SUPPORT GUIDE

Porting your application built on

Sunhillo' turnkey

WAN protocols from the PCI334/334A

to PCI Express®-based PCE335

OVERVIEW

The advantages of PCE architecture over PCI™ are very compelling because of its increasing ubiquity, feature rich hardware, greater bandwidth, etc. However, customers may have concerns about how many changes must be made to their existing applications in order to migrate.

One of the benefits of using Sunhillo WAN protocols is the MPSapi remains the same across both the PCI and the PCI Express platforms, making it easier to convert applications. This, in turn, will allow customers to quickly migrate applications to the new PCI Express controllers that are becoming the industry standard.

The paragraphs that follow will outline the steps required to migrate Sunhillo WAN protocol-based applications from the PCI334/PCI334A to the PCI Express PCE335 controller.

GETTING STARTED

Customers reading this document are likely running an older version of a protocol (X.25, Frame Relay, Radar, or others) for the PCI334/PCI334A. Over the years the protocols have evolved and the client applications need to account for that in order to run the new client side software and updated protocols.

For clients running the Linux 2.4 kernel, the host will have Linux® STREAMS (LiS) installed, since that is how the old device driver worked. The new driver supporting the PCE335 does not depend on STREAMS; therefore, it is not used. SolarisTM, of course, will continue to be STREAMS-based. The Windows®-based driver has never been STREAMS-based. For reference, the /usr/spt/config directory on the client has all the release notes for the updated MPSapi, driver, protocols, etc.

Along with the PCE335 card and new firmware, the developer must obtain a Right-To-Use (RTU) and turnkey protocol development kit (CD), which contains the host-side drivers, protocol header files, client-side utilities (with source code), and sample applications (also with source code). Cabling can be ordered with, or without, a console port connection for command line interface access. Please note that a console port will be useful for initial application development.

Client-side installation is approximately the same for the PCI334 and the PCE335. Installation instructions for the client and its specific operating system can be found in the MPS Software and Hardware Installation Manual (document #SUN2430), which is included on the CD for the protocol being used. The installation script builds the portions of the software distribution that execute only on the local host system. These portions include the MPSapi library and header files, host driver, configuration, utilities, and sample applications.

Another big change between the PCI334 and the PCE335 is the on-board operating system. The PCE335 card utilizes a Linux OS, whereas the PCI334 is xSTRa-based. This means that commload is no longer used to load the image to the card after powering up. Instead, the PCE335 has the advantage of having its image stored in flash and will automatically boot up and be operational in about 45 seconds.

The initial validation is performed through the commstats utility, as it will exercise the MPSapi and verify driver connectivity to the PCE335 controller. The commstats utility is available in the /usr/spt/bin directory and will provide useful information on the PCE335 controller and the WAN Protocol software executing on the controller.

The developer can issue the commstats command with the -i (information) option to view the driver version and wait for a valid up state. A 'Functional' state means that the PCE335 is functional and ready for use:

```
$ ./bin/commstats -i

2 Controllers Found –

SPT native MPS driver for PCI/PCIe bus, Version 2.2.0
Controller 0: PT-PCI334A (0x334a) - In RESET
Bus: 5 Device: 6 Function: 0
Controller 1: PT-PCE335 (0x3350) - Functional
Bus: 2 Device: 0 Function: 0
```

Note that the PCI334 controller is 'In RESET' as it has not yet received its download modules via the commload command.

Without the information flag the commstats command will return useful information on the PCE335 controller:

```
$ commstats -c 1

System Report for Server /dev/ucsclone, Controller: 1

Product code for kernel: SPT Kernel Image: RDR 810P0720.74.0
The date is Fri Sep 21 12:25:38 2018
Current system kernel version 2.6.20
System has been alive 264 hrs 54 mins 34 secs
Total system memory   121802752 bytes
Free system memory    43163648 bytes
Used system memory    78639104 bytes
Running Kernel 0
Vendor Id: 0x1214, Device Id: 0x3350
Board serial number: 599113
electrical interface per port:
      0 - RS232     1 - RS232     2 - RS232     3 - RS232
```

COMPILING AN EXISTING APPLICATION

As with the PCI334, Sunhillo provides the host-side software needed to access the PCE335. Because the new software has updated structures and drivers, customers must recompile their codes with the current library and header files. Here is an example of recompiling an old HDLC sample application from the /<old installation directory>/hdlc/client directory so that it may run on the PCE335:

First the old application source is copied into the new WAN protocol development environment:

```
# cd /usr/spt/hdlc
# mkdir old_client
# cp /<old installation directory>/hdlc/client/* old_client
```

Note: Only the (sample) application files should be copied into the new WAN Protocol development environment as the new WAN protocol development environment header files and MPSapi library must be used for the build process.

