PlayStation®2 IOP Library Overview
Release 2.4

Common Network Configuration Library

Publication date: October 2001

Sony Computer Entertainment Inc.
1-1, Akasaka 7-chome, Minato-ku
Tokyo 107-0052, Japan

Sony Computer Entertainment America
919 E. Hillsdale Blvd.
Foster City, CA 94404, U.S.A.

Sony Computer Entertainment Europe
30 Golden Square
London W1F 9LD, U.K.

Summary Table of Contents

About This Manual

This is the Runtime Library Release 2.4 version of the *PlayStation®2 IOP Library Overview - Common Network Configuration Library* manual.

The purpose of this manual is to provide overview-level information about the PlayStation®2 IOP common network configuration library. For related descriptions of the PlayStation®2 IOP netcnf library structures and functions, refer to the *PlayStation®2 IOP Library Reference - Common Network Configuration Library.*

Changes Since Last Release

Chapter 1: NETCNF

- In the "type" section of "ATTACH_CNF Keywords", a description of type nic has been added.

Related Documentation

Library specifications for the EE can be found in the *PlayStation®2 EE Library Reference* manuals and the *PlayStation®2 EE Library Overview* manuals.

Note: the Developer Support Web site posts current developments regarding the Libraries and also provides notice of future documentation releases and upgrades.

Typographic Conventions

Certain Typographic Conventions are used throughout this manual to clarify the meaning of the text:

Convention	Meaning
`courier`	Indicates literal program code.
italic	Indicates names of arguments and structure members (in structure/function definitions only).
medium bold	Indicates data types and structure/function names (in structure/function definitions only).
blue	Indicates a hyperlink.

Developer Support

Sony Computer Entertainment America (SCEA)

SCEA developer support is available to licensees in North America only. You may obtain developer support or additional copies of this documentation by contacting the following addresses:

Order Information	Developer Support
In North America:	*In North America:*
Attn: Developer Tools Coordinator Sony Computer Entertainment America 919 East Hillsdale Blvd. Foster City, CA 94404, U.S.A. Tel: (650) 655-8000	E-mail: PS2_Support@playstation.sony.com Web: http://www.devnet.scea.com/ Developer Support Hotline: (650) 655-5566 (Call Monday through Friday, 8 a.m. to 5 p.m., PST/PDT)

Sony Computer Entertainment Europe (SCEE)

SCEE developer support is available to licensees in Europe only. You may obtain developer support or additional copies of this documentation by contacting the following addresses:

Order Information	Developer Support
In Europe:	*In Europe:*
Attn: Production Coordinator Sony Computer Entertainment Europe 30 Golden Square London W1F 9LD, U.K. Tel: +44 (0) 20 7859-5000	E-mail: ps2_support@scee.net Web: https://www.ps2-pro.com/ Developer Support Hotline: +44 (0) 20 7859-5777 (Call Monday through Friday, 9 a.m. to 6 p.m., GMT)

Chapter 1:
Common Network Configuration Library

NETCNF Application Overview

netcnf.irx is a resident library that provides functions for managing the configuration management file and configuration files.

In addition to defining structures that correspond to configuration files, netcnf.irx provides functions for obtaining a list of configuration files that are saved on a memory card, reading the contents of a configuration file into structures in memory, etc.

The common network configuration library (netcnf.irx) is used to create network configuration applications for handling network configuration files, which are shared resources. Standards for creating these must be observed.

For creating network configuration applications, see the information under the EE/IOP common document directory (general/).

- Network Configuration Application Creation Standards (setapp_r)

NET Configuration File Overview

Individual information required for a network connection such as the IP address within the network or the telephone number of the access point for a dialup connection is maintained for each individual Playstation 2 as a configuration file conforming to the specifications in this document. All programs that create, edit, or interpret configuration files must conform to these specifications.

NET Configuration File Types

There are three types of NET configuration files as shown below. The keywords that can be specified within a configuration file are defined for each type of configuration file.

Table 1-1

File Type	Contents
NET_CNF	Information related to lines and filenames of other configuration files
ATTACH_CNF	Configuration information related to an individual network interface, that is, specifications of processing to be performed when a network device is connected
DIAL_CNF	Information about the dialing procedure of an individual modem

Undefined Keywords

To prepare for the addition of new keywords in the future, a program that edits NET configuration files must maintain any undefined or unsupported keywords that appear in the original file asis, without making changes.

Configuration Processing

Processing for setting up the configuration by interpreting the contents of NET configuration files is performed by starting up netcnf.irx. For details, refer to the netcnf.irx document.

Method for starting up netcnf.irx

Load netcnf.irx using the following method:

netcnf.irx icon=<icon-path> iconsys=<iconsys-path>

Syntax of NET Configuration Files

This section explains the basic rules for describing NET configuration files.

General Format

NET configuration files are interpreted in units of “lines”, that terminate with either a “\n” (LF:0x0a), or the end of the file. However, if “\\” (0x5c) occurs immediately before “\n”, the two bytes for that “\\” and “\n” are ignored, and the next line is treated as if it were a logical continuation.

Generally, a line within a NET configuration file has the following format.

```
keyword  arg1  arg2 ...
```

A keyword and its arguments can be described in a single set on one logical line. The keyword and arguments are each separated by one or more consecutive spaces (SP:0x20) or tabs (TAB:0x08). Besides spaces and tabs, any character that is considered to be white space by the isspace() macro, which is defined in the file ctype.h in the IOP development environment, will be treated as a delimiter.

