

FAX Programmer's Guide

for Windows 2000

Copyright © 2000 Seoul Commtech

PRINTED ON RECYCLED PAPER

Table of Contents

About This FAX Guide	6
FAX Guide Overview	
When To Use This FAX Guide	
FAX Product Terminology	
1. FAX Introduction	9
1.1. FAX Overview	9
1.2. FAX Library Interface Features.....	9
1.3. SCT Fax API Model	9
1.3.1. FAX Model for Stand-alone Applications	
1.3.2. FAX Model for SCbus Applications	
1.4. Voice/Fax Integration	11
1.5. FAX Library Data Structures.....	12
1.5.1. DF_IOTT Data Structure	
1.5.2. DF_UIO Data Structure - User Defined I/O	
1.5.4. DF_DCS Data Structure	
1.5.5. DF_DIS Data Structure.	
2. FAX Sending and Receiving	14
Fax Transfer Session Overview	
2.1. Normal Fax Transmission.....	14
2.2. Five Phases of a Fax Call	15
2.3. Opening and Closing a FAX Channel.....	16
2.4. Data Encoding Schemes	17
2.5. FAX File Formats.....	17
2.5.1. TIFF/F Files	
2.5.2. Raw Files.	
2.6. FAX Synchronous/Asynchronous Mode Operation	18
2.6.1. Synchronous Mode Operation	
2.6.2. Asynchronous Mode Operation	
3. FAX Library Function Overview.....	20
3.1. FAX Main Library Function Overview	20
3.1.1. Transmit Fax	
3.1.2. Receive Fax	
3.1.3. Set Initial Fax State	
3.1.4. Initialize DF_IOTT	

3.1.5. Configuration	
3.1.6. Extended Attribute	
3.1.7. Resource Management	
3.1.8. SCbus Routing	
3.1.9. Miscellaneous	
3.2. FAX Convenience Function Overview.....	25
3.3. FAX Function Reference Information Overview.....	26
3.4. FAX Library Error Handling	27
General Error Handling	
Synchronous Mode - Error Handling	
Asynchronous Mode - Error Handling	
3.5. FAX Include and Library Files.....	29
Include Files	
Library Files	
4. FAX Main Library Function Reference	31
FPMX_BADIOTT() - returns a pointer to an invalid DF_IOTT	31
FPMX_BADPAGE() - returns the fax page number (if error during processing)	33
FPMX_BADSCANLINES() - returns the number of bad scan lines	35
FPMX_BSTAT() - returns a bitmap to indicate Phase B status	38
FPMX_CHTYPE() - returns the FAX channel' s base hardware type.....	42
FPMX_CODING() - returns the most recently negotiated fax encoding scheme	44
FPMX_ESTAT() - returns Phase E information	47
FPMX_FXVERSION() - returns the FAX library version number string.....	49
FPMX_LASTIOTT() - returns a pointer to the last processed DF_IOTT	51
FPMX_PGXFER() - returns the number of transferred fax pages	53
FPMX_PHDCMD() - returns the Phase D command	55
FPMX_PHDRPY() - returns the Phase D reply	58
FPMX_RESLN() - returns the resolution of the page.....	61
FPMX_RTNPAGES() - returns the number of RTN pages	64
FPMX_SCANLINES() - returns the number of scan lines in the last page.....	67
FPMX_SPEED() - returns the fax transfer speed	70
FPMX_STATE() - returns the current state of the FAX channel.....	73
FPMX_TERMMSK() - returns a bitmap of termination reasons	76
FPMX_TFBADTAG() - returns the invalid TIFF/F tag number	79

FPMX_TFNOTAG() - returns missing TIFF/F mandatory tag number.....	81
FPMX_TFPGBASE() - returns the base page numbering scheme.....	83
FPMX_TRCOUNT() - returns the number of bytes transferred	85
FPMX_WIDTH() - returns the decimal value of the negotiated width	87
fpm_close() - closes a SCT FAX channel device	90
fpm_getDCS() - returns the most recent DCS message.....	92
fpm_getDIS() - returns the most recent DIS message.....	96
fpm_getNSF() - returns the remote station's NSF message.....	100
fpm_getparm() - returns the FAX parameter	105
fpm_initstat() - sets the initial fax state.....	109
fpm_open() - opens a FAX channel or board device	112
fpm_rcvfax() - receives fax data	115
fpm_rcvfax2() - receives fax data (file descriptor argument)	133
fpm_sendfax() - transmits fax data	137
fpm_setiott() - sets up default DF_IOTT structure values	156
fpm_setparm() - sets the FAX parameter.....	160
fpm_setuio() - registers user-defined I/O functions	164
fpm_stopch() - forces termination of a fax transmission or reception	167
5. FAX Convenience Function Reference	170
fpm_sendraw() - send a single page of raw fax data.....	171
fpm_sendtiff() - send pages of a single TIFF/F file	175
6. FAX Data Structures.....	179
6.1. FAX Data Structures Overview.....	179
6.2. Declaring FAX Data Structures.....	179
6.3. FAX Library Data Structures Reference.....	180
6.3.1. DF_IOTT - Fax Transmit Data Description	
6.3.3. DF_UIO - User-Defined I/O	
6.3.4. DF_DCS - Digital Command Signal Information	
6.3.5. DF_DIS - Digital Identification Signal Information	
Input to the Library (from Disk Storage)	
Appendix A - TIFF/F Tags and Values.....	186
Output from the Library (to Disk Storage)	
Appendix B - FAX Phase D Status Values	189
Phase D Command (from TRANSMITTER to RECEIVER)	
Phase D Reply (from RECEIVER to TRANSMITTER)	
Appendix C - FAX Phase E Status Values	190

Phase E Status Values Returned to the TRANSMITTER

Phase E Status Values Returned to the RECEIVER.

General Phase E Status Values Returned to RECEIVER or TRANSMITTER ..

Appendix D - FAX Error Codes..... 193

Appendix E - FAX Event Codes 195

About This FAX Guide

NOTE: SCT fax resource guide .

FAX Guide Overview

FAX Programmer's Guide for Windows 2000 windows 2000 Voice/fax fax
application fax parameter reference .
reference .

About This FAX Guide Guide , Guide .

Chapter 1 fax software Overview .

- ? FAX Library Interface Features
- ? SCT Fax API Model
- ? Voice/Fax Integration
- ? FAX Library Data Structure

Chapter 2 .

- ? Normal, polled and turnaround polled fax transmission/reception
- ? Five phases of a fax call
- ? Opening and closing a channel
- ? Fax data encoding schemes
- ? Fax file formats
- ? Fax retransmission capabilities
- ? Fax synchronous/asynchronous mode operation

Chapter 3 .

- ? FAX Main Library Functions
- ? FAX Convenience Functions
- ? FAX function reference information overview
- ? FAX library error handling
- ? Required library and "#include" files for FAX functionality

Chapter 4 FAX Main Library reference .

Chapter 5 FAX Convenience Function reference .

Chapter 6 FAX Library function data structure .

Appendices reference .

? TIFF/Ftags and Values

? FAX Phase D Status Values

? FAX Error Code Values

? Fax Phase E Status Values

? FAX Event Codes

Glossary Index Guide .

When To Use This FAX Guide

SCT System Release software SCT hardware install , Guide
reference .

? FAX library function details application fax

? , polled, turnaround polled fax operation FAX procedure

FAX Product Terminology

Guide .

VPM voice SCT Voice resource .

CT-V04A 4 channel Voice resource 가 .

CT-V08A 8 channel Voice resource 가 .

CT-V16A 16 channel Voice resource 가 .

FPM SCT fax resource .

FPM SCT fax resource . FPM Board 4 fax channel
, Daughter Card CT-V04A 1 , CT-V08A 2
FPM Board가 . Voice Board fax Board

Voice Baseboard	FAX Daughter card
CT-V04A	FPM
CT-V08A	
CT-V16A	

1. FAX Introduction

FAX library SCT FAX API model, Voice Fax , fax library
data structure overview가 .

1.1 FAX Overview

SCT Voice Driver, SCT Standard Run Time library, FAX Function library, Voice FAX
hardware application Voice fax .

1.2 FAX Library Interface Features

- ? channel fax
- ? fax
- ? page message format type fax
- ? file memory raw fax data
- ?
- ? resolution
- ? fax
- ? fax TIFF/F file raw image
- ?
- ? T.30 fax protocol phase B
- ? T.30 fax protocol phase D
- ? Non-Standard Facilities(NSF) 가 application
- ? Digital Command Signal(DCS) 가 application
- ? Digital Information signal 가 application

1.3 SCT Fax API Mode

SCT FAX API application design .

1.3.1. FAX Model for Stand-alone Applications

Stand-alone configuration application channel open , fax call
setup , FAX application FAX API 가 .

