device (<4095):111
Specify the ip address of VLAN device or empty string for no ip address: 192.168.1.100
Specify the netmask address of VLAN device or empty string for no netmask:
225.225.225.0

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 8
```

Enter "8" to quit "ncu" utility.

Removing VLAN interface

Login a root. Start "ncu" utility:

```
# ncu
CHARON Network Configuration Utility, STROMASYS (c) 2020 Version 1.7

Interfaces    Dedicated to    State
-----
eth0          host            connected to host
eth1          host            connected to host
lo            host            unmanaged from host

=====
bridge name      bridge id      STP enabled    interfaces
=====
eth1.111
=====

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 6
```

Enter "6" then enter the VLAN interface for remove:

```
Specify the VLAN interface, which be removed: eth1.
111
Removed VLAN -:eth1.111:-

select action:

1 - Dedicate to CHARON
2 - Release to host
3 - Create Bridge with TAPs
4 - Remove Bridge
5 - Add VLAN
6 - Remove VLAN
7 - Print status
8 - Exit

:> 8
```

Enter "8" to quit "ncu" utility.

CHARON Guest Utilities for OpenVMS

Table of Contents

- Description
- Installation
- Performance optimization
- Virtual tapes management
- Defining keys
- Displaying version

Description

The "CHARON Guest Utilities for OpenVMS" (CHARONCP) package contains several utilities for managing virtual tapes, changing the emulator speed and creating useful definitions for that operations.

This set of utilities is located in the "CHARONCP013.vdisk" disk file in the "/opt/charon/disks" folder.

Supported OpenVMS versions (depending on platform support): OpenVMS 6.1 and above.

 In case of OpenVMS upgrade, CHARONCP will have to be re-installed.

Installation

Specify this image in the CHARON configuration file, boot from the system disk and mount the disk with the following OpenVMS command:

```
$ MOUNT <device name> /OVERRIDE=IDENTIFICATION
```

Issue the following commands to install the package (example given for OpenVMS V8.4):

```
$ @SYS$UPDATE:VMSINSTAL
...
* Are you satisfied with the backup of your system disk [YES]? YES
* Where will the distribution volumes be mounted: <device name>:[CHARONCP013.KIT]

Enter the products to be processed from the first distribution volume set.
* Products: CHARONCP013

* Enter installation options you wish to use (none): <press enter>
...
Do you want to install this product [NO]? YES
...
* Where should the CHARONCP root directory be located ? [SYS$SYSDEVICE:[CHARONCP]]: <press enter>
...
* Do you want to purge files replaced by this installation [YES]? <press enter>
```


Select all the components included to the package:

```

                                Component Selection

Select the CHARONCP components you wish to install from the menu below.
An asterisk appears next to the packages that have already been
selected. You can remove a package from the list by selecting it
again. You may enter more than one selection by separating your
choices with commas.

1. [*] CHARONCP Guest Utility (REQUIRED)
2. [*] Compatability Utilities
3. [*] Install DCL Commands & Help

4. Exit

* Your choice [4]: 1,2,3

...

* Your choice [4]: 4

...

* Is this correct [YES]: <press enter>

...

* Products: <press enter>

                                VMSINSTAL procedure done at hh:mm

$

```

Proceed with installation using all the default options.

Once the installation is completed, add the following line to the "SYS\$STARTUP:SYSTARTUP_VMS.COM" ("SYS\$STARTUP:SYSTARTUP_V5.COM" for VMS 5.5) file for the package to be loaded automatically at system startup:

```
$ @SYS$STARTUP:CHARONCP_STARTUP
$ CHARONCP SET IDLE /ENABLE
```

After that the package will be loaded automatically on startup.

Performance optimization

CHARON takes 100% of host CPU even in case of idle state of guest OpenVMS operating system. To get rid of such resources consumption there is a specific option provided by CHARON Guest Utilities - "idle" mode.

To load the OpenVMS idle loop detection software, use:

```
$ CHARONCP SET IDLE /ENABLE
```

This allows CHARON to detect when the emulated CPU(s) are idle and use the host power saving instructions to reduce power usage.

To unload the OpenVMS idle loop detection software, use:

```
$ CHARONCP SET IDLE /DISABLE
```

Virtual tapes management

Specify mapping to tape container in the following way in the CHARON configuration file:

```
set <adapter name> container[<unit name>] = ".vtape" removable[<unit name>] = true
```



- It is mandatory to set the "removable" parameter to "true"
- The container name must be ".vtape", no name and path must be specified, only extension.

Example:

```
set PKA container[600] = ".vtape" removable[600] = true
```

Once it is done using the following commands it is possible to manage virtual tapes attached to CHARON.

To create the specified host-file (if it does not already exist) and attach it to the specified virtual tape device, use:

```
$ CHARONCP SET MAGTAPE <device> /LOAD="filename.vtape"
```



The container name specified in the /LOAD parameter must not be more than 255 characters

Example:

```
$ CHARONCP SET MAGTAPE MKA600: /LOAD="/charon/tapes/backup_01.vtape"
```

To detach any file currently attached to the specified virtual tape device, use:

```
$ CHARONCP SET MAGTAPE <device> /UNLOAD
```

Example:

```
$ CHARONCP SET MAGTAPE MKA600: /UNLOAD
```

Possible errors	Description
BADFILENAME	The filename specified as a value to the qualifier /LOAD was either too long or does not have a file extension of ".vtape".
DEVNOTDISM	Attempting to execute a SET MAGTAPE/LOAD when a file is already attached. Perform a SET MAGTAPE/UNLOAD first. If a SET MAGTAPE/LOAD command has not previously been executed, then the CHARON configuration container specification for the tape device may contain a full path. Doing this will create and attach an initial tape container file. To avoid this, remove the file name from the specification (leaving only a file extension of ".vtape" and optional directory).



If some tape container has been already specified in the CHARON configuration file use the command "CHARONCP SET MAGTAPE <device> /UNLOAD" to unload it first.

Defining keys

It is possible to define certain keys on the terminal keyboard for fast access to the CHARONCP functionality while you are in CHARONCP.

To define an equivalence string and a set of attributes with a key on the terminal keyboard, use:

```
$ CHARONCP
CHARONCP> DEFINE /KEY <key-name> <equivalence-string>
```

You can have a set of keys defined automatically for use with the CHARONCP utility by placing DEFINE/KEY commands in the file `SYS$LOGIN:CHARONCP_KEYDEFS.INI`.

Example:


```
$ CHARONCP
CHARONCP> DEFINE /KEY F1 "SET MAGTAPE MKA600: /UNLOAD"
```

To display key definitions created with the DEFINE/KEY command. Refer to the DCL help entry for SHOW KEY for further information, use:

```
$ CHARONCP
CHARONCP> SHOW KEY <key-name>
```

Example:

```
$ CHARONCP
CHARONCP> SHOW KEY F1
DEFAULT key state definitions:
F1 = "set magtape mka600: /unload"
CHARONCP>
```

 For more information refer to the OpenVMS DCL Dictionary (DEFINE/KEY section).

Displaying version

To display the CHARONCP package version number and architecture, use:

```
$ CHARONCP SHOW VERSION
```

This can be useful for customers reporting issues with the CHARONCP software.

Example:

```
$ CHARONCP SHOW VERSION
CHARONCP version id is: V1.3
```

CHARON-AXP for Linux configuration Details

Introduction

This chapter describes, in detail, all of the configuration parameters of the devices emulated by CHARON-AXP for Linux, with corresponding examples and parameters.

Emulated devices are loaded with the "load" command and parameters are made active with the "set" command. Some parameters can be specified directly in the "load" command.

CHARON-AXP configuration

- General Settings
- Core Devices
- Console
- Remote Management Console (RMC)
- Placement of peripheral devices on PCI bus
- Disks and tapes
 - KZPBA PCI SCSI adapter
 - KGPSA-CA PCI Fibre Channel adapter
 - Acer Labs 1543C IDE/ATAPI CD-ROM adapter
 - PCI I/O Bypass controller
 - Finding the target "/dev/sg" device
- Networking
- PBXDA PCI serial lines adapter
- AlphaStation Sound Card (AD1848) emulation
- PBXGA graphics card
- Sample configuration files

General Settings

Table of Contents

- Session
 - hw_model
 - configuration_name
 - log
 - log_method
 - log_file_size
 - log_rotation_period
 - log_flush_period
 - license_key_id
 - license_id
 - license_key_lookup_retry
 - affinity
 - n_of_cpus
 - n_of_io_cpus
- File inclusion

Session


General settings that control the execution of CHARON-AXP belong to an object called the "session". It is a preloaded object; therefore, only "set" commands apply.

Example:


```
set session <parameter>=<value>
```

The following sections describe all available "session" parameters, their meaning and examples of their usage:





hw_model

Parameter	hw_model
Type	Text string
Value	<p>Virtual Alpha system hardware model to be emulated.</p> <p>Use a named template for each model as a starting point for a custom configuration. This would ensure that this parameter is set correctly.</p> <p>Example:</p> <pre>set session hw_model="AlphaServer_ES40"</pre> <p>Available models are:</p> <ul style="list-style-type: none"> • AlphaServer_AS400 • AlphaServer_AS800 • AlphaServer_AS1000 • AlphaServer_AS1000A • AlphaServer_AS1200 • AlphaServer_AS2000 • AlphaServer_AS2100 • AlphaServer_AS4000 • AlphaServer_AS4100 • AlphaServer_DS10 • AlphaServer_DS10L • AlphaServer_DS15 • AlphaServer_DS20 • AlphaServer_DS25 • AlphaServer_ES40 • AlphaServer_ES45 • AlphaServer_GS80 • AlphaServer_GS160 • AlphaServer_GS320 <p> Refer to this section to find how to set a particular Alpha model supported by CHARON-AXP.</p>


configuration_name

Parameter	configuration_name
Type	Text string
Value	<p>Name of the CHARON-AXP instance (it must be unique).</p> <p>In most cases this is set to the local machine name (TCPIP node identifier for example).</p> <pre>set session configuration_name="MSCDV1"</pre> <p>The value of this parameter is used as a prefix to the event log file name. (see below).</p> <p>From the example above, the CHARON-AXP log file will have the following name:</p> <pre>MSCDV1-YYYY-MM-DD-hh-mm-ss-xxxxxxxxx.log</pre> <p>xxxxxxxx is an increasing decimal number starting from 00000000 to separate log files with the same time of creation (in case the log is being written faster than one log file per second).</p> <p> It is highly recommended to use the "configuration_name" parameter if more than one CHARON instance runs on the same server.</p>


log

Parameter	log
Type	Text string
Value	<p>The log file or directory name is where the log file for this CHARON-AXP session is stored.</p> <p style="text-align: center;">Log specified as a file name</p> <p>It is possible to overwrite the existing log file or to extend it using the "General Settings#log_method" parameter.</p> <p> The "log_method" parameter is effective only when a single log file is specified, not a directory.</p> <p>Example:</p> <pre>set session log="/charon/es40prod.log"</pre> <p style="text-align: center;">Log specified as a directory</p> <p>CHARON-AXP automatically creates individual log files for each CHARON-AXP execution session. If the log parameter is omitted, CHARON-AXP creates a log file for each CHARON-AXP execution session in the directory where the emulator was started. In these two cases, the log rotation mode is enabled, meaning a new log file is created each time the virtual machine is started and when the log file size exceeds the one specified (see General Settings#log_file_size) and/or when the log file is older than a specified number of days (see General Settings#log_rotation_period).</p> <p> A symbolic link located in the same directory will be created, pointing to the active log file. Its name is based on the hw_model parameter or the configuration_name parameter if specified.</p> <p>If the "configuration_name" parameter of the session is specified, the log file name is composed as follows:</p> <pre><configuration_name>-YYYY-MM-DD-hh-mm-ss-xxxxxxxxx.log</pre> <p>If the "configuration_name" parameter is omitted, the log file name will have the following format:</p> <pre><hw_model>-YYYY-MM-DD-hh-mm-ss-xxxxxxxxx.log</pre> <p>where "xxxxxxxxx" is an increasing decimal integer, starting from 00000000 to separate log files with the same time of creation (in case the log is being created faster than one log file per second).</p> <p> Only existing directory can be specified. If the directory specified does not exist, this will be considered as a flat file.</p> <p>Example:</p> <pre>set session configuration_name="es40prod" set session log="/charon/logs"</pre> <p>The execution of the virtual machine will create a log file, named /charon/logs/es40prod-2016-10-13-10-00-00-000000000.log (for example) and a symbolic link named /charon/logs/es40prod.log pointing to this file. The link will be updated when the log rotation will occur.</p> <p> Starting with version 4.11 build 204-11, log file numbering is implemented when log rotating is used: when starting the Charon emulator the first log file name will be like <configuration_name>-YYYY-MM-DD-hh-mm-ss-xxxxxxxxx.log where xxxxxxxxxxx will be 000000000, this number will increase with each log rotation to make it easier to find a log file set (1st log file at start + all its rotated files)</p>

log_method

Parameter	log_method
Type	Text string
Value	<ul style="list-style-type: none"> • "append" (default) • "overwrite" <p>Determines if the previous log information is maintained or overwritten.</p> <p> This parameter must be specified only in addition to "log" parameter and on the same line.</p> <p>This parameter is applicable only if the CHARON-AXP log is stored to a file that is specified explicitly with the "log" parameter.</p> <p>Example:</p> <pre>set session log="log.txt" log_method="overwrite"</pre>

log_file_size

Parameter	log_file_size
Type	Text string
Value	<p>If log rotation is enabled, the log_file_size parameter determines the log file size threshold at which a new log is automatically created. Rotating log file size is a multiple of 64K</p> <ul style="list-style-type: none"> • "unlimited" or "0" (default) - the feature is disabled • "default" - default size is used (4Mb) • <size>[KMG] - size of the current log file in bytes with additional multipliers: <ul style="list-style-type: none"> • K - Kilobyte - multiply by 1024 • M - Megabyte - multiply by 1024*1024 • G - Gigabyte - multiply by 1024*1024*1024 <p>Example 1:</p> <pre>set session log_file_size="default"</pre> <p>Example 2:</p> <pre>set session log_file_size=10M</pre> <p> Minimum LOG File size is 64K, maximum is 1G. Setting size less than 64K effectively makes the LOG File unlimited.</p>

log_rotation_period

Parameter	log_rotation_period
Type	Text string
Value	<ul style="list-style-type: none"> • "default" - default value, 7 days. This value is used even if the "log_rotation_period" is not specified. • "daily" or "1" • "weekly" or "7" • "never" • <N> - in N days where N is greater than 0 <p>If the rotation log mode is enabled this parameter controls switching to next log file based on period of time passed. If enabled the switching to next log file occurs at midnight.</p> <p>Example 1:</p> <pre>set session log_rotation_period="weekly"</pre> <p>Example 2:</p> <pre>set session log_rotation_period=14</pre>

log_flush_period

Parameter	log_flush_period
Type	Numeric
Value	<ul style="list-style-type: none"> • <period-in-seconds> <p>Defines a period of flushing log to disk. In most cases, the default is sufficient</p> <p>Default period is 60 seconds (it means that every minute log file is flushed to disk)</p> <p>Example:</p> <pre>set session log_flush_period=30</pre>

license_key_id

Parameter	license_key_id
Type	Text string
Value	<p>A set of Sentinel Key IDs that specifies the license keys to be used by CHARON. It is also possible to use the keyword "any" to force CHARON to look for a suitable license in all available keys.</p> <p>Example for a HASP license:</p> <pre>set session license_key_id = "1877752571,354850588,any"</pre> <p>Based on the presence of this parameter in the configuration file, CHARON behaves as follows:</p> <ol style="list-style-type: none"> No keys are specified (the parameter is absent) CHARON performs an unqualified search for any suitable key in unspecified order. If no key is found, CHARON exits. One or many keys are specified CHARON performs a qualified search for a regular license key in the specified order. If it is not found, CHARON exits (if the keyword "any" is not set). <p>If the keyword "any" is specified then if no valid license has been found in the keys with specified ID's all other available keys are examined for valid license as well.</p> <p>Please note: the order in which keys are specified is very important. If a valid license was found in the key which ID was not the first one specified in configuration file, then available keys are periodically rescanned and if the key with the ID earlier in the list than the current one is found CHARON tries to find a valid license there and in case of success switches to that key.</p> <p>Example for a VE license:</p> <pre>set session license_key_id = "VE://10.1.1.1:8083/1111-2222-3333-4444"</pre> <p>To configure a backup license server, add the backup license server information to the same line after the primary license server information:</p> <pre>set session license_key_id = "VE://<primary-licserv-IP-Address>[:<port>]/<passphrase>/, VE://<backup-licserv-IP-Address>[:<port>]/<passphrase>/"</pre>


license_id

Parameter	license_id
Type	Text string
Value	<p>A set of license identifiers that specifies the licenses to be used by CHARON. This parameter is applicable only to licenses on which Stromasys placed restrictions on what products can be combined on a single license key. Please contact your Stromasys representative or VAR for more information.</p> <p>Example:</p> <pre>set session license_id = "2718281828,314159265"</pre> <p>If this parameter is set, Charon considers for validation only the available licenses with license ID parameter set and equal to one of the license ID's specified in the configuration. This prioritized list corresponds to the "Product License Number" line in the Product section of the license.</p>

license_key_lookup_retry

Parameter	license_key_lookup_retry
Type	Text String
Value	<p>In case the CHARON-AXP license connection is not present when the guest starts up, this parameter specifies how many times CHARON-AXP will try to establish the connection and, optionally, a period of time between retries.</p> <p>Syntax:</p> <pre>set session license_key_lookup_retry = "N [, T]"</pre> <p>Options:</p> <ul style="list-style-type: none"> • N - Number of retries to look for license keys. • T - Time between retries in seconds. If not specified 60 seconds are used <p>Example 1:</p> <pre>set session license_key_lookup_retry = 1</pre> <p>If license key is not found during initial scan, do only one more attempt after 60 seconds.</p> <p>Example 2:</p> <pre>set session license_key_lookup_retry = "1,30"</pre> <p>Same as above but retry in 30 seconds.</p> <p>Example 3:</p> <pre>set session license_key_lookup_retry = "3,10"</pre> <p>If license key is not found during initial scan, do 3 more attempts waiting 10 seconds between them.</p> <p>Example 4:</p> <pre>set session license_key_lookup_retry = "5"</pre> <p>If license key is not found during the initial scan, do 5 more attempts waiting 60 seconds between them.</p>

affinity

Parameter	affinity
Type	Text string
Value	<p>Overrides any initial process affinity mask provided by the host operating system. Once specified it binds the running instance of the emulator to particular host CPUs.</p> <p>Used for soft partitioning of the host CPU resources and/or for isolating host CPUs for other applications.</p> <p>By default the CHARON-AXP emulator instance allocates as many host CPUs as possible. The "affinity" parameter overrides that and allows explicit specification on which host CPU the instance must run on.</p> <p> The "affinity" parameter defines the total number of host CPUs to be used both for emulated Alpha CPUs and for CHARON-AXP application itself (including the CPUs to be used for I/O - controlled by "n_of_io_cpus" parameter described below).</p> <p>Host CPUs are enumerated as a comma separated list of host system assigned CPU numbers:</p> <pre>set session affinity="0, 2, 4, 6"</pre>

n_of_cpus

Parameter	n_of_cpus
Type	Numeric
Value	<p>Limits the number of emulated CPUs. This parameter can be used to force testing of smaller configurations as well by limiting the available "hardware".</p> <p>Example:</p> <pre>set session n_of_cpus=3</pre> <p>The maximum number of CPUs enabled by CHARON-AXP is specified by the license key, but cannot exceed the original hardware restrictions. See table below.</p>

Alpha Model	MAX Number of emulated CPUs
AlphaServer_AS400	1
AlphaServer_AS800	1
AlphaServer_AS1000	1
AlphaServer_AS1000A	1
AlphaServer_AS1200	2
AlphaServer_AS2000	2
AlphaServer_AS2100	4
AlphaServer_AS4000	2
AlphaServer_AS4100	4
AlphaServer_DS10	1
AlphaServer_DS10L	1
AlphaServer_DS15	1
AlphaServer_DS20	2
AlphaServer_DS25	2
AlphaServer_ES40	4
AlphaServer_ES45	4
AlphaServer_GS80	8
AlphaServer_GS160	16
AlphaServer_GS320	32

n_of_io_cpus

Parameter	n_of_io_cpus
Type	Numeric
Value	<p>This parameter specifies how many host CPUs CHARON-AXP will use for I/O handling. Use of the "affinity" parameter may limit the number of CPUs available.</p> <p>By default the CHARON-AXP instance reserves one third of all available host CPUs for I/O processing (round down, at least one). The "n_of_io_cpus" parameter overrides that by specifying the number of CHARON I/O CPUs explicitly. Note: The total of N_of_cpus and N_of_io_cpus can not exceed the available host CPUs.</p> <p>Example:</p> <pre>set session n_of_io_cpus=2</pre>

File inclusion

It is possible to include a configuration file into an existing one using the "include" command. The file extension is usually `.icfg`.

This method is most helpful when large numbers of disks are configured across multiple Charon instances.

Format:

```
include "file.icfg"
```

Example:

```
include "/charon/commonpart.icfg"
```

Core Devices

Table of Contents

- CPU
 - enabled
 - cpu_architecture
 - cache_size
 - num_translators
 - host_options
 - Enabling the old style performance optimization
 - Enlarging ACE cache size
 - Setting specific ACE host options
- RAM
 - size
- TOY
 - container
 - sync_to_host
- ROM
 - container
 - system_name
 - system_serial_number
 - dsrdb
 - version
- Virtual Alpha interval timer
- Setting of a particular Alpha model
 - AlphaStation 200 - 400
 - AlphaServer 600 - 800
 - AlphaServer 1000
 - AlphaServer 1000A
 - AlphaServer 1200 and AlphaStation 1200
 - AlphaServer 2000
 - AlphaServer 2100
 - AlphaServer 4000
 - AlphaServer 4100
 - AlphaServer/AlphaStation DS10 and AlphaServer DS10L
 - AlphaServer DS15 and AlphaStation DS15
 - AlphaServer DS20 and AlphaStation DS20
 - AlphaServer DS25 and AlphaStation DS25
 - AlphaServer ES40 and AlphaStation ES40
 - AlphaServer ES45
 - AlphaServer GS80
 - AlphaServer GS160
 - AlphaServer GS320
- Auto Boot
 - auto_action restart
- Setting System Marketing Model (SMM)

CPU

The CHARON-AXP CPU can be calibrated with "set ace" directive and the following parameters:

enabled

Parameter	enabled
Type	Boolean
Value	<p>A CHARON-AXP emulated CPU is configured with the "enabled" command enabling the high performance Advanced CPU Emulation mode ("ACE"). The ACE option optimizes the Alpha instruction interpretation and significantly improves performance. It also requires approximately twice the amount of host memory allocated by CHARON instance itself to store the optimized code (Note that 2Gb of host memory + the amount of Alpha memory emulated per each CHARON instance is required).</p> <p>ACE optimization is performed dynamically during execution. It does not need to write optimized code back to disk, ACE provides its full capability instantly. The optimization does not compromise the Alpha instruction decoding; CHARON-AXP remains fully Alpha hardware compatible and completely transparent to the Alpha operating systems and applications.</p> <p>This configuration setting enables the ACE mode if the CHARON-AXP license permits it. If this configuration setting is omitted from the CHARON-AXP configuration file and the license permits it, "true" is the default, otherwise "false" is the default.</p> <p>Example:</p> <pre>set ace enabled = false</pre> <p>"set ace enabled=true" is ignored when the license does not permit ACE operation.</p>

cpu_architecture

Parameter	cpu_architecture
Type	Text String
Value	<p>Specifies the architecture of the virtual Alpha CPU. Can be one of the following: EV4, EV45, EV5, EV56, EV6, EV67, EV68</p> <p>Example:</p> <pre>set ace cpu_architecture = EV6</pre> <p>Refer to "Core Devices#Setting of a particular Alpha model" to find an appropriate value of the Alpha architecture per each Alpha model supported by CHARON-AXP.</p>

cache_size

Parameter	cache_size
Type	Value
Value	<p>This parameter may affect Charon performance. Do not change this parameter until advised by Stromasys.</p> <p>"cache_size" defines the amount of memory in megabytes allocated to the ACE cache.</p> <p>Default value is 1GB (1024 MB).</p> <p>Example:</p> <pre>set ace cache_size = 2048</pre> <p>This parameter may be changed for performance optimization in case of some guest OS specific tasks (see Enlarging ACE cache size section).</p>

num_translators

Parameter	num_translators
Type	Value
Value	<p>This parameter may affect Charon performance. Do not change this parameter until advised by Stromasys</p> <p>"num_translators" defines the number of ACE translators.</p> <p>Default value in most situations is number of I/O CPUs dedicated to CHARON (defined by "n_of_io_cpus" parameter) minus 1, but in some specific situations it may be set to some other value for better performance.</p> <p>Example:</p> <pre>set ace num_translators = 4</pre>

host_options

Parameter	host_options
Type	Text String
Value	<p>This parameter may affect Charon performance. Do not change this parameter until advised by Stromasys.</p> <p>"host_options" defines options of ACE (DIT) translator and code-generator. Those options affect Charon performance</p> <p>Default settings are set to optimize performance for most guest OS (VMS and Tru64) usage profiles. However there are some profiles (for example OpenVMS compilation tasks) where the default settings do not provide optimal performance.</p> <p>The following switches are available to user:</p> <ol style="list-style-type: none"> <code>--fixed-variant=[X]</code> <p>The value X can be one of three options: [-1, 0, 1]. This value defines the desired translation variant. Set -1 for dynamic (default) or 0 or 1 for the fixed number implemented by the translator.</p> <ol style="list-style-type: none"> <code>--x64-optimize or --x64-nooptimize</code> <p>This switch enables translation optimizations (the default is to optimize).</p> <p>Default parameters have been changed in Charon version 4.9 compared to previous versions (4.8 and below). If Charon system demonstrates lower performance after upgrade to version 4.9, please test the system with host_options switched to default 4.8 settings:</p> <pre>set ace host_options = "--fixed-variant=0 --x64-nooptimize"</pre> <p>Note that in this case the parameter "Core Devices#num_translators" must be set to the number of I/O CPUs dedicated to CHARON (see the "n_of_io_cpus" parameter). Also note that changing parameter "Core Devices#cache_size" can be an alternative solution too (see the section below).</p>

Enabling the old style performance optimization

Despite the fact that CHARON is already optimized for wide range of the guest OS tasks, there may be some situations (for example OpenVMS compilation tasks) when performance degradation may reach about 50% of the version 4.8.

In this case the following solutions can be applied:

Enlarging ACE cache size

Try this solution first. It is recommended to enlarge the ACE cache size at least in 2 times as it is shown in the following example:

```
set ace cache_size = 2048
```

If this solution does not work for the specific guest OS tasks, apply also the next solution.

Setting specific ACE host options

Set the following set of options in the configuration file:

```
set session n_of_io_cpus = <N>
set ace num_translators = <N>
set ace host_options = "--fixed-variant=0 --x64-nooptimize"
```

where <N> is number of the I/O CPUs dedicated to CHARON. If this value is not set (the default value is used) it is recommended to specify it explicitly.

RAM

The CHARON-AXP memory subsystem is permanently loaded and has the logical name "ram".

size

Parameter	size
Type	Numeric
Value	Size of the emulated memory in MB.

Example:

```
set ram size = 2048
```

The amount of memory is capped at a maximum, this is defined in the CHARON license key. If the host system cannot allocate enough memory to map the requested emulated memory, CHARON-AXP generates an error message in the log file and reduces its effective memory size.

The following table lists the values of emulated RAM for various hardware models of virtual Alpha systems:

Hardware Model	RAM size (in MB)			
	Min	Max	Default	Increment
AlphaServer 400	64	1024	512	64
AlphaServer 800	256	8192	512	256
AlphaServer 1000	256	1024	512	256
AlphaServer 1000A	256	1024	512	256
AlphaServer 1200	256	32768	512	256
AlphaServer 2000	64	2048	512	64
AlphaServer 2100	64	2048	512	64
AlphaServer 4000	64	32768	512	64
AlphaServer 4100	64	32768	512	64
AlphaServer DS10, DS10L	64	32768	512	64
AlphaServer DS15	64	32768	512	64
AlphaServer DS20	64	32768	512	64
AlphaServer DS25	64	32768	512	64
AlphaServer ES40	64	32768	512	64
AlphaServer ES45	64	32768	512	64
AlphaServer GS80	256	65536	512	256
AlphaServer GS160	512	131072	512	512
AlphaServer GS320	1024	262144	1024	1024

TOY

CHARON-AXP maintains its time and date using the "toy" (time-of-year) component. In order to preserve the time and date while a virtual system is not running, the TOY component uses a binary file on the host system to store the date and time relevant data. The name of the file is specified by the "container" option of the "toy" component.

container

Parameter	container
Type	Text string
Value	<p>Specifies a name for the file in which CHARON-AXP preserves the time and date during its "offline" period. This file also keeps some console parameters (such as the default boot device).</p> <p>By default it is left unspecified.</p> <p>Example:</p> <pre>set toy container="/Charon/my_virtual_system.dat"</pre> <p>It is recommended to specify the full path to the TOY file.</p>

sync_to_host

Parameter	sync_to_host
Type	Text string
Value	<p>Specifies whether and how the guest OS time is synchronized with the CHARON host time.</p> <p>Syntax:</p> <pre>set TOY sync_to_host = "{as_vms as_tru64 as_is}"</pre> <p>Value description:</p> <ul style="list-style-type: none"> ■ <code>as_vms</code> : If the guest OS is OpenVMS/AXP and its date and time must be set to the host's date and time each time it boots. ■ <code>as_tru64</code> : If the guest OS is Tru64 UNIX and its date and time must be set to the host's date and time each time it boots. ■ <code>as_is</code> : If the TOY date and time must be set to the host's UTC date and time. <p>Example:</p> <pre>set TOY sync_to_host = "as_vms"</pre> <p>To synchronize the guest OS with TOY, use the following commands (from "SYSTEM"/"root" account):</p> <ul style="list-style-type: none"> ■ on OpenVMS/AXP: <code>\$ set time</code> ■ on Tru64 UNIX: <code># date -u `consvar -g date cut -f 3 -d ' '`</code> <p>The default value is "not specified" - it means that by default CHARON does not synchronize its guest OS time with the CHARON host time but collects date and time from the file specified with "container" parameter.</p> <p>If "sync_to_host" parameter is specified there is no need to specify "container" parameter in addition.</p>

The CHARON-AXP time zone may be different from that of the host system. Correct CHARON time relies on the correctness of the host system time to calculate the duration of any CHARON "offline" periods. (i.e. while the virtual system is not running). Every time CHARON comes on line it calculates a Delta time (the system time is used if there is no TOY file). Therefore, if the host system time is changed while CHARON is not running, the CHARON time may be incorrect when CHARON is restarted and the CHARON time must be set manually.

ROM

The System Flash ROM file conserves specific parameters between reboots.

container

Parameter	container
Type	Text string
Value	<p>Specifies the name of a file in which CHARON-AXP stores an intermediate state of its Flash ROM. This state includes, for example, most of the console parameters.</p> <p>By default it is left unspecified.</p> <p>it is recommended to specify the full path to this file</p> <p>Example:</p> <pre>set rom container="/Charon/my_virtual_system.rom"</pre>

system_name

Parameter	system_name
Type	Text string
Value	<p>Allows changing the system name</p> <p>Example:</p> <pre>set rom system_name="Alpha Server 1000 4/200"</pre> <p>Refer to Core Devices#Setting of a particular Alpha model to find an appropriate value of the Alpha system name per each Alpha model supported by CHARON-AXP</p>

system_serial_number

Parameter	system_serial_number
Type	Text string
Value	<p>Allows changing the system serial number</p> <p>Example:</p> <pre>set rom system_serial_number = NY12345678</pre> <p>Any sequence of characters can be used as a serial number. Sequences longer than 16 characters are truncated.</p> <p>Serial Numbers should be according to DEC standard: 10 characters. First two characters are capital letters, remaining 8 characters are decimal digits.</p> <p>By default it is set to SN01234567</p>

dsrdb

Parameter	dsrdb[n]
Type	Numeric
Value	<p>DSRDB - Dynamic System Recognition Data Block. These parameters allow changing the emulated hardware model type.</p> <p>dsrdb[0] stands for SMM - System Marketing Model.</p> <p>Example:</p> <pre>set rom dsrdb[0]=1090</pre> <p>Core Devices#Setting of a particular Alpha model describes connection between "dsrdb" parameter and the rest of the parameters defining an exact Alpha model - including SMM.</p>

version

Parameter	version
Type	Text string
Value	<p>Sets Console and PAL code versions in the following way:</p> <ul style="list-style-type: none"> ■ SRM Console version to X.Y-Z: set rom version[0] = x.y-z ■ OpenVMS PAL code version to X.Y-Z: set rom version[1] = x.y-z ■ Tru64 UNIX PAL code version to X.Y-Z: set rom version[2] = x.y-z <p>Example:</p> <pre>set rom version[0] = 7.3-1 version[1] = 1.98-104 version[2] = 1.92-105</pre>

Virtual Alpha interval timer

The CHARON-AXP provides interval timer interrupts to virtual Alpha CPU(s) at frequency 100Hz (100 interrupts a second).

This is default behavior which may be changed through “clock_period” configuration parameter of virtual ISA or EISA bus, depending on emulated hardware model of virtual Alpha system.

Value of the parameter is interval timer period in microseconds. By default it is set to 10000. By changing it to 1000 frequency of virtual interval timer interrupts may be increased to 1000Hz (1000 interrupts per second).

Parameter	clock_period
Type	Numeric
Value	<p>Specifies period of interval timer, in microseconds. Only two values are supported:</p> <ul style="list-style-type: none"> • 10000 (which corresponds to 100Hz interval timer) • 1000 (which corresponds to 1000Hz interval timer) <p>By default it is set to 10000.</p>

Example for AlphaServer 400, DS, ES, GS:

```
set ISA clock_period=1000
```

Example for AlphaServer 800, 1000, 1000A, 1200, 2000, 2100, 4000, 4100:

```
set EISA clock_period=1000
```

Higher interval timer frequency creates higher load for virtual Alpha CPU which may cause degradation of overall virtual system performance.

Setting of a particular Alpha model

It is important to have the "system_name", "hw_model", "cpu_architecture" and "dsrdb[n]" (DSRDB - Dynamic System Recognition Data Block) parameters in sync. (see above for details) to configure CHARON-AXP for emulation of a particular Alpha model.

The following tables illustrate how to synchronize those values:

AlphaStation 200 - 400

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_400	AlphaStation 200 4/100	EV4	1156
AlphaServer_400	AlphaStation 200 4/133	EV4	1088
AlphaServer_400	AlphaStation 205 4/133	EV4	1250
AlphaServer_400	AlphaStation 255 4/133	EV4	1257
AlphaServer_400	AlphaStation 200 4/166	EV4	1087
AlphaServer_400	AlphaStation 205 4/166	EV4	1251
AlphaServer_400	AlphaStation 255 4/166	EV4	1258
AlphaServer_400	AlphaStation 400 4/166	EV4	1086
AlphaServer_400	AlphaStation 205 4/200	EV4	1252
AlphaServer_400	AlphaStation 255 4/200	EV4	1259
AlphaServer_400	AlphaStation 200 4/233	EV45	1151
AlphaServer_400	AlphaStation 205 4/233	EV45	1253
AlphaServer_400	AlphaStation 255 4/233	EV45	1260
AlphaServer_400	AlphaStation 400 4/233	EV45	1152
AlphaServer_400	AlphaStation 205 4/266	EV45	1254
AlphaServer_400	AlphaStation 255 4/266	EV45	1261
AlphaServer_400	AlphaServer 300 4/266	EV45	1593
AlphaServer_400	AlphaStation 400 4/266	EV45	1153
AlphaServer_400	AlphaStation 400 4/266	EV45	1154
AlphaServer_400	AlphaStation 200 4/300	EV45	1157
AlphaServer_400	AlphaStation 205 4/300	EV45	1255
AlphaServer_400	AlphaStation 255 4/300	EV45	1262
AlphaServer_400	AlphaStation 400 4/300	EV45	1160
AlphaServer_400	AlphaStation 205 4/333	EV45	1256
AlphaServer_400	AlphaStation 255 4/333	EV45	1263

AlphaServer 600 - 800

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_800	AlphaServer 600 5/333	EV56	1310
AlphaServer_800	AlphaServer 800 5/333	EV56	1310
AlphaServer_800	AlphaServer 800 5/400	EV56	1584

AlphaServer_800	AlphaStation 600A 5/500	EV56	1590
AlphaServer_800	AlphaServer 800 5/500	EV56	1585

AlphaServer 1000

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_1000	AlphaServer 1000 4/200	EV4	1090
AlphaServer_1000	AlphaServer 1000 4/233	EV45	1091
AlphaServer_1000	AlphaServer 1000 4/266	EV45	1264

AlphaServer 1000A

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_1000A	AlphaServer 1000A 4/266	EV45	1265

AlphaServer 1200 and AlphaStation 1200

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_1200	AlphaServer 1200 5/300	EV5	1722
AlphaServer_1200	AlphaServer 1200 5/300	EV5	1724
AlphaServer_1200	AlphaServer 1200 5/400	EV56	1726
AlphaServer_1200	AlphaServer 1200 5/400	EV56	1728
AlphaServer_1200	AlphaStation 1200 5/400	EV56	1758
AlphaServer_1200	AlphaStation 1200 5/400	EV56	1760
AlphaServer_1200	AlphaServer 1200 5/466	EV56	1730
AlphaServer_1200	AlphaServer 1200 5/466	EV56	1732
AlphaServer_1200	AlphaStation 1200 5/466	EV56	1762
AlphaServer_1200	AlphaStation 1200 5/466	EV56	1764
AlphaServer_1200	AlphaServer 1200 5/533	EV56	1734
AlphaServer_1200	AlphaServer 1200 5/533	EV56	1736
AlphaServer_1200	AlphaServer 1200 5/533	EV56	1746
AlphaServer_1200	AlphaServer 1200 5/533	EV56	1748
AlphaServer_1200	AlphaStation 1200 5/533	EV56	1766
AlphaServer_1200	AlphaStation 1200 5/533	EV56	1768
AlphaServer_1200	AlphaStation 1200 5/533	EV56	1778
AlphaServer_1200	AlphaStation 1200 5/533	EV56	1780
AlphaServer_1200	AlphaServer 1200 5/600	EV56	1738
AlphaServer_1200	AlphaServer 1200 5/600	EV56	1740
AlphaServer_1200	AlphaServer 1200 5/600	EV56	1750
AlphaServer_1200	AlphaStation 1200 5/600	EV56	1752
AlphaServer_1200	AlphaStation 1200 5/600	EV56	1770
AlphaServer_1200	AlphaStation 1200 5/600	EV56	1772
AlphaServer_1200	AlphaStation 1200 5/600	EV56	1782

AlphaServer_1200	AlphaStation 1200 5/600	EV56	1784
AlphaServer_1200	AlphaServer 1200 5/666	EV56	1742
AlphaServer_1200	AlphaServer 1200 5/666	EV56	1744
AlphaServer_1200	AlphaServer 1200 5/666	EV56	1754
AlphaServer_1200	AlphaServer 1200 5/666	EV56	1756
AlphaServer_1200	AlphaStation 1200 5/666	EV56	1774
AlphaServer_1200	AlphaStation 1200 5/666	EV56	1776
AlphaServer_1200	AlphaStation 1200 5/666	EV56	1786
AlphaServer_1200	AlphaStation 1200 5/666	EV56	1788

AlphaServer 2000

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_2000	AlphaServer 2000 4/200	EV4	1123
AlphaServer_2000	AlphaServer 2000 4/233	EV45	1171
AlphaServer_2000	AlphaServer 2000 4/275	EV45	1127

AlphaServer 2100

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_2100	AlphaServer 2100 4/200	EV4	1059
AlphaServer_2100	AlphaServer 2100 4/200	EV4	1135
AlphaServer_2100	AlphaServer 2100 4/233	EV45	1179
AlphaServer_2100	AlphaServer 2100 4/233	EV45	1187
AlphaServer_2100	AlphaServer 2100 4/275	EV45	1115
AlphaServer_2100	AlphaServer 2100 4/275	EV45	1139

AlphaServer 4000

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1409
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1411
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1421
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1423
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1433
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1435
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1445
AlphaServer_4000	AlphaServer 4000 5/266	EV5	1447
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1413
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1415
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1425
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1427

AlphaServer_4000	AlphaServer 4000 5/300	EV5	1437
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1439
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1449
AlphaServer_4000	AlphaServer 4000 5/300	EV5	1451
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1417
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1419
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1429
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1431
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1441
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1443
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1453
AlphaServer_4000	AlphaServer 4000 5/400	EV56	1455
AlphaServer_4000	AlphaServer 4000 5/466	EV56	1634
AlphaServer_4000	AlphaServer 4000 5/466	EV56	1636
AlphaServer_4000	AlphaServer 4000 5/466	EV56	1654
AlphaServer_4000	AlphaServer 4000 5/466	EV56	1656
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1638
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1640
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1642
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1644
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1658
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1660
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1662
AlphaServer_4000	AlphaServer 4000 5/533	EV56	1664
AlphaServer_4000	AlphaServer 4000 5/600	EV56	1646
AlphaServer_4000	AlphaServer 4000 5/600	EV56	1648
AlphaServer_4000	AlphaServer 4000 5/600	EV56	1666
AlphaServer_4000	AlphaServer 4000 5/600	EV56	1668
AlphaServer_4000	AlphaServer 4000 5/666	EV56	1650
AlphaServer_4000	AlphaServer 4000 5/666	EV56	1652
AlphaServer_4000	AlphaServer 4000 5/666	EV56	1670
AlphaServer_4000	AlphaServer 4000 5/666	EV56	1672

AlphaServer 4100

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1313
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1317
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1337
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1341
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1361
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1365

AlphaServer_4100	AlphaServer 4100 5/266	EV5	1385
AlphaServer_4100	AlphaServer 4100 5/266	EV5	1389
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1321
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1325
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1345
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1349
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1369
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1373
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1393
AlphaServer_4100	AlphaServer 4100 5/300	EV5	1397
AlphaServer_4100	AlphaServer 4100 5/400	EV56	1329
AlphaServer_4100	AlphaServer 4100 5/400	EV56	1333
AlphaServer_4100	AlphaServer 4000 5/400	EV56	1353
AlphaServer_4100	AlphaServer 4000 5/400	EV56	1357
AlphaServer_4100	AlphaServer 4000 5/400	EV56	1377
AlphaServer_4100	AlphaServer 4100 5/400	EV56	1381
AlphaServer_4100	AlphaServer 4100 5/400	EV56	1401
AlphaServer_4100	AlphaServer 4100 5/400	EV56	1405
AlphaServer_4100	AlphaServer 4100 5/466	EV56	1594
AlphaServer_4100	AlphaServer 4100 5/466	EV56	1598
AlphaServer_4100	AlphaServer 4100 5/533	EV56	1602
AlphaServer_4100	AlphaServer 4100 5/533	EV56	1606
AlphaServer_4100	AlphaServer 4100 5/533	EV56	1610
AlphaServer_4100	AlphaServer 4100 5/533	EV56	1614
AlphaServer_4100	AlphaServer 4100 5/600	EV56	1618
AlphaServer_4100	AlphaServer 4100 5/600	EV56	1622
AlphaServer_4100	AlphaServer 4100 5/666	EV56	1626
AlphaServer_4100	AlphaServer 4100 5/666	EV56	1630

AlphaServer/AlphaStation DS10 and AlphaServer DS10L

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_DS10	AlphaServer DS10 6/466	EV6	1839
AlphaServer_DS10	AlphaStation DS10 6/466	EV6	1879
AlphaServer_DS10	AlphaStation XP900 6/466	EV6	1879
AlphaServer_DS10L	AlphaServer DS10L 6/466	EV6	1961
AlphaServer_DS10L	AlphaServer DS10L 67/616	EV67	1962
AlphaServer_DS10	AlphaStation DS10 67/616	EV67	1962
AlphaServer_DS10	AlphaServer DS10 67/616	EV67	1970

AlphaServer DS15 and AlphaStation DS15

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_DS15	AlphaServer DS15 68CB/1000	EV68	2047
AlphaServer_DS15	AlphaStation DS15 68CB/1000	EV68	2048
AlphaServer_DS15	AlphaServer TS15 68CB/1000	EV68	2049

AlphaServer DS20 and AlphaStation DS20

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_DS20	AlphaServer DS20 6/500	EV6	1838
AlphaServer_DS20	AlphaServer DS20E 6/500	EV6	1840
AlphaServer_DS20	AlphaServer DS20 6/500	EV6	1920
AlphaServer_DS20	AlphaServer DS20 6/500	EV6	1921
AlphaServer_DS20	AlphaServer DS20E 67/667	EV67	1939
AlphaServer_DS20	AlphaStation DS20E 6/500	EV6	1941
AlphaServer_DS20	AlphaStation DS20E 67/667	EV57	1943
AlphaServer_DS20	AlphaServer DS20E 68A/833	EV68	1964
AlphaServer_DS20	AlphaServer DS20E 68A/833	EV68	1982
AlphaServer_DS20	AlphaServer DS20L 68A/833	EV68	2006

AlphaServer DS25 and AlphaStation DS25

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_DS25	AlphaServer DS25 68CB/1000	EV68	1994
AlphaServer_DS25	AlphaStation DS25 68CB/1000	EV68	1995

AlphaServer ES40 and AlphaStation ES40

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_ES40	AlphaServer ES40 6/500	EV6	1813
AlphaServer_ES40	AlphaServer ES40 6/500	EV6	1861
AlphaServer_ES40	AlphaServer ES40 6/500	EV6	1869
AlphaServer_ES40	AlphaServer ES40 6/500	EV6	1923
AlphaServer_ES40	AlphaServer ES40 6/500	EV6	1931
AlphaServer_ES40	AlphaServer ES40 6/667	EV6	1817
AlphaServer_ES40	AlphaServer ES40 6/667	EV6	1865
AlphaServer_ES40	AlphaServer ES40 6/667	EV6	1873
AlphaServer_ES40	AlphaServer ES40 6/667	EV6	1927
AlphaServer_ES40	AlphaServer ES40 6/667	EV6	1935
AlphaServer_ES40	AlphaStation ES40 67/667	EV67	1949
AlphaServer_ES40	AlphaStation ES40 67/667	EV67	1957
AlphaServer_ES40	AlphaStation ES40 68/833	EV68	1984
AlphaServer_ES40	AlphaStation ES40 68/833	EV68	1988

AlphaServer ES45

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=
AlphaServer_ES45	AlphaServer ES45/3B 68CB/1000	EV68	1971
AlphaServer_ES45	AlphaServer ES45/2 68CB/1000	EV68	1975
AlphaServer_ES45	AlphaServer ES45/2B 68CB/1000	EV68	1975
AlphaServer_ES45	AlphaServer ES45/1B 68CB/1000	EV68	2002
AlphaServer_ES45	AlphaServer ES45/3B 68CB/1250	EV68	2013
AlphaServer_ES45	AlphaServer ES45/2 68CB/1250	EV68	2017
AlphaServer_ES45	AlphaServer ES45/2B 68CB/1250	EV68	2017
AlphaServer_ES45	AlphaServer ES45/1B 68CB/1250	EV68	2021

AlphaServer GS80

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=	set rom dsrdb[1]=	set rom dsrdb[4]=
AlphaServer_GS80	AlphaServer GS80 67/728	EV67	1967		
AlphaServer_GS80	AlphaServer GS1280	EV67	2038	50	3050

AlphaServer GS160

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=	set rom dsrdb[1]=	set rom dsrdb[4]=
AlphaServer_GS160	AlphaServer GS160 67/728	EV67	1968		
AlphaServer_GS160	AlphaServer GS1280	EV67	2039	50	3050

AlphaServer GS320

set session hw_model=	set rom system_name=	set ace cpu_architecture=	set rom dsrdb[0]=	set rom dsrdb[1]=	set rom dsrdb[4]=
AlphaServer_GS320	AlphaServer GS320 67/728	EV67	1969		
AlphaServer_GS320	AlphaServer GS1280	EV67	2040	50	3050

Auto Boot

CHARON-AXP systems can be configured to boot the operating system automatically at start up.

auto_action restart

Parameter	auto_action restart
Type	Text string
Value	<p>Determines whether CHARON-AXP boots automatically if the correct boot flags are set (and saved in the Alpha console files).</p> <p>Example:</p> <pre>>>>set bootdef_dev dka0 >>>set auto_action restart</pre>

Setting System Marketing Model (SMM)

CHARON-AXP allows to set an exact System Marketing Model (SMM) for a given model of Alpha, for example:

```
set rom dsrdb[0]=1090
```

Refer to [Core Devices#Setting of a particular Alpha model](#) to find allowed values of SMM per each Alpha model supported by CHARON-AXP.

Console

Table of Contents


- General Description
- General parameters
 - alias
 - communication
 - baud
 - break_on
 - stop_on
 - log
 - log_file_size
 - log_flush_period
 - port
 - application
 - connection_override
 - access_control
 - Notes on "port" and "application" parameters
- Mapping Serial line controllers to system resources
 - line
 - "ttyY" notation specifics
- OPA0 console configuration examples
 - Using legacy syntax (not recommended)
 - Using new syntax (recommended)

General Description

CHARON-AXP offers two-port serial console on all supported AXP models.

Example for OPA0 console ("COM1" port):

```
set COM1 alias=OPA0
```

 Starting with Charon-AXP V4.12 build 204-13, the line above is optional. OPA0 is mapped by default to COM1.


Example when using the TTA0 console ("COM2" port):

```
set COM2 alias=TTA0
```


Refer to [Mapping Serial line controllers to system resources](#) for details of mapping.

General parameters

CHARON-AXP console lines COM1 and COM2 have the following parameters:

 All the values in the following tables are case insensitive.

alias

Parameter	alias
Type	Identifier
Value	<p>This parameter is used to set an useful name for COM1 or COM2 ports. It can be any name, for example "Console1", but usually it is "OPA0" for COM1 and "TTA0" for COM2.</p> <p>This name is logged in CHARON log file, it can also be used for parametrization in CHARON configuration file along with "COM1" and "COM2" identifiers.</p> <p>The main purpose of this parameter is migration from old CHARON systems (which do not have the described implementation of consoles) to the current design, since it allows retaining the original name used for parametrization, since the rest of the parameters stay the same in both implementations.</p> <p> This note applies to builds before 204-13 If the "alias" parameter is not specified CHARON log file will miss the name for the given console, for example " : Connected. Remote 127.0.0.1:63516" will be displayed instead of "OPA0 : Connected. Remote 127.0.0.1:63516". So it is always recommended to specify the "alias" parameter.</p> <p>Example:</p> <pre>set COM2 alias=TTA0</pre>



communication

Parameter	communication
Type	Text string
Value	<ul style="list-style-type: none"> • "ascii" - for connection to terminals (default) • "binary" - for binary (packet) protocols, which are used mainly for communicating with PLCs


baud

Parameter	baud
Type	Numeric
Value	<p>Forces the baud rate of the corresponding TTY port to a specified value. The variety of supported values depends on the underlying physical communication resource (TTY port). The most widely used values are: 300, 1200, 9600, 19200, 38400.</p> <p>Example:</p> <pre>set OPA0 baud=38400</pre>


break_on

Parameter	break_on
Type	Text string
Value	<p>Specifies what byte sequences received over the physical serial line will trigger a HALT command.</p> <p>This parameter works only for the console line.</p> <p>Specify the following values: "Ctrl-P", "Break" or "none" ("none" disables triggering a HALT condition).</p> <p> If your guest operating system is OpenVMS in addition to "none" setting you have to set a specific console parameter "controlp" to "off" in the following way:</p> <pre>>>> set controlp off >>> power off</pre> <p>The second line is to preserve the ROM settings.</p> <p>Example:</p> <pre>set OPA0 break_on="Ctrl-P"</pre> <p>The default value is "Break".</p> <p> This parameter can be specified only for COM1 (OPA0) console</p>


stop_on

Parameter	stop_on
Type	Text string
Value	<p>Specifies what byte sequences received over the physical serial line will trigger a STOP condition. The STOP condition causes CHARON-AXP to exit.</p> <p>Specify either "F6" or "none" ("none" disables triggering a STOP condition).</p> <p>Example:</p> <pre>set OPA0 stop_on="F6"</pre> <p>The default value is "none".</p> <p>Setting "F6" triggers the STOP condition upon receipt of the "<ESC>[17~" sequence. Terminals usually send these sequences by pressing the F6 button</p> <p> This parameter can be specified only for COM1 (OPA0) console</p>

log

Parameter	log
Type	Text string
Value	<p>A string specifying a file name to store the content of the console sessions or a directory where the log files for each individual session will be stored.</p> <p>If an existing directory is specified, CHARON-AXP automatically enables creation of individual log files, one for each session using the same scheme as used for the generation of the rotating log files. If the "log" parameter is omitted, CHARON-AXP does not create a console log.</p> <p>Example 1:</p> <pre>set OPA0 log="log.txt"</pre> <p>Example 2:</p> <pre>set OPA0 log="/Charon/Logs"</pre> <p> Only existing directory can be used as a value of the "log" parameter.</p>

log_file_size

Parameter	log_file_size
Type	Text string
Value	<p>If log rotation is enabled, the log_file_size parameter determines the log file size threshold at which the log is automatically rotated.</p> <ul style="list-style-type: none"> • "unlimited" or "0" (default) - the feature is disabled • "default" - default size is used (4Mb) • <size>[KMG] - size of the current log file in bytes with additional multipliers: <ul style="list-style-type: none"> • K - Kilobyte - multiply by 1024 • M - Megabyte - multiply by 1024*1024 • G - Gigabyte - multiply by 1024*1024*1024 <p>Example 1:</p> <pre>set OPA0 log_file_size="default"</pre> <p>Example 2:</p> <pre>set OPA0 log_file_size=10M</pre> <p> Minimum log file size is 64K, maximum is 1G. Setting size less than 64K effectively makes the log file unlimited.</p>

log_flush_period

Parameter	log_flush_period
Type	Numeric
Value	<ul style="list-style-type: none"> • <period-in-seconds> <p>Defines a period of flushing log to disk.</p> <p>Default period is 60 seconds (it means that every minute log file is flushed to disk)</p> <p>Example:</p> <pre>set OPA0 log_flush_period=30</pre>


port

Parameter	port
Type	Numeric
Value	<p>The TCP/IP port number for the virtual serial line. A virtual serial line always listens on this port for incoming connection requests. If multiple virtualized machines are running on a server, ensure the port number is unique across the platform.</p>


application


Parameter	application
Type	Text string
Value	<p>A command line for calling some host application for communication to Charon on a given COM line. Typically this parameter is used for mapping COM1 or COM2 or some "xterm" terminal emulator</p> <p>Example:</p> <pre>set OPA0 application = "xterm -title OPA0 -e telnet 127.0.0.1 10003"</pre> <p>If "putty" terminal emulator is going to be used as an option copy the following file to your home directory:</p> <pre># mkdir -p \$HOME/.config/putty/sessions (if it does not already exist) # cp /opt/charon/putty/sessions/CHTERM-VT100 \$HOME/.config/putty/sessions</pre> <p>Example:</p> <pre>set OPA0 application = "putty -load CHTERM-VT100 -title OPA0@XYZ -P 10003"</pre>

connection_override

Parameter	connection_override
Type	text string
Value	<p>"enable"</p> <p>Allows new connection to override existing connection, if any. Enabled connection override on OPA0 allows to intercept virtual serial console.</p> <p>When emulator detects new connection request on the port (10003 for the below example), it closes old connection, if any, and switches to the new one.</p> <p>Example:</p> <pre>set COM1 alias = OPA0 set OPA0 port = 10003 connection_override = enable</pre> <p> This is implemented only for serial lines using the new syntax, not for lines using the legacy syntax (load virtual_serial_line ...).</p>

access_control

 Available only since build 204-13

Parameter	access_control
Type	text string
Value	<p>"disable"</p> <p>Since build 204-13, Incoming connection requests are by default filtered for virtual serial lines and then allowed only for the localhost. This is to avoid security scanners that can block the port.</p> <p>Example:</p> <pre>set OPA0 access_control = disable</pre> <p> This is implemented only for serial lines using the new syntax, not for lines using the legacy syntax (load virtual_serial_line ...).</p>

Notes on "port" and "application" parameters



Use the combination of "port" and "application" parameters as follows to connect a 3rd party terminal emulator or similar program.