Comments

A “#” at the beginning of a line indicates a comment line. In addition, if a “#” is placed anywhere other than within a string, the portion from "#" until the end of that line will be treated as a comment.

CR Code

All CR codes (0x0d) within a NET configuration file are ignored except when they are specified with an escape.

Strings and Escape Processing Within a String

When an argument is a string, that entire string should be specified enclosed in double quotes ("). The setting for usr_name in the configuration management file need not be enclosed in double quotes ("). Although there are exceptional cases in which strings do not need to be enclosed in double quotes ("), this is generally not the case. Therefore, descriptions related to this condition will be omitted.

The following escape specifications can be used to specify control codes within a string that is enclosed in double quotes (").

Table 1-2

Escape Specification	Corresponding Character
\a	BEL (0x07)
\b	BS (0x08)
\t	TAB (0x09)
\n	LF (0x0a)
\v	VT (0x0b)
\f	FF (0x0c)

Escape Specification	Corresponding Character
\r	CR (0x0d)
\ooo	Octal code specification. *ooo* is a one- to three-digit octal number
\xxx	Hexadecimal code specification. *xx* is a one- or two-digit hexadecimal number
\X	Character *X* specification. *X* is a character other than the above ([^abtnvfr0-9xa-fA-F]), which is mainly used for specifying ‘\’ or ‘"’

When an actual escape character is specified in a configuration file, it should be specified as “\\a” in order to escape the “\”.

Kanji Code

UTF8 should be used for kanji code. Kanji and kana characters must not be used for configuration items (configuration contents) other than configuration names. Only hiragana and katakana can be used in a configuration name. Also, a comma “,” cannot be used because it is treated as a delimiter in the configuration management file.

Header Comment

The following comment must be entered at the beginning of a NET configuration file.

```
# <Sony Computer Entertainment Inc.> comment
```

The contents after the “>” up to the end of the line can be arbitrary.

The header comment is used by netcnf.irx for error detection during individual encoding/decoding. (Although only ATTACH_CNF is subject to individual encoding/decoding, the header comment is required in all NET configuration files.)

NET_CNF Keywords

A NET_CNF-format file, which is the root of a configuration file group, contains individual information for each user and the filenames of other configuration files. The keywords that can be specified within NET_CNF are as follows.

interface

This is a keyword for defining a network interface. More specifically, it is a keyword for specifying ATTACH_CNF. This keyword can be specified multiple times within NET_CNF.

Syntax

interface *“display_name” “ifc_cnf” [“dev_cnf”]*

Description

display_name is the display name that enables the user to distinguish each interface definition in a configuration file creation or editing program. As long as there is no specific limitation, an arbitrary character string can be specified.

ifc_cnf and *dev_cnf* specify the filenames of ATTACH_CNF files in which configuration information related to that network interface is specified. The filenames can be specified as a relative pathname based on the directory in which NET_CNF exists.

For a PPP connection, ISP-dependent configuration information is entered in *ifc_cnf* and modem-dependent configuration information is entered in *dev_cnf*. Separate usage conventions have been established for each individual file.

However, the type in the *dev_cnf* file alone is never merged during connection processing. Only the type in *ifc_cnf* is used.

If a certain keyword is duplicated in the three files NET_CNF, *ifc_cnf*, and *dev_cnf*, the last specification becomes effective. That is, the priority order for interpreting duplicate keywords is NET_CNF < *ifc_cnf* < *dev_cnf*.

chat_additional

This is a keyword for defining an additional script for a modem.

Syntax

chat_additional *"additional_script"*

Description

This is a keyword for defining an additional script to be sent to the modem following the initialization script that is defined by the chat_init keyword for the PPP interface. It is for user-specific support such as adjusting the modem transmission level. The definition in NET_CNF becomes a default additional script that does not depend on the interface. If this keyword is specified in the relevant ATTACH_CNF beforehand, when a definition that depends on an individual interface is required, the setting in NET_CNF will be overwritten.

For information about script specifications, see "Chat Script Specifications."

redial_count

This is a keyword for specifying the upper limit of the redial count.

Syntax

redial_count *redial_count*

Description

This is a keyword for specifying the maximum number of times redialing is to be attempted when the modem driver returns "busy" for the PPP interface. When this keyword is not specified, the default is 0 times and no redialing is attempted.

The specification in NET_CNF becomes the default redial count that does not depend on the interface. If this keyword is specified in the relevant ATTACH_CNF beforehand when the redial count is to be changed for each interface, the setting in NET_CNF will be overwritten.

redial_interval

This is a keyword for specifying the redial interval.

Syntax

redial_interval *redial_interval*

Description

This is a keyword for specifying the delay interval for redialing in seconds. The default when no redial_interval is specified is 10 seconds.

The specification in NET_CNF becomes the default redial interval that does not depend on the interface. If this keyword is specified in the relevant ATTACH_CNF beforehand when the redial interval is to be changed for each interface, the setting in NET_CNF will be overwritten.

outside_number

This is a keyword for specifying the outside line calling string.

Syntax

outside_number *“outside_number”*

Description

This is a keyword for specifying the outside line calling string that is to be dialed at the beginning of a telephone number when outside line calling is used. The default when no “outside_number” is specified is “” (empty string).

outside_delay

This is a keyword for specifying the outside line calling delay string.

Syntax

outside_delay *“outside_delay”*

Description

This is a keyword for specifying the outside line calling delay string that is to be inserted between the outside line calling string and the telephone number when outside line calling is used. Although this depends on the model and configuration of the modem, for many AT command modems, a comma “,” means a delay of approximately 2 seconds. The default when no “outside_delay” is specified is “” (empty string).