NOTE: stand-alone application Voice FAX resource channel
Network interface . time slot routing

Stand-alone application fax application ,

1. Open .
 ? Voice API Voice channel open
 ? FAX API FAX channel open .
2. Voice call setup .
3. fax .
 ? FAX channel .
 ? fax
4. application
5. application open channel close .
 ? Fax channel close
 ? Voice channel close

FAX channel voice channel board channel ,
 voice API fax API channel (Voice, play, record,
 fax send recv) , channel FAX Voice
 I/O func .

1.3.2. FAX Model for SCbus Applications

SCbus application , application resource device
 channel SCbus time slot data SCbus time slot
 , data device time slot listen

- ? fpm_listen()
- ? fpm_unlisten()
- ? fpm_getxmitslot()

SCbus model , fax application , 가 가

, VPM channel fax ,
 , SPM fax .

- (1) VPM channel fax
 channel open .
 ? Voice API Voice channel open ,
 ? fax API fax channel open .

Call setup .

SC routing .

? nr_scroute(vpmdev,SC_CODEC,fxdev,SC_FAX,SC_FULLLDEP);

fax .

? fpm_sendfax(), fpm_recvfax()

CAUTION : fpm_initstat() .

SCbus routing .

Call .

Application .

Application fax channel Voice channel close .

(2) SPM channel fax

channel open .

? SPM API SPM channel open ,

? fax API fax channel open .

Call setup .

SC routing .

? nr_scroute(spmdev,SC_SPM,fxdev,SC_FAX,SC_FULLLDEP);

fax .

? fpm_sendfax(), fpm_recvfax()

CAUTION : fpm_initstat() .

SCbus routing .

Call .

Application .

Application fax channel SPM channel close .

1.4 Voice/Fax Integration

Fax voice/fax application , Voice Driver Library FAX API 가
Voice library Voice software Reference fax library

Guide

SCT Voice Driver Voice Board(VPM) fax Board(FPM)가

Voice fax device channel

? Voice Fax I/O

NOTE: Voice Device channel open close SCT Voice Driver function

FAX library function SCT Voice Drive Standard Runtime Library
, fax application voice fax가 application 가
fax

1.5 FAX Library Data Structures

FAX library data structure

1.5.1. DF_IOTT Data Structure

FAX application DF_IOTT data structure data . DF_IOTT data
structure fax data field . Array linked list
DF_IOTT structure fax call source fax
data . DF_IOTT structure 6.3.1 DF_IOTT fax Transmit
Data Description

1.5.2. DF_UIO Data Structure – User Defined I/O

DF_UIO data structure Windows 2000 I/O read(), write(), lseek()
가 I/O function field 가 I/O function
fpm_setuio() DF_UIO structure install
. fpm_setuio() fpm_setuio reference . DF_UIO data structure
6.3.2 DF_UIO_user-Defined I/O

1.5.3. DF_DCS Data Structure

fpm_getDCS	DF_DCS	structure	T.30	Digital	Command
information	. DF_UIO data structure	6.3.4	DF_ DCS	? Digital	Command
Signal	.				

1.5.4. DF_DIS Data Structure

fpm_getDIS	DF_DIS structure	T.30 Digital Identification Signal
information	. DF_UIO data structure	6.3.5 DF_DIS ? Digital Identification
Signal Information	.	

2. FAX Sending and Receiving

FAX Transfer Session Overview

fax application

fax

- open
- time slot routing
- call setup

fax API 가 fax

Term	Definition
CALLER	call application
CALLED	call application
TRANSMITTER	fax application
RECEIVER	fax application

FAX library application caller가 called가 ,
Transmitter receiver가 .

fax TRANSMITTER , application procedure .

- page document boundary 가
- fax , fax data .

fax RECEIVER , application procedure .

- fax data .

2.1. Normal Fax Transmission

Caller application called application fax connection ,
caller application transmitter가 called application receiver가
fax .

? CALLER application CALLED application fax send

? CALLED application receive

? CALLER application CALLED application fax error
disconnect

CALLER APPLICATION	CALLED APPLICATION
Fax TRANSMITTER: Set initial fax state: CALLER Send function issued (send function completes)	Fax RECEIVER: Set initial fax state: CALLED Receive function issued (receive function completes)
Fax Transfer Status: Fax transmitted to CALLED	Fax Transfer Status: Fax received from CALLER

2.2. Five Phases of a Fax Call

ITU(CCITT) T.30 fax protocol fax session

 fax library data structure negotiation
 send receive SCT fax library

 fax session

- ? Phase A ? fax call set up (begin fax session)
- ? Phase B ? pre -message procedure
- ? Phase C ? message transmission
- ? Phase D ? post-message procedure
- ? Phase E ? fax call release ? disconnect (end fax session)

Phase A station(CALLER CALLED) service
 station dialing

NOTE: CALLER CALLED application connection fax API function

. (Voice software reference manual .)

CALLED party , phase A .

? Fax tone detection

? Digital handshake detection

Phase B , CALLER station fax Transmitter CALLED
station fax Receiver . (fax application fax fax session
send receive application .)

Phase B fax document page data parameter
Transmitter receiver .

Phase C Phase B CALLER CALLED application
parameter fax document .

Phase D fax document page 가
continuation value .

2.3. Opening and Closing a FAX Channel

Channel device operation fpm_open() fax channel
device open .

NOTE: FAX API fpm_open() device handle
application . Function reference section
fpm_open() , fpm_open voice open
.

fpm_open channel open , SCT
Device handle . SCT channel device handle dev , .
int dev;
dev = fpm_open(channel_name, mode)

fax library function , SCT channel device handle
. Channel name channel open , open
action dev handle .

NOTE: fax channel open board -channel open
 . Board channel .

Parent process child process application device handle
 , child process open .

가 FAX channel , SCT C language library

NOTE: FAX channel open , close 4 fpm_open,
 fpm_close reference .

2.4. Data Encoding Schemes

fax data , fax data ITU data encoding
 scheme .

·FPM

- ITU T.4 Group3, 1-Dimensional

Modified Huffman(MH) image 1-Dimensional encoding
 scheme .

2.5. FAX File Formats

Fax data class F Tagged Image File Format , unstructured(raw) format

2.5.1. TIFF/F Files

TIFF/F file fax data Appendix A tag .

TIFF/F file 가 tag .

2.5.2. Raw Files

Modified Huffman encoding raw fax data :

·fax library raw data가 , Least Significant Bit 가
 , byte aligned EOL sequence 가 .

NOTE: raw data Least Significant Bit가 .

Fax data가 raw , unformatted file fax data 1

가 .

Modified Huffman encoding raw fax data :
raw, unformatted file DF_IOTT width, resolution,
encoding scheme 가 fax data byte stream .

2.6. FAX Synchronous/Asynchronous Mode Operation

FAX operation / Overview .

2.6.1. Synchronous Mode Operation

operation application loading ,
channel , application channel .

Fax send receive 가 , fax data
error가 가 , application
channel FAX send fax data .

2.6.2. Asynchronous Mode Operation

mode operation program channel
task가 process . , single call
session task timing sequence .

fax send receive 가 ,
 , fax data .
application . Application
 . application channel open
 , process channel .
application event , channel
state machine . event ,
application .

SCT Standard Runtime library
event . fax SRL
TFX_FAXERROR . SRL ATDV_LASTERR() FAX Extended Attribute
FPMX_ESTAT() error error code . (FAX error
code appendix D , Phase E status value Appendix C

.)

FAX

event . (Event code fpm_sendfax() ,

fpm_rcvfax(), fpm_rcvfax2() reference Appendix E .)

NOTE: 1. application , SRL event

2. fpm_sendfax(), fpm_rcvfax, fpm_rcvfax2() mode argument가

fax

3. fax Library Function Overview

FAX main library	overview, fax convenience function	overview
------------------	------------------------------------	----------

3.1. FAX Main Library Function Overview

Fax main library(libfpm.lib)	SCT Voice Driver	interface	Voice/fax
application	building Blocks		. FAX main library
category			
Transmit Fax	- Transmit a fax document		
Receive Fax	- Receive a fax document or request to receive a fax document(polling)		
Set Initial Fax State	- Set application's initial fax state to either CALLER(transmit) or CALLED(receive)		
Initialize DF_IOTT	- Set defaults for DF_IOTT transfer table Structure		
Configuration	- Set and read FAX parameters		
Extended Attribute	- FAX status		
Resource Management	- Open and close FAX channel devices. Stop FAX resources		
SCbus Routing	- FAX specific SCbus routing functions		
Miscellaneous	- Install user-defined I/O functions Retrieve DCS, DIS and NSF messages		

3.1.1. Transmit FAX

fpm_sendfax()	·DF_IOTT transfer table structure	fax data
	- page TIFF/F fax	
	- raw fax data	
	·CALLER CALLED application	가

Transmitter application		, Transmit fax main library
·linked list, array	DF_IOTT entry	fax document Receiver

- operation function .
- Phase B event generation
- Phase D event generation
- DF_IOTT entry low high resolution .