```
set COM1 alias=OPA0 port=10003 application="xterm -title OPA0 -e chterm -h 127.0.0.1:10003"
```

In this example CHARON-AXP OPA0 console connects to port 10003 of localhost ("127.0.0.1") and at the same time it starts "xterm" with parameters "-title OPA0 -e chterm -h 127.0.0.1:10003", instructs it to connect to the port 10003 of the host with TCP/IP address "127.0.0.1" (localhost)

Mapping Serial line controllers to system resources

line

Parameter	line
Type	Text string
Value	<p>A defined TTY port on host system (or "(console)" value):</p> <ul style="list-style-type: none"> ■ (console) for CHARON console (the console from which it starts). It is the default setting for COM1 ("OPA0") ■ "/dev/tty<N>" for virtual console ■ "/dev/ttyS<N>" for onboard serial lines ■ "/dev/ttyUSB<N>" for modem or USB serial lines adapters ■ "/dev/tty<XXX>" for proprietary (depending on a driver) devices such as DIGI or MOXA cards <p> If a virtual console "/dev/tty<N>" is going to be used, it must be freed from all the processes running on it at first. Refer to your OS documentation for details, also some description on how to do it is available here.</p> <p> A specific account for running CHARON ("charon") does not allow usage of physical consoles "/dev/tty<N>" as CHARON consoles. If you plan to map CHARON console to "/dev/tty<N>" use only "root" account for CHARON running.</p>

"ttyY" notation specifics

Note that the "ttyY" notation can have different forms depending on the nature of the device used:

Mapping	Type	Commentary
"/dev/tty<N>" where N is from 0 to 11	Linux virtual tty	Those tty devices must be free from the Linux "getty/mgetty" and similar programs (specified in "/etc/inittab") Example: "/dev/tty1"
"/dev/ttyS<N>" where N is a number	Onboard serial lines	Example: "/dev/ttyS1"
"/dev/tty<XXX>" where XXX is a complex letter /number notation	Proprietary (depending on a driver) devices	Example for a first port of a MOXA card: "/dev/ttyR01" Example for a first port of a DIGI card: "/dev/ttyaa"
"/dev/ttyUSB<N>" where N is a number	Modem or USB serial lines adapters	Example: "/dev/ttyUSB1"

OPA0 console configuration examples

Using legacy syntax (not recommended)

This example maps OPA0 to port 10003, enable F6 key (emulator stop) and logs the console input/output to a rotating log file in /charon/myaxp/logs folder:

```
load virtual_serial_line OPA0
set OPA0 port=10003
set OPA0 stop_on="F6"
set OPA0 log="/charon/myaxp/logs"
```

Using new syntax (recommended)

This example maps OPA0 to port 10003, enable F6 key (emulator stop), logs the console input/output to a rotating log file in /charon/myaxp/logs folder and enables the connection_override feature:

```
set COM1 alias=OPA0 (optional starting with build 204-13)
set OPA0 port=10003
set OPA0 stop_on="F6"
set OPA0 log="/charon/myaxp/logs"
set OPA0 connection_override=enable
```

Remote Management Console (RMC)

Table of Contents

- General Description
 - Parameter
 - Examples
- Connection
- Available commands

General Description

The purpose of the Remote Management Console is to let the emulator trigger actions on the Charon host to properly unload and save a vtape container while the guest operating system is running.

Originally, the tape **offline** state in Charon-VAX and Charon-AXP was not persistent. An automatic, periodic loading function will bring the tape device **online** again using the same container. That means that subsequent operations might overwrite the tape container.

To improve this behavior, an initially persistent **offline** state was introduced that could be configured by providing an "empty tape drive" using a name of **.vtape**. With this improvement, a virtual tape device will remain **offline**, until a valid name is provided during the runtime of the emulator. This name can be provided in two ways:

- CHARONCP: SCSI devices on OpenVMS only (see "Charon Guest Utilities for OpenVMS" chapter)
- Remote Management Console (this section)

However, this first improvement did not allow to return to an "empty tape drive" once a valid name had been provided. Hence, the automatic loading process would once again create the problem of possibly overwriting an existing container. This problem triggered an improvement that will allow a **persistent offline state**. The persistent offline state is configured on platforms where it is supported by adding a new keyword to the removable parameter as shown in the example below from a configuration file:

```
set PKA container[0] = ".vtape" removable = "noauto"
```

With this configuration, a tape device in Charon behaves as follows. When the tape device is put into status **offline**, it clears the runtime value of the associated container turning it into an "empty tape drive". Any further operations on this tape device are only possible after changing value of the container to a meaningful value and loading the tape. This behavior makes the runtime **offline** state persistent.

To load the Remote Management Console use the following syntax:

```
load remote_management_console RMC
```

 Only one Remote Management Console can be set per Charon instance.

Parameter

Parameter	port
Type	Numeric
Value	The TCP/IP port number for the remote management console. If multiple Charon instances are running on a server, ensure the port number is unique across the platform.

Examples

Example 1:

```
load remote_management_console RMC port=13000
```

Example 2:

```
load remote_management_console RMC
set RMC port=13000
```

Connection

The RMC is reachable via the configured TCP port using (for example) a **telnet** client.

Notes:

- Line mode has to be used (default) and not character mode during telnet connection. If necessary, use the "-c" parameter to get rid of any defined .telnetrc file or press the escape character and enter "mode line" at the telnet prompt.
- If you use 'putty' to access the RMC, use "Raw" connection type and not "Telnet" and ensure the "Implicit CR in every LF" option in the "Terminal" settings is checked.
- If you access from a remote location to the RMC port, please define appropriate firewall rules if necessary.


Example:

```
# telnet -c localhost 13000
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
1, Ok
help
hello
help
set <name> {<parameter> = <value>}*
show removable [media]
acquire <name> <unit-no>
release <name> <unit-no>
show virtual [serial lines]
disconnect <name> <unit-no>
bye
1, Ok
bye
Connection closed by foreign host.
```

Available commands

Short description of the available commands:

Command	Description
help	Usage information as shown in the example above
hello	Currently no user-relevant function
set <i><name></i> { <i><parameter></i> = <i><value></i> }	The set command is used to set a meaningful name for a vtape container. Example: <pre>set PUA container[11] = mytape1.vtape</pre>
show removable	Lists the removable devices configured for the specific emulator instance.
acquire <i><name></i> <i><unit-no></i>	Brings a vtape online . Example: <pre>acquire pua 11</pre>
release <i><name></i> <i><unit-no></i>	Brings a vtape offline . If noauto is configured, an empty container name is set. Example: <pre>release pua 11</pre>
show virtual <i><serial line></i>	List configured serial lines. If run without parameters, all serial lines will be listed.
disconnect <i><name></i> <i><unit-no></i>	Disconnect virtual serial line. Example: <pre>disconnect com2 0</pre>
bye	Disconnect from RMC


 Commands cannot be shortened


Example:

```

# telnet -c localhost 13000
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
1, Ok
show removable
PUA, 11, offline, ".vtape"
PUA, 12, offline, ".vtape"
1, Ok
set PUA container[11] = xxx.vtape
1, Ok
show removable
PUA, 11, offline, "xxx.vtape"
PUA, 12, offline, ".vtape"
1, Ok
acquire pua 11
1, Ok
show removable
PUA, 11, available, "xxx.vtape"
PUA, 12, offline, ".vtape"
1, Ok
release pua 11
1, Ok
show removable
PUA, 11, offline, "xxx.vtape"
PUA, 12, offline, ".vtape"
1, Ok
bye
Connection closed by foreign host.

```

 Future versions may contain additional features to support the automation of the functionality that the Remote Management Console provides. Currently, if you need support for an automation project, please contact your Stromasys representative or your Stromasys VAR.

 Current connection can be closed disregard to connection_override settings.

Placement of peripheral devices on PCI bus

Table of Contents

- General Description
 - bus
 - device
 - function
 - irq_bus
 - irq
- Available PCI slots per each Alpha model emulated by CHARON-AXP
 - AlphaServer 400 (3 PCI slots)
 - AlphaServer 800 (4 PCI slots)
 - AlphaServer 1000 (3 PCI slots)
 - AlphaServer 1000A (7 PCI slots)
 - AlphaServer 1200 (6 PCI slots)
 - AlphaServer 2000 (3 PCI slots)
 - AlphaServer 2100 (3 PCI slots)
 - AlphaServer 4000 (16 PCI slots)
 - AlphaServer 4100 (8 PCI slots)
 - AlphaServer DS10 (4 PCI slots)
 - AlphaServer DS10L (1 PCI slot)
 - AlphaServer DS15 (4 PCI slots)
 - AlphaServer DS20 (6 PCI slots)
 - AlphaServer DS25 (6 PCI slots)
 - AlphaServer ES40 (10 PCI slots)
 - AlphaServer ES45 (10 PCI slots)
 - AlphaServer GS80 (8 PCI busses)
 - AlphaServer GS160 (16 PCI busses)
 - AlphaServer GS320 (32 PCI busses)

General Description

Each peripheral device of CHARON-AXP connects to an emulated PCI bus with the following configuration parameters:

bus

Parameter	bus
Type	Text string
Value	<p>When specified, the bus configuration parameter tells the CHARON-AXP software the virtual PCI bus to which virtual Alpha system shall connect a certain virtual PCI adapter.</p> <p>If the bus configuration parameter is not specified, CHARON-AXP software connects the virtual PCI adapter to the first available virtual PCI bus</p> <p>By default the bus configuration parameter is not specified.</p> <ul style="list-style-type: none"> • For AlphaServer 400-4100, DS, ES, format is: "pci_<X>" • For AlphaServer GS, format is: "qbb_<X>_pca_<Y>_pci_<Z>" <p>Example (AlphaServer ES40):</p> <pre>load KZPBA PKA bus=pci_1</pre> <p>Example (AlphaServer GS80):</p> <pre>load KZPBA PKA bus=qbb_1_pca_1_pci_0</pre>

device

Parameter	device
Type	Numeric
Value	<p>When specified, the device configuration parameter specifies position of a virtual PCI adapter on virtual PCI bus.</p> <p>If the device configuration parameter is not specified, the CHARON software connects the virtual PCI adapter at the first available position of the virtual PCI bus.</p> <p>By default the device configuration parameter is not specified.</p> <p>Example:</p> <pre>load KZPBA PKA device=2</pre>

function

Parameter	function
Type	Numeric
Value	<p>When specified, the function configuration parameter specifies position of a virtual PCI adapter on virtual PCI bus.</p> <p>If the function configuration parameter is not specified, the CHARON software connects the virtual PCI a dapter at the first available position of the virtual PCI bus.</p> <p>By default the function configuration parameter is not specified.</p> <p>Example:</p> <pre>load KZPBA PKA function=0</pre>

irq_bus

Parameter	irq_bus
Type	Text string
Value	<p>When specified, the "irq_bus" configuration parameter specifies virtual bus routing interrupt requests from virtual PCI adapter to CHARON-AXP virtual Alpha CPUs.</p> <p>The "irq_bus" configuration parameter must be set to "isa" for AlphaServer 400. For Alpha systems other than AlphaServer 400 the "irq_bus" configuration parameter must be left as is (i.e. not specified).</p> <p>By default the "irq_bus" configuration parameter is not specified.</p> <p>Example:</p> <pre>load KZPBA PKA irq_bus=isa</pre>

irq

Parameter	irq
Type	Numeric
Value	<p>When specified, the "irq" configuration parameter assigns interrupt request to the virtual PCI adapter in Alpha system.</p> <p>If the irq configuration parameter is not specified, the CHARON-AXP software uses the correct values depending on the selected PCI position of a virtual PCI adapter.</p> <p>By default the irq configuration parameter is not specified.</p> <p>Example:</p> <pre>load KZPBA PKA irq=24</pre>

Note that typically all or some of those parameters are specified on loading of some PCI controller in the following way:

```
load KZPBA PKA bus=pci_1 device=1 function=0 irq_bus=isa irq=24
```

Available PCI slots per each Alpha model emulated by CHARON-AXP

The tables below specifies a map of preloaded devices and available slots for each Alpha models emulated by CHARON-AXP.

AlphaServer 400 (3 PCI slots)

In addition to 3 PCI vacant slots there are 2 PCI positions occupied by on-board devices. All 5 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	6	0	11	NCR 53C810 PCI SCSI Adapter	PKA
-	0	7	0	-	Intel i82378 PCI ISA Bridge (SATURN)	
0	0	11	0	10	<option>	
1	0	12	0	15	<option>	
2	0	13	0	9	<option>	

The IRQ stands for ISA IRQ Number because all interrupts are routed through the Intel i82378 PCI ISA Bridge (SATURN) resident cascade of Intel i8259 interrupt controllers.

So far the CHARON-AXP emulators do not support virtual NCR 53C810 PCI SCSI adapter. Instead, virtual QLOGIC ISP1040B PCI SCSI adapter is used.

i No support for Multi-Function PCI devices in AlphaServer 400.

Example: Loading DE435 into slot 0

```
load DE435/dec21x4x EWA bus=pci_0 device=11 function=0 irq_bus=isa
```

i The "irq_bus=isa" setting is specific to AlphaServer 400 only.

AlphaServer 800 (4 PCI slots)

In addition to 4 PCI vacant slots there are 3 PCI positions occupied by on-board devices. All 7 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	5	0	0	QLOGIC ISP1020 PCI SCSI Adapter	PKA
-	0	6	0	0	S3 Trio32/64 Display Adapter	
-	0	7	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
0	0	11	0	1	<option>	
			1	2	<option>, function 1	
			2	17	<option>, function 2	
			3	18	<option>, function 3	
1	0	12	0	3	<option>	
			1	4	<option>, function 1	
			2	19	<option>, function 2	
			3	20	<option>, function 3	
2	0	13	0	5	<option>	
			1	6	<option>, function 1	
			2	21	<option>, function 2	
			3	22	<option>, function 3	
3	0	14	0	7	<option>	
			1	8	<option>, function 1	
			2	23	<option>, function 2	
			3	24	<option>, function 3	

The IRQ stands for input line of ASIC interrupt controllers. It has nothing to do with "EISA" style interrupts. So far, the CHARON-AXP emulators do not emulate S3 Trio32/64 Display Adapter. So position of the device 6, function 0 on the PCI 0 remains empty.

Example 1: Loading DE500BA into slot 0

```
load DE500BA/dec21x4x EWA bus=pci_0 device=11 function=0
```

Example 2: Loading multiple DE500BA's into slot 3, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_0 device=14 function=0
load DE500BA/dec21x4x EWB bus=pci_0 device=14 function=1
load DE500BA/dec21x4x EWC bus=pci_0 device=14 function=2
load DE500BA/dec21x4x EWD bus=pci_0 device=14 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_0 device=12 function=0
load DE500BA/dec21x4x EWA bus=pci_0 device=12 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located "closer" to CPU and therefore assigned name PKA.

AlphaServer 1000 (3 PCI slots)

In addition to 3 PCI vacant slots there are 2 PCI positions occupied by on-board devices. All 5 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	6	0	12	NCR 53C810 PCI SCSI Adapter	PKA
-	0	7	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
0	0	11	0	0	<option>	
			1	1	<option>, function 1	
			2	2	<option>, function 2	
			3	3	<option>, function 3	
1	0	12	0	4	<option>	
			1	5	<option>, function 1	
			2	6	<option>, function 2	
			3	7	<option>, function 3	
2	0	13	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	

The IRQ stands for input line of ASIC interrupt controllers. It has nothing to do with "EISA" style interrupts. So far, the CHARON-AXP emulators do not emulate NCR 53C810 PCI SCSI adapter. Instead, emulation of QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 0

```
load DE500BA/dec21x4x EWA bus=pci_0 device=11 function=0
```

Example 2: Loading multiple DE500BA's into slot 0, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_0 device=11 function=0
load DE500BA/dec21x4x EWB bus=pci_0 device=11 function=1
load DE500BA/dec21x4x EWC bus=pci_0 device=11 function=2
load DE500BA/dec21x4x EWD bus=pci_0 device=11 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 2, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_0 device=13 function=0
load DE500BA/dec21x4x EWA bus=pci_0 device=13 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located "closer" to CPU and therefore assigned name PKA.

AlphaServer 1000A (7 PCI slots)

In addition to 7 PCI vacant slots there are 3 PCI positions occupied by on-board devices. All 10 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	6	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
-	0	7	0	-	DECchip 21050 PCI-to-PCI Bridge)	
0	0	11	0	1	<option>	
			1	2	<option>, function 1	
			2	17	<option>, function 2	
			3	18	<option>, function 3	
1	0	12	0	2	<option>	
			1	3	<option>, function 1	
			2	19	<option>, function 2	
			3	20	<option>, function 3	
2	0	13	0	3	<option>	
			1	4	<option>, function 1	
			2	21	<option>, function 2	
			3	22	<option>, function 3	
<i>PCI1 (bus=pci_1)</i>						
-	1	0	0	0	NCR 53C810 PCI SCSI Adapter	PKA
3	1	1	0	7	<option>	
			1	8	<option>, function 1	
			2	23	<option>, function 2	
			3	24	<option>, function 3	
4	1	2	0	9	<option>	
			1	10	<option>, function 1	
			2	25	<option>, function 2	
			3	26	<option>, function 3	
5	1	3	0	11	<option>	
			1	12	<option>, function 1	
			2	27	<option>, function 2	
			3	28	<option>, function 3	
6	1	4	0	13	<option>	
			1	14	<option>, function 1	
			2	29	<option>, function 2	
			3	30	<option>, function 3	

The IRQ stands for input line of ASIC interrupt controllers. It has nothing to do with "EISA" style interrupts. So far, the CHARON-AXP emulators do not emulate NCR 53C810 PCI SCSI adapter. Instead, emulation of QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 0

```
load DE500BA EWA bus=pci_0 device=11 function=0
```

Example 2: Loading multiple DE500BA's into slot 0, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA EWA bus=pci_0 device=11 function=0
load DE500BA EWB bus=pci_0 device=11 function=1
load DE500BA EWC bus=pci_0 device=11 function=2
load DE500BA EWD bus=pci_0 device=11 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 3, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_1 device=1 function=0
load DE500BA EWA bus=pci_1 device=1 function=1
```

 In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA.

AlphaServer 1200 (6 PCI slots)

In addition to 6 PCI vacant slots there are 2 PCI positions occupied by on-board devices. All 8 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_1)</i>						
-	1	1	0	4	NCR 53C810 PCI SCSI Adapter	PKA
0	1	2	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
1	1	3	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	16	<option>, function 3	
2	1	4	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	
<i>PCI0 (bus=pci_0)</i>						
-	0	1	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
4	0	2	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
5	0	3	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
6	0	4	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	

So far, the CHARON-AXP emulators do not emulate NCR 53C810 PCI SCSI adapter. Instead, emulation of QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 4

```
load DE500BA/dec21x4x EWA bus=pci_0 device=2 function=0
```

Example 2: Loading multiple DE500BA's into slot 4, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_0 device=2 function=0
load DE500BA/dec21x4x EWB bus=pci_0 device=2 function=1
load DE500BA/dec21x4x EWC bus=pci_0 device=2 function=2
load DE500BA/dec21x4x EWD bus=pci_0 device=2 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_1 device=2 function=0
load DE500BA/dec21x4x EWA bus=pci_1 device=2 function=1
```

 In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA.

AlphaServer 2000 (3 PCI slots)

In addition to 3 PCI vacant slots there are 3 PCI positions occupied by on-board devices. All 6 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	0	0	2	DEC TULIP PCI Ethernet adapter	EWA
-	0	1	0	1	NCR 53C810 PCI SCSI Adapter	PKA
-	0	2	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
0	0	6	0	0	<option>	
			1	24	<option>, function 1	
			2	26	<option>, function 2	
			3	29	<option>, function 3	
1	0	7	0	4	<option>	
			1	25	<option>, function 1	
			2	27	<option>, function 2	
			3	30	<option>, function 3	
2	0	8	0	5	<option>	
			1	20	<option>, function 1	
			2	28	<option>, function 2	
			3	31	<option>, function 3	

The IRQ stands for input line of T2 resident cascade of Intel i8259 interrupt controllers. It has nothing to do with "EISA" style interrupts.

So far the CHARON-AXP emulators do not support virtual NCR 53C810 PCI SCSI adapter. Instead, virtual QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 0

```
load DE500BA/dec21x4x EWB bus=pci_0 device=6 function=0
```

Example 2: Loading multiple DE500BA's into slot 0, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWB bus=pci_0 device=6 function=0
load DE500BA/dec21x4x EWC bus=pci_0 device=6 function=1
load DE500BA/dec21x4x EWD bus=pci_0 device=6 function=2
load DE500BA/dec21x4x EWE bus=pci_0 device=6 function=3
```

i In the above examples device name is EWB as there is a built-in EW-like PCI Ethernet Adapter located "closer" to CPU and therefore assigned name EWA.

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_0 device=7 function=0
load DE500BA/dec21x4x EWB bus=pci_0 device=7 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located "closer" to CPU and therefore assigned name PKA, and device name is EWB as there is a built-in EW-like PCI Ethernet Adapter located "closer" to CPU and therefore assigned name EWA.

AlphaServer 2100 (3 PCI slots)

In addition to 3 PCI vacant slots there are 3 PCI positions occupied by on-board devices. All 6 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	0	0	2	DEC TULIP PCI Ethernet adapter	EWA
-	0	1	0	1	NCR 53C810 PCI SCSI Adapter	PKA
-	0	2	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
0	0	6	0	0	<option>	
			1	24	<option>, function 1	
			2	26	<option>, function 2	
			3	29	<option>, function 3	
1	0	7	0	4	<option>	
			1	25	<option>, function 1	
			2	27	<option>, function 2	
			3	30	<option>, function 3	
2	0	8	0	5	<option>	
			1	20	<option>, function 1	
			2	28	<option>, function 2	
			3	31	<option>, function 3	

The IRQ stands for input line of T2 resident cascade of Intel i8259 interrupt controllers. It has nothing to do with "EISA" style interrupts.

So far the CHARON-AXP emulators do not support virtual NCR 53C810 PCI SCSI adapter. Instead, virtual QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 0

```
load DE500BA/dec21x4x EWB bus=pci_0 device=6 function=0
```

Example 2: Loading multiple DE500BA's into slot 0, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWB bus=pci_0 device=6 function=0
load DE500BA/dec21x4x EWC bus=pci_0 device=6 function=1
load DE500BA/dec21x4x EWD bus=pci_0 device=6 function=2
load DE500BA/dec21x4x EWE bus=pci_0 device=6 function=3
```

i In the above examples device name is EWB as there is a built-in EW-like PCI Ethernet Adapter located "closer" to CPU and therefore assigned name EWA.

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_0 device=7 function=0
load DE500BA/dec21x4x EWB bus=pci_0 device=7 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located "closer" to CPU and therefore assigned name PKA, and device name is EWB as there is a built-in EW-like PCI Ethernet Adapter located "closer" to CPU and therefore assigned name EWA.

AlphaServer 4000 (16 PCI slots)

In addition to 16 PCI vacant slots there are 2 PCI positions occupied by on-board devices. All 18 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_1)</i>						
-	1	1	0	4	NCR 53C810 PCI SCSI Adapter	PKA
1	1	2	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
2	1	3	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
3	1	4	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	
4	1	5	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	
<i>PCI0 (bus=pci_0)</i>						
-	0	1	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
5	0	2	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
6	0	3	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
7	0	4	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	
8	0	5	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	

PCI3 (bus=pci_3)					
9	3	2	0	8	<option>
			1	9	<option>, function 1
			2	10	<option>, function 2
			3	11	<option>, function 3
10	3	3	0	12	<option>
			1	13	<option>, function 1
			2	14	<option>, function 2
			3	15	<option>, function 3
11	3	4	0	16	<option>
			1	17	<option>, function 1
			2	18	<option>, function 2
			3	19	<option>, function 3
12	3	5	0	20	<option>
			1	21	<option>, function 1
			2	22	<option>, function 2
			3	23	<option>, function 3
PCI2 (bus=pci_2)					
13	2	2	0	8	<option>
			1	9	<option>, function 1
			2	10	<option>, function 2
			3	11	<option>, function 3
14	2	3	0	12	<option>
			1	13	<option>, function 1
			2	14	<option>, function 2
			3	15	<option>, function 3
15	2	4	0	16	<option>
			1	17	<option>, function 1
			2	18	<option>, function 2
			3	19	<option>, function 3
16	2	5	0	20	<option>
			1	21	<option>, function 1
			2	22	<option>, function 2
			3	23	<option>, function 3

So far the CHARON-AXP emulators do not support virtual NCR 53C810 PCI SCSI adapter. Instead, virtual QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 4

```
load DE500BA/dec21x4x EWA bus=pci_1 device=5 function=0
```

Example 2: Loading multiple DE500BA's into slot 4, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_1 device=5 function=0
load DE500BA/dec21x4x EWB bus=pci_1 device=5 function=1
load DE500BA/dec21x4x EWC bus=pci_1 device=5 function=2
load DE500BA/dec21x4x EWD bus=pci_1 device=5 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_1 device=2 function=0
load DE500BA/dec21x4x EWA bus=pci_1 device=2 function=1
```

 In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA.

AlphaServer 4100 (8 PCI slots)

In addition to 8 PCI vacant slots there are 2 PCI positions occupied by on-board devices. All 10 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_1)</i>						
-	1	1	0	4	NCR 53C810 PCI SCSI Adapter	PKA
1	1	2	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
2	1	3	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
3	1	4	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	
4	1	5	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	
<i>PCI0 (bus=pci_0)</i>						
-	0	1	0	-	Intel i82375 PCI EISA Bridge (MERCURY)	
5	0	2	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
6	0	3	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
7	0	4	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	
8	0	5	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	

So far the CHARON-AXP emulators do not support virtual NCR 53C810 PCI SCSI adapter. Instead, virtual QLOGIC ISP1040B PCI SCSI adapter is used.

Example 1: Loading DE500BA into slot 4

```
load DE500BA/dec21x4x EWA bus=pci_1 device=5 function=0
```

Example 2: Loading multiple DE500BA's into slot 4, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_1 device=5 function=0
load DE500BA/dec21x4x EWB bus=pci_1 device=5 function=1
load DE500BA/dec21x4x EWC bus=pci_1 device=5 function=2
load DE500BA/dec21x4x EWD bus=pci_1 device=5 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_1 device=2 function=0
load DE500BA/dec21x4x EWA bus=pci_1 device=2 function=1
```

 In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA.

AlphaServer DS10 (4 PCI slots)

In addition to 4 PCI vacant slots there are 5 PCI positions occupied by on-board devices. All 9 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_0)</i>						
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	9	0	29	DECchip 21143 PCI Ethernet Adapter	EWA
-	0	11	0	30	DECchip 21143 PCI Ethernet Adapter	EWB
-	0	13	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA, DQB
1	0	14	0	35	<option>	
			1	34	<option>, function 1	
			2	33	<option>, function 2	
			3	32	<option>, function 3	
2	0	15	0	39	<option>	
			1	38	<option>, function 1	
			2	37	<option>, function 2	
			3	36	<option>, function 3	
3	0	16	0	43	<option>	
			1	42	<option>, function 1	
			2	41	<option>, function 2	
			3	40	<option>, function 3	
4	0	17	0	47	<option>	
			1	46	<option>, function 1	
			2	45	<option>, function 2	
			3	44	<option>, function 3	
-	0	19	0	11	ALi M1543C PCI USB adapter	

Example 1: Loading DE500BA into slot 1

```
load DE500BA/dec21x4x EWC bus=pci_0 device=14 function=0
```

Example 2: Loading multiple DE500BA's into slot 1, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWC bus=pci_0 device=14 function=0
load DE500BA/dec21x4x EWD bus=pci_0 device=14 function=1
load DE500BA/dec21x4x EWE bus=pci_0 device=14 function=2
load DE500BA/dec21x4x EWF bus=pci_0 device=14 function=3
```

i In the above examples device name is EWC as there are built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB.

