Packet Processing Language Virtual Machine

Installation Guide

January 2005

IP Fabrics, Inc.
14964 NW Greenbrier Parkway
Beaverton, OR 97006
503-444-2400
503-444-2401 FAX
www.ipfabrics.com

Table of Contents

Table of Figures

Table of Tables

Welcome to PPL

PPL is a very-high-level language for describing the processing of network packets. Although it could apply in concept to any type of packet, the language is oriented toward layer 3 IP packets, toward specific protocols at layer 4 (e.g., TCP and UDP), and toward "deep" packet processing at layer 7. In most senses, PPL is a *functional* language as opposed to a procedural language such as C. PPL also defines explicitly several concepts of concurrent processing.

In addition to being applicable to broad types of packet processing, PPL contains specific features oriented toward applications such as encryption, authentication, content inspection, stateless and stateful firewall filtering, detection of intrusions and denial-of-service attacks, layer 7 filtering, traffic management, and content-based load balancing.

PPL is intended to be used in an implementation with one or more network processors with high-speed packet-classification capabilities, and as such it represents a very-high-level language for writing concurrent data-plane software or microcode. Although PPL could be compiled in the traditional sense to the low-level code of these network processors, an implementation that is generally more effective is compiling it to a software virtual machine atop the network processor(s).

Overview

This document describes installation procedures for an IXP28XX system and what external resources are required to begin writing and executing PPL programs, as well as specific setup procedures required.

PPL is extremely flexible and provides multi-platform support. For the simplicity, the Intel IXMB2851 is used as a representative platform example throughout this document.

Typical Components for ATCA Platforms

Configuration 1 (Intel IXDP28x1 Development System)

- 5 slot ATCA chassis
- External chassis power supply
- ATCA Shelf Manager
- Intel IXMB28x1 NPU ACTA blade
- IXD4GET02 Quad Gigabit Ethernet Mezzanine Card
- RadiSys 6315 400MHZ Celeron PrPMC
- 1GB Compact Flash
- CDROM
- AC power cord
- RJ-45 to dual DB9 serial cable
- RJ-45 to RJ-45 10/100Mb Ethernet cable
- User's Manual, August 2004, Part number: C58425-002 Rev. 01

Configuration 2 (ATCA Blade in Pentair/Schroff 2-slot chassis)

- 2 slot ATCA chassis
- Intel IXMB28x1 NPU ACTA blade
- IXD4GET02 Quad Gigabit Ethernet Mezzanine Card
- RadiSys 6315 400MHZ Celeron PrPMC
- 1GB Compact Flash
- CDROM
- AC power cord
- RJ-45 to dual DB9 serial cable
- RJ-45 to RJ-45 10/100Mb Ethernet cable
- User's Manual, August 2004, Part number: C58425-002 Rev. 02

Host Requirements

The XScale processor on the NPU blade runs the Linux operating system and in its current debug configuration, mounts a file system from a remote host over its 10Mb Ethernet port. This remote host is generally either an embedded processor, such as a

PrPMC or other blade, or a user-supplied remote-host computer. If using a user-supplied remote host computer, the minimum requirements are:

- BOOTP or DHCP support
- NFS support
- 10/100Mb TP Ethernet port
- RS232 port capable of 115Kbaud, 8N1, no flow control (DB9 connector)
- Terminal emulation software
- 1GB of storage space for the root filesystem image of the NPU blade

Additional Requirements

- PPL Compiler Host Computer. This can be either:
  - embedded PrPMC boot host, running Pentium®-based Linux (RedHat 7.3), or
  - a separate Pentium®-based computer, running either Linux (RedHat 7.3 or later) or Windows
- Packet generator (computer system or dedicated equipment) 10/100Mb (half duplex) or 1Gb (full duplex)
- Appropriate network cabling

CDROM Contents

The PPL VM CDROM is an ISO9660 filesystem that should be readable on most any computer system. The directory structure is as follows:

Directory	Description
/documentation	Documentation files in Adobe PDF format
/host	Files for host computer configuration
/host/tools	Files for PPL development
/host/samples	Sample PPL programs (source code) and demo software
/target	Files for the target PPL blade
/demo	Files for the Windows®-based demo client program
/license	Files detailing the licenses
/README	UNIX text file pointing to /documentation directory

Table 1. CDROM Contents

Setup Process

The setup process consists of the following general steps:

- Setup of hardware including cable connections and jumper settings
- *[Optional]* Setup of software and network services on the remote host computer(s)
- *[Optional]* Setup of software and firmware on the PPL blade

If you received an ATCA chassis with the PPL blade preinstalled, the only hardware configuration you should need to perform is cable connection. It is still a good idea to check the jumper settings, however.