CALLED application

Transmit Fax main library

- CALLED application linked list array DF_IOTT fax document caller .
- Phase B event generation
- Phase D event generation
- DF_IOTT entry low high resolution .

3.1.2. Receive Fax

fpm_rcvfax()	·fax data , file
	: page TIFF/F fax file
	: raw image data(fpm_rcvfax() page
)
fpm_rcvfax2()	· 가 file name file descriptor가 argument
	fpm_rcvfax()

Receiver application

Receive FAX Main library function

-
- fax data device
- fax data file name(fpm_rcvfax()) file descriptor(fpm_rcvfax2())
- fax data가 format type(TIFF/F or raw)
-
- Phase B event generation
- Phase D event generation
-
- user defined I/O function (fpm_rcvfax2())

3.1.3. Set Initial Fax State

fpm_initstat()	· fax
	(CALLER = transmit state
	CALLED = receive state)

FPMX_LASTIOTT()	·	DF_IOTT structure pointer	
FPMX_PGXFER()	·	page	
FPMX_PHDCMD()	·TFX_PHASED event	Phase D	
	Command		
FPMX_PHDRPY()	·TFX_PHASED event	Phase D	
	reply		
FPMX_RESLN()	·Phase D	page	resolution
FPMX_RTNPAGES()	·fax	Transmitter	RTN
	page		
FPMX_SCANLINES()	·	page	scan line
FPMX_SPEED()	·Phase B가		
	(bound)		
FPMX_STATE()	·FAX device channel		
FPMX_TERMMSK()	·	bitmap	
FPMX_TFBADTAG()	·ATDV_LASTERR()	EFX_BADTAG가	
		TIFF/F tag	
FPMX_TFNOTAG()	· ATDV_LASTERR()	EFA_BADTIFF	
	,	TIFF/F tag	
FPMX_TFPGBASE()	·	TIFF/F file	base page
		numbering scheme	
FPMX_TRCOUNT()	·	fax session	byte
FPMX_WIDTH()	·Phase D가		width
	(line Pixel)		

FAX Extended Attribute FAX Channel device handle parameter 가
, FAX Extended Attribute FAX library file .

NOTE: FAX Extended Attribute case sensitive , .

FAX Extended Attribute .

3.1.7. Resource Management

fpm_open()	·fax channel	board device	open
fpm_close()	·fax channel device	close	
fpm_stopch()	· channel	I/O	stop

Resource Management FAX main library function FAX resource start stop
stop .

FAX library FAX channel device open . fpm_open
, SCT device handle . channel open
, handle device .

fpm_close handle device close .

fpm_open() fpm_close() device busy . device가
busy idle .

- NOTES: 1. FAX device가 process , fpm_open() fpm_close()
FAX device operation
.
2. device handle SCT , windows 2000 file
descriptor가 .
3. process process application , device handle
. Device child process open .

fpm_stopch() FAX channel device stop .

3.1.8. SCbus Routing

fpm_listen()	·FAX Listen channel	SCbus time slot	.
fpm_unlisten()	·SCbus	FAX listen channel	.
fpm_getxmitslot()	·FAX device channel	SCbus time slot	

FAX specific SCbus routing SCbus routing

resource SCbus routing function .

FAX SCbus routing FAX library , SCbus Routing Function
Reference .

3.1.9. Miscellaneous

fpm_getDCS()	·Digital Command Signal data
fpm_getDIS()	·Digital Information Signal data
fpm_getNSF()	·Non-Standard Facilities data
fpm_setuio()	·user-defined I/O function install

fpm_getDCS() miscellaneous function channel 가 T.30
Digital Command signal message . DCS Message Transmitter receiver
Phase B information .

fpm_getDIS() miscellaneous function channel 가 T.30
Digital Information signal message . DIS message Receiver capability
. DIS Phase B , Receiver Transmitter
.

fpm_getNSF() miscellaneous function channel 가 T.30
Non-Standard Facilities message . NSF message fax machine 가
가 message .
message . NSF message
Phase B .

Miscellaneous function fpm_setuio() application read(), write(),
lseek() I/O . DF_UIO() data structure
I/O pointer .

3.2. FAX Convenience Function Overview

FAX main library . FAX convenience
category .
Transmit Fax - TIFF/F raw file

FAX convenience fpm_sendfax DF_IOTT data structure
 source code FAX convenience function reference

Transmitter application	raw data	TIFF/Fax data file
Transmit Fax convenience		

3.3. FAX Function Reference Information Overview

Function Reference FAX function

Reference Header Information

```

Name:
Inputs:                                parameter
Returns:
Includes:                                header file
Category:    가
Mode:        가            mode

```

Description

function reference description

- .
- parameter value
- 가 include , 가 , FAX board
가

Cautions

function reference caution

.

Example

function reference example 가 application

- fax parameter bold .
- .

Source Code

function reference source code가 .

See Also

reference .

3.4. FAX Library Error Handling

error handling

.

General Error Handling

SCT FAX Library .

0 .

-1 .

:

pointer Extended Attribute ,

AT_FAILURE .

Pointer Extended Attribute AT_FAILURE

가 , ATDV_LASTERR() ,
 ATDV_ERRMSGP() , Standard Runtime
 Library Programmer's Guide

NOTES: 1. fpm_open fpm_close error handling rule
 가 ?1 , error.h errno

2. ATDV_LASTERR() EDX_SYSTEM , error.h
 errno

Fax FAX Extended Attribute . Phase E
 Appendix C . Phase E
 reporting mechanism

Synchronous Mode – Error Handling
 FAX library

0
 ?1

FAX library 가 , error code가

SCT error code ATDV_LASTERR()
 errno string pointer ATDV_ERRMSGP()
 . SCT defined FAX error code Appendix D

Asynchronous Mode – Error Handling
 FAX library

0
 ?1

fax library 가 , application flow
 가 , error

event SRL .

fpm_rcvfax() fpm_rcvfax(), fpm_sendfax() event

Event	Description
TFX_FAXERROR	Error in processing
TFX_FAXRECV	Successful completion of fpm_rcvfax() or fpm_rcvfax2()
TFX_FAXSEND	Successful completion of fpm_sendfax()

FAX library 가 , TFX_FAXERROR event
가 .

3.5. FAX Include and Library Files

Include files

FAX library Application .
#include <srllib.h> /* For Voice and FAX development purposes. */
#include <dxxlib.h> /* For voice development purposes. */
#include <faxlib.h> /* For FAX development purposes. */

NOTE: dxxlib.h faxlib.h srllib.h가 include .

Library Files

library가 compile include .
libvpm.lib
libsrl.lib
libfpm.lib

NOTE: link library list가 .

Windows 2000 Library Files

Fax header library file System Release FAX Software .
proto type header file . Voice SRL library header
file library file System Release Voice Software .

file program install hard disk install .

NOTE: faxconv.c file FAX Convenience , application

compile faxconv.c link . Faxconv.c file System
Release diskette , software가 install install .

4. FAX Main Library Function Reference

FAX Main library reference

FAX Convenience function Reference 5

FPMX_BADIOTT() returns a pointer to an invalid DF_IOTT

Name: DF_IOTT * FPMX_BADIOTT(dev)
Inputs: int dev -FAX channel device handle
Returns: DF_IOTT structure pointer
Error , AT_FAILURE
Include: srllib.h
dxxplib.h
faxlib.h
Category: Extended Attribute
Mode: synchronous

Description

DF_IOTT structure, pointer , ,
DF_IOTT structure , TFX_FAXERROR가 , ATDV_LASTERR()
, EFX_BADIOTT .

Parameter	Description
Dev	channel open FAX channel device handle

FPMX_BADIOTT() fax send가 channel

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
```

```

#include <srllib.h>
#include <dxxxlib.h>
#include <faxlib.h>
DF_IOTT iott[10];
DF_IOTT * lastiotp;
long pagenum;
int dev;
extern int errno;
/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/* Call fpm_sendfax( ) after setting up the DF_IOTT array. */
if (fpm_sendfax(dev, iott, EV_SYNC) == -1) {
    /*
     * Get pointer to DF_IOTT being processed when error
     * occurred.
     */
    lastiotp = FPMX_LASTIOTT(dev);
    /*
     * Page being processed within this DF_IOTT when error
     * occurred.
     */
    pagenum = FPMX_BADPAGE(dev);
.
.
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	
·	channel fpm_sendfax()	, EFX_BADIOTT error가

FPMX_BADPAGE() returns the fax page number (if error during processing)

Name:	long FPMX_BADPAGE(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, DF_IOTT structure bad page number , AT_FAILURE
Includes:	srllib.h dxxplib.h Faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

error가 , DF_IOTT structure fax page number

.

