Newconfig: a dynamic-configuration framework for FreeBSD

Atsushi Furuta
Software Research Associates, Inc.
<furuta@sra.co.jp>

Jun-ichiro Hagino
Research Laboratory, Internet Initiative Japan Inc.
<itojun@itojun.org>

June 9, 1999

Overview

- What is newconfig?
- Motivations
- Design
- Implementation
- Future work
- New-bus vs. newconfig
- Conclusion

What is newconfig?

- Originnaly developed by Chris Torek in 4.4BSD.
- NetBSD, OpenBSD, BSD/OS
- We port this framework to FreeBSD-current

Motivations

- PAO development
- CardBus support (hybrid of PCMCIA and PCI)
- There is no IRQ abstraction.
- There is no way to give configuration hint to PCI devices. (such as PCIC on PCI bus)

Design

- A target is to merge to FreeBSD-current

  ⇒ develop based on FreeBSD-current

- To implement dynamic configuraition.

  ⇒ add feature to handle kernel device driver tree dynamically.

- Support any drivers and any buses.

Design (continued)

To remove old config ···

- Old auto-configuration mechanism appeared in 4.1BSD.

- config(8) is bus/machine dependent

- config(8) knows bus structure

Design (continued)

Configuration hint and Plug-and-play bus

- No need configuration hint on PnP bus.

  ··· if every devices keep the PnP spec.

- Many rotten devices (or BIOSes).

- Device framework should provide "overriding" configuration hint.

  ··· or driver writers tend to "hard coding"

Design (continued)

To support separation bus-dependent part

`foo.c`	foo driver core
`foo_isa.c`	foo driver ISA dependent part
`foo_pci.c`	foo driver PCI dependent part

Experience AMD 53C974 PCscsi controller.

⇒ 3-days by a beginner of driver programming.

Design (continued)

Our idea of auto-configuration requirements:

1. Configuration hint information to device drivers
2. Bus/device hierarchy information
3. Inter-module dependency information
4. Device name → object file name mapping information

Newconfig handles all them, but static way.

Implementation

config.new(8) reads 2 group of files

→ generate configuration data

→ statically linked to kernel

Implementation (continued)

1. "files" file
   - `/usr/src/sys/conf/files.newconf`
   - `/usr/src/sys/i386/conf/files.i386.newconf`
   - provided by programmer
2. CONFIG file
   - `/usr/src/sys/i386/conf/NEWCONF`
   - provided by user
   - like "GENERIC", or "LINT"

Future work

- Dynamic configuration for newconfig
- When are device configuration information given?

Compile time	(static)
Boot time	(a.k.a. UserConfig)
Run time	(dynamic)

- The weakest point of the current implementation of newconfig.

Future work (continued)

Implementation of dynamic config

- A utility that parse "files" file for dynamic module.
- A module loader with parsed "files" file and config hint.
- Kernel codes that handles loaded config information and registers new bus/device hierarchy.

New-bus vs. newconfig

New-bus

http://www.freebsd.org/~dfr/devices.html

1. Configuration hint information to device drivers
2. Bus/device hierarchy information
3. Inter-module dependency information
4. Device name → object file name mapping information

Conclusion

- Static vs. dynamic: both config required
- To provide the way of overriding PnP information
- Explicit syntax of configuration data are useful

References

newconfig information

http://www.jp.freebsd.org/newconfig/

presentation draft

http://home.jp.freebsd.org/~furuta/freenix/