Hardware Configuration with Intel IXMB28x1 Development board

The following is a representative example of the hardware configuration required to install PPL.

Hardware configuration of the blade consists mainly of attaching cables as shown in the following diagram:

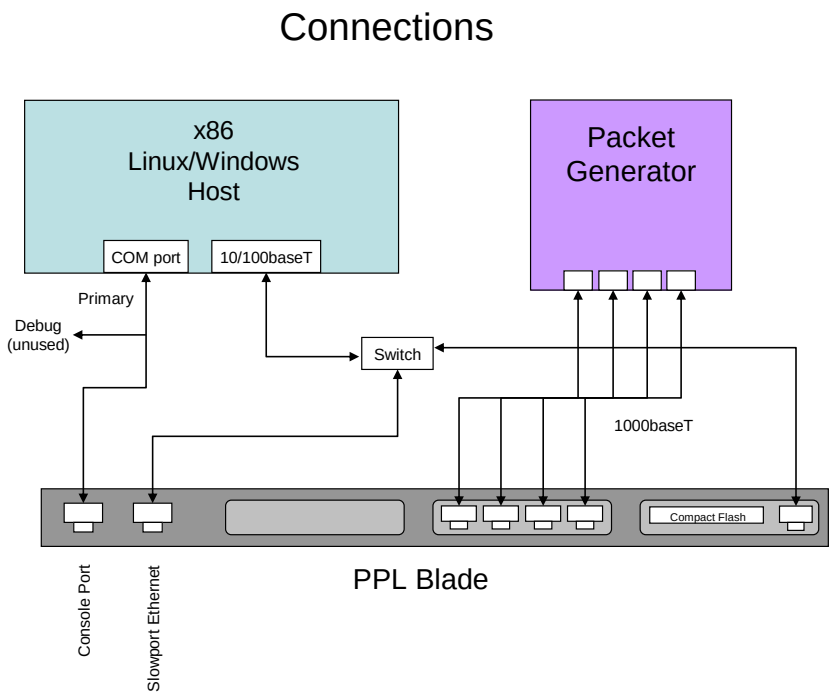

Figure 1 Connection Diagram using Embedded PrPMC as Remote File System

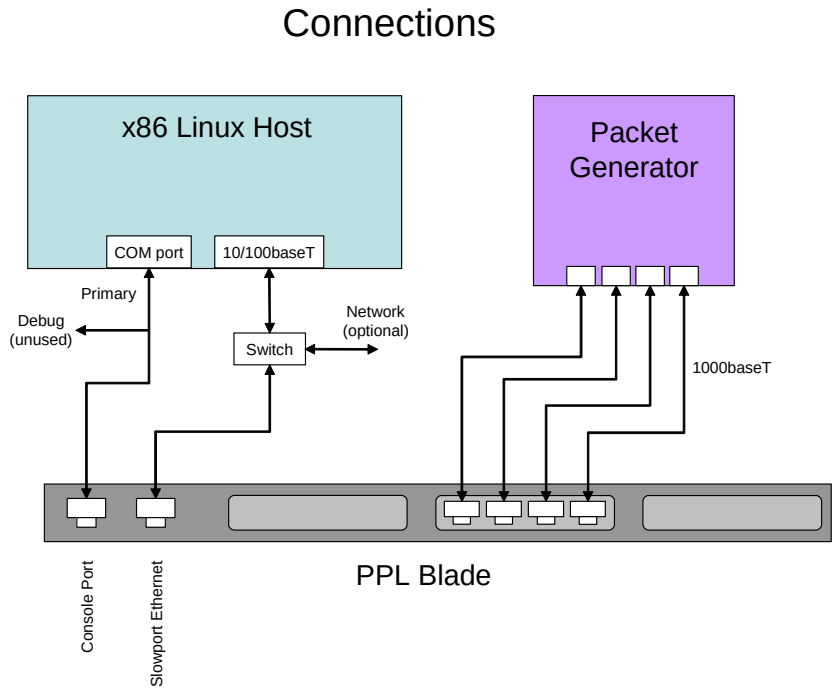

Figure 2 Connection Diagram Using User-Supplied Server as Remote File System

Follow the instructions in the IXDP2801 System User's Manual [1] to connect the external power supply to the chassis and to install the Shelf Manager (only required when using the 5-Slot ATCA chassis). Before installing the IXMB2801 Network Processor ATCA blade, follow the instructions in chapter 5 of the IXDP2801 System's User Manual to install the IXD4GET0C Quad Gigabit Ethernet Mezzanine Card into the PN6 and PN7 connectors on the IXMB2801 Network Processor ATCA blade. All of the IXMB2801 Network Processor ATCA blade's jumpers and switches should remain in their factory default positions (as documented in System User's Manual), except for the following jumpers and switches (see page 14 of the System's User Manual for the proper identification and location of the switches and jumpers on both the main board and the Network Processor daughterboard):

(1) On the main board, the jumper block labeled J2 should be configured as follows: 1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 19-20 are off (no shunts installed). A shunt should be installed on pins 13-14 and on pins 15-17 (see Figure 3 below for proper configuration - shaded areas indicate that the jumper should be installed). This

[1] Intel IXDP2801 and Intel IXDP2851 Advanced Development Platforms, System User's Manual, August 2004, Part number: C58425-002 Rev. 01.

configures the BMC to “legacy” mode, board reset from BMC active, reset of BMC: reset inactive, EEPROM : write disable, 12V, and power-on switched by the BMC.

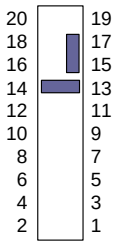

Figure 3 Jumper Settings

(2) On the Network Processor “daughter-board”, SW2 position 7 should be switched to “OFF”. This will configure the Network Processor to run at 1.4GHz instead of the default 1.0GHz. Be sure to “reconfigure” SW2 position 7 on the network processor card only (SW2 on the main board must stay at the factory default settings) .

Install IXMB2801 Network Processor ATCA blade into one of the “Node” slots (slots 3 to 5 are Node slots). Finally, connect the RJ-45 to DB9 serial cable from the IXMB2801 Network Processor ATCA card’s “Console” port to COM port on the host computer using the DB9 connector labeled “Primary”. Also connect the port labeled “10M Eth.” (next to the Console port) into your normal Ethernet network or directly into an Ethernet port on the host computer. The system should now be ready for power-on and software configuration.