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_0 device=14 function=0
load DE500BA/dec21x4x EWC bus=pci_0 device=14 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA, as there are two built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB.

AlphaServer DS10L (1 PCI slot)

In addition to 1 PCI vacant slots there are 5 PCI positions occupied by on-board devices. All 6 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_0)</i>						
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	9	0	29	DECchip 21143 PCI Ethernet Adapter	EWA
-	0	11	0	30	DECchip 21143 PCI Ethernet Adapter	EWB
-	0	13	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA, DQB
1	0	17	0	47	<option>	
			1	46	<option>, function 1	
			2	45	<option>, function 2	
			3	44	<option>, function 3	
-	0	19	0	11	ALi M1543C PCI USB adapter	

Example 1: Loading DE500BA into slot 1

```
load DE500BA/dec21x4x EWC bus=pci_0 device=17 function=0
```

Example 2: Loading multiple DE500BA's into slot 1, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWC bus=pci_0 device=17 function=0
load DE500BA/dec21x4x EWD bus=pci_0 device=17 function=1
load DE500BA/dec21x4x EWE bus=pci_0 device=17 function=2
load DE500BA/dec21x4x EWF bus=pci_0 device=17 function=3
```

i In the above examples device name is EWC as there are built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB.

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKB bus=pci_0 device=17 function=0
load DE500BA/dec21x4x EWC bus=pci_0 device=17 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA, as there are two built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB

AlphaServer DS15 (4 PCI slots)

In addition to 4 PCI vacant slots there are 7 PCI positions occupied by on-board devices. All 11 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	8	0	-	Adaptec AIC-7899 (channel 0)	PKA
			1	-	Adaptec AIC-7899 (channel 1)	PKB
-	0	9	0	-	Intel i82559 PCI Ethernet Adapter	EIA (EWA)
-	0	10	0	-	Intel i82559 PCI Ethernet Adapter	EIB (EWB)
-	0	13	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA, DQB
-	0	19	0	-	ALi M1543C PCI USB adapter	
<i>PCI2 (bus=pci_2)</i>						
1	2	7	0	40	<option>	
			1	41	<option>, function 1	
			2	42	<option>, function 2	
			3	43	<option>, function 3	
2	2	8	0	36	<option>	
			1	37	<option>, function 1	
			2	38	<option>, function 2	
			3	39	<option>, function 3	
3	2	9	0	24	<option>	
			1	25	<option>, function 1	
			2	26	<option>, function 2	
			3	27	<option>, function 3	
4	2	10	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	

The IRQ stands for bit position in DRIR of TITAN chip. It has nothing to do with "ISA" style interrupts which are routed to IRQ 55 (including ALi M1543C PCI IDE/ATAPI controller).

So far the CHARON-AXP emulators do not emulate Adaptec AIC-7899. Instead, emulation of QLOGIC ISP1040B is used.

So far the CHARON-AXP emulators do not emulate Intel i82559. Instead, emulation of DECchip 21143 is used.


So far the CHARON-AXP emulators do not emulate ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 remains empty.

Example 1: Loading DE500BA into slot 1

```
load DE500BA/dec21x4x EWC bus=pci_2 device=7 function=0
```


Example 2: Loading multiple DE500BA's into slot 2, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWC bus=pci_2 device=8 function=0
load DE500BA/dec21x4x EWD bus=pci_2 device=8 function=1
load DE500BA/dec21x4x EWE bus=pci_2 device=8 function=2
load DE500BA/dec21x4x EWF bus=pci_2 device=8 function=3
```

 In the above examples device name is EWC as there are built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB.

Example 3: Loading mixture of KZPBA and DE500BA into slot 3, populating 2 functions out of 4

```
load KZPBA PKC bus=pci_2 device=9 function=0
load DE500BA/dec21x4x EWC bus=pci_2 device=9 function=1
```

 In the above example device name is PKC as there are 2 built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA and PKB, as there are two built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB

AlphaServer DS20 (6 PCI slots)

In addition to 6 PCI vacant slots there are 5 PCI positions occupied by on-board devices. All 11 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_1)</i>						
4	1	7	0	47	<option>	
			1	46	<option>, function 1	
			2	45	<option>, function 2	
			3	44	<option>, function 3	
5	1	8	0	43	<option>	
			1	42	<option>, function 1	
			2	41	<option>, function 2	
			3	49	<option>, function 3	
6	1	9	0	39	<option>	
			1	38	<option>, function 1	
			2	37	<option>, function 2	
			3	36	<option>, function 3	
<i>PCI0 (bus=pci_0)</i>						
-	0	5	0	-	ALi M1543C PCI ISA bridge	
-	0	6	0	19	Adaptec AIC-7895 (channel 0)	PKA
			1	18	Adaptec AIC-7895 (channel 1)	PKB
-	0	15	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA, DQB
-	0	19	0	-	ALi M1543C PCI USB adapter	
1	0	7	0	31	<option>	
			1	30	<option>, function 1	
			2	29	<option>, function 2	
			3	28	<option>, function 3	
2	0	8	0	27	<option>	
			1	26	<option>, function 1	
			2	25	<option>, function 2	
			3	24	<option>, function 3	
3	0	9	0	23	<option>	
			1	22	<option>, function 1	
			2	21	<option>, function 2	
			3	20	<option>, function 3	

The IRQ stands for bit position in DRIR of Tsunami/Typhoon Chip. It has nothing to do with "ISA" style interrupts which are routed to IRQ 55 (including ALi M1543C PCI IDE/ATAPI controller).

Unless SCSI option is plugged into PCI slot 4, 5, or 6, the onboard SCSI controllers appear as PKA (pka7.0.0.6.0) and PKB (pkb7.0.0.106.0) respectively.

So far the CHARON-AXP emulators do not support virtual Adaptec AIC-7895 PCI SCSI adapter. Instead, virtual QLOGIC ISP1040B PCI SCSI adapter is used.

So far the CHARON-AXP emulators do not support virtual ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 remains empty

Example 1: Loading DE500BA into slot 4

```
load DE500BA/dec21x4x EWA bus=pci_1 device=7 function=0
```

Example 2: Loading multiple DE500BA's into slot 4, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_1 device=7 function=0
load DE500BA/dec21x4x EWB bus=pci_1 device=7 function=1
load DE500BA/dec21x4x EWC bus=pci_1 device=7 function=2
load DE500BA/dec21x4x EWD bus=pci_1 device=7 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKC bus=pci_0 device=7 function=0
load DE500BA/dec21x4x EWA bus=pci_0 device=7 function=1
```

i In the above example device name is PKC as there are two built-in PK-like PCI SCSI Adapters located "closer" to CPU and therefore assigned names PKA and PKB.

AlphaServer DS25 (6 PCI slots)

In addition to 6 PCI vacant slots there are 7 PCI positions occupied by on-board devices. All 13 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	8	0	-	Intel i82559 PCI Ethernet Adapter	EIA (EWA)
1	0	9	0	24	<option>	
			1	25	<option>, function 1	
			2	26	<option>, function 2	
			3	27	<option>, function 3	
2	0	10	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
-	0	16	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA, DQB
-	0	19	1	-	ALi M1543C PCI USB adapter	
<i>PCI1 (bus=pci_1)</i>						
3	1	1	0	28	<option>	
			1	29	<option>, function 1	
			2	30	<option>, function 2	
			3	31	<option>, function 3	
4	1	2	0	32	<option>	
			1	33	<option>, function 1	
			2	34	<option>, function 2	
			3	35	<option>, function 3	
<i>PCI2 (bus=pci_2)</i>						
-	2	1	0	-	Adaptec AIC-7899 (channel 0)	PKA
			1	-	Adaptec AIC-7899 (channel 1)	PKB
-	2	5	0	-	BroadCom BCM5703 PCI Ethernet Adapter	EIB (EWB)
<i>PCI3 (bus=pci_3)</i>						
5	3	1	0	36	<option>	
			1	37	<option>, function 1	
			2	38	<option>, function 2	
			3	39	<option>, function 3	
6	3	2	0	40	<option>	
			1	41	<option>, function 1	
			2	42	<option>, function 2	
			3	43	<option>, function 3	

The IRQ stands for bit position in DRIR of TITAN Chip. It has nothing to do with “ISA” style interrupts which are routed to IRQ 55 (including ALi M1543C PCI IDE/ATAPI controller).

So far the CHARON-AXP emulators do not emulate Intel i82559. Instead, emulation of DECchip 21143 is used.

So far the CHARON-AXP emulators do not emulate ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 remains empty.

Unless SCSI option is plugged into PCI slot 1, 2, 3, or 4, the onboard SCSI controllers appear as PKA (pka7.0.0.1.2) and PKB (pkb7.0.0.101.2) respectively.

So far the CHARON-AXP emulators do not emulate Adaptec AIC-7899. Instead, emulation of QLOGIC ISP1040B is used.

So far the CHARON-AXP emulators do not emulate BroadCom BCM5703. Instead, emulation of DECchip 21143 is used.

Example 1: Loading DE500BA into slot 5

```
load DE500BA/dec21x4x EWC bus=pci_3 device=1 function=0
```

Example 2: Loading multiple DE500BA's into slot 5, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWC bus=pci_3 device=1 function=0
load DE500BA/dec21x4x EWD bus=pci_3 device=1 function=1
load DE500BA/dec21x4x EWE bus=pci_3 device=1 function=2
load DE500BA/dec21x4x EWF bus=pci_3 device=1 function=3
```

i In the above examples device name is EWC as there are built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB.

Example 3: Loading mixture of KZPBA and DE500BA into slot 6, populating 2 functions out of 4

```
load KZPBA PKC bus=pci_3 device=2 function=0
load DE500BA/dec21x4x EWC bus=pci_3 device=2 function=1
```

i In the above example device name is PKB as there is a built-in PK-like PCI SCSI Adapter located “closer” to CPU and therefore assigned name PKA, as there are two built-in EW-like PCI Ethernet Adapters located “closer” to CPU and therefore assigned names EWA and EWB

AlphaServer ES40 (10 PCI slots)

In addition to 10 PCI vacant slots there are 3 PCI positions occupied by on-board devices. All 13 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI1 (bus=pci_1)</i>						
5	1	1	0	24	<option>	
			1	25	<option>, function 1	
			2	26	<option>, function 2	
			3	27	<option>, function 3	
6	1	2	0	28	<option>	
			1	29	<option>, function 1	
			2	30	<option>, function 2	
			3	31	<option>, function 3	
7	1	3	0	32	<option>	
			1	33	<option>, function 1	
			2	34	<option>, function 2	
			3	35	<option>, function 3	
8	1	4	0	36	<option>	
			1	37	<option>, function 1	
			2	38	<option>, function 2	
			3	39	<option>, function 3	
9	1	5	0	40	<option>	
			1	41	<option>, function 1	
			2	42	<option>, function 2	
			3	43	<option>, function 3	
10	1	6	0	44	<option>	
			1	45	<option>, function 1	
			2	46	<option>, function 2	
			3	47	<option>, function 3	
<i>PCI0 (bus=pci_0)</i>						
1	0	1	0	8	<option>	
			1	9	<option>, function 1	
			2	10	<option>, function 2	
			3	11	<option>, function 3	
2	0	2	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
3	0	3	1	16	<option>	
			1	17	<option>, function 1	

			2	18	<option>, function 2	
			3	19	<option>, function 3	
4	0	4	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	
-	0	5	0	-	ALi M1543C PCI ISA bridge	
-	0	15	0	-	ALi M1543C PCI ISA bridge	DQA, DQB
-	0	19	0	-	ALi M1543C PCI USB adapter	

The IRQ stands for bit position in DRIR of Tsunami/Typhoon chip. It has nothing to do with "ISA" style interrupts which are routed to IRQ 55 (including ALi M1543C PCI IDE/ATAPI controller).

So far the CHARON-AXP emulators do not support virtual ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 remains empty.

Example 1: Loading DE500BA into slot 5

```
load DE500BA/dec21x4x EWA bus=pci_1 device=1 function=0
```

Example 2: Loading multiple DE500BA's into slot 5, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_1 device=1 function=0
load DE500BA/dec21x4x EWB bus=pci_1 device=1 function=1
load DE500BA/dec21x4x EWC bus=pci_1 device=1 function=2
load DE500BA/dec21x4x EWD bus=pci_1 device=1 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKA bus=pci_0 device=1 function=0
load DE500BA/dec21x4x EWA bus=pci_0 device=1 function=1
```

AlphaServer ES45 (10 PCI slots)

In addition to 10 PCI vacant slots there are 3 PCI positions occupied by on-board devices. All 13 PCI positions are listed in the following table in the order in which Alpha SRM console enumerates them.

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>PCI0 (bus=pci_0)</i>						
-	0	7	0	-	ALi M1543C PCI ISA bridge	
1	0	8	0	20	<option>	
			1	21	<option>, function 1	
			2	22	<option>, function 2	
			3	23	<option>, function 3	
2	0	9	0	24	<option>	
			1	25	<option>, function 1	
			2	26	<option>, function 2	
			3	27	<option>, function 3	
3	0	10	0	12	<option>	
			1	13	<option>, function 1	
			2	14	<option>, function 2	
			3	15	<option>, function 3	
4	0	11	0	16	<option>	
			1	17	<option>, function 1	
			2	18	<option>, function 2	
			3	19	<option>, function 3	
-	0	16	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA, DQB
-	0	19	0	-	ALi M1543C PCI USB adapter	
<i>PCI1 (bus=pci_1)</i>						
5	1	1	0	28	<option>	
			1	29	<option>, function 1	
			2	30	<option>, function 2	
			3	31	<option>, function 3	
6	1	2	0	32	<option>	
			1	33	<option>, function 1	
			2	34	<option>, function 2	
			3	35	<option>, function 3	
<i>PCI2 (bus=pci_2)</i>						
7	2	1	0	0	<option>	
			1	1	<option>, function 1	
			2	2	<option>, function 2	
			3	3	<option>, function 3	
8	2	2	0	4	<option>	
			1	5	<option>, function 1	

			2	6	<option>, function 2	
			3	7	<option>, function 3	
PCI3 (bus=pci_3)						
9	3	1	0	36	<option>	
			1	37	<option>, function 1	
			2	38	<option>, function 2	
			3	39	<option>, function 3	
10	3	2	0	40	<option>	
			1	41	<option>, function 1	
			2	42	<option>, function 2	
			3	43	<option>, function 3	

The IRQ stands for bit position in DRIR of TITAN chip. It has nothing to do with "ISA" style interrupts which are routed to IRQ 55 (including ALi M1543C PCI IDE/ATAPI controller).

So far the CHARON-AXP emulators do not support virtual ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 remains empty.

Example 1: Loading DE500BA into slot 5

```
load DE500BA/dec21x4x EWA bus=pci_1 device=1 function=0
```

Example 2: Loading multiple DE500BA's into slot 5, populating all 4 functions (gives 4 Ethernet ports)

```
load DE500BA/dec21x4x EWA bus=pci_1 device=1 function=0
load DE500BA/dec21x4x EWB bus=pci_1 device=1 function=1
load DE500BA/dec21x4x EWC bus=pci_1 device=1 function=2
load DE500BA/dec21x4x EWD bus=pci_1 device=1 function=3
```

Example 3: Loading mixture of KZPBA and DE500BA into slot 1, populating 2 functions out of 4

```
load KZPBA PKA bus=pci_0 device=8 function=0
load DE500BA/dec21x4x EWA bus=pci_0 device=8 function=1
```

AlphaServer GS80 (8 PCI busses)

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
QBB0.PCA0.PCI0 (bus=qbb_0_pca_0_pci_0)						
0/1	0	1	0	36	QLOGIC ISP1040B PCI SCSI Adapter	PKA
2	0	2	0	40	<option>	
3	0	3	0	44	<option>	
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	15	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA
-	0	19	0	-	ALi M1543C PCI USB adapter	
QBB0.PCA0.PCI1 (bus=qbb_0_pca_0_pci_1)						
4	1	4	0	48	<option>	
5	1	5	0	52	<option>	
6	1	6	0	56	<option>	
7	1	7	0	60	<option>	
QBB0.PCA1.PCI0 (bus=qbb_0_pca_1_pci_0)						
0/1	2	0	0	32	<option>	

2	2	2	0	40	<option>	
3	2	3	0	44	<option>	
QBB0.PCA1.PCI1 (bus=qbb_0_pca_1_pci_1)						
4	3	4	0	48	<option>	
5	3	5	0	52	<option>	
6	3	6	0	56	<option>	
7	3	7	0	60	<option>	
QBB1.PCA0.PCI0 (bus=qbb_1_pca_0_pci_0)						
0/1	8	0	0	32	<option>	
2	8	2	0	40	<option>	
3	8	3	0	44	<option>	
QBB1.PCA0.PCI1 (bus=qbb_1_pca_0_pci_1)						
4	9	4	0	48	<option>	
5	9	5	0	52	<option>	
6	9	6	0	56	<option>	
7	9	7	0	60	<option>	
QBB1.PCA1.PCI0 (bus=qbb_1_pca_1_pci_0)						
0/1	10	0	0	32	<option>	
2	10	2	0	40	<option>	
3	10	3	0	44	<option>	
QBB1.PCA1.PCI1 (bus=qbb_1_pca_1_pci_1)						
4	11	4	0	48	<option>	
5	11	5	0	52	<option>	
6	11	6	0	56	<option>	
7	11	7	0	60	<option>	

PCI 2 and 3 on each QBB are not populated.

So far the CHARON-AXP emulators do not support virtual ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 remains empty.

Total number of PCI devices configured through CFG file may not exceed 20.

Example: Loading DE500BA into slot 2 of QBB0.PCA0

```
load DE500BA/dec21x4x EWA bus=qbb_0_pca_0_pci_0 device=2 function=0
```

AlphaServer GS160 (16 PCI busses)

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>QBB0.PCA0.PCI0 (bus=qbb_0_pca_0_pci_0)</i>						
0/1	0	1	0	36	QLOGIC ISP1040B PCI SCSI Adapter	PKA
2	0	2	0	40	<option>	
3	0	3	0	44	<option>	
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	15	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA
-	0	19	0	-	ALi M1543C PCI USB adapter	
<i>QBB0.PCA0.PCI1 (bus=qbb_0_pca_0_pci_1)</i>						
4	1	4	0	48	<option>	
5	1	5	0	52	<option>	
6	1	6	0	56	<option>	
7	1	7	0	60	<option>	
<i>QBB0.PCA1.PCI0 (bus=qbb_0_pca_1_pci_0)</i>						
0/1	2	0	0	32	<option>	
2	2	2	0	40	<option>	
3	2	3	0	44	<option>	
<i>QBB0.PCA1.PCI1 (bus=qbb_0_pca_1_pci_1)</i>						
4	3	4	0	48	<option>	
5	3	5	0	52	<option>	
6	3	6	0	56	<option>	
7	3	7	0	60	<option>	
<i>QBB1.PCA0.PCI0 (bus=qbb_1_pca_0_pci_0)</i>						
0/1	8	0	0	32	<option>	
2	8	2	0	40	<option>	
3	8	3	0	44	<option>	
<i>QBB1.PCA0.PCI1 (bus=qbb_1_pca_0_pci_1)</i>						
4	9	4	0	48	<option>	
5	9	5	0	52	<option>	
6	9	6	0	56	<option>	
7	9	7	0	60	<option>	
<i>QBB1.PCA1.PCI0 (bus=qbb_1_pca_1_pci_0)</i>						
0/1	10	0	0	32	<option>	
2	10	2	0	40	<option>	
3	10	3	0	44	<option>	
<i>QBB1.PCA1.PCI1 (bus=qbb_1_pca_1_pci_1)</i>						
4	11	4	0	48	<option>	
5	11	5	0	52	<option>	
6	11	6	0	56	<option>	

7	11	7	0	60	<option>	
<i>QBB2.PCA0.PCI0 (bus=qbb_2_pca_0_pci_0)</i>						
0/1	16	0	0	32	<option>	
2	16	2	0	40	<option>	
3	16	3	0	44	<option>	
<i>QBB2.PCA0.PCI1 (bus=qbb_2_pca_0_pci_1)</i>						
4	17	4	0	48	<option>	
5	17	5	0	52	<option>	
6	17	6	0	56	<option>	
7	17	7	0	60	<option>	
<i>QBB2.PCA1.PCI0 (bus=qbb_2_pca_1_pci_0)</i>						
0/1	18	0	0	32	<option>	
2	18	2	0	40	<option>	
3	18	3	0	44	<option>	
<i>QBB2.PCA1.PCI1 (bus=qbb_2_pca_1_pci_1)</i>						
4	19	4	0	48	<option>	
5	19	5	0	52	<option>	
6	19	6	0	56	<option>	
7	19	7	0	60	<option>	
<i>QBB3.PCA0.PCI0 (bus=qbb_3_pca_0_pci_0)</i>						
0/1	24	0	0	32	<option>	
2	24	2	0	40	<option>	
3	24	3	0	44	<option>	
<i>QBB3.PCA0.PCI1 (bus=qbb_3_pca_0_pci_1)</i>						
4	25	4	0	48	<option>	
5	25	5	0	52	<option>	
6	25	6	0	56	<option>	
7	25	7	0	60	<option>	
<i>QBB3.PCA1.PCI0 (bus=qbb_3_pca_1_pci_0)</i>						
0/1	26	0	0	32	<option>	
2	26	2	0	40	<option>	
3	26	3	0	44	<option>	
<i>QBB0.PCA0.PCI1 (bus=qbb_3_pca_1_pci_1)</i>						
4	27	4	0	48	<option>	
5	27	5	0	52	<option>	
6	27	6	0	56	<option>	
7	27	7	0	60	<option>	

PCA 2 and 3 on each QBB are not populated in emulator.

So far the CHARON-AXP emulators do not emulate ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 on QBB 0 remains empty.

Total number of PCI devices configured through CFG file may not exceed 20.

Example: Loading DE500BA into slot 2 of QBB0.PCA0

```
load DE500BA/dec21x4x EWA bus=qbb_0_pca_0_pci_0 device=2 function=0
```

AlphaServer GS320 (32 PCI busses)

Slot	pci_<N>	device	function	irq	Description	Preloaded Name
<i>QBB0.PCA0.PCI0 (bus=qbb_0_pca_0_pci_0)</i>						
0/1	0	1	0	36	QLOGIC ISP1040B PCI SCSI Adapter	PKA
2	0	2	0	40	<option>	
3	0	3	0	44	<option>	
-	0	7	0	-	ALi M1543C PCI ISA bridge	
-	0	15	0	-	ALi M1543C PCI IDE/ATAPI controller	DQA
-	0	19	0	-	ALi M1543C PCI USB adapter	
<i>QBB0.PCA0.PCI1 (bus=qbb_0_pca_0_pci_1)</i>						
4	1	4	0	48	<option>	
5	1	5	0	52	<option>	
6	1	6	0	56	<option>	
7	1	7	0	60	<option>	
<i>QBB0.PCA1.PCI0 (bus=qbb_0_pca_1_pci_0)</i>						
0/1	2	0	0	32	<option>	
2	2	2	0	40	<option>	
3	2	3	0	44	<option>	
<i>QBB0.PCA1.PCI1 (bus=qbb_0_pca_1_pci_1)</i>						
4	3	4	0	48	<option>	
5	3	5	0	52	<option>	
6	3	6	0	56	<option>	
7	3	7	0	60	<option>	
<i>QBB1.PCA0.PCI0 (bus=qbb_1_pca_0_pci_0)</i>						
0/1	8	0	0	32	<option>	
2	8	2	0	40	<option>	
3	8	3	0	44	<option>	
<i>QBB1.PCA0.PCI1 (bus=qbb_1_pca_0_pci_1)</i>						
4	9	4	0	48	<option>	
5	9	5	0	52	<option>	
6	9	6	0	56	<option>	
7	9	7	0	60	<option>	
<i>QBB1.PCA1.PCI0 (bus=qbb_1_pca_1_pci_0)</i>						
0/1	10	0	0	32	<option>	
2	10	2	0	40	<option>	
3	10	3	0	44	<option>	
<i>QBB1.PCA1.PCI1 (bus=qbb_1_pca_1_pci_1)</i>						

4	11	4	0	48	<option>	
5	11	5	0	52	<option>	
6	11	6	0	56	<option>	
7	11	7	0	60	<option>	
QBB2.PCA0.PCI0 (bus=qbb_2_pca_0_pci_0)						
0/1	16	0	0	32	<option>	
2	16	2	0	40	<option>	
3	16	3	0	44	<option>	
QBB2.PCA0.PCI1 (bus=qbb_2_pca_0_pci_1)						
4	17	4	0	48	<option>	
5	17	5	0	52	<option>	
6	17	6	0	56	<option>	
7	17	7	0	60	<option>	
QBB2.PCA1.PCI0 (bus=qbb_2_pca_1_pci_0)						
0/1	18	0	0	32	<option>	
2	18	2	0	40	<option>	
3	18	3	0	44	<option>	
QBB2.PCA1.PCI1 (bus=qbb_2_pca_1_pci_1)						
4	19	4	0	48	<option>	
5	19	5	0	52	<option>	
6	19	6	0	56	<option>	
7	19	7	0	60	<option>	
QBB3.PCA0.PCI0 (bus=qbb_3_pca_0_pci_0)						
0/1	24	0	0	32	<option>	
2	24	2	0	40	<option>	
3	24	3	0	44	<option>	
QBB3.PCA0.PCI1 (bus=qbb_3_pca_0_pci_1)						
4	25	4	0	48	<option>	
5	25	5	0	52	<option>	
6	25	6	0	56	<option>	
7	25	7	0	60	<option>	
QBB3.PCA1.PCI0 (bus=qbb_3_pca_1_pci_0)						
0/1	26	0	0	32	<option>	
2	26	2	0	40	<option>	
3	26	3	0	44	<option>	
QBB3.PCA1.PCI1 (bus=qbb_3_pca_1_pci_1)						
4	27	4	0	48	<option>	
5	27	5	0	52	<option>	
6	27	6	0	56	<option>	
7	27	7	0	60	<option>	
QBB4.PCA0.PCI0 (bus=qbb_4_pca_0_pci_0)						

0/1	32	0	0	32	<option>	
2	32	2	0	40	<option>	
3	32	3	0	44	<option>	
QBB4.PCA0.PCI1 (bus=qbb_4_pca_0_pci_1)						
4	33	4	0	48	<option>	
5	33	5	0	52	<option>	
6	33	6	0	56	<option>	
7	33	7	0	60	<option>	
QBB4.PCA1.PCI0 (bus=qbb_4_pca_1_pci_0)						
0/1	34	0	0	32	<option>	
2	34	2	0	40	<option>	
3	34	3	0	44	<option>	
QBB4.PCA1.PCI1 (bus=qbb_4_pca_1_pci_1)						
4	35	4	0	48	<option>	
5	35	5	0	52	<option>	
6	35	6	0	56	<option>	
7	35	7	0	60	<option>	
QBB5.PCA0.PCI0 (bus=qbb_5_pca_0_pci_0)						
0/1	40	0	0	32	<option>	
2	40	2	0	40	<option>	
3	40	3	0	44	<option>	
QBB5.PCA0.PCI1 (bus=qbb_5_pca_0_pci_1)						
4	41	4	0	48	<option>	
5	41	5	0	52	<option>	
6	41	6	0	56	<option>	
7	41	7	0	60	<option>	
QBB5.PCA1.PCI0 (bus=qbb_5_pca_1_pci_0)						
0/1	42	0	0	32	<option>	
2	42	2	0	40	<option>	
3	42	3	0	44	<option>	
QBB5.PCA1.PCI1 (bus=qbb_5_pca_1_pci_1)						
4	43	4	0	48	<option>	
5	43	5	0	52	<option>	
6	43	6	0	56	<option>	
7	43	7	0	60	<option>	
QBB6.PCA0.PCI0 (bus=qbb_6_pca_0_pci_0)						
0/1	48	0	0	32	<option>	
2	48	2	0	40	<option>	
3	48	3	0	44	<option>	
QBB6.PCA0.PCI1 (bus=qbb_6_pca_0_pci_1)						
4	49	4	0	48	<option>	

5	49	5	0	52	<option>	
6	49	6	0	56	<option>	
7	49	7	0	60	<option>	
QBB6.PCA1.PCI0 (bus=qbb_6_pca_1_pci_0)						
0/1	50	0	0	32	<option>	
2	50	2	0	40	<option>	
3	50	3	0	44	<option>	
QBB6.PCA1.PCI1 (bus=qbb_6_pca_1_pci_1)						
4	51	4	0	48	<option>	
5	51	5	0	52	<option>	
6	51	6	0	56	<option>	
7	51	7	0	60	<option>	
QBB7.PCA0.PCI0 (bus=qbb_7_pca_0_pci_0)						
0/1	56	0	0	32	<option>	
2	56	2	0	40	<option>	
3	56	3	0	44	<option>	
QBB7.PCA0.PCI1 (bus=qbb_7_pca_0_pci_1)						
4	57	4	0	48	<option>	
5	57	5	0	52	<option>	
6	57	6	0	56	<option>	
7	57	7	0	60	<option>	
QBB7.PCA1.PCI0 (bus=qbb_7_pca_1_pci_0)						
0/1	58	0	0	32	<option>	
2	58	2	0	40	<option>	
3	58	3	0	44	<option>	
QBB7.PCA1.PCI1 (bus=qbb_7_pca_1_pci_1)						
4	59	4	0	48	<option>	
5	59	5	0	52	<option>	
6	59	6	0	56	<option>	
7	59	7	0	60	<option>	

PCA 2 and 3 on each QBB are not populated in emulator.

So far the MSC/AXP emulators do not emulate ALi M1543C PCI USB adapter. So position of the device 19, function 0 on the PCI 0 on QBB 0 remains empty.

Total number of PCI devices configured through CFG file may not exceed 20.

Example: Loading DE500BA into slot 2 of QBB0.PCA0