The specification in NET_CNF becomes the default outside line calling delay string that does not depend on the interface. If this keyword is specified in the relevant ATTACH_CNF beforehand when the outside line calling delay string is to be changed for each interface, the setting in NET_CNF will be overwritten.

This outside line calling delay string is not used when the outside line calling string (outside_number) is an empty string.

dialing_type

This is a keyword for specifying the line type.

Syntax

dialing_type [tone | pulse | any]

Description

The line type is specified with the following arguments.

Table 1-3

Argument	Meaning
tone	Tone line (analog) [default]
pulse	Pulse line (analog)
any	Line other than the above (such as digital)

The specification in NET_CNF becomes the default line type that does not depend on the interface. If this keyword is specified in the relevant ATTACH_CNF beforehand when the line type is to be changed for each interface, the setting in NET_CNF will be overwritten.

ATTACH_CNF Keywords

ATTACH_CNF is the configuration file format corresponding to the NETDEV layer. The keywords that are specified in ATTACH_CNF are divided into two groups. One group consists of keywords for specifying the NETDEV layer modules and network devices to which the configuration file is to be applied. The keywords that belong to this group are shown below.

Table 1-4

Keyword	Specification Contents
type	NETDEV layer type specification (USB-Ether or PPP)
vendor	Network device’s vendor specification
product	Network device’s product name specification
location	Network device’s destination port specification

The other group consists of keywords for specifying settings related to the network interface and network device. This group has many keywords that are specific to PPP connections (indicated by [*] in the following table).

Table 1-5

Keyword	Specification Contents
dhcp	Whether or not to use DHCP
dhcp_host_name	Specifies the host name to be reported to the DHCP server
dhcp_host_name_null_terminated	Specifies whether the terminating NUL character is to be included in the host name that is reported to the DHCP server
dhcp_release_on_stop	Specifies whether to report the release of the IP address to the DHCP server when an interface goes down
address	Specifies the IP address
netmask	Specifies the subnet mask
nameserver	Specifies the name server
route	Specifies routing information
phy_config	Specifies the method of configuring the physical layer chip
mtu	Specifies the default values for want.mru, allow.mru and the value of MTU
pppoe	Specifies whether PPPoE (PPP over Ethernet) is enabled
pppoe_service_name	Specifies the service name for PPPoE (PPP over Ethernet)
pppoe_ac_name	Specifies the access concentrator name for PPPoE(PPP over Ethernet)

Keyword	Specification Contents
pppoe_host_uniq_auto	Specifies whether host unique data should be automatically created to distinguish clients for PPPoE(PPP over Ethernet)
chat_additional	Definition of additional script for the modem
redial_count	Specifies the upper limit of the redial count
redial_interval	Specifies the redial interval
outside_delay	Specifies the outside line calling delay string
dialing_type	Specifies the line type
phone_number	Specifies the calling destination telephone number (*)
answer_mode	Specifies whether or not to establish a network connection because of an incoming call
answer_timeout	Specifies the timeout interval when the connection is established by answering
chat_login	Specifies the login procedure (*)
auth_name / auth_key / peer_name / peer_key	Authentication name and authentication key (*)
want.auth / allow.auth	Authentication method (*)
lcp_timeout	LCP timeout interval (*)
ipcp_timeout	IPCP timeout interval (*)
idle_timeout	Idle timeout interval (*)
connect_timeout	Line connection timeout interval (*)
mru_nego / mru	Maximum reception data length (*)
accm_nego / accm	Asynchronous control character map (ACCM:Async-Control-Character-Map) negotiation (*)
magic_nego	Magic number negotiation (*)
prc_nego	Protocol field compression (PRC:Protocol-Field-Compression) negotiation (*)
acc_nego	Address and control field compression (Address-and-Control-field-Compression) negotiation (*)
address_nego / ip_address / ip_mask	IP address and subnet mask negotiation
vjcomp_nego	VJ compression negotiation (*)
dnsn_nego / dnsn	Name server negotiation (*)
force_chap_type	Limit the CHAP authentication algorithms that are allowed
omit_empty_frame	Empty frame omission (*)
log_flags	Log display contents (*)

The PPP connection keywords are further divided into those for making settings related to the ISP (those that should be entered in *ifc.cnf*) and those for making settings related to the modem (those that should be entered in *dev.cnf*). Separate usage conventions have been established for each individual file.

type

This is a keyword for specifying the type of layer that is supported by ATTACH_CNF among the layers that are beneath the INET layer.

Syntax

[-]type [eth | ppp | nic]

Any of the following can be specified.

-type	No lower level layer type specification [default]
type eth	Supports USB-Ethernet
type ppp	Supports PPP connections
type nic	Supports Ethernet (Network Adaptor)

vendor

This is a keyword for specifying the vendor name of an interface device supported by ATTACH_CNF.

Syntax

[-]vendor [*vendor*]

Any of the following can be specified.

-vendor	No vendor specification [default]
vendor *vendor*	Supports only devices having vendor name “*vendor*”

Description

The vendor name is not the one that is defined as a name with the USB String Descriptor. It is the string specified by the driver of the device.

product

This is a keyword for specifying the product name of a device supported by ATTACH_CNF.

Syntax

[-]product [*product*]

-product	No product name specification [default]
product *product*	Supports only devices having product name “*product*”

Description

The product name is not the one that is defined as a name with the USB String Descriptor. It is the string specified by the driver of the device.

location

This is a keyword for specifying location information, namely the port to which the device is connected, of a device supported by ATTACH_CNF.

Syntax

[-]location [*location*]

Any of the following can be specified.