NOTE: DF_IOTT structure FAX Extended Attribute

FPMX_LASTIOTT() .

Parameter	Description
-----------	-------------

dev	channel open FAX channel device handle
-----	---

FPMX_BADPAGE()	bad page number fax send가
channel	.

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
```

```

#include <faxlib.h>
DF_IOTT iott[10];
DF_IOTT * lastiotp;
long pagenum;
int dev;
extern int errno;
/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/* Call fpm_sendfax( ) after setting up the DF_IOTT array. */
if (fpm_sendfax(dev, iott, EV_SYNC) == -1) {
    /*
     * Get pointer to DF_IOTT being processed when error
     * occurred.
     */
    lastiotp = FPMX_LASTIOTT(dev);
    /*
     * Page being processed within this DF_IOTT when error
     * occurred.
     */
    pagenum = FPMX_BADPAGE(dev);
    .
    .
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	
·	channel fpm_sendfax()	, EFX_BADPAGE error가

FPMX_BADSCANLINES() returns the number of bad scan lines

Name :	long FPMX_BADSCANLINES(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	page bad scan line
	AT_FAILURE
Includes:	srllib.h
	dxxxlib.h
	faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description	page bad scan line
	page Phase D

Parameter	Description
-----------	-------------

dev	channel open FAX channel device handle
	data T.30 protocol Phase D가 update
. Application	bad scan line Phase D event ,
TFX_PHASED event가	FPMX_BADSCANLINES() .
Phase D event가	, send receive Phase D가
TFA_FAXSEND	TFX_RECV event .
NOTE:	fax session , Phase D가 ,
FPMX_BADSCANLINES()	page bad scan line information

Cautions
None.

Example

```
#include <stdio.h>
```

```

#include <errno.h>
#include <srllib.h>
#include <dxxxlib.h>
#include <faxlib.h>
int dev;
extern int errno;
/* Handler for Phase D events. */
int phd_hdlr( );
main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
.
/*
* Install handler (phd_hdlr( )) using sr_enbhdlr( ) to service
* TFX_PHASED events.
*/
.
.
/*
* Call fpm_rcvfax( ) in asynchronous mode to receive
* TIFF/F file. Set DF_PHASED bit in mode field
* to enable generation of Phase D events.
*/
if (fpm_rcvfax(dev,"fax.tif", EV_ASYNC|DF_PHASED) == -1) {
    printf("Error - %s (error code %d)\n",ATDV_ERRMSGP(dev),
        ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev)==EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
.

```

```

}
/*
 * Handler registered with SRL to handle TFX_PHASED events.
 */
int phd_hdlr( )
{
int dev = sr_getevtdev( );
if (sr_getevtttype( ) == TFX_PHASED) {
    /*
     * Number of bad scan lines of the page just
     * received is available at this point.
     */
    printf("Bad scan lines in page received: %ld\n",
FPMX_BADSCANLINES(dev));
}
.
.
return(0);
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	
·fax session	page가	

returns a bitmap to indicate Phase B status FPMX_BSTAT()

Name:	long FPMX_BSTAT(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, TFX_PHASEB event가 Phase B bitmap AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

Phase B	bitmap .
DFS_REMOTEID	Remote station ID가
DFS_NSF	Remote station NSF(nonstandard facilities) message available
DFS_DIS	DIS (Digital Information Signal) message available (sent from Receiver to Transmitter)
DFS_DCS	DCS (Digital Command Signal) message available (sent from Transmitter to Receiver)

NOTES: 1. FPMX_BSTAT() NSF, DIS, DCS message 가 bit . DCS, DIS, NSF message , fpm_getDCS(), fpm_gdtDIS(), fpm_getNSF() .

Parameter	Description
dev:	channel open FAX channel device handle
	data T.30 protocol Phase D가 update
	. fax session fax session
	.
Phase B information	notify application , Phase B

event(fpm_rcvfax(), fpm_rcvfax2(), fpm_sendfax() DF_PHASEB) enable
Phase B event(TFX_PHASEB)가 FPMX_BSTAT() issue .
(programming example .)

NOTE: 1. fax session Phase B negotiation FPMX_BSTAT()
Phase B negotiation Phase B availability
information .
2. fpm_sendfax(), fpm_rcvfax(), fpm_rcvfax2() 가 ,
SRL sr_enbhdr() Phase B event event
handler .

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
int dev;
extern int errno;
/* Handler for Phase B events. */
int phb_hdlr( );
main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
.
```

```

/*
 * Install handler (phb_hdlr( )) using sr_enbhdlr( ) to service
 * TFX_PHASEB events.
 */
.
.
/*
 * Call fpm_sendfax( ) in asynchronous mode after setting
 * up the DF_IOTT array. Set DF_PHASEB bit in mode field
 * to enable generation of Phase B events.
 */
if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
    printf("Error - %s (error code %d)\n",
           ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
.
}
/*
 * Handler registered with SRL to handle TFX_PHASEB events.
 */
int phb_hdlr( )
{
    int dev = sr_getevtdev( );
    if (sr_getevtttype( ) == TFX_PHASEB) {
        if (FPMX_BSTAT(dev) & DFS_REMOTEID) {
            /*
             * Remote ID available - get remote id using
             * fpm_getparm( ).
            */
            .
            .
        }
    }
}

```



```

        /* Remote data rate capability. */
        printf("Data rate for fax transmission: %ld\n",
               FPMX_SPEED(dev));
    }
    .
    .
return(0);
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	
·	Phase B event가	가

See Also

- fpm_getDCS()
- fpm_getDIS()
- fpm_getNSF()
- fpm_getparm()

FPMX_CHTYPE()

Description

Parameter	Description
-----------	-------------

Cautions

Example

42

```

#include <srllib.h>
#include <dxxxlib.h>
#include <faxlib.h>
int dev;
extern int errno;
main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
.
/* Determine channel hardware type. */
switch(FPMX_CHTYPE(dev)) {
    .
    .
    break;
    .
    break;
}
.
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

See Also

·fpm_sendfax()

FPMX_CODING()returns the most recently negotiated fax encoding scheme

Name:	long FPMX_CODING(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, line enclding scheme AT_FAILURE(or if initial Phase B not complete)
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

channel 가	fax encoding scheme .
.	
DFS_MH	Modified Huffman line encoding scheme
DFS_MR	Modified Read line encoding scheme
DFS_MMR	Modified Read line encoding scheme

Parameter Description

dev:	channel open	FAX channel device handle
	data T.30 protocol Phase D가	update
. fax session		fax session
		.
Phase B information	notify application	, Phase B
event(fpm_rcvfax(), fpm_rcvfax2(), fpm_sendfax())		DF_PHASEB) enable
Phase B event(TFX_PHASEB)가	FPMX_BSTAT()	issue .
(programming example .)		

NOTE: 1. fax session Phase B negotiation FPMX_BSTAT()
 Phase B negotiation Phase B availability

information .
 2. FPM Board Modified Huffman , DFS_MR, DFS_MMR
 reserved .

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
int dev;
extern int errno;
/* Handler for Phase B events. */
int phb_hdlr( );
main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
.
/*
* Install handler using sr_enbhdlr( ) to service
* TFX_PHASEB events.
*/
.
.
/*
* Call fpm_sendfax( ) in asynchronous mode after setting
* up the DF_IOTT array. Set DF_PHASEB bit in mode field
```

```

* to enable generation of Phase B events.
*/
if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
    printf("Error - %s (error code %d)\n",
           ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
.
}
/*
* Handler registered with SRL to handle TFX_PHASEB events.
*/
int phb_hdlr( )
{
    int dev = sr_getevtdev( );
    if (sr_getevttype( ) == TFX_PHASEB) {
        /* Negotiated line encoding scheme. */
        printf("Negotiated data encoding scheme: %ld\n",
               FPMX_CODING(dev));
    }
    .
    .
    return(0);
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

See Also

- fpm_sendfax()
- fpm_sendfax()
- fpm_setparm() (FPM_RXCODING and FPM_TXCODING parameters)

FPMX_ESTAT()

return Phase E information

Name: long FPMX_ESTAT(dev)
Inputs: int dev FAX channel device handle
Returns: , Phase E
, AT_FAILURE
Includes: srllib.h
dxxplib.h
faxlib.h
Category: Extended Attribute
Mode: synchronous

Description

T.30 fax protocol error .

NOTES: 1. fpm_rcvfax(), fpm_rcvfax2(), fpm_sendfax() 가 ?1
disconnection .

2. T.30 protocol error가 , FPMX_ESTAT() 0

Phase E Appendix C .