Using the sample applications provides the benefit where the diff command may be used to determine the changes that will need to be incorporated into any existing applications:

```
# diff old_client/hdlcrecv.c client
1c1
< /* @(#) hdlcrecv.c 03/05/06 Version 1.54 */
---
> /* @(#) hdlcrecv.c 07/07/16 Version 1.62 */
84a85,88
> 27. 13-Nov-06 kls Added support for QNX6.
> 28. 08-Jun-07 hhn Added support for timestamp options.
> 29. 10-jul-07 hhn Multi Thread hooks in place.
> 30. 16-jul-07 kls Added actualBaud bind parameter.
```

First the old example applications are rebuilt without making any changes. This yields the results:

```
# cd old_client
# make clean; make solaris
rm -f htest hdlcrecv hdlcsend hfull *.o
/opt/gnu/bin/gcc -I../../include -I../include -c hdlcutils.c
hdlcutils.c: In function `hdlc_response':
hdlcutils.c:191: `MAX_STATUS_VAL' undeclared (first use in this
function)
hdlcutils.c:191: (Each undeclared identifier is reported only once
hdlcutils.c:191: for each function it appears in.)
hdlcutils.c: In function `hdlc_open':
hdlcutils.c:510: `MAX_STATUS_VAL' undeclared (first use in this
function)
```

From review of the updated hdlcutils.c application one can find the following change notice:

```
8. 25-jul-06 mpb Make our 'MAX_STATUS_VAL' unique since mcc_api.h has
its own #define for that.
```

This has been defined in the updated hdlchdr.h header file:

```
# grep MAX_STATUS_VAL ../include/*
#define WAN_MAX_STATUS_VAL 12 /* Max status value */
```

After the old application is changed to use the new define, the compilation is successful as shown below,

```
# make clean
rm -f htest hdlcrecv hdlcsend *.o
# make linux
make[1]: Entering directory `/usr/spt/hdlc/client'
cc -DLINUX -I../../include -I../include -c hdlcutils.c
cc -DLINUX -I../../include -I../include -o htest \
hdlcutils.o htest.c -lMPS
cc -DLINUX -I../../include -I../include -o hdlcrecv \
hdlcutils.o hdlcrecv.c -lMPS
cc -DLINUX -I../../include -I../include -o hdlcsend \
hdlcutils.o hdlcsend.c -lMPS
make[1]: Leaving directory `/usr/spt/hdlc/client'
```

Following successful compilation the (sample) applications may be executed following the instructions in the appropriate WAN Protocol Software User's Guide. Testing the PCE335 and then the PCI334, and finally test between the PCE335 and the PCI334, if desired.

The sample applications are used to demonstrate the use of the MPSapi to send data from the client application on your Host system, to the controller, out the physical interface, through loopback connectors, back into the controller and back to the client application. The received data can be compared against the transmitted data to ensure end to end data integrity. In addition to being used in the above described loopback mode most example applications may also be used to send data between the PCE335 and the PCI334. This compatibility testing is useful if both servers are to be maintained with the same application.

Note: Remember that the protocol modules loaded on the PCI334 must be version compatible with the PCE335 firmware image. You will need to contact Sunhillo if the PCI334 files need to be updated.

CLIENT UTILITIES

The PCE335 controller offers expanded utilities that make monitoring and maintaining the WAN Protocol software running on the card much easier. The client utilities are installed in the /usr/spt/bin directory and run from the Host system as client applications. Further explanation of these utilities can be found in the MPS WAN Protocol Configuration Guide (document #SUN2298).

DOWNLOAD PROGRAM (wcpLoad)

Firmware updates on the card are a common procedure when Sunhillo makes enhancements or bug fixes to the protocol stack. Also, customers may want to modify the image on the card or an image may have become corrupted. The **wcpLoad** utility allows a new image to be easily flashed down to the card.