-location	No location specification [default]
location *location*	Supports only the device at the location specified by "*location*"

Currently, *location* is represented by the following kinds of strings.

Table 1-6

location	Meaning
"USB-1"	USB port 1 of the PlayStation 2 console
"USB-2"	USB port 2 of the PlayStation 2 console
"USB-1,3"	Port 3 of the Hub connected to USB port 1 of the PlayStation 2 console

Description

The *location* specification is provided for supporting situations when more than one instance of the same product made by the same manufacturer is to be used simultaneously. For practical purposes, it is probably unnecessary.

dhcp

This is a keyword for specifying whether or not a setting such as the IP address is to be made using DHCP.

Syntax

[-]dhcp

Any of the following can be specified.

-dhcp	DHCP is not used. [default]
dhcp	DHCP is used.

dhcp_host_name

This is a keyword for specifying the host name to be reported to the DHCP server.

Syntax

Any of the following can be specified.

-dhcp_host_name	No host name is reported [default]
dhcp_host_name *host_name*	*host_name* is reported as the host name

Description

Depending on the DHCP server, a host name specified from the server side may have to be included in DHCP messages. This is a keyword for specifying the host name for this kind of DHCP server. The host name that is set is only used during DHCP processing.

dhcp_host_name_null_terminated

This is a keyword for specifying whether or not the terminating NUL character is to be included in the host name that is reported to the DHCP server.

Syntax

Any of the following can be specified.

-dhcp_host_namenull_terminated	NUL character is not included in host name
dhcp_host_namenull_terminated	NUL character is included in host name [default]

Description

Although a terminating NUL character essentially need not be included in the host name reported to the DHCP server, since many DHCP clients that are generally used today include a NUL character, a configuration that matches this situation is the default. As long as no particular problem occurs, leave the default unchanged.

dhcp_release_on_stop

This is a keyword for specifying whether or not to report the release of the IP address to the DHCP server when an interface goes down.

Syntax

Any of the following can be specified.

-dhcp_release_on_stop	Release of IP address is not reported [default]
dhcp_release_on_stop	Release of IP address is reported

address

This is a keyword for specifying the IP address.

Syntax

[-]address [<address>]

-address	The IP address is not directly specified [default]
address *address*	The IP address is set to "*address*"

For information about the syntax of "*address*" see the description of sceInetName2Address().

Description

If DHCP is not used and the connection is not a PPP connection, the IP address must be specified directly.

When DHCP is used, even if "address *address*" is specified, the directly specified address is overwritten by the IP address that was assigned by DHCP.

netmask

This is a keyword for specifying the subnet mask.

Syntax

[-]netmask [<netmask>]

-netmask	The subnet mask is not specified directly [default]
netmask *netmask*	The subnet mask is set to "*netmask*"

For information about the syntax of “*netmask*” see the description of sceInetName2Address().

Description

If DHCP is not used and the connection is not a PPP connection, the subnet mask must be specified directly. However, If the subnet mask is Class A, B, or C, the subnet mask is automatically set by INET.

When DHCP is used, even if “netmask *netmask*” is specified, the directly specified subnet mask is overwritten by the subnet mask that was assigned by DHCP.

nameserver

This is a keyword for adding or deleting a name server.

Syntax

nameserver [add | del] *address*

route

This is a keyword for adding or deleting routing information.

Syntax

Any of the following can be specified.

route add *address* [gw address] [netmask *netmask*]

Adds the specified routing information.

route add default gw *address*

Adds the default gateway.

route del *address*

Deletes the specified routing information.

route add default

Adds the default routing information.

phy_config

This is a keyword for specifying the method of configuring a physical layer chip. It is valid only for an Ethernet-connection ATTACH_CNF(ATTACH_CNF for which type eth is specified).

Syntax

Any of the following can be specified.

-phy_config	The configuration method for the physical layer chip is not specified [default]
phy_config auto	Auto Negotiation Mode
phy_config 10	10BaseT, Half-Duplex
phy_config 10_fd	10BaseT, Full-Duplex, No-Flow-Control
phy_config 10_fd_pause	10BaseT, Full-Duplex, Flow-Control
phy_config tx	100BaseTX, Half-Duplex
phy_config tx_fd	100BaseTX, Full-Duplex, No-Flow-Control
phy_config tx_fd_pause	100BaseTX, Full-Duplex, Flow-Control

mtu

Specifies the default values for want.mru, allow.mru for the PPP and PPPoE(PPP over Ethernet) interfaces, and the value of the MTU for the Ethernet interface.

Syntax

mtu *mtu*

Description

For PPP or PPPoE(PPP over Ethernet), the default values for want.mru as well as allow.mru are specified in *mtu*. If want.mru, allow.mru have been set, the setting of want.mru, allow.mru takes precedence. The MTU value of the Ethernet interface is specified in *mtu*. If *mtu* is not specified, it takes a value of 1500 for the PPP and Ethernet interfaces, and a value of 1492 for PPPoE(PPP over Ethernet).

pppoe

Specifies whether or not PPPoE(PPP over Ethernet) is enabled.

Syntax

[-]pppoe

Description

If pppoe is specified, PPPoE(PPP over Ethernet) will be used. If -pppoe is specified, PPPoE(PPP over Ethernet) will not be used. The following keywords beginning with pppoe are valid only when PPPoE(PPP over Ethernet) is used. In all other cases, they are ignored even if they have been specified.

pppoe_service_name

Specifies the service name for PPPoE(PPP over Ethernet).