To test the system, the host system’s terminal emulator software should be invoked and should be set to 115Kbaud, 8N1, and no flow control. At power-up, you should see the message “Recovery Loader version x.xx” (followed by a stream of other diagnostic/bootup messages) output to the terminal emulator after about a 10 second delay. If you don’t see this message, recheck all of the configuration steps.

Host Software Configuration

The following is a representative example of the host software configuration required to install PPL.

The PPL blade in its current configuration boots its operating system from flash and mounts a remote file system over the network using its slowport Ethernet connection (10Mb). This requires that a host on the network to which the blade is connected supply the blade with its IP address and be the NFS server for the root filesystem used by the blade. Both of these host functions can, of course, reside on a single or on multiple servers.

Booting Using an Embedded PrPMC

The simplest solution is to use the supplied Pentium®-based Linux PrPMC to perform all the required server functions and to also optionally execute the PPL compiler. To do this, no further configuration is required. However, when using the PrPMC, special care must be taken prior to powering-down or hot-swapping the NPU blade. **Please refer to the *Important PrPMC Shutdown Procedure* in this document.**

The PrPMC will assign the NPU blade IP address `192.168.20.107 .`

Booting Using a User-Supplied Server

Another solution is to run a separate Pentium®-based Linux machine to perform all the required server functions and to also execute the PPL compiler.

A host running RedHat Linux version 7.3 has been extensively used for this purpose at IP Fabrics although other versions of Linux or UNIX should work just as well. The following instructions are specific to RedHat 7.3, although other versions of Linux or UNIX should be similar. Also, the instructions assume that the required services are already running, which is the default when RedHat 7.3 is installed on a PC.

RedHat 7.3 was used as this is the version specified by Intel for use with its SDK used to build the PPL VM.

You will need superuser privileges on the host to perform this configuration.

Linux Kernel Booting

The file /host/zImage.pro31.2801on the supplied CDROM is already pre-loaded into the NPU blade's flash memory. This is the compressed kernel image that the PPL blade executes.

NFS Configuration

The file /host/pplfs.tar.gz is a compressed tarfile containing the root filesystem image for the PPL blade. This must be untar'ed into a directory on the host. In the examples that follow, it is assumed that this directory is named "/pplfs" on the host.