Parameter Description		
dev:	channel open	FAX channel device handle

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
```

```

int dev;

extern int errno;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/* Call fpm_sendfax( ) after setting up the DF_IOTT array. */
if (fpm_sendfax(dev, iott, EV_SYNC) == -1) {
    printf("Error - %s (error code %d)\n",
        ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    } else if (ATDV_LASTERR(dev) == EFX_DISCONNECT) {
        /*
         * Additional error processing - check Phase E status to
         * determine cause of error during fax protocol.
         */
        printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
    }
}
.

```

Errors

dev FAX channel device가 , AT_FAILURE

FPMX_FXVERSION() returns the FAX library version number string

Name: char * FPMX_FXVERSION(dev)
Inputs: int dev -FAX channel device handle
Returns: , FAX library version number string
 , AT_FAILURE

Includes: srllib.h
 dxxplib.h
 faxlib.h
Category: Extended Attribute
Mode: synchronous

Description
FPMX_FXVERSION() FAX library version number string .

Parameter Description			
dev:	channel	open	FAX channel device handle

Cautions
None.

Example

```
#include <stdio.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int dev;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
```

```

/*
 * Optional display of version number of SCT Fax
 * library.
 */
printf("%s\n", FPMX_FXVERSION(dev));

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	


```

* FAX device handle in dev.
*/
/* Call fpm_sendfax( ) after setting up the DF_IOTT array. */

if (fx_sendfax(dev, iott, EV_SYNC) == -1) {
    /* Get pointer to DF_IOTT being processed when error
    * occurred.
    */
    lastiotp = ATFX_LASTIOTT(dev);
    /*
    * Page being processed within this DF_IOTT when error
    * occurred.
    */
    pagenum = ATFX_BADPAGE(dev);
}

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_PGXFER() returns the number of transferred fax pages

Name:	long FPMX_PGXFER(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, page , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

 fax call fax page

 fpm_rcvfax() , fpm_rcvfax2(), fpm_sendfax() page

 . FPMX_PGXFER() 가 , fax

session page, .

fax session FPMX_PGXFER() count

value channel send receive가 application .

Parameter Description

dev:	channel	open	FAX channel	device handle
------	---------	------	-------------	---------------

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
```

```

int dev;

extern int errno;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/*
 * Call fpm_rcvfax( ) to receive a fax into the file
 * "myfax.tif".
 */
if (fpm_rcvfax(dev, "myfax.tif", DF_TIFF|DF_NOPOLL|EV_SYNC) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    /*
     * Additional error processing - check Phase E status to
     * determine cause of error during fax protocol.
     */
    printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
    .
    .
}
printf("Number of pages received: %ld\n", FPMX_PGXFER(dev));

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

FPMX_PHDCMD()returns the Phase D command

Name:long FPMX_PHDCMD(dev)

Inputs:int dev·FAX channel device handle

Returns:·TFX_PHASED가Phase D command
·AT_FAILURE

Includes:srllib.h
dxxplib.h
faxlib.h

Category:Extended Attribute

Mode:synchronous

Description

FPMX_PHDCMD()Phase D command·Phase D command·

DFS_EOPEnd Of Procedure

DFS_MPSMulti-page Signal

DFS_EOMEnd Of Message

dataT.30 protocolPhase D가update

·Phase D command monitor application TFX_PHASED event가

Phase D event enable FPMX_PHDCMD() issue·Phase D

Phase D event가send receive,

TFX_FAXSEND TFX_FAXRECV event·

Fax session FPMX_PHDCMD()Phase D command

value channel send receive가application

·

Phase D commandAppendix B·

Parameter Description

dev:channel openFAX channel device handle

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int dev;

extern int errno;

/* Handler for Phase D events. */
int phd_hdlr( );

main( )
{
/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/*
 * Install handler (phd_hdlr( )) using sr_enbhdlr( ) to service
 * TFX_PHASED events.
 */
.
.
/*
 * Call fpm_rcvfax( ) in synchronous mode to receive
 * TIFF/F file. Set DF_PHASED bit in mode field
 * to enable generation of Phase D events.
 */
if (fpm_rcvfax(dev,"fax.tif", EV_SYNC|DF_PHASED) == -1) {
    printf("Error - %s (error code %d)\n",ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev)==EDX_SYSTEM) {
```



```

        printf("errno = %d\n", errno);
    }
}

/*
 * Examine Phase D command for last page.
 */
printf("Phase D command: %ld\n", FPMX_PHDCMD(dev));

.
.
}

/*
 * Handler registered with SRL to handle TFX_PHASED events.
 */
int phd_hdlr( )
{
    int dev = sr_getevtdev( );
    if (sr_getevttype( ) == TFX_PHASED) {
        /*
         * Examine Phase D command - e.g., DFS_MPS, DFS_EOM,
         * DFS_EOP.
         */
        phdcmd = FPMX_PHDCMD(dev);
        printf("Phase D command: %ld\n", phdcmd);
    }

    .
    .
    return(0);
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

FPMX_PHDRPY() returns the Phase D reply

Name:	long FPMX_PHDRPY(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, TFX_PHASED가 Phase D reply , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description	
FPMX_PHDRPY()	Phase D replay . Phase D reply .
DFS_MCF	Message confirmation
DFS_RTN	Retrain negative
DFS_RTP	Retrain positive

Phase D reply Appendix B .

Parameter Description	
dev:	channel open FAX channel device handle
	data T.30 protocol Phase D가 update
. Phase D reply	monitor application TFX_PHASED event가
Phase D event	enable FPMX_PHDRPY() . Phase D Phase D
event가	send receive ,
TFX_FAXSEND	TFX_FAXRECV event .
Fax session	FPMX_PHDRPY() Phase D command
value channel	send receive가 application
.	

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
int dev;
extern int errno;

/* Handler for Phase D events. */
int phd_hdlr( );

main( )
{
/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/*
 * Install handler (phd_hdlr( )) using sr_enbhdlr( ) to service
 * TFX_PHASED events.
 */
.
.
/*
 * Call fpm_sendfax( ) in synchronous mode after setting
 * up the DF_IOTT array. Set DF_PHASED bit in mode field
 * to enable generation of Phase D events.
 */
if (fpm_sendfax(dev, iott, EV_SYNC|DF_PHASED) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
```

```

        printf("errno = %d\n", errno);
    }
}

/*
 * Examine Phase D reply for last page.
 */
printf("Phase D reply: %ld\n", FPMX_PHDRPY(dev));

.
.
}

/*
 * Handler registered with SRL to handle TFX_PHASED events.
 */
int phd_hdlr( )
{
    long phdrpy;
    int dev = sr_getevtdev( );
    if (sr_getevttype( ) == TFX_PHASED) {
        /*
         * Examine Phase D reply - e.g., DFS_MCF, DFS_RTN,
         * DFS_RTP.
         */
        phdrpy = FPMX_PHDRPY(dev);
        printf("Phase D reply: %ld\n", phdrpy);
    }
    .
    .
    return(0);
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

FPMX_RESLN() returns the resolution of the page

Name:	long FPMX_RESLN(dev)
Inputs:	int dev FAX channel device handle
Returns:	page resolution , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description	
FPMX_RESLN(dev)	page resolution .
DF_RES_HI	High resolution (fine) - 196
DF_RES_LO	Low resolution (coarse) - 98

Parameter Description	
dev:	channel open FAX channel device handle
	data T.30 protocol Phase D가 update
. Phase D reply	monitor application TFX_PHASED event가
Phase D event	enable FPMX_RESLN() . Phase D Phase D event
가	send receive ,
TFX_FAXSEND	TFX_FAXRECV event .
Fax session	FPMX_RESLN() Phase D command value
channel	send receive가 application .

Cautions
None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

DF_IOTT iott[10];

int dev;

extern int errno;

/* Handler for Phase D events. */
int phd_hdlr( );

main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
/*
* Install handler (phd_hdlr( )) using sr_enbhdlr( ) to service
* TFX_PHASED events.
*/
.
/*
* Call fpm_sendfax( ) in synchronous mode after setting
* up the DF_IOTT array. Set DF_PHASED bit in mode field
* to enable generation of Phase D events.
*/
if (fpm_sendfax(dev, iott, EV_SYNC|DF_PHASED) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
/*
* Resolution of the page just transferred is
```

```

* available at this point.
*/
printf("Page was transferred at resolution: %ld\n", FPMX_RESLN(dev));
.
}
/*
* Handler registered with SRL to handle TFX_PHASED events.
*/
int phd_hdlr( )
{
long phdrpy;
int dev = sr_getevtdev( );
if (sr_getevttype( ) == TFX_PHASED) {
    /*
    * Resolution of the page just transferred is
    * available at this point.
    */
    printf("Page was transferred at resolution: %ld\n",
        FPMX_RESLN(dev));
}
.
return(0);
}

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_RTNPAGES()

Mode: synchronous

TIFF/F file , page bad Fax Lines tag , page
image quality .

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int dev;

extern int errno;

long badpages;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/*
 * Call fpm_rcvfax( ) to receive a fax into the file
 * "myfax.tif".
 */
if (fpm_rcvfax(dev, "myfax.tif", DF_TIFF|EV_SYNC) == -1) {
    printf("Error - %s (error code %ld)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    /*
     * Additional error processing - check Phase E status to
     * determine cause of error during fax protocol.
     */
    printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
```

```

    .
}
/*
 * Check if the received file has any pages for
 * which a RTN was returned.
 */
badpages = FPMX_RTNPAGES(dev);

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_SCANLINES() returns the number of scan lines in the last page

Name:	long FPMX_SCANLINES(dev)
Inputs:	int dev FAX channel device handle
Returns:	page scan line , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description
FPMX_SCANLINES() page scan line page Phase D

Parameter Description
dev: channel open FAX channel device handle
data T.30 protocol Phase D가 update RTN page monitor application TFX_PHASED event가 Phase D event enable FPMX_SCANLINES() Phase D Phase D event가 send receive , TFX_FAXSEND TFX_FAXRECV event
Fax session FPMX_SCANLINES() Phase D command value channel send receive가 application

NOTE: fax session Phase D completion , FPMX_SCANLINES()
completed page scan line total number return

Cautions
None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int dev;

extern int errno;

/* Handler for Phase D events. */
int phd_hdlr( );

main( )
{
/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
/*
 * Install handler (phd_hdlr( )) using sr_enbhdlr( ) to service
 * TFX_PHASED events.
 */
.
/*
 * Call fpm_rcvfax( ) in asynchronous mode to receive
 * TIFF/F file. Set DF_PHASED bit in mode field
 * to enable generation of Phase D events.
 */
if (fpm_rcvfax(dev,"fax.tif", EV_ASYNC|DF_PHASED) == -1) {
    printf("Error - %s (error code %d)\n",ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev)==EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
/*
 * Handler registered with SRL to handle TFX_PHASED events.
```

```

*/
int phd_hdlr( )
{
    int dev = sr_getevtdev( );
    if (sr_getevtttype( ) == TFX_PHASED) {
        /*
         * Total number of scan lines on the page just
         * received is available at this point.
         */
        printf("Total number of scan lines in page received: %ld\n",
            FPMX_SCANLINES(dev) );
    }
    .
    return(0);
}

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_SPEED()

Description	page	speed
FPMX_SPEED()		
Phase D가		

dev:	channel	open	FAX channel	device handle
		data	T.30 protocol	Phase D가
. fax session	,		fax transfer	update
				fax session