Syntax

pppoe_service_name *service_name*

Description

If *service_name* is not specified, it will be empty.

pppoe_ac_name

Specifies the access concentrator name for PPPoE(PPP over Ethernet).

Syntax

pppoe_ac_name *ac_name*

Description

If *ac_name* is not specified, it will be empty.

pppoe_host_uniq_auto

Specifies whether host unique data should be automatically created to distinguish clients for PPPoE(PPP over Ethernet).

Syntax

[-]pppoe_host_uniq_auto

Description

If pppoe_host_uniq_auto is specified, host unique data is created automatically and reported to the server. pppoe_host_uniq_auto is specified by default. When -pppoe_host_uniq_auto is specified, host unique data will not be reported to the server.

chat_additional

This is a keyword for defining an additional script for the modem. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

chat_additional “*chat*”

Description

This is a keyword for defining an additional script for the modem for a PPP interface. It is used for user-specific support such as adjusting the modem transmission level.

Although the default additional script is defined in NET_CNF, ATTACH_CNF defines an additional script that depends on an individual interface. If an additional script is defined in ATTACH_CNF, any additional script that was defined in NET_CNF is ignored.

redial_count

This is a keyword for specifying the upper limit of the redial count. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

redial_count *redial_count*

Description

This is a keyword for specifying the maximum number of times redialing is to be attempted when “busy” is returned by the modem driver in a PPP interface. The default when no redial_count is specified is 0, and no redialing is performed.

Although the default redial count is set in NET_CNF, ATTACH_CNF sets a redial count that depends on an individual interface. If the redial count is set in ATTACH_CNF, any redial count that was set in NET_CNF is ignored.

redial_interval

This is a keyword for specifying the redial interval. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

redial_interval *redial_interval*

Description

This is a keyword for specifying the interval in seconds to wait before redialing when “busy” is returned by the modem driver. The default when no “redial_interval” is specified is 10 seconds.

Although the default redial interval is set in NET_CNF, ATTACH_CNF sets a redial interval that depends on an individual interface. If the redial interval is set in ATTACH_CNF, any redial interval that was set in NET_CNF is ignored.

outside_delay

This is a keyword for specifying the outside line calling delay string. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

outside_delay “*outside_delay*”

Description

This is a keyword for specifying the outside line calling delay string that is to be inserted between the outside line calling string and the telephone number when outside line calling is used. Although this depends on the model and configuration of the modem, for many AT command modems, a comma “,” means a delay of approximately 2 seconds. The default when no “outside_delay” is specified is “” (empty string).

Although the default outside line calling delay string is set in NET_CNF, ATTACH_CNF sets an outside line calling delay string that depends on an individual interface. If the outside line calling delay string is set in ATTACH_CNF, any outside line calling delay string that was set in NET_CNF is ignored.

The outside line calling delay string is not used when the outside line calling string is an empty string.

dialing_type

This is a keyword for specifying the line type. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

dialing_type [tone | pulse | any]

Description

The line type is specified with the following arguments.

Table 1-7

Argument	Meaning
tone	Tone line (analog) [default]
pulse	Pulse line (analog)
any	Line other than the above (such as digital)

Although the default line type is set in NET_CNF, ATTACH_CNF sets a line type that depends on an individual interface. If the line type is set in ATTACH_CNF, any line type that was set in NET_CNF is ignored.

phone_number

This is a keyword for specifying the calling destination telephone number. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

phone_number “*phone_number*”

phone_number0 “*phone_number*”

phone_number1 “*phone_number*”

(omitted)

phone_number9 “*phone_number*”

Description

This is a keyword for specifying the calling destination telephone number.

phone_number0 to phone_number9 indicate the calling destination priority order, that is, the redialing order. Redial processing is performed in the following order. First, an attempt is made to call the telephone number specified by phone_number0. If phone_number0 has not been specified or is busy, an attempt is made to call the telephone number specified by phone_number1, and so on. (If only one telephone number is specified, that same number is redialed repeatedly.)

phone_number is equivalent to phone_number0.

answer_mode

This is a keyword for specifying whether or not to establish a network connection due to an incoming call. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which "type ppp" is specified).

Syntax

Any of the following can be specified.

-answer_mode	Connection is established due to calling [default]
answer_mode	Connection is established due to answering

answer_timeout

This is a keyword for specifying the timeout interval when the connection is established due to answering. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which "type ppp" is specified).

Syntax

answer_timeout *answer_timeout*

Description

This is a keyword for specifying the timeout interval in seconds when the connection is established due to answering. If no "answer_timeout" is specified, the timeout interval will be the same as the calling timeout interval specified by the connect_timeout keyword.

chat_login

This is a keyword for specifying the script string that specifies the login procedure. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which "type ppp" is specified).

Syntax

chat_login "*chatscript*"

Description

The default when no "chatscript" is specified is "", which means that no login procedure is executed.

For information related to the specifications for this chat script, see "Chat Script Specifications".

auth_name, auth_key, peer_name, and peer_key

These are keywords for specifying the authentication name and authentication key. They are valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which "type ppp" is specified).

Syntax

auth_name "*auth_name*"

auth_key "*auth_key*"

peer_name "*peer_name*"

peer_key "*peer_key*"

Description

The user name and password that are assigned by a general ISP correspond to auth_name and auth_key. peer_name and peer_key are the authentication name and authentication key of the connection destination. No default setting exists for any of these keywords. Each parameter is used as follows according to the authentication method.