Use the following commands to create the directory and untar the file into it ("$" is the shell prompt and is not typed in; "/mnt/cdrom" is the directory that the CDROM is mounted to).

```
$ mkdir /pplfs
$ cd /pplfs
$ tar xvfz /mnt/cdrom/host/pplfs.tar.gz
```

Table 2. Root Filesystem Creation

The file /etc/exports must be edited to specify the path to the root filesystem and the IP address that will attempt to mount it. Modify the IP address as appropriate for your setup.

```
/pplfs 192.168.100.100(rw,no_root_squash,no_all_squash)
```

Table 3. NFS Configuration File Example

DHCP Configuration

The file /etc/dhcpd.conf must be edited to allow the host to assign an IP address to the PPL blade when it sends its BOOTP request to the host. An example of this file follows:

```
subnet 192.168.100.0 netmask 255.255.255.0 {
    host ppl.mydomain.com {
        hardware ethernet 00:07:e9:2d:37:82;
        fixed-address 192.168.100.100;
        option root-path "/pplfs";
    }
}
```

Table 4. DHCP Configuration File Example

This example configures the system to assign the static IP address 192.168.100.100 to the PPL blade (with the specified MAC address) and use the root filesystem located at "/pplfs" on the host. Static IP addresses are generally easier to use in a development environment.

The MAC address of the slowport Ethernet is displayed in the serial terminal emulator during RedBoot execution if this is connected to the blade and host system.

Restarting the Services

Use the following commands to restart the services on the server required by the PPL blade:

```
$ /etc/init.d/dhcpd restart
$ /etc/init.d/nfs restart
$ /etc/init.d/xinetd restart
```

Table 5. Restarting Linux Network Services

Alternatively, the host can be simply rebooted to "restart" the services.

Preparing for Use

If you plan to write XScale applications using the IPF API, use the following commands to prepare the system for use:

$ ln -s libipfsystem.so.x.y.z libipfsystem.so
$ ln -s libipfsystemdbg.so.x.y.z libipfsystemdbg.so

where "x", "y", and "z" are the version numbers of the shared libraries shipped in the release you received.

PPL Development Setup – Installing the Eclipse IDE

Prerequisites

Java RunTime Environment

If you are installing Eclipse onto a machine without a Java RunTime Environment (JRE) 1.4 or greater, you'll need to download and install the JRE prior to installing Eclipse.

The Sun JRE can be downloaded from java.sun.com.

Eclipse

Before installing the PDT plug-in, Eclipse version 3.0 or higher must be installed.

Eclipse comes in two flavors. The *SDK* version contains Java IDE components and is much larger than the *Platform* version. If you only want to use Eclipse as a PPL IDE the Platform version is sufficient. If you plan to use the Eclipse for other development, use the SDK version.

Eclipse can be downloaded from www.eclipse.org.

Plug-in Installation

Linux Setup

To use the compiler under the PPL WorkBench, unzip the file pdt.zip into the Eclipse root directory. Copy the remaining contents of the /host/tools/Linux to a directory in the system path on an x86 Linux host computer. This may be the same host as that providing the above network services, but does not have to be.

Windows Setup

Execute the program, `ppl-setup.exe` contained in the /host/tools/windows directory. This installation file installs the PPL Compiler under C:\Program Files\ppl, the PPL WorkBench plugins under the eclipse directory (typically, C:\eclipse) and the PPL debugger in the Java Runtime home directory (typically C:\Program Files\Java\j2re1.2_xx). Be sure to set the system path prior to using the workbench.

Running the Compiler

In order to build programs for the simulator, debugger or PPL VM, the PPL compiler must be included in the path. In Windows, the *ppl_setup.exe* installer will place the compiler under <*Program Files*>*ppl*.

The compiler can be invoked standalone via the command line once the compiler is included in the shell path.

Running Eclipse

To run Eclipse, double-click on *eclipse.exe* or the eclipse binary found in the eclipse root directory.

PPL Blade Software Configuration

Booting Using an Embedded PrPMC

No additional configuration is required. However, when using the PrPMC, special care must be taken prior to powering-down or hot-swapping the NPU blade. **Please refer to the *Important PrPMC Shutdown Procedure* in this document.**

Booting Using a User-Supplied Server

The PPL Blade requires no Linux configuration to execute PPL programs but some files must be copied from the CDROM to a working directory in its root filesystem.