```

Phase B negotiation transfer speed monitor application
, Phase B event(fpm_rcvfax(), fpm_rcvfax2(), fpm_sendfax()
DF_PHASEB), Phase B event(TFX_PHASEB)가 FPMX_SPEED()

```

70

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

DF_IOTT iott[10];

int dev;

extern int errno;

/* Handler for Phase B events. */
int phb_hdlr( );

main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
.
/*
* Install handler (phb_hdlr( )) using sr_enbhdlr( ) to service
* TFX_PHASEB events.
*/
.
.
/*
* Call fpm_sendfax( ) in asynchronous mode after setting
* up the DF_IOTT array. Set DF_PHASEB bit in mode field
* to enable generation of Phase B events.
*/
if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
```

```

        if (ATDV_LASTERR(dev)==EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
    }
    .
    .
}
/*
 * Handler registered with SRL to handle TFX_PHASEB events.
 */
int phb_hdlr( )
{
    int dev = sr_getevtdev( );
    if (sr_getevttype( ) == TFX_PHASEB) {
        /* Remote data rate capability. */
        printf("Data rate for fax transmission: %ld\n",
            FPMX_SPEED(dev));
    }
    .
    .
    return(0);
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

FPMX_STATE() returns the current state of the FAX channel

Name:	long FPMX_STATE(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, FAX channel device , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description
FPMX_STATE() dev FAX channel device .

Parameter Description
dev: channel open FAX channel device handle
fax channel 가 .
CS_IDLE FAX channel is idle
CS_SENDFAX FAX channel is transmitting (fpm_sendfax() active)
CS_RECVFAX FAX channel is receving (fpm_rcvfax() or fpm_rcvfax2() active)
CS_FAXIO fax , fax session

Cautions
None.

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
```

```

int dev;

extern int errno;

/* Handler for Phase B events. */
int phb_hdlr( );

main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
.
/*
* Install handler using sr_enbhdlr( ) to service
* TFX_PHASEB events.
*/
.
.
/*
* Check state of the FAX channel.
* If idle, call fpm_sendfax( ) in asynchronous mode after setting
* up the DF_IOTT array. Set DF_PHASEB bit in mode field
* to enable generation of Phase B events.
*/
if (FPMX_STATE(dev) == CS_IDLE) {
    if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
        printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),
            ATDV_LASTERR(dev));
        if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
    }
}
.
.
}

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_TERMMSK() returns a bitmap of termination reasons

Name:	long FPMX_TERMMSK(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, bitmap , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

FPMX_TERMMSK() bitmap .

fpm_rcvfax(), fpm_rcvfax2(), fpm_sendfax()가
 . .

TM_FXTERM	Normal completion of fax send/receive
TM_POLLED	Poll request received from TRANSMITTER
TM_VOICEREQ	Voice request issued/received

 send receive가 .

Parameter Description

dev:	channel	open	FAX channel	device handle
------	---------	------	-------------	---------------

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
```

```

#include <dxxilib.h>
#include <faxlib.h>

int dev;

extern int errno;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
.
/*
 * Call fpm_rcvfax( ) to receive a fax into the file
 * "myfax.tif".
 */
if (fpm_rcvfax(dev, "myfax.tif", DF_TIFF|EV_SYNC) == -1) {
    printf("Error - %s (error code %ld)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    /*
     * Additional error processing - check Phase E status to
     * determine cause of error during fax protocol.
     */
    printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
    .
    .
}
/* Check termination reasons. */
if (FPMX_TERMMSK(dev) & TM_POLLED) {
    printf("Poll received\n");
    /* Respond to poll by issuing a fpm_sendfax( ). */
    .
    .
}
if (FPMX_TERMMSK(dev) & TM_VOICEREQ) {
    printf("Voice request received\n");

```

```

        /* Respond to voice request (PRI_EOP). */
        .
        .
    }

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_TFBADTAG() returns the invalid TIFF/F tag number

Name:	long FPMX_TFBADTAG(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, TIFF/F tag , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

FPMX_TFBADTAG() ATDV_LASTERR() 가 EFX_BADTAG error
 TIFF/F tag . EFX_BADTAG error FAX library ,
TIFF/F tag가 TIFF/F file .

Parameter Description

dev:	channel	open	FAX channel	device handle
------	---------	------	-------------	---------------

fax session	FPMX_TFBADTAG()	Bad TIFF/F tag value
send가 channel	application	.

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott;
int dev;
```

```

extern int errno;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
/* Call fpm_sendfax( ) after setting up the DF_IOTT. */
if (fpm_sendfax(dev, &iott, EV_SYNC) == -1) {
    if (ATDV_LASTERR(dev) == EFX_BADTAG) {
        printf("Bad Tag in TIFF/F file. Tag number %ld\n", FPMX_TFBADTAG(dev));
    }
}
.

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

FPMX_TFNOTAG() returns missing TIFF/F mandatory tag number

Name:	long FPMX_TFNOTAG(dev)
Inputs:	int dev ·FAX channel device handle
Returns:	, tag가 , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description

FPMX_TFNODTAG() ATDV_LASTERR() 가 EFX_BADTIFF error
, TIFF/F tag가 .

Parameter Description

dev:	channel	open	FAX channel	device handle
fax session	FPMX_TFBADTAG()		Bad TIFF/F tag value	
send가 channel		application		.

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott;
int dev;
extern int errno;
```

```

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
/* Call fpm_sendfax( ) after setting up the DF_IOTT. */
if (fpm_sendfax(dev, &iott, EV_SYNC) == -1) {
    if (ATDV_LASTERR(dev) == EFX_BADTIF) {
        printf("Missing Tag in TIFF/F file. Tag number %ld\n",
               FPMX_TFNOTAG(dev));
    }
}
.

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	


```

int dev;

extern int errno;

long tfpgbase;

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
/*
 * Call fpm_sendfax( ) after setting up the DF_IOTT to
 * send the TIFF/F file.
 */
if (fpm_sendfax(dev, &iott, EV_SYNC) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    .
}

/* Determine page numbering scheme. */
tfpgbase = FPMX_TFPGBASE(dev);