- When the remote side issues a PAP authentication request to the local side (remote side is the server)

  auth_name Sent as authentication name

  auth_key Sent as authentication key

- When the local side issues a PAP authentication request to the remote side (remote side is the server)

  peer_name Compared with received authentication name

  peer_key Compared with received authentication key

- When the remote side issues a CHAP authentication request to the local side (local side is the server)

  auth_name Authentication name to be sent in Response

  auth_key Authentication key to be used in Response digest

  peer_name Compared with remote side name that is sent in Challenge (if peer_name is "*", no comparison is performed)

- When the local side issues a CHAP authentication request to the remote side (local side is the server)

  peer_name Authentication name to be used for Response verification

  peer_key Authentication key to be used for Response verification

  auth_name Local authentication name to be sent in Challenge

want.auth

This is a keyword for specifying the authentication method for issuing a request to the remote side. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which "type ppp" is specified).

Syntax

want.auth [any | pap | chap | pap/chap | chap/pap]

Description

This keyword specifies the authentication method for issuing a request to the remote side.

Table 1-8

Argument	Authentication Method
any	Authentication by PAP or CHAP is not requested [default]
pap	Only authentication by PAP is requested
chap	Only authentication by CHAP is requested
pap/chap	First authentication by PAP is requested, and if the resulting connection is rejected, then authentication by CHAP is requested next
chap/pap	First authentication by CHAP is requested, and if the resulting connection is rejected, then authentication by PAP is requested next

The algorithm for requesting CHAP authentication is MD5.

allow.auth

This is a keyword for specifying the authentication method to be allowed by the local side. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

allow.auth [any | pap | chap | pap/chap | chap/pap]

Description

This is a keyword for specifying the authentication method to be allowed when authentication is requested by the local side.

Table 1-9

Argument	Authentication Method
any	Authentication by PAP or CHAP is not allowed [default]
pap	Only authentication by PAP is allowed
chap	Only authentication by CHAP is allowed
pap/chap	Authentication by PAP or CHAP is allowed
chap/pap	Authentication by PAP or CHAP is allowed

pap/chap and chap/pap have the same meaning. The three algorithms that are allowed for CAHP authentication are MD5, MS (Version 1), and MS (Version 2).

lcp_timeout

This is a keyword for specifying the timeout interval for LCP. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

lcp_timeout *sec*

Description

This is a keyword for specifying the timeout interval for LCP in seconds. The default when no timeout interval is specified is one second.

This timeout interval will be the resend interval when there is no response to a request.

ipcp_timeout

This is a keyword for specifying the timeout interval for IPCP. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

ipcp_timeout *sec*

Description

This is a keyword for specifying the timeout interval for IPCP in seconds. The default when no timeout interval is specified is one second.

This timeout interval will be the resend interval when there is no response to a request.

idle_timeout

This is a keyword for specifying the PPP idle timeout interval. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

idle_timeout *sec*

Description

The timeout interval in seconds is specified for the argument. PPP automatically disconnects the communication with the remote destination if there are no sends or receives during the time that the number of seconds specified here elapses. The default when no timeout interval is specified is 90 seconds.

connect_timeout

This is a keyword for specifying the line connection timeout interval. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

connect_timeout *sec*

Description

A timeout occurs if the modem does not report a connection completion within the number of seconds specified here from the time when the chat_dial and chat_login processing is completed. The default when no timeout interval is specified is 10 seconds.

mru_nego and mru

These are keywords for configuring the MRU (Maximum-Receive-Unit: Maximum receive data length) negotiation. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.mru_nego

[-]allow.mru_nego

want.mru *mru*

allow.mru *mru*

Description

Whether or not MRU negotiation is requested is specified with want.mru_nego, and whether or not MRU negotiation is allowed is specified with allow.mru_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

The maximum receive data length is specified for want.mru and allow.mru. The default value when either of these data lengths is not specified is 1500 bytes.

accm_nego and accm

These are keywords for configuring ACCM (Async-Control-Character-Map: Asynchronous control character map) negotiation. They are valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.accm_nego

[-]allow.accm_nego

want.accm_value *accm*

allow.accm_value *accm*

Description

Whether or not ACCM negotiation is requested is specified with want.accm_nego, and whether or not ACCM negotiation is allowed is specified with allow.accm_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

The accm value is specified for want.accm and allow.accm. The default value when these accm values are not specified are 0 for want.accm and 0xffffffff for allow.accm.

magic_nego

This is a keyword for configuring magic number negotiation. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.magic_nego

[-]allow.magic_nego

Description

Whether or not magic number negotiation is requested is specified with want.magic_nego, and whether or not magic number negotiation is allowed is specified with allow.magic_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

prc_nego

This is a keyword for configuring PRC (Protocol-Field-Compression) negotiation. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.prc_nego

[-]allow.prc_nego

Description

Whether or not PRC negotiation is requested is specified with want.prc_nego, and whether or not PRC negotiation is allowed is specified with allow.prc_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

acc_nego

This is a keyword for configuring ACC (Address-and-Control-Field-Compression) negotiation. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.acc_nego

[-]allow.acc_nego

Description

Whether or not ACC negotiation is requested is specified with want.acc_nego, and whether or not ACC negotiation is allowed is specified with allow.acc_nego. Appending “-” at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

address_nego, ip_address, and ip_mask

These are keywords for configuring IP address and subnet mask negotiation. They are valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.address_nego

[-]allow.address_nego

want.ip_address *address*

want.ip_mask *netmask*

allow.ip_address *address*

allow.ip_mask *netmask*

Description

Whether or not IP address negotiation is requested when a PPP connection is established is specified with want.address_nego, and whether or not IP address negotiation is allowed is specified with allow.address_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

The IP address is determined as follows according to the address_nego, ip_address, and ip_mask specifications and negotiation results.