Target Files Setup

Copy the files from the /target directory on the CDROM to the /home/ppl directory in the root filesystem image on the NFS server. If the directory names used in the previous host configuration instructions were used, then the following command would accomplish this:

```
$ cp /mnt/cdrom/target/* /pplfs/home/ppl
```

Table 6. Copying Target Files

RedBoot Setup

If the same server is used to provide the NPU blade with an IP address (via BOOTP) and the remote file system (using NFS), no further configuration is required.

However, if separate BOOTP and NFS serves are used, the NPU blade's RedBoot script must be changed.

To change the RedBoot startup script (this is required), the serial console must be displayed in a terminal emulator. After powering on the PPL blade, RedBoot will execute, and eventually (after several minutes) will print:

```
== Executing boot script in 2.000 seconds - enter ^C to abort
```

Table 7. RedBoot Startup Script Prompt

Type <CTRL>-c at that time (you have two seconds to do so) to get to the RedBoot command prompt. The following dialog can then take place (commands and responses that you type are in bold):

```
== Executing boot script in 2.000 seconds - enter ^C to abort
^C
ITP-RedBoot> fconfig –i
Initialize non-volatile configuration - are you sure (y/n)? y
Run script at boot: false true
Boot script:
Enter script, terminate with empty line
>> fis load linux
>> exec -c "console=ttyS00,115200n8
nfsaddrs=192.168.100.108:192.168.100.43:0.0.0.0:255.255.255.0:ipfhost1::
nfsroot=192.168.100.43:/opt/hardhat/devkit/arm/xscale_be/target3"
0x0d008000
>> <Enter>
Boot script timeout (1000ms resolution): 0 5
Use BOOTP for network configuration: true <Enter>
BOOTP requests numbers <1..30>: 3 20
GDB connection port: 9000 <Enter>
Network debug at boot time: false <Enter>
Skip POST execution: false <Enter>
Update RedBoot non-volatile configuration - are you sure (y/n)? y
... Unlock from 0xcdfc0000-0xcdfe0000: .
... Erase from 0xcdfc0000-0xcdfe0000: .
... Program from 0x0017a2f0-0x0017b2f0 at 0xcdfc0000: .
... Lock from 0xcdfc0000-0xcdfe0000: .
ITP-RedBoot>
```

Table 8. RedBoot Startup Script

The "exec" command needs the "-c" parameter to pass a command line to the kernel. The command line is contained within double quotes and is all on one line.

The "nfsaddrs" parameter takes the following form:

```
nfsaddrs=<client IP>:<server IP>:<gateway
IP>:<netmask>:<server hostname>::
```

The "nfsroot" parameter takes the following form:

```
nfsroot=<server IP>:<mount path>
```


Note that there are no whitespace characters *within* the console, nfsaddrs, or nfsroot parameters but there is whitespace (a space character) *between* those parameters.

If you don't specify the "console" parameter, the serial console won't work after the kernel starts up.

A Typing the “res” command (reset) at the prompt will reboot the system and attempt to execute the script. If the host configuration is complete, the blade should boot Linux and mount the root filesystem. If it fails, the blade will crash (the exec command in the startup script causes it to execute code that isn’t there) and will require power cycling of the ATCA chassis to reboot it in order to try again.

Preparing for Use

Before you can invoke SysApp, use the following command to prepare the system for use:

$ ldconfig .

Accessing the Embedded PrPMC

The embedded PrPMC is a standard Pentium®-based embedded computer running Linux. Due to the small form-factor of the PrPMC, there are no connectors for keyboard, mouse, serial, or video devices. As such, network connectivity (e.g., telnet, ssh, ftp, etc) is the sole means for interacting with the PrPMC. The PrPMC’s IP address is statically set to 192.168.20.200.

Logging into the PrPMC

To Login to the PrPMC:

```
telnet 192.168.20.200
> Login = root
> Password  = (no password required)
```

Figure 4 Logging into the PrPMC

Important PrPMC Shutdown Procedure

Since the PrPMC uses CompacFlash for its file system, special care must be taken when powering-off or shutting-down. In order to prevent damaging the CompactFlash, the following steps **must** be taken prior to powering-off the chassis or hot-swapping the NPU blade:

Login to the PrPMC
Telnet 192.168.20.200
> Login = user
> Password = password

SU to root:
> su root
> Password = password

Change directories and shutdown the system
> /sbin/shutdown –h now

Wait 1 minute before powering-down or hot-swapping

Figure 5 Shutdown Procedure