Errors
                                AT_FAILUREP
    .dev          FAX channel device handle

```

FPMX_TRCOUNT() returns the number of bytes transferred

Name:	long FPMX_TRCOUNT(dev)
Inputs:	int dev .FAX channel device handle
Returns:	, fax byte , AT_FAILURE
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Extended Attribute
Mode:	synchronous

Description			
FPMX_TRCOUNT()	dev	channel	fax
byte	.		

Parameter Description			
dev:	channel	open	FAX channel device handle

Cautions
None.

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
int dev;
extern int errno;
/* Handler for Phase B events. */
int phb_hdlr( );
main( )
{
```

```

/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX device handle in dev.
 */
.
/*
 * Call fpm_sendfax( ) in synchronous mode after setting
 * up the DF_IOTT array.
 */
if (fpm_sendfax(dev, iott, EV_SYNC) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
/* Check the transfer count */
printf("Transfer count is %d\n", FPMX_TRCOUNT(dev));
.
}

```

Errors

	AT_FAILUREP	.
·dev	FAX channel device handle	

See Also

·vpm_close() (refer to the Voice Software Reference)

application	Phase D event	generation	, FPMX_WIDTH()
page		Phase D event	handler routine

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
int dev;
extern int errno;
/* Handler for Phase D events. */
int phd_hdlr( );
main( )
{
/*
* Open the channel using fpm_open( ) and obtain the SCT
* FAX device handle in dev.
*/
.
/*
* Install handler using sr_enbhdlr( ) to service
* TFX_PHASED events.
*/
.
/*
* Call fpm_sendfax( ) in synchronous mode after setting
* up the DF_IOTT array. Set DF_PHASED bit in mode field
* to enable generation of Phase D events.
*/
if (fpm_sendfax(dev, iott, EV_SYNC|DF_PHASED) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
```



```

        if (ATDV_LASTERR(dev)==EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
    }
    /*
    * Width of the page just transferred is available at
    * this point.
    */
    printf("Page width: %ld\n", FPMX_WIDTH(dev));
    .
}
/*
* Handler registered with SRL to handle TFX_PHASED events.
*/
int phd_hdlr( )
{
    int dev = sr_getevtdev( );
    if (sr_getevttype( ) == TFX_PHASED) {
        /*
        * Width of the page just transferred is available at
        * this point.
        */
        printf("Page width: %ld\n", FPMX_WIDTH(dev));
    }
    .
    return(0);
}

```

Errors

	AT_FAILUREP	.
-dev	FAX channel device handle	

closes a SCT FAX channel device

fpm_close()

Name:	int fpm_close(dev)
Inputs:	int dev . SCT FAX channel device
Returns:	, 0 , -1
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Resource Management
Mode:	synchronous

Description

fpm_close() fpm_open() open SCT fax channel device close .
 device handle , device가 busy idle
 handle .

NOTE: fpm_close FAX event . (channel
 hook , Voice channel device parameter
 .)

FAX channel device close , vpm_close .

Parameter Description

dev:	channel open	FAX channel device handle
------	-----------------	------------------------------

Cautions

fpm_close() handle FAX event .

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
```

```

main()
{
    int dev; /* Fax channel device handle. */
    /* Open the Voice channel device using vpm_open( ). */
    .
    /* Open the FAX channel device. */
    if ((dev = fpm_open("vpmB1C1", NULL)) == -1) {
        /* Error opening device */
        printf("Error opening channel, errno = %d\n", errno);
        exit(1);
    }
    /* FAX transfers (send/receive) calling FAX API functions using dev. */
    .
    /* Close the FAX channel device. */
    if (fpm_close(dev) == -1) {
        /* Error closing device. */
        printf("Error closing channel\n");
        printf("Error - %s (error code %d)\n",
            ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
        if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
        exit(1);
    }
    .
}

```

Errors

가 ?1 . errno ,

EIN Invalid Argument
 EBADF Invalid file descriptor
 EINTR A signal was caught

See Also

·fpm_open()
 ·vpm_close() (refer to the Voice Software Reference)

fpm_faxtitle ()

add fax header

Name : int faxtitle (LPSTR *src, LPSTR *dst, LPSTR *fmt)

Inputs :

LPSTR *src : header TIFF
LPSTR *dst : TIFF
LPSTR *fmt : header

Returns :

0 :
-1 :

Includes: SmartSdk.h

Category:

Mode : synchronous

□

fpm_faxtitle() fax tif header 가 .
Src dst Header 가 .

Parameter	Description
Parameter	
src : header	가 tif full-pass
dst : header	가 가 tif full-pass
fmt : 가	header .
&D –	(10 Byte)
&T –	(5 Byte)
&P –	(3 Byte)
Header	header 88 Byte

Cautions

LibFaxTitle.lib가

Example

#include "SmartSdk.h"

```

        char titlefrm[] = {"&D  &T          Fax  Test  Header  Title
Page.&P"};
int main(int argc, char* argv[])
{
    if (argc < 3) {
        printf("USAGE: %s srcfile dstfile\n",argv[0]);
        exit(1);
    }
    faxtitle(argv[1], argv[2], titlefrm);
    exit(0);
}

```

□ Error

```

0 :
-1 :          file  (tif          Read/Write  )

```


fpm_getDCS() returns the most recent DCS message

Name: int fpm_getDCS(dev)

Inputs: int dev ·FAX channel device handle
DF_DCS * dcs_buf ·DF_DCS structure pointer

Returns: , 0
, -1

Includes: srllib.h
dxxplib.h
faxlib.h

Category: Miscellaneous

Mode: synchronous

Description

fpm_getDCS() channel(dev) 가 DCS Message(Digital Command Signal) .

Parameter Description

dev: channel open FAX channel device handle
dcs_buf: DCS message가 DF_DCS structure pointer

DCS message Transmitter Receiver
. DCS message fax Phase B , Transmitter Receiver
.

NOTE: application DCS message Phase B
. application FAX Extended Attribute
가 DCS message , fpm_getDCS()
. (FPMX_RESLN(), FPMX_SPEED(),
FPMX_WIDTH() reference .) Transmitter 가
DCS message Phase B .

NOTE: DCS message가 , FPMX_BSTAT()
. FPMX_BSTAT() Transmitter DCS message가 가

DFS_DCS bit 가 Bitmap .

DCS message가 , DCS message fpm_sendfax(), fpm_rcvfax(),
fpm_rcvfax2() Phase B . DCS message
Phase B , fax send
receive가 .

NOTE: Phase B fax send receive T.30 EOM message가
Transmitter .

DCS message ITU publication Procedures for Document Facsimile
Transmission in the General Switched Telephone Network, Recommendation T.30

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
extern int errno;
/* Handler for Phase B events. */
int phb_hdlr( );
main( )
{
    int voxdev; /* Voice channel device handle. */
    int dev; /* Fax channel device handle. */
    /*
     * Open the channel using vpm_open( ) to obtain the SCT
     * VOICE device handle in voxdev.
     * Open the channel using fpm_open( ) to obtain the FAX channel
     * device handle in dev.
     */
    .
    /*
     * Install handler using sr_enbhdlr( ) to service
```



```

* TFX_PHASEB events.
*/
.
/*
* Call fpm_sendfax( ) in asynchronous mode after setting
* up the DF_IOTT array. Set DF_PHASEB bit in mode field
* to enable generation of Phase B events.
*/
if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
}
/*
* Handler registered with SRL to handle TFX_PHASEB events.
*/
int phb_hdlr( )
{
    int dev = sr_getevtdev( );
    DF_DCS dcs_buf;
    if (sr_getevttype( ) == TFX_PHASEB) {
        if (FPMX_BSTAT(dev) & DFS_DCS) {
            /* T.30 DCS available. */
            if (fpm_getDCS(dev, &dcs_buf) == -1) {
                /* Error processing */
                .
            } else {
                /* Application specific analysis of the DCS */
                .
            }
        }
    }
}
.

```