Table 1-10

Negotiation Request	Negotiation Result	Local IP Address
Request negotiation want.address_nego	Ack	want.ip_address is used
Request negotiation want.address_nego	Nak	The logical AND of the assigned IP address and want.ip_mask is taken, and if that matches the logical AND of want.ip_address and want.ip_mask, that IP address is used
Do not request negotiation -want.address_nego	--	want.ip_address is used

Table 1-11

Negotiation Request	Negotiation Result	Remote IP Address
Allow negotiation allow.address_nego	Ack	allow.ip_address is used
Allow negotiation allow.address_nego	Nak	The logical AND of the IP address that was sent and allow.ip_mask is taken, and if that matches the logical AND of allow.ip_address and allow.ip_mask, that IP address is used
Do not allow negotiation - allow.address_nego	--	allow.ip_address is used

The default value of want.ip_address or allow.ip_address is “0.0.0.0”, and the default value of want.ip_mask and allow.ip_mask is “0.0.0.0”.

vjcomp_nego

This is a keyword for configuring negotiation related to VJ compression. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.vjcomp_nego

[-]allow.vjcomp_nego

Description

Whether or not VJ compression negotiation is requested is specified with want.vjcomp_nego, and whether or not VJ compression negotiation is allowed is specified with allow.vjcomp_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

dns1_nego, dns1, dns2_nego, and dns2

These are keywords for configuring negotiation related to the name server. They are valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]want.dns1_nego [-]want.dns2_nego

[-]allow.dns1_nego [-]allow.dns2_nego

want.dns1 *address* want.dns2 *address*

allow.dns1 *address* allow.dns2 *address*

Description

Whether or not name server negotiation is requested is specified with want.dns1_nego or want.dns2_nego, and whether or not name server negotiation is allowed is specified with allow.dns1_nego or allow.dns2_nego. Appending “-“ at the beginning of each keyword means that negotiation is not requested or allowed. Not appending “-“ means that negotiation is requested or allowed.

The IP address of the name server is specified for want.dns1, want.dns2, allow.dns1, and allow.dns2.

Although an unlimited number of DNS addresses can be set in INET, at most two DNS addresses per network interface can be reported from the PPP layer to the INET layer.

force_chap_type

This is a keyword for limiting the CHAP authentication algorithms that are to be allowed. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

force_chap_type [no | md5 | ms | ms-v1 | ms-v2]

Description

This is a keyword for limiting the algorithms when CHAP authentication is allowed. The meanings of the arguments are as follows.

Table 1-12

Argument	Meaning
no	Authentication algorithms are not limited [default]
md5	Limited to only MD5
ms	Limited to only MS (Version 1)
ms-v1	Limited to only MS (Version 1)
ms-v2	Limited to only MS (Version 2)

When this keyword is not specified, the authentication algorithms are not limited.

This keyword is provided for forcing a specific algorithm to be used when the connection cannot be made by using the authentication algorithm that was requested by the remote side.

omit_empty_frame

This is a keyword for controlling the transmission of empty frames. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which “type ppp” is specified).

Syntax

[-]omit_empty_frame

Description

If omit_empty_frame is specified, an empty frame (Flag-only frame) is not added immediately after a PPP frame. If -omit_empty_frame is specified by appending “-“ at the beginning of the keyword, an empty frame is added. When this keyword is not specified, an empty frame is added.

During reception, data can be received regardless of whether or not an empty frame exists.

log_flags

This is a keyword for specifying the PPP log display contents. It is valid only for a PPP-connection ATTACH_CNF (ATTACH_CNF for which "type ppp" is specified).

Syntax

log_flags *flag…*

Description

This is a debugging keyword that specifies the PPP log display contents using inetlog.c. Any of the following can be specified for "*flag*". The default value when this keyword is not set is "phase cp auth chat".

Table 1-13

flag	Log Display Contents
phase	PPP state
cp	LCP and IPCP states
auth	PAP and CHAP states
chat	Chat processing and its responses
private	Private information within the chat script
dll	The exchange of data at the DLL level
dump	DLL data packet dump
timer	Timer-related information
event	Events sent to PPP

Note: Each of these flags can also be specified with a numeric value.

DIAL_CNF Keywords

DIAL_CNF is a configuration file for specifying modem-dependent settings using general parameters of a PPP connection.

dialing_type_string

This is a keyword for defining the dialing type string.

Syntax

dialing_type_string "*tone*" "*pulse*" "*any*"

Description

This keyword defines the contents of the special string \D that is used in a chat script. Any of the values "*tone*", "*pulse*", and "*any*" that are specified here will be assigned for the special string \D according to the line type that is used by the user.

For example, by specifying dialing_type_string "ATDT" "ATDP" "ATD", \D will be replaced by "ATDT" when dialing_type is tone within NET_CNF.

redial_string

This is a keyword for specifying the reserved string that is returned from the modem when the circuit is busy.

Syntax

redial_string “*redial_string*”

Description

This is a keyword for specifying the reserved string that is returned from the modem when the circuit is busy. With many AT command modems, this string is “BUSY”.

chat_init

This is a keyword for defining the initialization part of the script for the calling/answering procedure.

Syntax

chat_init “*init_script*”

Description

This keyword defines the script string that specifies the required initialization procedure for both calling and answering. It must contain a description that aborts processing if the string specified by redial_string is received.

For information related to script string specifications, see “Chat Script Specifications.”

chat_dial

This is a keyword for defining the calling procedure.