```
load DE500BA/dec21x4x EWA bus=qbb_0_pca_0_pci_0 device=2 function=0
```


Disks and tapes

Contents

- KZPBA PCI SCSI adapter
- KGPSA-CA PCI Fibre Channel adapter
- Acer Labs 1543C IDE/ATAPI CD-ROM adapter
- PCI I/O Bypass controller
- Finding the target `/dev/sg` device

KZPBA PCI SCSI adapter

Table of Contents

- General description
- Loading KZPBA storage adapter
- Configuration parameters
 - `scsi_id`
 - `host, port`
 - `container`
 - `media_type`
 - `removable`
 - `geometry`
 - `use_io_file_buffering`
 - `io_queue_depth`
 - `min_n_of_threads`
 - Dismountable disks/tapes

General description

KZPBA is a PCI SCSI adapter based on the QLogic ISP1040 Fast Wide SCSI adapter chip for Alpha.

In CHARON-AXP environment it supports up to 120 disks and tapes.

For systems with more than 16 heavily used units it is recommended to configure several virtual KZPBA PCI SCSI adapters and distribute the heavily loaded units evenly between the adapters.

Loading KZPBA storage adapter

Syntax for loading KZPBA storage adapter:

```
load KZPBA <controller name>
```

Example:

```
load KZPBA PKA
```

Please note:

In AlphaStation 400 configuration use the following syntax for KZPBA storage adapter loading:

```
load KZPBA PKB irq_bus=isa
```

The adapter instance name ("PKA" in the example above) is used then for parametrization, for example:

```
set PKA container[602]="/Mydisks/vms_distributive.vdisk"
```

The numbers in the square brackets represent SCSI ID and LUN of the devices on the virtual KZPBA SCSI bus.

They have the following format: **XXYY**, where:

Parameter	Range	Description
XX	0..15	SCSI ID
YY	00..07	LUN

By default KZPBA adapter uses first available PCI slot. If instead some particular slot is needed, refer to [this section](#) for details of specific placement of PCI peripherals on CHARON-AXP PCI bus.

By default each loaded KZPBA SCSI PCI adapter has SCSI ID=7. This setting can be changed with "scsi_id" parameter, for example:

```
set PKA scsi_id=0
```

CHARON-AXP HP Alpha models may have one or two KZPBA adapters preloaded.

Configuration parameters

The KZPBA PCI SCSI adapter emulation has the following configuration parameters:

scsi_id

Parameter	scsi_id
Type	Numeric
Value	Specifies SCSI ID of KZPBA PCI SCSI Adapter in a range 0..7 By default the "scsi_id" configuration parameter is set to 7. Example: set PKA scsi_id=6

host, port

Parameter	host, port
Type	Text String
Value	<p>These parameters are used in SCSI cluster configurations:</p> <ul style="list-style-type: none"> ■ host : specifies remote end-point (remote host name and, optionally, TCP/IP port on remote host) of SCSI connection between this KZPBA PCI SCSI adapter and remote KZPBA PCI SCSI adapter on some host. ■ port : specifies local end-point (TCP/IP port on local host) of SCSI connection between this KZPBA PCI SCSI adapter and remote KZPBA PCI SCSI adapter on some host. <p>By default the "host" and "port" configuration options are not specified.</p> <p>Syntax:</p> <pre>port[connection-number]=<local port> host[connection-number]="<host-name{:tcpip-port-no}>"</pre> <p>Where: connection-number = remote_scsi_id * 100 + lun_id</p> <p>Example - 2 members cluster on the same Charon server - Node 1:</p> <pre>set PKA scsi_id=6 set PKA port[700]=11067 host[700]="localhost:11076"</pre> <p>Example - 2 members cluster on the same Charon server - Node 2:</p> <pre>set PKA scsi_id=7 set PKA port[600]=11076 host[600]="localhost:11067"</pre>

container

Parameter	<p>container[N]</p> <p>N is "XXYY" number, where:</p> <ul style="list-style-type: none"> ■ XX - SCSI ID (0..15) ■ YY - LUN (00..07)
Type	Text String
Value	<p>Possible values of the parameter are strings in one of the following forms:</p> <ul style="list-style-type: none"> ■ Physical disk <ul style="list-style-type: none"> ■ <code>/dev/sd<L></code> where "L" is letter, for example <code>/dev/sdb</code> ■ <code>/dev/disk/by-id/...</code> - addressing by the disk ID, for example <code>/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4</code> ■ <code>/dev/disk/by-label/...</code> - addressing by the disk label, for example <code>/dev/disk/by-label/MyStorage</code> ■ <code>/dev/disk/by-uuid/...</code> - addressing by the disk UUID, for example <code>/dev/disk/by-uuid/0e808a2f-cdd3-4944-a245-f729ffd73882</code> Please note that existing data on such a disk may be destroyed, depending on how it is used in the emulator. <p><code>/dev/sd<L></code> addressing is not persistent, so it is strongly recommended to use <code>/dev/disk/by-id/wwn-*</code> syntax instead to refer the disk by its WWID - especially in the environments utilizing FC and SAN storages.</p> <p>These disks must not be formatted by the host OS.</p> <p>Example:</p> <pre>set PKA container[0]="/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4"</pre> <p>It is also possible to use not a whole disk, but previously created partitions on it. In this case the syntax is the following: <code>/dev/sd<L><N></code> where N is the number of partition to be used.</p> <p>Example:</p> <pre>set PKA container[0]="/dev/sdc1"</pre> ■ Multipath disk <ul style="list-style-type: none"> ■ <code>/dev/dm-<N></code> ■ <code>/dev/mapper/mpath<N></code> ■ <code>/dev/mapper/disk<N></code> <p>Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake.</p> <p>These disks must not be formatted by the host OS.</p> <p>Example:</p> <pre>set PKA container[100]="/dev/dm-0"</pre> ■ Loop (virtual block) devices <ul style="list-style-type: none"> ■ <code>/dev/loop<N></code> <p>Example:</p> <pre>set PKA container[200]="/dev/loop0"</pre> ■ Direct mapping to some SCSI device, for example, a SCSI disk, tape reader or tape changer <ul style="list-style-type: none"> ■ <code>/dev/sg<N></code> (do not use <code>/dev/st<N></code> devices, only <code>/dev/sg<N></code> for tape drives) <p>Example:</p> <pre>set PKA container[300]="/dev/sg0"</pre> ■ CD-ROM device <ul style="list-style-type: none"> ■ <code>/dev/sr<N></code> ■ <code>/dev/cdrom</code> ■ <code>/dev/cdrom<N></code>

Example:

```
set PKA container[400]="/dev/sr0"
```

- **ISO file for reading distribution CD-ROM image**

- `[<path-name>]/<file-name>[".iso"]`

Mapping may also include the full path (recommended), for example: `"/my_disks/vms_distributive.iso"`

Example:

```
set PKA container[600]="/my_disks/vms_distributive.iso"
```

- **File representing a physical disk of the HP Alpha system (disk image)**

- `[<path-name>]/<file-name>[".vdisk"]`

These files can be created from scratch with "mkdiskcmd" utility. Data and OS disk backups are transferred from the original system via tapes or network and restored into these container files.

Mapping may also include the full path, for example: `"/my_disks/my_boot_disk.vdisk"`

Example:

```
set PKA container[401]="my_dka401.vdisk"
```

- **File representing the tape (tape image)**

- `[<path-name>]/<path-name>[""]<file-name>".vtape"`

These files are created automatically.

Mapping may also include a full path (recommended), for example: `"/my_tapes/backup.vtape"`

If the "[CHARON Guest Utilities for OpenVMS](#)" (CHARONCP) package is used the syntax is different. Please read the corresponding chapter.

Example:

```
set PKA container[500]="my_mka500.vtape"
```

How the Emulator Maps Guest-OS Operations to the Virtual Tape Drive:

Guest-OS operations	Emulator Action
Open device for reading	Create a container file if none exists. open for reading and lock container file
Open device for writing	Create a container file if necessary; open for writing and lock the container file
Unload (eject) tape from drive	Close a container file if open and unlock it - this allows copy/move/delete operations on CHARON host

Please note:

The container file associated with a virtual tape drive can be compared to the tape cartridge used in a physical tape drive. Both store the data written to the tape device by the guest OS.

The size of virtual tape container files is limited only by space available in the emulator host file system.

Prerequisite to the examples below: a virtual tape device has been configured in the CHARON configuration file and it is not in use by the guest OS.

To perform backup:

1. The tape device may be issued the "unload" command and the container-file moved/deleted to insure proper status
2. Initialize the tape device using standard guest OS procedure.
3. Perform backup.
4. Issue "unload" command to the tape device in the guest OS.
5. On the emulator host, move the *.vtape container file containing backup data for storage or further backup.

To restore from a backup:

1. The tape device may be issued the "unload" command to insure proper status.
2. On the emulator host, move or copy a *.vtape container file containing backup data onto the filename specified in the CHARON configuration file.
3. Perform restore.
4. Issue the "unload" command to the tape device in the guest OS.
5. Delete or move the container file in preparation for the next vtape operation.

CHARON does not support multi-volume backup for tape images. If some multi-volume set (in form of tape images) has to be restored it is recommended to configure several tape drives in CHARON configuration file, assign each tape image to each tape drive and use them in the following way (OpenVMS example):

```
$ BACKUP MKA100:BACKUP.BCK,MKA200,MKA300,MKA4000/SAVE_SET DKA0:...
```

This parameter is initially not set, thus creating NO storage elements on the controller.

media_type

Parameter	<p>media_type[N]</p> <p>N is "XXYY" number, where:</p> <ul style="list-style-type: none"> ■ XX - SCSI ID (0..15) ■ YY - LUN (00..07)
Type	Text String
Value	<p>Instructs CHARON-AXP to use the supplied value as the PRODUCT field in the SCSI INQUIRY data returned to a software running on virtual HP Alpha system in response to SCSI INQUIRY command.</p> <p>If not specified, CHARON-AXP attempts to guess the SCSI INQUIRY data based on virtual SCSI device type and underlying container (which is specified in the corresponding container configuration parameter).</p> <p>Initially is not specified.</p> <p>Example:</p> <pre>set PKA media_type[0]="HSZ70"</pre>

removable

Parameter	<p>removable[N]</p> <p>N is "XXYY" number, where:</p> <ul style="list-style-type: none"> ■ XX - SCSI ID (0..15) ■ YY - LUN (00..07)
Type	Boolean
Value	<p>When set to "true", the removable configuration parameter instructs CHARON-AXP to report the corresponding virtual SCSI device as removable.</p> <p>By default the removable configuration parameter is set to "false".</p> <p>Example:</p> <pre>set PKA removable[400]=true</pre> <p>Note that virtual SCSI tapes and CD-ROM devices are always reported as removable regardless of the "removable" configuration parameter.</p>

geometry

Parameter	<p>geometry[N]</p> <p>N is "XXYY" number, where:</p> <ul style="list-style-type: none"> ■ XX - SCSI ID (0..15) ■ YY - LUN (00..07)
Type	Text String
Value	<p>This formatted string value specifies the explicit geometry of the disk storage element. This parameter is not applicable to tape storage elements.</p> <p>The string format is <X>"/"<Y>["/"<Z>] or <X>"/"<Y>["/"<Z>]["/"] where:</p> <ul style="list-style-type: none"> ■ X corresponds to the number of sectors per track ■ Y corresponds to the number of tracks per cylinder ■ Z corresponds to the number of cylinders on the unit. If omitted, Z is calculated based on X, Y and the total number of sectors on the unit that reflects the size of the disk storage element. This is an optional parameter. ■ B corresponds to the total size of the disk (in blocks) reported to the guest OS. If omitted it is calculated automatically. This is an optional parameter. <p>If this parameter is not set, CHARON-AXP will configure the geometry based on the most probable disk type.</p> <p>Initially not set.</p> <p>Please note:</p> <p style="padding-left: 40px;">It is possible to specify each parameter independently of another one. The following syntax is used for that:</p> <pre style="padding-left: 40px;">set PKA geometry[300]="*,*,*,16777210"</pre> <p>The syntax described above is applicable only to disk storage elements. If the container is a tape image, the following format is used instead:</p> <p>Syntax:</p> <pre>"<image-size>[, <early-warning-zone-size>]"</pre> <p>where:</p> <ul style="list-style-type: none"> ■ <code>image-size</code> corresponds to the tape size in MB ■ <code>early-warning-zone-size</code> corresponds to the size (in KB) of the space left on the tape when a warning to the OS is issued. If omitted, 64K is assumed. <p>Example:</p> <pre>set PKA geometry[603] = "255/255"</pre>

use_io_file_buffering

Parameter	<p>use_io_file_buffering[N]</p> <p>N is "XXYY" number, where:</p> <ul style="list-style-type: none"> ■ XX - SCSI ID (0..15) ■ YY - LUN (00..07)
Type	Boolean
Value	<p>When set to "true", instructs CHARON-AXP to enable host operating system I/O cache on read/write operations.</p> <p>Note that this caching has a significant effect only in case of mapping to disk and tape containers, not physical drives.</p> <p>When enabled, host operating system I/O cache may significantly improve I/O performance of the virtual system. At the same time maintaining I/O cache requires additional host resources (CPU and memory) which may negatively affect overall performance of the virtual system.</p> <p>Initially is set to "false".</p> <p>Example:</p> <pre>set PKA use_io_file_buffering[603]=true</pre>

io_queue_depth

Parameter	<p>io_queue_depth[N]</p> <p>N is "XXYY" number, where:</p> <ul style="list-style-type: none"> ● XX - SCSI ID (0..15) ● YY - LUN (00..07)
Type	Numeric
Value	<p>Specifies KZPBA I/O requests (read or write) for a given unit in a range 2..128</p> <p>Setting this parameter enables KZPBA instance to run up-to the specified numbers of I/O requests (read or write) for unit N in parallel, thus improving the performance.</p> <p>The default value set by controller is optimal for most of the cases. It may be needed to enlarge this number if guest OS I/O queue for a certain unit contains too many pending entries. In this case the value should be equal to an average size of the queue, collected over time.</p> <p>Please do not set this parameter without clear understanding of the purpose.</p> <p>By default parallel execution of I/O requests is disabled.</p> <p>Example:</p> <pre>set PKA io_queue_depth[603]=4</pre>

min_n_of_threads

Parameter	min_n_of_threads
Type	Numeric
Value	<p>Instructs KZPBA I/O to reserve a given number of working threads in a range 1..64, thus improving the performance.</p> <p>All units of KZPBA instance share the I/O threads.</p> <p>The default value is equal to number of units plus 2.</p> <p>For optimization it is possible to set this parameter to sum of the "io_queue_depth" parameters for each unit plus 2. This assumption seems optimal for most of the cases.</p> <p>Please do not set this parameter without clear understanding the purpose.</p> <p>Example:</p> <pre>set PKA min_n_of_threads=16</pre>

Dismountable disks/tapes

When a dismountable tape or disk image connected to an emulated KZPBA controller is dismounted by OpenVMS, it is disconnected from CHARON-AXP and can be manipulated. It can be replaced with a different disk image if it keeps the same name. This capability may be useful when designing back-up and restore procedures. When copying CHARON-AXP disk images while CHARON-AXP is running, please take care to minimize the risk of overloading a heavily loaded CHARON-AXP host system. For example, using a sequential series of simple ftp binary copies is less resource intensive and thus less disruptive than multiple, simultaneous copies.

Empty disk images are created with the "mkdiskcmd" utility. Tape images (*.vtape) will be created automatically if they don't exist (no utility needed).

CHARON-AXP is able to boot from disk images of any OpenVMS/Alpha and Tru64 version.

The virtual KZPBA storage controller examines the file extension (vdisk or vtape) to distinguish between a disk image and a tape image.

Configured physical devices or tape/disk images that do not exist on the host system will, in general, cause OpenVMS/Alpha to report the unit offline. In some cases this will result in a VMS BUG CHECK. In this case, an error message will be written to the log file.

KGPSA-CA PCI Fibre Channel adapter

Table of Contents

- General description
- Loading KGPSA storage adapter
- Configuration parameters
 - host_bus_location
 - wwid
 - container
 - media_type
 - removable
 - geometry
 - use_io_file_buffering
 - io_queue_depth
 - min_n_of_threads
- Mapping to host resources
 - Fabric virtualization mode
 - Pass Through mode
 - Pass Through mode establishing sequence
 - Building PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC
 - Installation of PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC
 - Adding PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC to Linux startup
 - Configuration of KGPSA-CA in pass through mode
 - FCMGR utility description
 - Configuration steps using FCMGR utility
 - Fabric presentation mode
 - FC HBA interfaces finding
 - List available adapters
 - Load HBA's kernel modules (drivers)
 - Discover the Linux SCSI hosts the FC HBAs are mapped to
 - Discover the connected FC HBAs (ports)
 - Discover the HBAs port/node names
 - Charon Configuration
 - By port name
 - By host id name
 - Configuration parameters
 - interface
 - trace_level
 - io_model
 - port_name
 - node_name
 - link_up_delay
 - link_poll_period
 - emu_fabric
 - lun_exclude
 - lun_include
 - scsi_version
 - device_id
 - vendor_id
 - revision_id
 - scsi_serialize_io
 - scsi_max_queue_depth

General description

CHARON-AXP supports emulation of DEC-KGPSA-CA PCI Fibre Channel adapter.

Every instance of KGPSA-CA works in one of the two following modes:

- [KGPSA-CA PCI Fibre Channel adapter#Fabric virtualization mode](#) (creating virtual fabric in combination with virtual FC-3 Storage Controller). This is default mode.
- [KGPSA-CA PCI Fibre Channel adapter#Pass Through mode](#) (using a specific CHARON PCI Pass Through driver)
- [KGPSA-CA PCI Fibre Channel adapter#Fabric presentation mode](#) (using Linux FC HBA directly)

Loading KGPSA storage adapter

Syntax for loading KGPSA-CA storage adapter:

```
load KGPSA <name>
```

Example:

```
load KGPSA FGA
```

Please note:

In AlphaStation 400 configuration use the following syntax for KGPSA-CA storage adapter loading:

```
load KGPSA FGA irq_bus=isa
```

The adapter instance name ("FGA" in the example above) is used then for parametrization, for example:

```
set FGA container[100]="/my_disks/vms_distributive.vdisk"
```

Numbers in the square brackets represent KGPSA-CA units. They can be in the range 0..32766, but no more than 255 units can be configured on a single controller.

By default KGPSA-CA adapter uses first available PCI slot. If instead some particular slot is needed, refer to [this section](#) for details of specific placement of PCI peripherals on CHARON-AXP PCI bus.

Configuration parameters

The KGPSA-CA PCI FC adapter emulation has the following configuration parameters:

host_bus_location

Parameter	host_bus_location
Type	Text String
Value	<p>KGPSA-CA PCI Fibre Channel adapter in Pass Through mode only !</p> <p>Establish connection between virtual DEC-KGPSA-CA PCI FC adapter and physical EMULEX LightPulse PCI/PCI-X/PCle FC adapter (KGPSA-CA PCI Fibre Channel adapter in Pass Through mode)</p> <p>Syntax:</p> <pre>load KGPSA <controller name> host_bus_location="/dev/kgpsa<X>"</pre> <p>Example:</p> <pre>load KGPSA FGA host_bus_location="/dev/kgpsa0"</pre>

wwid

Parameter	<p>wwid[N]</p> <p>N is 0..32766 (no more than 255 units)</p>
Type	Text String
Value	<p>Sets WWID for emulated KGPSA adapter unit.</p> <p>Syntax:</p> <pre>set <controller name> wwid[unit-number]="XXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXX"</pre> <p>Example:</p> <pre>set FGA wwid[2]="6008-05F3-0005-2950-BF8E-0B86-A0C7-0001"</pre>

container

Parameter	container[N] N is 0..32766 (no more than 255 units)
Type	Text String
Value	<p>Possible values of the parameter are strings in one of the following forms:</p> <ul style="list-style-type: none"> ■ Physical disk <ul style="list-style-type: none"> ■ <code>"/dev/sd<L>"</code>, where "L" is letter, for example <code>"/dev/sdb"</code> ■ <code>"/dev/disk/by-id/..."</code> - addressing by the disk ID, for example <code>"/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4"</code> ■ <code>"/dev/disk/by-label/..."</code> - addressing by the disk label, for example <code>"/dev/disk/by-label/MyStorage"</code> ■ <code>"/dev/disk/by-uuid/..."</code> - addressing by the disk UUID, for example <code>"/dev/disk/by-uuid/0e808a2f-cdd3-4944-a245-f729ffd73882"</code> Please note that existing data on such a disk may be destroyed, depending on how it is used in the emulator. <p><code>"/dev/sd<L>"</code> addressing is not persistent, so it is strongly recommended to use <code>"/dev/disk/by-id/wwn-..."</code> syntax instead to refer the disk by its WWID - especially in the environments utilizing FC and SAN storages.</p> <p>These disks must not be formatted by the host OS.</p> <p>Example:</p> <pre>set FGA container[0]="/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4"</pre> <p>It is also possible to use not a whole disk, but previously created partitions on it. In this case the syntax is the following: <code>"/dev/sd<L><N>"</code> where N is the number of partition to be used.</p> <p>Example:</p> <pre>set FGA container[0]="/dev/sdc3"</pre> <ul style="list-style-type: none"> ■ Loop (virtual block) devices <ul style="list-style-type: none"> ● <code>"/dev/loop<N>"</code> <p>Example:</p> <pre>set FGA container[100]="/dev/loop0"</pre> ■ Multipath disk <ul style="list-style-type: none"> ■ <code>"/dev/dm-<N>"</code> ■ <code>"/dev/mapper/mpath<N>"</code> ■ <code>"/dev/mapper/disk<N>"</code> <p>Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake.</p> <p>These disks must not be formatted by the host OS.</p> <p>Example:</p> <pre>set FGA container[200]="/dev/dm-0"</pre> ■ File representing a physical disk of the Alpha system (disk image) <ul style="list-style-type: none"> ■ <code>["<drive>:\<path-name>\"]<file-name>[".vdisk"]</code> <p>These files can be created from scratch with "mkdiskcmd" utility. Data and OS disk backups are transferred from the original system via tapes or network and restored into these container files.</p> <p>Mapping may also include the full path (recommended), for example: <code>"/my_disks/my_boot_disk.vdisk"</code></p> <p>Example:</p> <pre>set FGA container[401]="my_dka401.vdisk"</pre> <p>This parameter is initially not set, thus creating NO storage elements on the controller.</p>

media_type

Parameter	media_type[N] N is 0..32766 (no more than 255 units)
Type	Text String
Value	<p>Instructs CHARON-AXP to use the supplied value as the PRODUCT field in the FC INQUIRY data returned to a software running on virtual Alpha system in response to FC INQUIRY command.</p> <p>If not specified, CHARON-AXP attempts to guess the FC INQUIRY data based on virtual FC device type and underlying container (which is specified in the corresponding container configuration parameter).</p> <p>Initially is not specified.</p> <p>Example:</p> <pre>set FGA media_type[0]="HSZ70"</pre>

removable

Parameter	removable[N] N is 0..32766 (no more than 255 units)
Type	Boolean
Value	<p>When set to "true", the removable configuration parameter instructs CHARON-AXP to report the corresponding virtual FC device as removable.</p> <p>By default the removable configuration parameter is set to "false".</p> <p>Example:</p> <pre>set FGA removable[400]=true</pre>

geometry

Parameter	geometry[N] N is 0..32766 (no more than 255 units)
Type	Text String
Value	<p>This formatted string value specifies the explicit geometry of the disk storage element.</p> <p>The string format is <X>"/<Y>["/<Z>] or <X>"/<Y>["/<Z>][["] where:</p> <ul style="list-style-type: none"> ■ X corresponds to the number of sectors per track ■ Y corresponds to the number of tracks per cylinder ■ Z corresponds the number of cylinders on the unit. If omitted, Z is calculated based on X, Y and the total number of sectors on the unit that reflects the size of the disk storage element. This is an optional parameter. ■ B corresponds to the total size of the disk (in blocks) reported to the guest OS. If omitted it is calculated automatically. This is an optional parameter. <p>If this parameter is not set, CHARON-AXP will configure the geometry based on the most probable disk type. Initially not set.</p> <p>Example:</p> <pre>set FGA geometry[201] = "255/255"</pre> <p>It is possible to specify each parameter independently of another one. The following syntax is used for that: <code>set FGA geometry[300] = "* , * , * , 16777210 "</code></p>

use_io_file_buffering

Parameter	use_io_file_buffering[N] N is 0..32766 (no more than 255 units)
Type	Boolean
Value	<p>When set to "true", instructs CHARON-AXP to enable host operating system I/O cache on read/write operations.</p> <p>Note that this caching has a significant effect only in case of mapping to disk containers, not physical drives.</p> <p>When enabled, host operating system I/O cache may significantly improve I/O performance of the virtual system. At the same time maintaining I/O cache requires additional host resources (CPU and memory) which may negatively affect overall performance of the virtual system.</p> <p>Initially is set to "false".</p> <p>Example:</p> <pre>set FGA use_io_file_buffering[300]=true</pre>

io_queue_depth

Parameter	io_queue_depth[N] N is 0..32766 (no more than 255 units)
Type	Numeric
Value	<p>Specifies KGPSA I/O requests (read or write) for a given unit in a range 2..128</p> <p>Setting this parameter enables KGPSA instance to run up-to the specified numbers of I/O requests (read or write) for unit N in parallel, thus improving the performance.</p> <p>The default value set by controller is optimal for most of the cases. It may be needed to enlarge this number if guest OS I/O queue for a certain unit contains too much pending entries. In this case the value should be equal to an average size of the queue, collected statistically.</p> <p>Please do not set this parameter without clear understanding of the purpose.</p> <p>By default parallel execution of I/O requests is disabled.</p> <p>Example:</p> <pre>set FGA io_queue_depth[603]=4</pre>

min_n_of_threads

Parameter	min_n_of_threads
Type	Numeric
Value	<p>Instructs KGPSA I/O to reserve a given number of working threads in a range 1..64, thus improving the performance.</p> <p>All units of KGPSA instance share the I/O threads.</p> <p>The default value is equal to number of units plus 2.</p> <p>For optimization it is possible to set this parameter to sum of the "io_queue_depth" parameters for each unit plus 2. This assumption seems optimal for most of the cases.</p> <p>Please do not set this parameter without clear understanding the purpose.</p> <p>Example:</p> <pre>set FGA min_n_of_threads=16</pre>

When a disk image connected to an emulated KGPSA-CA controller is dismounted by OpenVMS, it is disconnected from CHARON-AXP and can be manipulated. It can be replaced with a different disk image if it keeps the same name. This capability may be useful when designing back-up and restore procedures. When copying CHARON-AXP disk images while CHARON-AXP is running, please take care to minimize the risk of overloading a heavily loaded CHARON-AXP host system. For example, using a sequential series of simple ftp binary copies is less resource intensive and thus less disruptive than multiple, simultaneous copies.

Empty disk images are created with the "mkdiskcmd" utility.

CHARON-AXP is able to boot from disk images of any OpenVMS/Alpha and Tru64 version.

Mapping to host resources

Fabric virtualization mode

In this mode KGPSA-CA PCI FC adapter can be directly mapped to physical disks (both local and iSCSI) and disk images as shown in the following example:

```
set FGA container[0]="/my_disks/my_dka401.vdisk"
set FGA container[100]="/dev/sdb"
set FGA container[200]="/dev/sdc2"
set FGA container[300]="/dev/dm-0"
```

See the *KGPSA-CA PCI Fibre Channel adapter - Configuration parameters* section for details.

Pass Through mode

The CHARON PCI Pass Through mode allows connection between virtual DEC-KGPSA-CA PCI FC adapter and physical EMULEX LightPulse PCI/PCI-X /PCIe FC adapter plugged into host's PCI/PCI-X/PCIe slot.

Syntax:

```
load <controller name> host_bus_location="/dev/kgpsa<N>"
```

Example:

```
load KGPSA FGA host_bus_location="/dev/kgpsa0"
```

The following is a list of EMULEX LightPulse PCI/PCI-X/PCIe FC adapters supported by CHARON-AXP PCI Pass Through driver and suitable for emulation of KGPSA-CA PCI FC adapter in CHARON PCI Pass Through mode:

Supported	Not Supported	Not tested
LP8000 LP9000 LP9002 LP9802 LP10000 LP10000DC LP10000-S LPX1000 LP11002 LPe11002 (FC2242SR, A8003A) LPe1105 LPe12002 (AJ762B)	LPe1150 (FC2142SR, A8002A)	LPe11000

Pass Through mode establishing sequence

To establish "pass through" mode do the following:

1. Install the EMULEX LightPulse PCI/PCI-X/PCIe FC adapter (see above for a list of supported models) to some spare PCI/PCI-X/PCIe slot of the host system
2. Build PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC
3. Install PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC
4. Add PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC to Linux startup
5. Map KGPSA-CA adapter(-s) to EMULEX LightPulse PCI/PCI-X/PCIe FC adapter instance(-s) in CHARON-AXP configuration file

If kernel of the host system has been upgraded or reinstalled all the steps of the PPT KGPSA driver installation must be repeated

Building PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC

To build PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC do the following:

Step	Description
1	<p>Make sure that the required building tools and include files are installed. If they are absent install them:</p> <pre># yum groupinstall "Development Tools"</pre> <pre># yum install kernel kernel-headers kernel-devel</pre> <p>The kernel version must match the version of the installed kernel-headers (i.e. this packages must have same versions. It can be verified via "<code>rpm -q -a grep kernel-</code>")</p> <p>Check that the "kernel", the "kernel-devel" and the "kernel-headers" have the same version, and ensure that system is booted from this kernel version (not from some older one and etc) with "<code>uname -a</code>" command.</p>
2	<p>Open xterm and change the default directory to "<code>/opt/charon/drivers/kgpsa</code>":</p> <pre># cd /opt/charon/drivers/kgpsa</pre>
3	<p>Issue "make clean; make" commands to build kernel object:</p> <pre># make clean; make</pre> <p>It is prohibited to use a module built on a certain version of kernel on another one.</p>
4	<p>Check that there are no compilation errors and the file "<code>ppt_kgpsa.ko</code>" has been built</p>

Installation of PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC

To install PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC do the following:

Step	Description
1	Unload standard "lpfc" driver; to do that issue the following command: <pre># rmmod lpfc</pre>
2	Load "ppt_kgpsa.ko" driver; to do that issue the following command: <pre># insmod ppt_kgpsa.ko</pre>
3	Issue "dmesg" command and check that no error appeared during the driver loading, also check that the driver has found all KGPSA devices. Example: <pre># dmesg grep KGPSA; ls -la /dev/kgpsa* Found KGPSA with VENDOR_ID=10DF DEVICE_ID=FE00 Found KGPSA with VENDOR_ID=10DF DEVICE_ID=FE00 crw-----. 1 root root 240, 0 Nov 28 17:47 /dev/kgpsa0 crw-----. 1 root root 240, 1 Nov 28 17:47 /dev/kgpsa1</pre>

Adding PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC to Linux startup

To add PPT driver for EMULEX LightPulse PCI/PCI-X/PCIe FC to Linux startup do the following:

Step	Description
1	Disable auto-loading of Linux standard "lpfc" driver on boot. To do that add "blacklist lpfc" to the black list file "/etc/modprobe.d/blacklist.conf"
2	Copy the KGPSA-CA kernel module to the location of Linux kernel modules, for example: <pre># cp /opt/charon/drivers/kgpsa/ppt_kgpsa.ko /lib/modules/3.10.9-200.fc20.x86_64/kernel/drivers/scsi/</pre> The particular path may be different, depending on the kernel version and Linux distribution.
3	Enable auto load of the module: <pre># echo ppt_kgpsa > /etc/modules-load.d/ppt_kgpsa.conf</pre>
4	Regenerate new "initramfs" image with "mkinitrd": <pre># mkinitrd -f /boot/initramfs-3.10.9-200.fc20.x86_64.img 3.10.9-200.fc20.x86_64</pre> The particular path may be different, depending on the kernel version and Linux distribution.

Configuration of KGPSA-CA in pass through mode

FCMGR utility description

To configure KGPSA-CA adapter in pass through mode a special SRM console utility "FCMGR" is used (it has the same functionality as the "WWIDMGR" utility of the native Alpha hardware).

It provides the following functionality:

Command	Description
<code>fc rescan {/verbose}</code>	Scans connected SAN using FC adapters, discovers FC ports, storage controllers, logical units and then builds volatile FC database. The "/verbose" qualifier enables FC communication trace on console (for diagnostic and troubleshooting).
<code>fc show {adapter port device}</code>	Displays corresponding part of volatile FC database.
<code>fc set {boot dump} udid <X></code>	Fills the environment variables <code>wwid0..wwid3</code> and <code>n1..n4</code> to identify path(s) to logical unit with the specified UDID. These variables are later used by "INIT" to create device database entries and by OpenVMS/Tru64 to get access to boot and dump disks. This command does NOT make any change to other environment variables.
<code>fc set {boot dump} wwid <X></code>	Fills the environment variables <code>wwid0..wwid3</code> and <code>n1..n4</code> to identify path(s) to logical unit with the specified WWID. These variables are later used by "INIT" to create device database entries and by OpenVMS/Tru64 to get access to boot and dump disks. This command does NOT make any change to other environment variables. This parameter is useful if UDID is absent. Please note: Only right part of the displayed WWID is used for specification, for example: <pre>P00>>>fc res polling for units on kgpsa0, slot 2, bus 0, hose 0 ... pga0.0.0.2.0 PGA0 F/W Rev 2.72A2 WWN 1000-0000-0263-0040 WWN 2003-0060-0263-0040 fabric directory WWN 20fc-0060-0263-0040 port 020100 CED 8GSH 0 F88V WWN 5000-1fe0-0000-0bf1 DEC HSG80CCL V88F lun 0000000000000000 UDID:-1 WWID:01000010:6000-1fe0-0000-0bf0-3030-3030-373f- 3f3f lun 0000000000000100 UDID:-1 WWID:01000010:6000-1fe0-0000-0bf0-3030-3030-3030- 3030 P00>>>fc set boot wwid 6000-1fe0-0000-0bf0-3030-3030-3030-3030</pre>
<code>fc clear</code>	Clears environment variables <code>wwid0..wwid3</code> and <code>n1..n4</code> , which automatically disable (but do NOT delete) device database entries representing FC attached devices. This command does NOT affect volatile FC database.

Example of usage:

```
P00>>>fc show devices
  UDID:110 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0081 (ev:none)
    via adapter      via fc_port      con
[0]      pga0.0.0.5.1   5000-1fe1-000b-6bf1 yes (ev:none)
[1]      pga0.0.0.5.1   5000-1fe1-000b-6bf4 yes (ev:none)
  UDID:108 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039 (ev:none)
    via adapter      via fc_port      con
[12]     pga0.0.0.5.1   5000-1fe1-000b-6bf1 yes (ev:none)
[13]     pga0.0.0.5.1   5000-1fe1-000b-6bf4 yes (ev:none)
```

Configuration steps using FCMGR utility

Once the configuration steps described above are done, start the CHARON VM and wait for the **P00>>>** prompt.

Please refer to the following example with two FC adapters PGA and PGB defined:

```

initializing ...

polling for units on kzpba0, slot 4, bus 0, hose 0 ...
  pka0.0.0.4.0          PKA0          Q-Logic/ISP PCI SCSI HBA

polling for units on kgpsa0, slot 5, bus 0, hose 0 ...
  pga0.0.0.5.0          PGA0          WWN 1000-0000-C92E-97C9
    fabric              WWN 2003-0060-6920-4682
    directory          WWN 20fc-0060-6920-4682
    port 021400        WWN 5000-1fe1-000b-6bf1
      lun 000000000000100    DEC    HSG80          V88F
    UDID:100          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
      lun 000000000000200    DEC    HSG80          V88F
    UDID:200          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
      lun 000000000000300    DEC    HSG80          V88F
    UDID:300          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
      lun 000000000000400    DEC    HSG80          V88F
    UDID:400          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
      lun 0000000000006c00    DEC    HSG80          V88F
    UDID:108          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
      lun 0000000000006d00    DEC    HSG80          V88F
    UDID:208          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
      lun 0000000000000000

    port 021500          WWN 5000-1fe1-000b-6bf4
      lun 000000000000100    DEC    HSG80          V88F
    UDID:100          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
      lun 000000000000200    DEC    HSG80          V88F
    UDID:200          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
      lun 000000000000300    DEC    HSG80          V88F
    UDID:300          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
      lun 000000000000400    DEC    HSG80          V88F
    UDID:400          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
      lun 0000000000006c00    DEC    HSG80          V88F
    UDID:108          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
      lun 0000000000006d00    DEC    HSG80          V88F
    UDID:208          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
      lun 0000000000000000

polling for units on kgpsa1, slot 6, bus 0, hose 0 ...
  pgb0.0.0.6.0          PGB0          WWN 1000-0000-C92D-8D00
    fabric              WWN 2003-0060-6920-45ff
    directory          WWN 20fc-0060-6920-45ff
    port 011400        WWN 5000-1fe1-000b-6bf2
      lun 000000000000100    DEC    HSG80          V88F
    UDID:100          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
      lun 000000000000200    DEC    HSG80          V88F
    UDID:200          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
      lun 000000000000300    DEC    HSG80          V88F
    UDID:300          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
      lun 000000000000400    DEC    HSG80          V88F
    UDID:400          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
      lun 0000000000006c00    DEC    HSG80          V88F
    UDID:108          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
      lun 0000000000006d00    DEC    HSG80          V88F
    UDID:208          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
      lun 0000000000000000

    port 011500          WWN 5000-1fe1-000b-6bf3
      lun 000000000000100    DEC    HSG80          V88F
    UDID:100          WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
      lun 000000000000200    DEC    HSG80          V88F
    UDID:200          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
      lun 000000000000300    DEC    HSG80          V88F
    UDID:300          WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
      lun 000000000000400    DEC    HSG80          V88F

```

```
UDID:400 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
  lun 00000000000006c00          DEC    HSG80    V88F
UDID:108 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
  lun 00000000000006d00          DEC    HSG80    V88F
UDID:208 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
  lun 00000000000000000
port 011100                          failed port login
```

... enter console

CHARON-AXP (AlphaServer ES40) emulator. Version 4.11
Copyright (C) 2020, STROMASYS (www.stromasys.com)

P00>>>

The next step is to configure paths for the FC storage:

```
P00>>>fc show devices

    UDID:100 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038 (ev:none)
        via adapter      via fc_port      con
[0]      pga0.0.0.5.0      5000-1fe1-000b-6bf1 no (ev:none)
[1]      pga0.0.0.5.0      5000-1fe1-000b-6bf4 yes (ev:none)
[2]      pgb0.0.0.6.0      5000-1fe1-000b-6bf2 no (ev:none)
[3]      pgb0.0.0.6.0      5000-1fe1-000b-6bf3 yes (ev:none)
    UDID:200 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074 (ev:none)
        via adapter      via fc_port      con
[4]      pga0.0.0.5.0      5000-1fe1-000b-6bf1 no (ev:none)
[5]      pga0.0.0.5.0      5000-1fe1-000b-6bf4 yes (ev:none)
[6]      pgb0.0.0.6.0      5000-1fe1-000b-6bf2 no (ev:none)
[7]      pgb0.0.0.6.0      5000-1fe1-000b-6bf3 yes (ev:none)
    UDID:300 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b (ev:none)
        via adapter      via fc_port      con
[8]      pga0.0.0.5.0      5000-1fe1-000b-6bf1 no (ev:none)
[9]      pga0.0.0.5.0      5000-1fe1-000b-6bf4 yes (ev:none)
[10]     pgb0.0.0.6.0      5000-1fe1-000b-6bf2 no (ev:none)
[11]     pgb0.0.0.6.0      5000-1fe1-000b-6bf3 yes (ev:none)
    UDID:400 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080 (ev:none)
        via adapter      via fc_port      con
[12]     pga0.0.0.5.0      5000-1fe1-000b-6bf1 no (ev:none)
[13]     pga0.0.0.5.0      5000-1fe1-000b-6bf4 yes (ev:none)
[14]     pgb0.0.0.6.0      5000-1fe1-000b-6bf2 no (ev:none)
[15]     pgb0.0.0.6.0      5000-1fe1-000b-6bf3 yes (ev:none)
    UDID:108 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039 (ev:none)
        via adapter      via fc_port      con
[16]     pga0.0.0.5.0      5000-1fe1-000b-6bf1 yes (ev:none)
[17]     pga0.0.0.5.0      5000-1fe1-000b-6bf4 no (ev:none)
[18]     pgb0.0.0.6.0      5000-1fe1-000b-6bf2 yes (ev:none)
[19]     pgb0.0.0.6.0      5000-1fe1-000b-6bf3 no (ev:none)
    UDID:208 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a (ev:none)
        via adapter      via fc_port      con
[20]     pga0.0.0.5.0      5000-1fe1-000b-6bf1 yes (ev:none)
[21]     pga0.0.0.5.0      5000-1fe1-000b-6bf4 no (ev:none)
[22]     pgb0.0.0.6.0      5000-1fe1-000b-6bf2 yes (ev:none)
[23]     pgb0.0.0.6.0      5000-1fe1-000b-6bf3 no (ev:none)

P00>>>fc set boot udid 400

P00>>>INIT

initializing ...

polling for units on kzpba0, slot 4, bus 0, hose 0 ...
    pka0.0.0.4.0      PKA0      Q-Logic/ISP PCI SCSI HBA

polling for units on kgpsa0, slot 5, bus 0, hose 0 ...
    pga0.0.0.5.0      PGA0      WWN 1000-0000-C92E-97C9
    fabric            WWN 2003-0060-6920-4682
    directory         WWN 20fc-0060-6920-4682
    port 021400      WWN 5000-1fe1-000b-6bf1
        lun 0000000000000100      DEC      HSG80      V88F
    UDID:100      WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
        lun 0000000000000200      DEC      HSG80      V88F
    UDID:200      WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
        lun 0000000000000300      DEC      HSG80      V88F
    UDID:300      WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
        lun 0000000000000400      DEC      HSG80      V88F
    UDID:400      WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
        lun 00000000000006c00      DEC      HSG80      V88F
    UDID:108      WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
        lun 00000000000006d00      DEC      HSG80      V88F
    UDID:208      WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
        lun 0000000000000000
    port 021500      WWN 5000-1fe1-000b-6bf4
        lun 0000000000000100      DEC      HSG80      V88F
    UDID:100      WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
        lun 0000000000000200      DEC      HSG80      V88F
```



```

UDID:200      WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
  lun 0000000000000300      DEC      HSG80      V88F
UDID:300 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
  lun 0000000000000400      DEC      HSG80      V88F
UDID:400 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
  lun 00000000000006c00      DEC      HSG80      V88F
UDID:108 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
  lun 0000000000006d00      DEC      HSG80      V88F
UDID:208 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
  lun 0000000000000000

```

polling for units on kgpsal, slot 6, bus 0, hose 0 ...

```

pgb0.0.0.6.0      PGB0      WWN 1000-0000-C92D-8D00
  fabric
  directory      WWN 20f3c-0060-6920-45ff
  port 011400      WWN 5000-1fe1-000b-6bf2
    lun 0000000000000100      DEC      HSG80      V88F
UDID:100 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
  lun 0000000000000200      DEC      HSG80      V88F
UDID:200 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
  lun 0000000000000300      DEC      HSG80      V88F
UDID:300 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
  lun 0000000000000400      DEC      HSG80      V88F
UDID:400 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
  lun 00000000000006c00      DEC      HSG80      V88F
UDID:108 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
  lun 0000000000006d00      DEC      HSG80      V88F
UDID:208 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
  lun 0000000000000000
  port 011500      WWN 5000-1fe1-000b-6bf3
    lun 0000000000000100      DEC      HSG80      V88F
UDID:100 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0038
  lun 0000000000000200      DEC      HSG80      V88F
UDID:200 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0074
  lun 0000000000000300      DEC      HSG80      V88F
UDID:300 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-007b
  lun 0000000000000400      DEC      HSG80      V88F
UDID:400 WWID:01000010:6000-1fe1-000b-6bf0-0009-9081-1283-0080
  lun 00000000000006c00      DEC      HSG80      V88F
UDID:108 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-0039
  lun 0000000000006d00      DEC      HSG80      V88F
UDID:208 WWID:01000010:6000-1fe1-000b-6bf0-0009-0440-4014-003a
  lun 0000000000000000
  port 011100      failed port login

```

... enter console

CHARON-AXP (AlphaServer ES40) emulator. Version 4.11
 Copyright (C) 2020, STROMASYS (www.stromasys.com)

P00>>>SHOW DEV

```

sys0.0.0.0.0      SYS0      System ROOT Device
ewa0.0.0.3.0      EWA0      00-51-71-F5-8E-D8
pka0.0.0.4.0      PKA0      Q-Logic/ISP PCI SCSI HBA
pga0.0.0.5.0      PGA0      WWN 1000-0000-C92E-97C9
pgb0.0.0.6.0      PGB0      WWN 1000-0000-C92D-8D00
pqa0.0.0.15.0     PQA0      ALi 1553C Integrated IDE Controller
pqb0.0.1.15.0     PQB0      ALi 1553C Integrated IDE Controller
dqa0.0.0.15.0     DQA0      Virtual ATAPI - TEAC DW-224E-V
dka0.0.0.4.0      DKA0      Virtual SCSI Disk (C)SRI
dga400.1001.0.5.0  $1$DGA400  DEC      HSG80      V88F
dga400.1002.0.5.0  $1$DGA400  DEC      HSG80      V88F
dgb400.1003.0.6.0  $1$DGA400  DEC      HSG80      V88F
dgb400.1004.0.6.0  $1$DGA400  DEC      HSG80      V88F

```

P00>>>BOOT \$1\$DGA400

```

dga400.1001.0.5.0: failed to open device
(boot dga400.1002.0.5.0)
jumping to bootstrap code

```

OpenVMS (TM) Alpha Operating System, Version V7.3-2
 © Copyright 1976-2003 Hewlett-Packard Development Company, L.P.

```
%SMP-I-CPUTRN, CPU #02 has joined the active set.
%SMP-I-CPUTRN, CPU #03 has joined the active set.
%SMP-I-CPUTRN, CPU #01 has joined the active set.
Please enter date and time (DD-MMM-YYYY HH:MM)
```

Fabric presentation mode

The CHARON-AXP FC Fabric presentation mode allows to use Linux FC HBA directly. When using this mode, there is no need to load KGPSA adapter (s) as it was described before. The following syntax has to be used instead:

Example:

```
load kgpsa_generic_storage PGA interface="host3"
```

Below please find step-by-step explanation on how to configure "kgpsa_generic_storage" instance.

This mode is available only for CHARON-AXP on RHEL and CentOS 7.x and later versions.

FC HBA interfaces finding

First we need to find Linux FC HBA for mapping.

List available adapters

```
# lspci | grep Fibre
03:00.0 Fibre Channel: Emulex Corporation Lancer-X: LightPulse Fibre Channel
Host Adapter (rev 30)
03:00.1 Fibre Channel: Emulex Corporation Lancer-X: LightPulse Fibre Channel
Host Adapter (rev 30)
0a:04.0 Fibre Channel: Emulex Corporation LP8000 Fibre Channel Host Adapter (rev 02)
```

So, 2 HBAs, 3 ports are available.

Load HBA's kernel modules (drivers)

```
# lspci -vvnn -s 03:00.0 | grep modules
Kernel modules: lpfc

# modprobe -v lpfc
```

Discover the Linux SCSI hosts the FC HBAs are mapped to

```
# ll /sys/class/scsi_host/
..
lrwxrwxrwx 1 root root 0 &#1086;&#1082;&#1090; 22 14:29 host3 ->
../../../../devices/pci0000:00/0000:00:01.0/0000:03:00.0/host3/scsi_host/host3
lrwxrwxrwx 1 root root 0 &#1086;&#1082;&#1090; 22 14:29 host4 ->
../../../../devices/pci0000:00/0000:00:01.0/0000:03:00.1/host4/scsi_host/host4
lrwxrwxrwx 1 root root 0 &#1086;&#1082;&#1090; 22 14:29 host6 ->
../../../../devices/pci0000:00/0000:00:1e.0/0000:0a:04.0/host6/scsi_host/host6
```

So, the available ports are mapped to hosts 3, 4 and 6.

Discover the connected FC HBAs (ports)

```
# lsscsi
..
[2:0:0:0] disk ATA Hitachi HDS72101 A610 /dev/sda
[3:0:0:0] storage MSA CONTROLLER 7.20 -
[3:0:0:2] disk MSA VOLUME 7.20 /dev/sdb
...
[5:0:0:2] disk Generic- USB3.0 CRW-MS/HG 1.00 /dev/sdi
[6:0:0:0] storage MSA CONTROLLER 7.20 -
```

So the connected hosts are 3 and 6 ("MSA CONTROLLER").

Discover the HBAs port/node names

```
# cat /sys/class/scsi_host/host3/port_name
0x10000090faa00a31
# cat /sys/class/scsi_host/host4/port_name
0x10000090faa00a32
# cat /sys/class/scsi_host/host6/port_name
0x10000000c923ea51
```

So the names are:

Host	Mapping	Port name
3	Lancer-X, port 0 ("3:0:0:0")	0x10000090faa00a31
4	Lancer-X, port 1 ("3:0:0:1")	0x10000090faa00a32
6	LP8000 single port	0x10000000c923ea51

Charon Configuration

2 ways of specification in Charon configuration file are available:

By port name

Specify Linux port name "0x10000090faa00a31" in Charon configuration file:

```
load kgpsa_generic_storage PGA interface="0x10000090faa00a31"
```

The configuration given with port name is **independent** on hardware change, modules load order, etc

By host id name

```
load kgpsa_generic_storage PGA interface="fc_host3"
```

or

```
load kgpsa_generic_storage PGA interface="host3"
```

The configuration given with host id name **might change** on hardware change, modules load order, etc

Configuration parameters

Never use the following parameters (except the "interface") without well defined reason. Most of the parameters are implemented for solving possible problems and tracing at customer sites, so they might harm if not used appropriately.

interface

Name of FC, SCSI host or adapter's port/node name, defining a connection to Linux FC HBA, for example:

Mapping	Description
fc_host6	Emulation over Linux SG (FC) host 6
host6	Emulation over Linux SG (FC) host 6
2000-0000-C923-EA51	Emulation over Linux SG (FC) host given with port/node name
0x20000000C923EA51	Emulation over Linux SG (FC) host given with port/node name
wwn-0x20000000C923EA51	Emulation over Linux SG (FC) host given with port/node name

Example:

```
load kgpsa_generic_storage PGA interface="host3"
```

trace_level

Value	Description
0	none (default)
1	All failures and suspects
2	SCSI, ELS/CT request flow
3	MB/IOCB request flow

Example:

```
load kgpsa_generic_storage PGA trace_level=1
```

io_model

I/O model is used for request processing.

This parameter is needed for debugging purposes only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Value	Description
sync	synchronous
async	asynchronous (default)

Example:

```
load kgpsa_generic_storage PGA io_model="sync"
```

port_name

The port name in format: %04x-%04x-%04x-%04x. Option to fake guest's port name.

This parameter is needed for debugging purposes only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA port_name=2000-0000-C923-EA51
```

By default this value is obtained from the specified hardware port (see above).

node_name

The node name in format: %04x-%04x-%04x-%04x. Option to fake guest's node name.