The original image that was flashed at the factory can be found in the images director of the MPS WAN Protocol CD that was purchased and can be copied to the utilities directory for flashing. The following command line would flash kernel 0 on controller 1 with the wanStrSer8260.img in verbose mode:

```
wcpLoad -c 1 -k 0 -v -f wanStrSer8260.img
wcpLoad for controller: 1 (0x3350)
Transfer of SPT Kernel Image: RDR 810P0720.74.0 to board in progress.
Percent Complete 100.0%
Image file copied to board.
Board will now be flashed with image file. This may take a while.
Controller will now be reset.
Now rebooting controller: 1
```

The **wcpLoad -r** command can be used to reboot the controller while keeping the host device driver informed. This is should be used to start the card in a known state.

EVENTS MONITOR PROGRAM (eventsClient)

The **eventsClient** utility uses the MPSapi to monitor all asynchronous events detected by the PCE335. To start this utility:

```
./bin/eventsClient -c 1
```

To exit this utility, use <Ctrl-c>

NWSL COMMAND LINE INTERFACE UTILITY (nwslCli)

The **nwslCli** utility provides access by issuing simple command-line instructions to the PCE335 over the PCE interface. These are the same commands and subsequent replies that would be used if a console cable were connected.

Developers will find this useful in debugging and modifying the behavior of the PCE335. The only limit of **nwslCli** is that no additional input is allowed. In other words, a "vi /tmp/foo.bar" would cause **nwslCli** to wait forever. In any case, press <Ctrl-c> to exit the **nwslCli** prompt. Here is an example of using **nwslCli**:

```
nwslCli "pti_info all"
Switch Settings     : 0x0F
Board Type          : PCE335
Board Rev           : 0
Build Options       : 3584 (RS232)
Baud Rate           : 9600
CPU Type            : 8260
CPU Revision        : HiP7_A.0
CPM Microcode Rev   : 0x0
System Clock        : 66 MHz
CPU Clock           : 264 MHz
CPM Clock           : 198 MHz
BRG Clock           : 24.750 MHz
Usable Memory       : 120 Mbytes
Image Load Addr     : 0xFE020000
```

```
Port A MAC addr     : 00:00:00:00:00:00
Port B MAC addr     : 00:00:00:00:00:00
Serial Number       : 599113
Image Identifier    : Ser8260 SPT Kernel Image: RDR 810P0720.74.0
Bootcode Revision   : 810Q093710 Nov  1 2007  10:15:29
OS Version          : 2.6.20
Factory Jumper      : 0
PCI Device ID       : 0x3350
PCI Vendor ID       : 0x1214
PCI Active bit      : Yes
PCI Memory size     : 8 Mbytes
Last reset reason   : Hardware
CPU number          : 1
Number of CPUs      : 1
Front Ejector Switch: ERROR -19
scc_drv             : L 810s0001.A6-033117 cksum:3182774161
bicc                : L cksum:3638614246
Boot String         : root=/dev/ram0 panic=3 console=ttyS0,9600
```

The **show boot options** command can also be retrieved by using **nwslCli**.

```
nwslCli –c 1 show-bootoptions
Autoboot:       On
WDOG boot:      On
Diag Boot:      Off
Diag Flag:      On
Boot Addr:      0xFE020000  (/dev/kernel0)
PCI Aperture:   8M
/dev/kernel0:        0xFE020000   -   Ser8260   SPT   Kernel   Image:   RDR
810P0720.74.0
```

NWSL COPY UTILITY (nwslCp)

The **nwslCp** utility provides a means of transferring a file from a client across the PCE interface and to the PCE335. Files can also be transferred back to the client. The utility can be used in getting the messages file when errors are suspected. Here is an example on controller 0:

```
./bin/nwslCp -c 1 -v -g /var/log/messages pce335_c1.msgs
```

CONCLUSION

In conclusion, porting applications from the PCI334/PCI334A to a PCE335 solution is straightforward. The same MPSapi is used, as was used with the PCI334, and updated protocols are provided. Recompiling is necessary, but familiar sample applications exist just as with the PCI334. Additionally, new utilities are provided that make development and maintenance much easier.

The following items relate to differences of note between the PCI334 and PCE335.

Multi-threaded applications:

Older MPSapi – Multi-threaded applications that overlapped reads and writes between threads were tolerated, although data could be lost. This created the situation where the host side application had a potential for crashing.

Updated MPSapi Symbol Strictly enforces well behaved multi-threaded applications.

Initial firmware load:

PCI334/334A required commload to download its protocol software. PCE335 is factory programmed with the WAN protocol software (field upgradeable via the wcpLoad utility).

For additional assistance, please contact Sunhillo at +1.844.767.7676 or email sales@sunhillo.com.