Syntax

chat_dial “*dial_script*”

Description

This keyword defines the script string that specifies the calling procedure. For information related to script string specifications, see “Chat Script Specifications.”

If -answer_mode is specified in NET_CNF or ATTACH_CNF, the calling script is generated as follows.

init_script + *additional_script* + *dial_script*

chat_answer

This is a keyword for defining the answering procedure.

Syntax

chat_answer “*answer_script*”

Description

This keyword defines the script string that specifies the answering procedure. For information related to script string specifications, see “Chat Script Specifications.”

If answer_mode is specified in NET_CNF or ATTACH_CNF, the answering script is generated as follows.

init_script + *additional_script* + “TIMEOUT *answer_timeout*” + *dial_script*

Chat Script Specifications

A chat script is used to specify the following five procedures for a PPP connection.

- Modem initialization (chat_init)

  Script for setting the timeout interval for a modem connection, setting the abort string, and specifying the modem initialization script.

- Modem additional script (chat_additional)

  Script for performing user-specific configuration for the modem.

- Modem calling script (chat_dial)

  Script for specifying calling processing. This script is supplied by the modem manufacturer together with the modem driver.

- Modem answering script (chat_answer)

  Script for specifying answer processing. This script is supplied by the modem manufacturer together with the modem driver.

- Login procedure (chat_login)

  Script used when a login procedure for exchanging strings without PPP authentication is required.

Script Specifications

Each script is specified as a single string. Note that when you specify a script in a file, you generally must specify the slash "\" as an escaped double slash "\\" as is done when coding in C.

General Syntax

Generally, strings for receiving and transmitting are specified alternately in a script, separated by spaces.

```
recv1 send1 recv2 send2
```

In this example, processing waits until "recv1" is received, then "send1" is sent. Next, processing waits until "recv2" is received, then finally "send2" is sent.

Special Strings

Within a script, the following special characters are interpreted as if they were replaced with the corresponding entries shown below.

Table 1-14

Special Character	Meaning
\s	Space (SP:0x20)
\t	Tab (HT:0x09)
\r	Return (CR:0x0d)
\n	Line feed (LF:0x0a)
\U	Authentication name (auth_name)
\P	Authentication key (auth_key)
\T	Telephone number (phone_number)
\D	Line type-specific dialing string
^*X*	CTRL-*X* (0x00-0x1f)
\c	\r is not appended at the end of the string (transmission string only)

Special Character	Meaning
\d	Wait 2 seconds (delay) (transmission string only)
\p	Wait 0.25 seconds (pause) (transmission string only)
\X	Character *X* itself (special character escape: transmission string only)

Empty String

To send strings without waiting at all, specify an empty string as the receive string. An empty string is specified by two consecutive double quotes.

Timeout Interval Specification

The upper limit of the interval for waiting for a receive string is specified as follows.

TIMEOUT *n* (*n* is a decimal number, units: seconds)

The default value when no timeout interval is specified is 30 seconds. If the receive string cannot be received within the timeout interval, the entire PPP connection process is terminated with an error.

Retry Specification

Retry processing to be performed if the receive string cannot be received within the timeout interval is specified as follows.

recv1-send1-recv2 send2

If “recv1” is received within the timeout interval, “send2” is sent directly.

If “recv1” was not received within the timeout interval, “send1” is sent and processing waits for “recv2” to be received. If “recv2” is received within the timeout interval, “send2” is sent. If it is not, processing terminates with an error.

Abort String Specification

The following specification can cause processing to be terminated with an error when a given specified string is received.

ABORT *string*

Chat Script Setup Examples

In the following setup examples, one pair consisting of receive and send strings has been specified per line. This simplifies verification and helps reduce specification errors.

Although a NET configuration file can be written directly as text, these scripts are shown in the specification format used in C-language programs to facilitate explanations.

```
chat_init =                  // < Initialization procedure >
     TIMEOUT 60              // Sets timeout interval to 60 seconds
     “ ABORT NO\\sCARRIER”   // Terminates with an error if “NO CARRIER” is
     received"
     ABORT ERROR"            // Terminates with an error if "ERROR" is received
     “ ABORT BUSY”           // Terminates with an error if "BUSY" is received
     “ ABORT DELAYED”        // Terminates with an error if "DELAYED" is received
     “ \"\" \\dA\\p"”        // Sends "AT\r" with a delay
     “ OK \"\"";             // Waits for "OK"

chat_additional =            // < Additional procedure >
```

```
          "";                        // Sets an empty string when nothing is to be done

     chat_dial =                     // < Calling procedure >
          "\"\" \\D\\T"              // Initiates a call
          " CONNECT \\d\\c";         // Waits for "CONNECT" and then waits an additional
          2 seconds

     chat_answer =                    // < Answering procedure >
          "RING ATA"                  // Waits for "RING" and then sends "ATA"
          " CONNECT \\d\\c";          // Waits for "CONNECT" and then waits an additional
          2 seconds

     chat_login =                     // < Login procedure >
          "TIMEOUT 10"                // Sets timeout interval to 10 seconds
          " ogin:--ogin: \\U"         // Waits for "ogin:" to be resent and sends
          authentication name
          " ssword: \\P"              // Waits for "ssword:" and sends authentication key
          " PPP\\sstart";             // Finally waits for "PPP start"
```

Precautions

- As long as “\c” is not specified in the send string, “\r” will be automatically appended to the end when the string is sent. However, this kind of automatic appending processing is not performed with receive strings.
- The maximum length of a send or receive string is 255 bytes. There is no specific limit (in the NET configuration file specifications) on the length of the entire script.
- The receive string buffer is cleared when the send string is sent.