This parameter is needed for debugging purposes only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA node_name=2000-0000-C923-EA51
```

By default this value is obtained from the specified hardware port (see above).

link_up_delay

The delay in seconds, port suspend reporting the link up waiting for Linux FC infrastructure creation. Default is 120 seconds.

When FC wire is removed and then plugged in, Linux rebuilds all the FC objects with new devices names, etc. Fabric presentation mode senses the link up and down events, but optionally it may delay reporting the event (for the given amount of seconds) to let Linux fully initialize FC. If delay is 0, guest OS may start initialization too early and thus fail to find some or all objects.

Example:

```
load kgpsa_generic_storage PGA link_up_delay=80
```

link_poll_period

The link check poll interval in seconds Default is 1 second.

When FC wire is removed and then plugged in, Linux rebuilds all the FC objects with new devices names, etc. Fabric presentation mode senses the link up and down events, but optionally it may delay reporting the event (for the given amount of seconds) to let Linux fully initialize FC. If delay is 0, guest OS may start initialization too early and thus fail to find some or all objects.

Example:

```
load kgpsa_generic_storage PGA link_poll_period=2
```

emu_fabric

The way the emulation deals with fabric:

Value	Description
auto	Delegates to Linux if supported, otherwise emulates
no	Delegates to Linux (real physical fabric port connected too)
yes	Emulates

Qlogic does not map SG devices for ELS/CT traffic. To work around this, Fabric presentation mode implements emulation of the whole non FCP (scsi) communication over fabric. If fabric is emulated, guest OS will not see non FCP target ports (for example initiators) and will not be able to communicate with ELS/CT over fabric.

Example:

```
load kgpsa_generic_storage PGA emu_fabric="no"
```

lun_exclude

The list of the LUNs to be excluded. If empty (default), nothing is excluded.

No multi line is possible for this parameter; the maximum number of symbols inside the double quotes is 255. Use "lun_include" parameter instead if a lot of LUNs must be excluded to specify only the included LUNs.

Example:

```
load kgpsa_generic_storage PGA lun_exclude="6008-05F3-0005-2950-C758-BCC2-E88B-0007,6008-05F3-0005-2950-8B03-E26B-E231-0005"
```

Please note: lun_exclude parameter cannot exclude storage mapped to LUN 0 because of its special function.

lun_include

The list of the LUNs to be included. If empty (default), all LUNs are included.

Please note: no multi line is possible for this parameter; the maximum number of symbols inside the double quotes is 255. Use "lun_exclude" parameter instead if a lot of LUNs must be included to specify only the excluded LUNs.

Example:

```
load kgpsa_generic_storage PGA lun_include="6008-05F3-0005-2950-C758-BCC2-E88B-0007,6008-05F3-0005-2950-8B03-E26B-E231-0005"
```

scsi_version

This approach of FC emulation downgrades all attached devices to the specifically defined version of SCSI. This is done to make legacy guest operating systems happy with attached modern devices.

Value (ANSI SCSI version)	Description
any	Any version
v1	SCSI-1
v2	SCSI-2
v3	SCSI-3
v3_SPC1	SCSI-3, primary command level 1
v3_SPC2	SCSI-3, primary command level 2 (default)
v3_SPC3	SCSI-3, primary command level 3
v3_SPC4	SCSI-3, primary command level 4

Refer to your guest OS documentation for more details and public resources for general specification of SCSI.

Example:

```
load kgpsa_generic_storage PGA scsi_version="v3_SPC3"
```

device_id

A fake device id (if not "lp8000" is needed), the default is "0xF800" ("lp8000").

This parameter is needed for some specific cases only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA device_id=0xFA00
```

vendor_id

A fake vendor id (if not "lp8000" is needed), the default is "0x10DF" ("lp8000").

This parameter is needed for some specific cases only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA vendor_id=0x10EA
```

revision_id

A fake revision id (if not "lp8000" is needed), the default is "0x02" ("lp8000").

This parameter is needed for some specific cases only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA revision_id=0x01
```

scsi_serialize_io

This parameter controls serialization of the SCSI requests.

Value	Description
false	Responses delivered to the emulated KGPSA card in order they were submitted to Linux. This is the default value.
true	Responses delivered to the emulated KGPSA card in order they were completed by Linux

This parameter is needed for some specific cases only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA scsi_serialize_io = true
```


scsi_max_queue_depth

This parameter sets limitation of the SCSI queue depth per LUN

Value	Description
unlimited	Linux SG I/O applies limitation of 16 I/Os. Requests are submitted unless write to SG I/O fails. This is the default value.
1...16	Only the specified number of the requests are submitted in parallel.

This parameter is needed for some specific cases only, so it is strongly recommended not to specify any value without particular instructions from Stromasys.

Example:

```
load kgpsa_generic_storage PGA scsi_max_queue_depth = 1
```

Acer Labs 1543C IDE/ATAPI CD-ROM adapter

Table of Contents

- General description
- Loading Acer Labs 1543C IDE/ATAPI adapter
- Configuration parameters
 - container

General description

*** THIS IS NO LONGER SUPPORTED. DO NOT USE ***

CHARON-AXP supports emulation of an integrated virtual Acer Labs 1543C IDE/ATAPI controller.

Loading Acer Labs 1543C IDE/ATAPI adapter

By default the integrated virtual Acer Labs 1543C IDE/ATAPI controller is preloaded with a name "ide".


Example:

```
set ide container="/dev/sg0"
```

Configuration parameters

The Acer Labs 1543C IDE/ATAPI adapter emulation has only one configuration parameter:

container


Parameter	container
Type	Text String
Value	<ul style="list-style-type: none"> • "/dev/sg<N>" <p>Specifies a physical device correspondent to ATAPI or SATA CD/DVD-ROM drive attached to the host system.</p> <p>By default it is left unspecified.</p> <p> Read this article on how to find the physical device name for mapping.</p>

Example:

```
set ide container="/dev/sg0"
```

When running HP OpenVMS/Alpha Operating System on top of CHARON-AXP virtualization layer the specified CD/DVD-ROM drive is available as DQA0: device.

CHARON-AXP is able to boot any OpenVMS/Alpha and Tru64 version from Acer Labs 1543C IDE/ATAPI CD-ROM.

 Virtual Acer Labs 1543C IDE/ATAPI can be mapped only to physical CD-ROM drives. If a CD-ROM container or an ISO file should be used, it is required to utilize [KZPBA-CA](#) controller as it offers full support of both physical and virtual mappings to system resources.


PCI I/O Bypass controller

Table of Contents

- General description
- Prerequisites
- Installation
- Loading PCI I/O bypass controller
- Configuration parameters
 - container
 - removable
 - geometry
 - use_io_file_buffering
- Usage example
- Deinstallation

General description

CHARON-AXP supports PCI I/O bypass controller for accessing to disk images and host physical disks. PCI I/O bypass controller requires a specific driver to be installed.

 PCI I/O bypass controller support is available only for OpenVMS guest operating system.

Prerequisites

This release supports VMS version V6.2-1H3 and higher. Bypass disks can not be used as a boot device in V6.2-1H3, higher versions do not have this restriction.

Make sure that the latest Bypass controller kit has been installed, especially for VMS versions before V7.3-2.

Installation

1. Open CHARON-AXP configuration file and attach the virtual disk "ovms_tool.vdisk" located by default in the "/opt/charon/disks" directory:

```
set PKA container[400] = "/opt/charon/disks/ovms_tools.vdisk"
```

2. Run CHARON, boot guest OpenVMS operating system.
3. Mount the disk, its label is "TOOLS" (Example: \$ MOUNT DKA400: TOOLS)
4. Use the POLYCENTER Software Installation (PCSI) utility to install the Bypass Driver. The following example demonstrates the "PCSI PRODUCT INSTALL" command to execute and the expected output (the example assumes the utilities virtual disk image is attached as DKA400:):

```
$ PRODUCT INSTALL CHARON_DISK/SOURCE=DKA400:[BYPASS]
%PCSI-I-CANNOTVAL, cannot validate DKA400:[BYPASS]SRI-AXPVMS-CHARON_DISK-V0104--1.PCSI;1
-PCSI-I-NOTSIGNED, product kit is not signed and therefore has no manifest file

The following product has been selected:
  SRI AXPVMS CHARON_DISK V1.4 Layered Product

Do you want to continue? [YES] YES

Configuration phase starting ...

You will be asked to choose options, if any, for each selected product and for any products that may be
installed to satisfy software dependency requirements.

SRI AXPVMS CHARON_DISK V1.4: Charon disk driver V1.4 for OpenVMS Alpha.

  Copyright (C) 1976, 2009 Software Resources International

  CHARON_DISK was produced by Software Resources International

* This product does not have any configuration options.

Execution phase starting ...

The following product will be installed to destination:
  SRI AXPVMS CHARON_DISK V1.4 DISK$TARDISSYSTEM:[VMS$COMMON.]

Portion done: 0%...10%...20%...80%...100%

The following product has been installed:
  SRI AXPVMS CHARON_DISK V1.4 Layered Product
```

Loading PCI I/O bypass controller

Syntax for loading PCI I/O bypass storage adapter:

```
load pci_io_bypass <name>
```

The <name> can be DI<x>, DR<x> or DU<x>

where x is selected according to VMS naming scheme, i.e. A stands for the first controller of given type, B - for the second, etc.

Example:

```
load pci_io_bypass DIA
```



In AlphaStation 400 configuration use the following syntax for PCI I/O bypass storage adapter loading:

```
load pci_io_bypass DIA irq_bus=isa
```

The adapter instance name ("DIA" in the example above) is used then for parametrization, for example:

```
set DIA container[0]="/Mydisks/vms_distributive.vdisk"
```

The numbers in the square brackets represent a number of device on PCI I/O Bypass controller.

The maximum number of I/O Bypass controller devices is 64.

By default I/O Bypass controller uses PCI slot corresponded to the <x> parameter (see above). If instead some particular slot is needed, refer to [this section](#) for details of specific placement of PCI peripherals on CHARON Virtual Machine (VM) PCI bus (note that "irq_bus" and "irq" parameters are ignored for I/O Bypass controller). In this case the <x> will be changed automatically according to custom position of I/O Bypass controller on PCI bus.



I/O Bypass controller is implemented for OpenVMS only.

Configuration parameters

The I/O Bypass controller has the following configuration parameters:

container

Parameter	container[N] N is 0..32766 (no more than 64 units)
Type	Text String
Value	<p>Possible values of the parameter are strings in one of the following forms:</p> <ul style="list-style-type: none"> Physical disk <ul style="list-style-type: none"> ■ <code>"/dev/sd<L>"</code> where "L" is letter. ■ <code>"/dev/disk/by-id/..."</code> - addressing by the disk ID, for example <code>"/dev/disk/by-id/ata-ST1000DM003-9YN162_S1D01QJ4"</code> ■ <code>"/dev/disk/by-label/..."</code> - addressing by the disk label, for example <code>"/dev/disk/by-label/MyStorage"</code> ■ <code>"/dev/disk/by-uuid/..."</code> - addressing by the disk UUID, for example <code>"/dev/disk/by-uuid/0e808a2f-cdd3-4944-a245-f729ffd73882"</code> <ul style="list-style-type: none"> ⚠ Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake. <p>⚠ <code>"/dev/sd<L>"</code> addressing is not persistent, so it is strongly recommended to use <code>"/dev/disk/by-id/wwn-..."</code> syntax instead to refer the disk by its WWID - especially in the environments utilizing FC and SAN storages.</p> <p>These disks must not be formatted by the host OS.</p> <p>Example:</p> <pre>set DIA container[0]="/dev/sdb"</pre> <p>It is also possible to use not a whole disk, but previously created partitions on it. In this case the syntax is the following: <code>"/dev/sd<L><N>"</code> where N is the number of partition to be used.</p> <p>Example:</p> <pre>set DIA container[0]="/dev/sdc1"</pre> Multipath disk <ul style="list-style-type: none"> ■ <code>"/dev/dm-<N>"</code> ■ <code>"/dev/mapper/mpath<N>"</code> ■ <code>"/dev/mapper/disk<N>"</code> <p>⚠ Be careful not to destroy all the information from the disk dedicated to CHARON-AXP by mistake.</p> <p>These disks must not be formatted by the host OS.</p> <p>Example:</p> <pre>set DIA container[100]="/dev/dm-0"</pre> Loop (virtual block) devices <ul style="list-style-type: none"> ■ <code>"/dev/loop<N>"</code> <p>Example:</p> <pre>set DIA container[200]="/dev/loop0"</pre> Direct mapping to some SCSI disks <ul style="list-style-type: none"> ■ <code>"/dev/sg<N>"</code> <p>Example:</p> <pre>set DIA container[300]="/dev/sg0"</pre> File representing a physical disk of the Alpha system (disk image) <ul style="list-style-type: none"> ■ <code>["<drive>":"<path-name>"<file-name>[".vdisk"]"</code> <p>These files can be created from scratch with "mkdiskcmd" utility. Data and OS disk backups are transferred from the original system via tapes or network and restored into these container files.</p> <p>Mapping may also include the full path, for example: <code>"/my_disks/my_boot_disk.vdisk"</code></p> <p>Example:</p> <pre>set DIA container[401]="my_dka401.vdisk"</pre>

This parameter is initially not set, thus creating NO storage elements on the controller.


removable

Parameter	removable[N] N is 0..32766 (no more than 64 units)
Type	Boolean
Value	<p>When set to "true", the removable configuration parameter instructs CHARON-AXP to report the corresponding virtual SCSI device as removable.</p> <p>By default the removable configuration parameter is set to "false".</p> <p>Example:</p> <pre>set DIA removable[400]=true</pre>

geometry

Parameter	geometry[N] N is 0..32766 (no more than 64 units)
Type	Text String
Value	<p>This formatted string value specifies the explicit geometry of the disk storage element. This parameter is not applicable to tape storage elements.</p> <p>The string format is <code>"*/*/"</code> or <code>"*,*,"</code> where B is the total size of the disk (in blocks) reported to the guest OS. If omitted it is calculated automatically.</p> <p>If this parameter is not set, CHARON VM will configure the geometry based on the most probable disk type.</p> <p>Initially not set.</p>

use_io_file_buffering

Parameter	use_io_file_buffering[N] N is 0..32766 (no more than 64 units)
Type	Boolean
Value	<p>When set to "true", instructs CHARON-AXP to enable host operating system I/O cache on reading/writing operations.</p> <p> Note that this caching has a significant effect only in case of mapping to disk and tape containers, not physical drives.</p> <p>When enabled, host operating system I/O cache may significantly improve I/O performance of the virtual system. At the same time maintaining I/O cache requires additional host resources (CPU and memory) which may negatively affect overall performance of the virtual system.</p> <p>Initially is set to "false".</p> <p>Example:</p> <pre>set DIA use_io_file_buffering[603]=true</pre>

When a disk image connected to an I/O Bypass controller is dismounted by OpenVMS, it is disconnected from CHARON-AXP and can be manipulated. It can be replaced with a different disk image if it keeps the same name. This capability may be useful when designing back-up and restore procedures. When copying CHARON-AXP disk images while CHARON-AXP is running, please take care to minimize the risk of overloading a heavily loaded CHARON-AXP host system. For example, using a sequential series of simple ftp binary copies is less resource intensive and thus less disruptive than multiple, simultaneous copies.

Empty disk images are created with the "mkdiskcmd" utility. Tape images (*.vtape) will be created automatically if they don't exist (no utility needed).

CHARON-AXP is able to boot from disk images of any OpenVMS/Alpha version.

Usage example

Example using DR disk devices (disk drives connected to a SWXCR raid controller):

```
load pci_io_bypass DRA
set DRA container[1] = "/Mydisks/dra1.vdisk"
set DRA container[2] = "/Mydisks/dra2.vdisk"
set DRA container[3] = "/Mydisks/dra3.vdisk"
set DRA container[4] = "/Mydisks/dra4.vdisk"
```

Deinstallation

1. Do a conversational boot. Please refer to your OpenVMS system administration guide for instructions.
2. Set the NOAUTOCONFIG system parameter to 1
3. Boot OpenVMS
4. Remove the product with the "\$ PRODUCT REMOVE CHARON_DISK" command.
5. Set the NOAUTOCONFIG system parameter to 0 and reboot.

Finding the target "/dev/sg" device

Table of Contents

- General description
- Procedures of finding the target "/dev/sg" device
 - Method 1
 - Method 2
 - Method 3

General description

This section describes how to find proper "/dev/sg" device for CHARON mapping

Procedures of finding the target "/dev/sg" device

Method 1

Open a terminal console and issue:

```
# lsscsi -g
```

i If the "lsscsi" command is not installed on your system, use "yum install lsscsi" to make it available. If you cannot install "lsscsi", use [method 2](#) or [method 3](#) described below.

Output example1:

```
[0:0:0:0]    disk    VMware    Virtual disk    1.0    /dev/sda    /dev/sg0
[0:0:10:0]   tape    COMPAQ    SDLT320         5F5F    /dev/st0    /dev/sg9
[0:0:11:0]   tape    COMPAQ    SDLT320         5F5F    /dev/st1    /dev/sg10
[0:0:12:0]   mediumx COMPAQ    MSL5000 Series  0520    /dev/sch0   /dev/sg11
```

Output example2:

```
[0:0:0:0]    disk    VMware,    VMware Virtual S 1.0    /dev/sda    /dev/sg0
[4:0:0:0]    cd/dvd  NECVMWar  VMware SATA CD01 1.00    /dev/sr0    /dev/sg1
```

Method 2

Open a terminal console and issue:

```
# cat /proc/scsi/sg/device_hdr; cat /proc/scsi/sg/devices
```

Output example:

host	chan	id	lun	type	opens	qdepth	bus	online
4	0	0	0	5	1	1	0	1
5	0	0	0	0	1	1	0	1

The fifth column ("type") value has the following correspondence:

Value	Device
0	Disk
1	Tape
5	CD-ROM
8	Tape changer

The "N" in the "/dev/sgN" device is the line number (starting from 0) corresponding to the output provided by the commands above without taking the header into account so here "/dev/sg0" corresponds to the the CD-ROM.

Method 3

On a freshly booted system, issue the following command:

```
# dmesg | grep sg
```

The output will look like that:

```
[ 1.503622] sr 4:0:0:0: Attached scsi generic sg0 type 5
[ 1.780897] sd 5:0:0:0: Attached scsi generic sg1 type 0
```

 This table lists all the devices, not only the real SCSI ones (SATA/IDE for example). CHARON supports only real SCSI devices.

Networking

Table of Contents

- General description
- Configuration steps
- Configuration parameters
 - interface
 - station_address
 - rx_fifo_size
 - adapter_mode
 - Example
 - DE602 and DE602AA network adapters link speed and duplex settings
- Packet Port
 - interface
 - port_enable_mac_addr_change
 - port_retry_on_tx
 - port_pending_rx_number
 - port_pending_tx_number
 - log
 - log_flush_period
 - Example

General description


CHARON-AXP supports emulation of the following network adapters:


- DE435
- DE450
- DE500AA
- DE500BA
- DE602
- DE602AA

Each of them is a PCI Ethernet adapter based on the DEC21040 (DE435, DE450, DE500AA and DE500BA) and the Intel i8255x (DE602 and DE602AA) PCI Ethernet adapter chips for the Alpha.

CHARON-AXP maps the virtual adapter to a dedicated Ethernet adapter in the Linux host system.

All the emulated controllers are loaded and configured in the same way.

 The Ethernet adapter in the Linux host system must support dynamic changes of its MAC address (i.e. no reboot of the host system is required to change the MAC address), which is the case with nearly all modern Ethernet adapters.

 By default the PCI Ethernet adapters use first available PCI slot. If instead some particular slot is needed, refer to [this section](#) for details of specific placement of PCI peripherals on CHARON-AXP PCI bus.

Configuration steps

To configure CHARON-AXP networking, follow these 3 steps:

1. Load network adapter (if required)

Use the "load" command as shown below.

Example:

For DEC21040 adapters	For Intel i8255x adapters
<code>load DE500BA/dec21x4x NIC</code>	<code>load DE602/i8255x NIC</code>



In AlphaStation 400 configuration use the following syntax for network adapter loading:

```
load DE500AA/dec21x4x NIC irq_bus=isa
```

i By default each loaded virtual network adapter uses first available PCI slot. If instead some particular slot is needed, refer to [this section](#) for details of specific placement of PCI peripherals on CHARON-AXP PCI bus.

2. Load "packet_port" or "tap_port"

Load "packet_port" or "tap_port" to connect network adapter to the host hardware network card (or to a virtual network interface).

Example:

```
load packet_port/chnetwrk NDIS interface = "eth1"
```

3. Connect the loaded "packet_port" ("tap_port") to the loaded virtual network adapter

Connect the network adapter to the "packet_port" ("tap_port") by setting the interface name.

Example:

```
set NIC interface = NDIS
```




The interface name can be either "(disabled)" for a disabled interface or "<network interface name>"

Examples:


```
load packet_port/chnetwrk NIC1 interface="(disabled)"
```

```
load packet_port/chnetwrk NIC2 interface="ens33"
```

 The AlphaServer DS15 and DS25 contain two built-in PCI Ethernet adapters. Models and names (EI* or EW*) of them depend on configuration add-on. Choose one of the two or none, but not both. The first instantiates onboard network interfaces as EIA and EIB. While the second - EWA and EWB (enabled by default for backward compatibility)

Example:

```
#include ds25-onboard-nics.icfg
include ds25-onboard-nics-ew.icfg
```

 It could be necessary to specify the path to the .icfg file if you use your own configuration file using the "include /opt/charon/cfg/ds25-onboard-nics-ew.icfg" syntax.

Configuration parameters

Each virtual network controller has the following parameters that are specified with the "set" command:

interface

Parameter	interface
Type	Text String
Value	Name of the corresponding instance of the "packet_port" or "tap_port" component

station_address

Parameter	station_address
Type	Text String
Value	<p>The "station_address" provides the ability to configure the adapter's permanent address. By default the adapter's permanent address is read from the host system's NIC.</p> <p>Format:</p> <p>XX-XX-XX-XX-XX-XX</p> <p>or</p> <p>XX:XX:XX:XX:XX:XX</p> <p>Example:</p> <pre>set EWA station_address="AF:01:AC:78:1B:CC"</pre>

rx_fifo_size

Parameter	rx_fifo_size
Type	Numeric
Value	<p>"rx_fifo_size" sets the receive FIFO size.</p> <p>The value is specified in Kb and, by default, is pre-calculated from the connected port's size of the receive queue.</p> <p>Typically, you do not need to change the "rx_fifo_size" parameter. It is available for extended tuning and debugging purposes.</p>

adapter_mode

Parameter	adapter_mode
Type	Text String
Value	<p>Defines the link speed and the duplex settings of the virtual network adapter (except for DE602/DE602AA - see Networking#DE602 and DE602AA network adapters link speed and duplex settings).</p> <p>The values are:</p> <ul style="list-style-type: none"> ■ "Auto" for auto-negotiate (default) ■ "10BaseT-HD" for 10Mbps half duplex ■ "10BaseT-FD" for 10Mbps full duplex ■ "100BaseT-HD" for 100Mbps half duplex ■ "100BaseT-FD" for 100Mbps full duplex <p>Example:</p> <pre>set EWA adapter_mode="100BaseT-HD"</pre>

Example

```
load packet_port/chnetwrk EWA0 interface="eth1"
set EWA interface = EWA0
set EWA adapter_mode = "100BbaseT-FD"
set EWA station_address="0C:FE:35:AA:67:3B"
```

DE602 and DE602AA network adapters link speed and duplex settings

Regardless of the "adapter_mode" setting in CHARON-AXP configuration file (see above), DE602 and DE602AA network adapters remains in "Auto-negotiation" mode, since the EIDRIVER of OpenVMS checks for EIX0_MODE environment variable when configuring the network card.

So mode propagation is implemented in CHARON-AXP via SRM console EIX0_MODE environment variable ("x" is A, B, C... depending on CHARON-AXP configuration), for example:

```
>>>help set

usage: set <variable-name> <value>
set <variable-name> ""

set eia0_mode { Twisted | Full | Fast | FastFD | Auto* }

>>>
```

⚠ The EIX0_MODE variable name is case insensitive, while its values are case sensitive! This is feature of OpenVMS EIDRIVER.

The values are:

Parameter	Description
"Auto"	Auto-negotiate (default)
"Twisted"	10Mbps half duplex
"Full"	10Mbps full duplex
"Fast"	100Mbps half duplex
"FastFD"	100Mbps full duplex

Example:

```
>>>set eia0_mode FastFD
```

Packet Port

The CHARON-specific "packet_port" interface establishes a connection between an Ethernet adapter in the Linux host system and a network adapter in the virtual Alpha system.

For every virtual adapter instance loaded, one dedicated host Ethernet physical adapter is required.

To create instances of the "packet_port", use the "load" command in the configuration file as follows:

```
load packet_port/chnetwrk <instance-name>
```

Example:

```
load packet_port/chnetwrk ppl
```

"packet_port" uses several configuration parameters to control its behavior.

interface

Parameter	interface
Type	Text string
Value	<p>This parameter identifies an Ethernet adapter of the host system dedicated to CHARON-AXP.</p> <p>Syntax:</p> <pre>set <name> interface="<adapter>"</pre> <p>Example:</p> <pre>set ppl interface="eth1"</pre>

port_enable_mac_addr_change

Parameter	port_enable_mac_addr_change
Type	Boolean
Value	<p>If "true" is specified (default value), CHARON-AXP sets the appropriate Ethernet address automatically.</p> <p>If "false" is specified, set the Ethernet address manually.</p> <p>Example:</p> <pre>set ppl port_enable_mac_addr_change=false</pre>

port_retry_on_tx

Parameter	port_retry_on_tx
Type	Numeric
Value	<p>The "port_retry_on_tx" parameter controls the number of times a port will attempt to transmit a packet before giving up.</p> <p>By default, the value is 3.</p> <p>Increasing this value may introduce problems in carrier loss logic, because not all NIC drivers support a carrier status query.</p> <p>Typically, you do not need to increase the value.</p> <p>Example:</p> <pre>set ppl port_retry_on_tx=8</pre>


port_pending_rx_number

Parameter	port_pending_rx_number
Type	Numeric
Value	<p>The "port_pending_rx_number" parameter sets the number of pending receive buffers.</p> <p>The default value is 63. The maximum value allowed is 195.</p> <p>You may want to increase the "port_pending_rx_number" when you have very busy networking and experience problems like losing connections not related to the carrier loss.</p> <p>Typically, you do not need to change this parameter.</p> <p>Example:</p> <pre>set pp1 port_pending_rx_number=128</pre>

port_pending_tx_number

Parameter	port_pending_tx_number
Type	Numeric
Value	<p>The "port_pending_tx_number" parameter sets the number of buffers the port uses to transmit.</p> <p>The default value is 62.</p> <p>You may want to increase the "port_pending_tx_number" value if the log file indicates dropped TX packets due to TX queue overflow.</p> <p>Typically, you do not need to change this parameter.</p> <p>Example:</p> <pre>set pp1 port_pending_tx_number=128</pre>

log


Parameter	log
Type	Text string
Value	<p>If this parameter is set to some valid file name or a directory where the log files for each individual session will be stored CHARON logs Recv and Xmit packets at the emulated port layer.</p> <p>If an existing directory is specified, CHARON automatically enables creation of individual log files, one for each session using the same scheme as used for the generation of the rotating log files. If the "log" parameter is omitted, CHARON does not create log.</p> <p>In certain situations enabling this parameter may help to detect loss of packets.</p> <p>Example 1:</p> <pre>set ppl log="ppl.log"</pre> <p>Example 2:</p> <pre>set ppl log="/charon/logs"</pre> <p> Only existing directory can be used as a value of the "log" parameter.</p>

log_flush_period

Parameter	log_flush_period
Type	Numeric
Value	<ul style="list-style-type: none"> • <period-in-seconds> <p>Defines a period of flushing log to disk.</p> <p>Default period is 60 seconds (it means that every minute log file is flushed to disk)</p> <p>Example:</p> <pre>set ppl log_flush_period=30</pre>

Example

```
load DE500BA/dec21x4x EWA
load packet_port/chnetwrk ppl interface="eth1"
set EWA interface=ppl
```

 CHARON-AXP supports VLAN adapters. If for some reasons you are going to use them, proceed with their installation and configuration according to the network adapter's vendor's User's Guide and then use the resulting VLAN interface the same way as the regular network interface.

[More information on VLAN](#)

PBXDA PCI serial lines adapter

Table of Contents

- [General description](#)
- [Loading PBXDA serial lines adapter](#)
- [Configuration parameters](#)
 - [port](#)
 - [line](#)
 - [log](#)
 - ["ttyY" notation specifics](#)

General description

PBXDA is a PCI serial lines adapter based on the AccelePort 2R 920, 4R 920, 8R 920 and Xem DIGI adapters.

Loading PBXDA serial lines adapter

Syntax for loading PBXDA (AccelePort 2r 920) serial lines adapter:

```
load PBXDA/DIGI <name>
```

Syntax for loading PBXDA_BA (AccelePort 4r 920) serial lines adapter:

```
load PBXDA_BA/DIGI <name>
```

Syntax for loading PBXDA_BB (AccelePort 8r 920) serial lines adapter:

```
load PBXDA_BB/DIGI <name>
```

Syntax for loading PBXDA_AC (AccelePort Xem) serial lines adapter:

```
load PBXDA_AC/DIGI <name>
```

Example:

```
load PBXDA/DIGI TXA
```

The adapter instance name ("TXA" in the example above) is used then for parametrization, for example:

```
set TXA line[2]="/dev/tty0"
```

The numbers in the square brackets represent line number on the virtual PBXDA adapter starting from 0.

Controller type	Maximum number of lines
PBXDA	2
PBXDA_BA	4
PBXDA_BB	8
PBXDA_AC	16

All the parameters described in the ["Placement of peripheral devices on PCI bus"](#), such as "bus", "device", "function", "irq", "irq_bus" are applicable for PBXDA controller.

Please note:

DIGI drivers for OpenVMS and Tru64 are sensitive to PBXDA location on PCI, therefore it is recommended to fix PBXDA location with explicit configuration.

Example:

```
load PBXDA TXA bus=pci_1 device=4 function=0
```

Configuration parameters

The PBXDA serial lines adapter emulation has the following configuration parameters:

port

Parameter	port
Type	Text String
Value	<p>Specifies a local port for incoming telnet connections</p> <p>By default the "port" configuration option is not specified.</p> <p>Syntax:</p> <pre>port[line-number]=<local port></pre> <p>Example:</p> <pre>set TXA port[2]=17060</pre>

line

Parameter	line
Type	Text string
Value	<p>A defined TTY port on host system:</p> <ul style="list-style-type: none"> ■ <code>"/dev/tty<N>"</code> for virtual console ■ <code>"/dev/ttyS<N>"</code> for onboard serial lines ■ <code>"/dev/ttyUSB<N>"</code> for modem or USB serial lines adapters ■ <code>"/dev/tty<XXX>"</code> for proprietary (depending on a driver) devices such as DIGI or MOXA cards <p>Example:</p> <pre>set TXA line[2]="/dev/ttyS1"</pre> <p>Please note:</p> <ul style="list-style-type: none"> • If a virtual console <code>"/dev/tty<N>"</code> is going to be used, it must be freed from all the processes running on it at first. Refer to your OS documentation for details, also some description on how to do it is available here. • A specific account for running CHARON ("charon") does not allow usage of physical consoles <code>"/dev/tty<N>"</code> as CHARON consoles. If you plan to map CHARON console to <code>"/dev/tty<N>"</code> use only "root" account for CHARON running.

log

Parameter	log
Type	Text string
Value	The optional log file or directory name is where the log file for the serial line is stored. Example: set TXA log[0]="/charon/logs/txa0.log"

"ttyY" notation specifics

Note that the "ttyY" notation can have different forms depending on the nature of the device used:

Mapping	Type	Commentary
"/dev/tty<N>" where N is from 0 to 11	Linux virtual tty	Those tty devices must be free from the Linux "getty/mgetty" and similar programs (specified in "/etc/inittab") Example: "/dev/tty1"
"/dev/ttyS<N>" where N is a number	Onboard serial lines	Example: "/dev/ttyS1"
"/dev/tty<XXX>" where XXX is a complex letter /number notation	Proprietary (depending on a driver) devices	Example for a first port of a MOXA card: "/dev/ttyR01" Example for a first port of a DIGI card: "/dev/ttyaa"
"/dev/ttyUSB<N>" where N is a number	Modem or USB serial lines adapters	Example: "/dev/ttyUSB1"

AlphaStation Sound Card (AD1848) emulation

Table of Contents

- [General description](#)
- [Loading PCXBJ audio adapter](#)

General description

PCXBJ is a PCI controller for emulation of AlphaStation Sound Card AD1848.

It is available for the following hardware models:

- AlphaServer 400

Please note:

- The latest "PulseAudio" package must be installed on Charon host to allow audio playback. Refer for "[PulseAudio](#)" page for obtaining the package and details of its installation.
- PCXBJ supports only sound playback on OpenVMS operating system at the moment. Its functionality will be extended in future releases.

Loading PCXBJ audio adapter

Syntax for loading PCXBJ audio adapter:

```
load PCXBJ <name>
```

Example:

```
load PCXBJ AUA
```

Please note:

- All the parameters described in the "[Placement of peripheral devices on PCI bus](#)", such as "[bus](#)", "[device](#)", "[function](#)", "[irq](#)", "[irq_bus](#)" are applicable for PCXBJ controller
- No extra parameters need to be set/changed.
- Once the controller is loaded it uses the host default audio card for audio playback via "PulseAudio" package.
- On OpenVMS side one must have MultiMedia services for OpenVMS package (MMOV) installed (available with OpenVMS CDL) along with MMOV-RT license.

PBXGA graphics card

Table of Contents

- General description
 - Emulated cards
- Important notes
- Virtual PBXGA PCI graphics card settings
 - Loading virtual PBXGA graphics card
 - Configuration parameters
 - size
 - cpu_draw
 - application
 - Using PBXGA_TV
 - General Information
 - Command-Line Parameters
 - Multiple Screens
- Examples
 - Example 1: 8-bpp graphics on AlphaServer 400
 - Example 2: 24-bpp graphics on AlphaServer DS20
 - Example 3: two display units, running independently
 - Example 4: dual head display, 2x 1280x1024
 - Example 5: triple head display, 3x 1280x1024
 - Example 6: Two screens consolidated in a column in one window

General description

Charon-AXP supports emulation of PBXGA graphics card(s) by direct virtualization and provides a virtual console for displaying graphics.

The console starts automatically upon starting the emulator. Its resolution is 800 x 600 by default.

Support for AXP graphics is provided with two components:

- emulation of PBXGA PCI graphics adapter and
- emulation of display unit (display unit is emulated with separate executable PBXGA_TV)

Emulated cards

- PBXGA_AA emulates 8bpp graphics card (256 colors, pseudo-color) with 4MB frame buffer - ZLXp-E1
- PBXGA_BA emulates 24bpp graphics card (16M colors, TrueColor) with 16MB frame buffer - ZLXp-E2

Important notes

- PBXGA graphics console cannot be used as the system console. Use OPA0 for this purpose.
- If PBXGA graphics card is going to be used, the resolution of CHARON host screen must be sufficient to provide ability to display the graphics console properly.
- If PBXGA graphics card is configured, starting with version 4.12, Charon can now run as a service and is therefore compatible with the Linux Toolkit. Alternatively the emulator can also be run in detached mode (`# <emulator exe> -d <configuration file>`)
- If the window of the PBXGA graphics card console is closed by user or killed as a process, it will not re-appear automatically (💡 the Charon emulator will continue to work). In this situation the user has to restart the pbxga_tv program with the correct parameters, as defined in the configuration file, to restart the graphical display (see [PBXGA graphics card#application](#) chapter further)

Virtual PBXGA PCI graphics card settings

Loading virtual PBXGA graphics card

Syntax for loading PBXGA graphics card:

```
load <emulated card> <name>
```

Syntax for loading PBXGA graphics card on AlphaServer 400:

```
load <emulated card> <name> irq_bus = isa
```

Example (more examples [below](#)):

```
load PBXGA_BA GYA
```

On Charon startup a window will appear on CHARON host monitor to display graphics.

All the parameters described in the "[Placement of peripheral devices on PCI bus](#)" chapter, such as "bus", "device", "function", "irq", "irq_bus" are applicable for PBXGA graphics card.

Configuration parameters

size

Parameter	size
Type	Text string
Value	<p>Predefine the screen size.</p> <p>Possible values: 640x480, 800x600 (default), 1024x768, 1152x900, 1280x1024, 1920x1080</p> <p>Example:</p> <pre>set GYA size = 1024x768</pre>

cpu_draw

Parameter	cpu_draw
Type	boolean
Value	<p>Offloading raster operations. The default value is "true"</p> <p>Example:</p> <pre>set GYA cpu_draw = false</pre>

application

Parameter	application
Type	Text string
Value	<p>Specifies the application to open for displaying the emulated graphics device.</p> <p>The pbxga_tv executable file is provided for this. It is located in the same folder as the Charon emulator executable file.</p> <p>Syntax:</p> <pre>pbxga_tv -c <configuration_name> -n <device name(s)></pre> <p>When several graphics cards are loaded, the <device name(s)> have to be separated by a pipe character and the application has to be defined for the latest loaded PBXGA card. See examples below. Note that correct consolidation is only possible when screens are of the same depth and size.</p> <p>Example:</p> <pre>set GYA application = "pbxga_tv -c DS20 -n GYA"</pre> <p>There is no need to supply display size to PBXGA_TV application.</p> <p>Using Panoramix/XINERAMA extensions to DECwindows is also possible (topics of proper configuration OpenVMS/Tru64 are not covered here). These extensions allow to join multiple heads in one big virtual screen. The PBXGA_TV virtual display unit has special notation allowing to consolidate multiple frame buffers in single window (topology is flexible, to some extent):</p> <ul style="list-style-type: none"> • Screens are joined horizontally when separated by " " • Screens are joined vertically when separated by "/" <p>Example:</p> <pre>set GYD application = "pbxga_tv -c MYCONF -n GYA GYB/GYC GYD"</pre>

Using PBXGA_TV

General Information

The PBXGA_TV display unit starts in passive mode. It displays content of frame buffer but does not react on keyboard input and/or mouse movements. In order to switch it to active mode, place mouse cursor with the window and click once. When PBXGA_TV is active, mouse cursor is stuck to PBXGA_TV window. Press CTRL+ALT+R to release.

The PBXGA_TV display unit starts in Window Mode. Use CTRL+ALT+F to toggle Full Screen Mode.

The PBXGA_TV application uses the SDL2 library to draw its window. The PBXGA_TV driver is statically linked with SDL2. You may need to install the "libdecor" package additionally.

Running PBXGA_TV remotely is not supported.

Please note: PBXGA_TV can only consolidate multiple screens from a single emulator instance.

Command-Line Parameters

The "pbxga_tv" application accepts the following parameters:

Parameter	Description
-c, --configuration-name <CONF>	Specify the configuration-name in the emulator configuration file
-n, --device-name <DEVICE>	Specify the device-name in the emulator configuration file
--guest-os <GUEST> -Tru64 -OpenVMS	Specify guest operating-system running in emulator. Values for <GUEST> are Tru64 or OpenVMS . -Tru64 and -OpenVMS are the same as --guest-os Tru64 and --guest-os OpenVMS
-u, --screen-update-period-ms <PERIOD>	Screen update period in milliseconds. If omitted, the default is 100

Multiple Screens

The PBXGA_TV application supports using multiple PBXGA adapters in screen consolidation mode, provided that guest OS is configured for PANORAMIX/XINERAMA extension. In order to join multiple PBXGA displays in a single window the value of device-name must list the desired device names as follows:

```
-n <device-name-1>[<device-name-2>...][<device-name-3>[<device-name-4>...]]
```

Example of dual display (two displays joined horizontally):

```
set session configuration_name = DUAL
...
load PBXGA GYA
load PBXGA GYB
set GYA application = "pbxga_tv -c DUAL -n GYA|GYB"
```

If the two displays must be joined vertically:

```
set GYA application = "pbxga_tv -c DUAL -n GYA/GYB"
```

The PBXGA_TV application can consolidate up to 3 displays in a row (or column) or up to 4 displays in a matrix of 2x2. All displays must have the same geometry and bits-per-pixel.

The PBXGA_TV application can run independently, from command line, or using shortcuts created by the user. Terminating the PBXGA_TV application does not destroy the graphics context (which is drawn by AXP emulator). But it must run on the same host as the AXP emulator (including Remote Desktop or VNC sessions).

Examples

Example 1: 8-bpp graphics on AlphaServer 400



```
...
set session hw_model = AlphaServer_400
set session configuration_name = AS400
...
load PBXGA_AA GYA irq_bus = isa size = 1024x768
set GYA application = "pbxga_tv -c AS400 -n GYA"
```

Example 2: 24-bpp graphics on AlphaServer DS20

```
...
set session hw_model = AlphaServer_DS20
set session configuration_name = DS20
...
load PBXGA_BA GYA size = 1280x1024
set GYA application = "pbxga_tv -c DS20 -n GYA"
```

Example 3: two display units, running independently

```
...
set session configuration_name = MYCONF
...
load PBXGA_BA GYA
set GYA application = "pbxga_tv -c MYCONF -n GYA"
load PBXGA_BA GYB size = 1280x1024
set GYB application = "pbxga_tv -c MYCONF -n GYB"
```

In the above example, display units have different resolutions.

Example 4: dual head display, 2x 1280x1024

```
...
set session hw_model = AlphaServer_ES45
set session configuration_name = ES45
...
load PBXGA_BA GYA size = 1280x1024
load PBXGA_BA GYB size = 1280x1024
set GYB application = "pbxga_tv -c ES45 -n GYA|GYB"
```

Example 5: triple head display, 3x 1280x1024

```
...
set session hw_model = AlphaServer_ES45
set session configuration_name = ES45
...
load PBXGA_BA GYA size = 1280x1024
load PBXGA_BA GYB size = 1280x1024
load PBXGA_BA GYC size = 1280x1024
set GYC application = "pbxga_tv -c ES45 -n GYA|GYB|GYC"
```

Example 6: Two screens consolidated in a column in one window

```
...
set session configuration_name = MYCONF
...
load PBXGA_BA GYA size = 1280x1024
load PBXGA_BA GYB size = 1280x1024
set GYB application = "pbxga_tv -c MYCONF -n GYA/GYB"
```

Sample configuration files

The template files can be found after product installation in the following directory:

```
/opt/charon/cfg/
```

The template file names have the form `<Alpha-model>.cfg.template`.

CHARON-AXP for Linux deinstallation

Deinstallation procedure

To uninstall the CHARON-AXP product:

1. Stop all running CHARON-AXP instances, [remove all CHARON-AXP services](#).
2. Login as "root" user.
3. Issue the following command:

```
# yum remove aksusbd charon-*
```

Appendixes

Contents

- [glibc.i686 installation without Internet connection](#)
- [How to implement time synchronisation between CHARON-AXP Host OS and Guest OS](#)

glibc.i686 installation without Internet connection

**** PLEASE NOTE: As of Charon-AXP 4.12 you no longer need to install the 32-bit glibc package.**

If you need to configure a local repository using a CDROM/DVD ISO image, see below.

Contents

- [Description](#)
- [Step-by-step guide](#)
 - [Introduction](#)
 - [Installation steps](#)
 - [Mounting the DVD](#)
 - [Installation using the rpm utility](#)
 - [Installation using the yum utility](#)

Description

The `glibc.i686` package is required for the CHARON-AXP, CHARON-VAX, CHARON-SSP and CHARON-HPA products. When connected to the Internet, and – for Red Hat Enterprise Linux – registered, this package can be installed as described in the product documentation. Usually the following command is used:

```
# yum install glibc.i686
```

When an Internet connection is not available, or if a server running Red Hat Enterprise Linux is not registered, this package can be installed using the operating system installation DVD or ISO file.

This document explains how to install the package from the installation DVD for the following Linux distributions:

- Red Hat Enterprise Linux 6.x and 7.x
- CentOS 7.x - Everything distribution DVD only.
The Standard distribution DVD of CentOS 7.x does not contain the `glibc.i686` package so the Everything distribution DVD is mandatory if no internet connection is available.

There are several different solutions for installing `glibc.i686` without Internet connection. This document explains some of them. For more information, please contact your system administrator or refer to the administrator's guide for the Linux distribution installed on your CHARON host system.

Step-by-step guide

Introduction

Check first if the `glibc.i686` package is installed using the following command:

```
# yum list installed glibc.i686
```

If the package is not installed, the command will report the following message (examples given for Red Hat Enterprise Linux 7.2)

```
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.
Error: No matching Packages to list
```


If the package is installed, the command will report the following message:

```
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager
This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.
Installed Packages
glibc.i686                2.17-105.e17                @<source>
```

Alternatively, the following rpm command can be used:

```
# rpm -qa | grep glibc | grep i686
```

If the command reports nothing, the package is not installed.

Installation steps

Mounting the DVD

Please either insert the installation DVD in the drive or use an ISO file.

If the DVD is mounted automatically, we recommend to unmount it and mount it manually on a mount point with no spaces in its name (the yum-config-manager utility does not handle spaces correctly).

Example / Red Hat Enterprise Linux 7.2:

```
# df -k
Filesystem            1K-blocks    Used Available Use% Mounted on
/dev/mapper/rhel-root 37300436 4376776 32923660 12% /
devtmpfs              1925960      0 1925960  0% /dev
tmpfs                 1941228      96 1941132  1% /dev/shm
tmpfs                 1941228     9132 1932096  1% /run
tmpfs                 1941228      0 1941228  0% /sys/fs/cgroup
/dev/sdal             508588 160220 348368 32% /boot
tmpfs                 388248      12 388236  1% /run/user/1000
tmpfs                 388248      0 388248  0% /run/user/0
/dev/sr0              3947824 3947824      0 100% /run/media/stromasys/RHEL-7.2 Server.x86_64

# umount /dev/cdrom
# mkdir -p /media/cdrom
# mount -r /dev/cdrom /media/cdrom
```

By default `/dev/cdrom` is linked to `/dev/sr0`.

If you have an ISO image of the distribution CD, you can mount it using a loopback device:

```
# mount /path/to/ISO-image.iso /media/cdrom -o loop
```

Installation using the rpm utility

If you use rpm to install the package, any packages on which it depends must be installed manually.

To do so:

1. Switch to the directory containing the packages. This directory depends on your Linux distribution.

Example:

```
# cd /media/cdrom/Packages/
```

2. Locate the target "glibc.i686" package and the package on which it depends:

```
# ls -l glibc*i686* nss-softokn-freebl*
```

3. Install the "glibc.i686" package and any others on which it depends (package versions may differ):

Example / Red Hat Enterprise Linux 7.2:

```
# rpm -i glibc-2.17-105.el7.i686.rpm nss-softokn-freebl-3.16.2.3-13.el7_1.i686.rpm
```

Example / CentOS7:

```
# rpm -i glibc-2.17-157.el7.i686.rpm nss-softokn-freebl-3.16.2.3-14.4.el7.i686.rpm
```

If the command returns a warning message related to RSA/SHA256 signature, please ignore it. The successful installation of the package can be checked using the "yum list installed glibc.i686" command.

If the installation command above reports additional unsatisfied dependencies, add the corresponding packages to the above command line.

4. Unmount the CD-ROM or ISO file if necessary:

```
# cd -
# umount /media/cdrom
```

Installation using the yum utility

The yum utility will check and install all the necessary dependencies.

Define the operating system installation DVD as a new repository:

Example / Red Hat Enterprise Linux 7.2:

```
# yum-config-manager --add-repo=file:///media/cdrom

Loaded plugins: langpacks, product-id
adding repo from: file:///media/cdrom

[media_cdrom]
name=added from: file:///media/cdrom
baseurl=file:///media/cdrom
enabled=1

# yum --nogpgcheck install glibc.i686

...
Is this ok [y/d/N]: y

...
Complete!
```

Once the installation is complete, the repository can be disabled (if no other package has to be installed):

```
# yum-config-manager --disable media_cdrom
```

Using this method, installing other packages could require the gpg check to be disabled. To do so:

- either add the `--nogpgcheck` parameter to the `yum install` commands.

Example:

```
# yum --nogpgcheck install <package>
```

- or disable the GPG check for the repository in the `/etc/yum.repos.d/media_cdrom.repo` file, by adding the `gpgcheck=0` line
- or disable the GPG check in the `/etc/yum.conf` file, replacing the `gpgcheck=1` line by `gpgcheck=0`

Example / CentOS 7:

```
# yum --disablerepo=* --enablerepo=c7-media install glibc.i686

...
Is this ok [y/d/N]: y

...
Is this ok [y/d/N]: y

...
Complete!
```

The `c7-media` repository is a default repository on CentOS 7.x pointing to the `/media/cdrom` folder. Please refer to the administrator's documentation for more information.

How to implement time synchronisation between CHARON-AXP Host OS and Guest OS

Table of contents

- Description
- Step-by-step guide
 - Configuration file settings
 - Virtual machine operating system settings
 - On OpenVMS/AXP
 - Using a batch queue
 - Using a detached process
 - Considerations using DECnet-Plus software
 - On Tru64 UNIX
- Related articles

Description

This document will explain how to implement the time synchronization feature using the "sync_to_host" parameter in the configuration file. This parameter allows to keep TOY time always synchronized with the host's time and disable undesirable updates to the TOY from guest OS.

Restrictions: Minimum product versions/builds required:

- **Windows:**
 - CHARON-AXP V4.4 Build 148-02 with patch 148-09 installed
 - CHARON-AXP V4.6 Build 166-03 and later
 - Note: CHARON-AXP V4.5 Build 153-03 and 153-05 (patched) are not supported
- **Linux:**
 - CHARON-AXP V4.6 Build 168-03 and later


Step-by-step guide

Configuration file settings


Update the configuration file with the following settings:

Syntax:

```
set TOY sync_to_host = "{as_vms | as_tru64 | as_is}{[, nowrite]"
```

 If "sync_to_host" parameter is specified there is no need to specify "container" parameter in addition.

where:

Parameter	Description
as_vms	If the guest OS is OpenVMS/AXP and its date and time must be set to the host's date and time each time it boots.
as_tru64	If the guest OS is Tru64 UNIX and its date and time must be set to the host's date and time each time it boots.
as_is	If the TOY date and time must be set to the host's UTC date and time
nowrite	Forbid updates to the TOY from the guest OS  If you want guest to synchronize itself using DTSS or NTP for example, remove "nowrite"

Example:

```
set TOY sync_to_host = "as_vms, nowrite"
```

To synchronize the guest OS with TOY, use the following commands (from "SYSTEM"/"root" account):

On OpenVMS/AXP	On Tru64 UNIX
\$ <code>set time</code>	# <code>date -u `consvar -g date cut -f 3 -d ' '`</code>

The default value is "not specified" - it means that by default CHARON does not synchronize its guest OS time with the CHARON host time but collects date and time from the file specified with "container" parameter.

⚠ If "sync_to_host" parameter is specified there is no need to specify "container" parameter in addition.

ℹ The CHARON virtual machine must be restarted in order to take the new parameter into account

Virtual machine operating system settings

The commands mentioned above used to synchronize the guest OS with TOY are effective only when they are executed. To avoid time difference, these commands must be executed at specified intervals.

You will find below examples on how to implement scripts to perform time synchronization for OpenVMS and Tru64 UNIX.

💡 If you have NTP running on your OpenVMS or Tru64 UNIX system, you can keep it running even if `sync_to_host` is enabled

On OpenVMS/AXP

You need first to perform a manual synchronization between the CHARON server and the CHARON virtual machine using the `SET TIME=` command:

```
$ SET TIME=12:30:00
```

You can use either a batch queue or a detached process to synchronize time. The two methods are described below.

Using a batch queue

Create a simple script containing the following lines. The example below will sync time every hour:

ℹ In our example, we will create the script in the `SYS$MANAGER` folder and name it `CHARON_SYNCTIME.COM`. The OS version used is OpenVMS 7.3-2. Its content is:

```
$ SET NOON
$ SET VERIFY
$LOOP:
$ SHOW TIME
$ SET TIME
$ SHOW TIME
$ WAIT 01:00:00
$ GOTO LOOP
```

ℹ The "\$ SET VERIFY" line is optional, just used for verifying commands are correctly executed

A batch queue will be required to create the job:

- Find an available batch queue or create a new one
- Execute the following command to view available batch queues (refer to OpenVMS documentation)

```
$ SHOW QUEUE /ALL /BATCH
```

If the command returns the following error message: "%JBC-E-JOBQUEDIS, system job queue manager is not running", you will need to initialize the queue manager:

```
$ START /QUEUE /MANAGER /NEW
%%%%%%%%% OPCOM 29-MAY-2015 12:30:07.36 %%%%%%%%%%
Message from user SYSTEM on VMS732
%JBC-I-CREATED, SYS$COMMON:[SYSEXE]QMAN$MASTER.DAT; created
```

- Create a dedicated batch queue for the synchronization job (recommended):


```
$ INIT /QUEUE /BATCH /START SYS$SYNCTIME /JOB_LIMIT=1
$ SHOW QUEUE SYS$SYNCTIME /FULL
Batch queue SYS$SYNCTIME, idle, on VMS732::
  /BASE_PRIORITY=4 /JOB_LIMIT=1 /OWNER=[SYSTEM] /PROTECTION=(S:M,O:D,G:R,W:S)
```

- Submit the job:

```
$ SUBMIT /QUEUE=SYS$SYNCTIME SYS$MANAGER:CHARON_SYNCTIME
Job CHARON_SYNCTIME (queue SYS$SYNCTIME, entry 1) started on SYS$SYNCTIME
```


- Update the systartup script (SYS\$STARTUP:SYSTARTUP_VMS.COM) to start the SYS\$SYNCTIME queue and the job at system boot. The two following lines will have to be added at the very end of the script (for example):

```
...
$ START /QUEUE SYS$SYNCTIME
$ SUBMIT /QUEUE=SYS$SYNCTIME SYS$MANAGER:CHARON_SYNCTIME
$
$ EXIT
```


 For OpenVMS version 5, the systartup script will be named: SYS\$STARTUP:SYSTARTUP_V5.COM

Using a detached process

Create a simple script containing the following lines. The example below will sync time every hour

 In our example, we will create the script in the `SYS$MANAGER` folder and name it `CHARON_SYNCNCTIME.COM`. The OS version used is OpenVMS 7.3-2. Its content is:

```
$ SET NOON
$ SET VERIFY
$LOOP:
$ SHOW TIME
$ SET TIME
$ SHOW TIME
$ WAIT 01:00:00
$ GOTO LOOP
```

 The "`$ SET VERIFY`" line is optional, just used for verifying commands are correctly executed. If you let it active, please replace the "`NL:`" device above by a log file name


Update the systartup script (`SYS$STARTUP:SYSTARTUP_VMS.COM`) to start the detached process at system boot. The following lines will have to be added at the very end of the script (for example):

```
...
$ RUN SYS$SYSTEM:LOGINOUT /AUTHORIZE /DETACH /UIC=[SYSTEM] -
  /PROCESS_NAME="TIME SYNC" /OUTPUT=NL: /ERROR=NL: -
  /INPUT=SYS$MANAGER:CHARON_SYNCNCTIME.COM
$
$ EXIT
```



To start the job manually without a reboot, just execute the line above from an interactive session.


Considerations using DECnet-Plus software

 If you have DECnet-Plus software installed, you will have to disable DTSS before setting time in order to avoid errors like:

```
%SET-E-NOTSET, error modifying time
-SYSTEM-E-TIMENOTSET, time service enabled; enter a time service command to update the time
```

 To disable DTSS, you will have to update the CHARON_SYNCNCTIME.COM script:

```
$ SET NOON
$! SET VERIFY
$LOOP:
$ SHOW TIME
$ RUN SYS$SYSTEM:NCL
DISABLE DTSS
DELETE DTSS
EXIT
$ SET TIME
$ SHOW TIME
$ @SYS$STARTUP:DTSS$STARTUP
$ WAIT 01:00:00
$ GOTO LOOP
```

 It is recommended here not to use "SET VERIFY"

On Tru64 UNIX


Restrictions: The synchronization requires the "consvar" command to be available on the Tru64 operating system thus Tru64 UNIX version 4.0F minimum is required


You need first to perform a manual synchronization between the CHARON server and the CHARON virtual machine using the `date` command. Example:

```
# date -u 05291724
```


Create an entry in the root's `crontab` file using "`crontab -e`" as shown below:

```
...
# CHARON time sync_to_host
00 * * * * /sbin/date -u ` /sbin/consvar -g date | cut -f 3 -d ' '`
```

 Full path to `date` and `consvar` commands must be specified

 The above command will be executed at minute 0 of every hour. If you want to execute this every 15 minutes for example, use the following line instead:

```
00/15 * * * * /sbin/date -u ` /sbin/consvar -g date | cut -f 3 -d ' '`
```

 More information at [Wikipedia.org](https://en.wikipedia.org/wiki/Cron) - Cron

Related articles

 [How to implement time synchronisation between CHARON-AXP Host OS and Guest OS \(sync_to_host\)](#)