```
return(0);  
}
```

Errors

ATDV_LASTERR()

EFX_NODATA

Phase B

FAX error code

가

가 call

error code Appendix D

returns the most recent DIS message fpm_getDCS()

Name:	int fpm_getDIS(dev, dis_buf)
Inputs:	int dev ·FAX channel device handle
	DF_DIS * dis_buf ·DF_DIS structure pointer
Returns:	, 0
	, -1
Includes:	srllib.h
	dxxplib.h
	faxlib.h
Category:	Miscellaneous
Mode:	synchronous

Description			
fpm_getDIS()	channel(dev)	가	DIS Message(Digital Information Signal)
DIS message Receiver capability	가	DIS message fax	
Phase B	Receiver Transmitter	.	
NOTE:			
application	DIS message	Receiver capability	
	application	DIS message	

Parameter Description			
dev:	channel open	FAX channel	device handle
dis_buf:	DIS 가	DF_DIS structure	pointer
Receiver	가	DIS message	Phase B
NOTE: DIS message가			
	, FPMX_BSTAT()		
. FPMX_BSTAT()	Transmitter	DIS message가	가
DFS_DIS bit	가	Bitmap	.

DIS message가 , DIS message fpm_sendfax(), fpm_rcvfax(), fpm_rcvfax2()
Phase B . DIS message
Phase B , fax send receive가

NOTE: Phase B fax send receive T.30 EOM message가
Transmitter .

DIS message ITU publication Procedures for Document Facsimile
Transmission in the General Switched Telephone Network, Recommendation T.30

Cautions

None.

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
DF_IOTT iott[10];
extern int errno;
/* Handler for Phase B events. */
int phb_hdlr( );
main( )
{
int voxdev; /* Voice channel device handle. */
int dev; /* Fax channel device handle. */
/*
* Open the channel using vpm_open( ) to obtain the SCT
* VOICE device handle in voxdev.
* Open the channel using fpm_open( ) to obtain the FAX channel
* device handle in dev.
*/
.
```

```

/*
 * Install handler using sr_enbhdlr( ) to service
 * TFX_PHASEB events.
 */
.
/*
 * Call fpm_sendfax( ) in asynchronous mode after setting
 * up the DF_IOTT array. Set DF_PHASEB bit in mode field
 * to enable generation of Phase B events.
 */
if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
}
/*
 * Handler registered with SRL to handle TFX_PHASEB events.
 */
int phb_hdlr( )
{
    int dev = sr_getevtdev( );
    DF_DIS dis_buf;
    if (sr_getevttype( ) == TFX_PHASEB) {
        if (FPMX_BSTAT(dev) & DFS_DIS) {
            /* T.30 DIS available. */
            if (fpm_getDIS(dev, &dis_buf) == -1) {
                /* Error processing. */
                .
            } else {
                /* Application specific analysis of the DIS. */
                .
            }
        }
    }
}

```

```

}
.
return(0);
}

```

Errors

ATDV_LASTERR() FAX error code .

EFX_NODATA Phase B 가 가 call

error code Appendix D .

fpm_getNSF() returns the remote station's NSF message

Name:	int fpm_getNSF(dev, nsf_length, nsf_data)			
Inputs:	int dev	·FAX channel device handle		
	Unsigned short nsf_length	·NSF message	가	byte
	Char * nsf_data	·NSF data	가	buffer pointer
Returns:	, 0			
	, -1			
Includes:	srllib.h			
	dxxplib.h			
	faxlib.h			
Category:	Miscellaneous			
Mode:	synchronous			

Description

fpm_getNSF() remote station NSF message (T.30 Non-Standard Facilities)

dev channel nsf_length data .

Parameter Description

dev:	channel	open	FAX channel	device handle
nsf_length:	NSF message	byte	NSF message	byte
nsf_data:	nsf_length	byte	NSF message	가 buffer pointer

NSF message information Phase B

option , 가 .

NSF message fax hardware .

가 .

NSF message가 remote station fax machine , message

Phase B application .

NOTE: FPMX_BSTAT() , remote station NSF message

. NSF message가 , FPMX_BSTAT() remote

station NSF message가

DFS_NSF 가 Bitmap

NSF message가 , NSF message fpm_sendfax(), fpm_rcvfax(),
fpm_rcvfax2() Phase B . DIS message
Phase B , fax send
receive가 .

NOTE: Phase B fax send receive T.30 EOM message가
Transmitter .

nsf_length argument Buffer NSF data byte .

nsf_data word NSF message 가 .

nsf_data byte NSF message 가 .

Example:

nsf_length = 10 bytes

nsf_data format

2 : NSF message
NSF message 8byte

NOTE: NSF message가 nsf_length , byte
blank , NSF message가 nsf_length
byte 가 , NSF message .

Cautions

None.

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
#define NSFMAX 128
```



```

DF_IOTT iott[10];

extern int errno;

/* Handler for Phase B events. */
int phb_hdlr( );

main( )
{
    int voxdev; /* Voice channel device handle. */
    int dev; /* Fax channel device handle. */

    /*
     * Open the channel using vpm_open( ) to obtain the SCT
     * VOICE device handle in voxdev.
     * Open the channel using fpm_open( ) to obtain the FAX channel
     * device handle in dev.
     */
    .
    /*
     * Install handler using sr_enbhdlr( ) to service
     * TFX_PHASEB events.
     */
    .
    /*
     * Call fpm_sendfax( ) in asynchronous mode after setting
     * up the DF_IOTT array. Set DF_PHASEB bit in mode field
     * to enable generation of Phase B events.
     */
    if (fpm_sendfax(dev, iott, EV_ASYNC|DF_PHASEB) == -1) {
        printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
        if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
    }
    .
}

/*
 * Handler registered with SRL to handle TFX_PHASEB events.
 */

```

```

int phb_hdlr( )
{
char nsf_data[NSFMAX];
char * nsfp;
unsigned short nsflen;
int dev = sr_getevtdev( );
if (sr_getevtttype( ) == TFX_PHASEB) {
    if (FPMX_BSTAT(dev) & DFS_NSF) {
        /* Remote NSF available. */
        if (fpm_getNSF(dev, NSFMAX, nsf_data) == -1) {
            /* Error processing */
            .
        } else {
            /* Obtain number of bytes of NSF returned */
            nsflen = * ((unsigned short *)&nsf_data[0]);
            if (nsflen > NSFMAX) {
                /*
                 * More NSF data available -- call fpm_getNSF( )
                 with larger data buffer if needed.
                 */
                .
            }
            /* Set pointer to NSF data. */
            nsfp = &nsf_data[2];
            /* Display NSF (application specific handling). */
            .
        }
    }
}
.
return(0);
}

```

Errors

ATDV_LASTERR()	error code	.
----------------	------------	---

EFX_NODATA	Phase B	가	,	NSF message가
	remote station			가 call
EFX_NSFBUFF	nsf_length	2		
			error code	Appendix D .

fpm_getparm() returns the FAX parameter

Name:	int fpm_getparm(dev, parm, valuep)			
Inputs:	int dev	·FAX channel device handle		
	unsigned long parm	·Parameter		
	Char *valuep	·parameter value가	location	pointer
Returns:	, 0			
	, -1			
Includes:	srllib.h			
	dxxplib.h			
	faxlib.h			
Category:	Configuration			
Mode:	synchronous			

Description

fpm_getparm()	dev	fax channel device	parm	fax
parameter valuep	location	.		

Parameter Description

dev:	channel open	FAX channel device handle
parm:	FAX Parameter (alphabet parameter
	.)	
valuep:	parm value가	void* cast location pointer

Parm values:

Define	Description
<hr/>	
FPMCH_MYFAXNO	Default: NULL
	Local identification – null terminated 10 character
FPMCH_SOURCEFAXNO	Remote identification – null terminated 10 character
FPM_RXBAUDRATE	Bytes: 2

Default: DF_MAXBAUD

fax data baud rate.

DF_MAXBAUD : byte

DF_9600BAUD : 9600baud

DF_7200BAUD : 7200baud

DF_4800BAUD : 4800baud

FPM_RXCODING

Bytes: 2

Default: DF_MH

fax data encoding scheme

DF_MH Modified Huffman

DF_MMR future use

FPM_TXBAUDRATE

Bytes: 2

Default: Maximum baud rate:

FPM=DF_14400BAUD

maximum baud rate

DF_MAXBAUD: maximum baud rate

DF_9600BAUD : 9600baud

DF_7200BAUD : 7200baud

DF_4800BAUD : 4800baud

DF_2400BAUD : 2400baud

FPM_TXCODING

Bytes: 2

Default: DF_MH

Fax encoding scheme

DF_MH Modified Huffman

DF_MR Modified Read

DF_MMR Modified Modified Read

Details

FPM_MYFAXNO parameter FAX application

. T.30 Call Subscriber Identification(CSI), Transmitting Subscriber Identification(TSI), Calling Subscriber Identification(CIG) message data .

10 NULL termination character . FPM_MYFAXNO parameter application .

FPM_SOURCEFAXNO parameter application fax remote fax machine . T.30 Call Subscriber Identification(CSI), Transmitting Subscriber Identification(TSI), Calling Subscriber Identification(CIG) message data . 10 null termination character ,

FPM_RXBAUDRATE parameter fax baud rate .

FPM_RXCODING parameter fax data가 TIFF/F RAW encoding scheme .

FPM_TXBAUDRATE parameter baud rate 가 . FAX hardware .

FPM_TXCODING parameter fax encoding scheme 가 .

Cautions

fpm_getparm() valuep (void*) cast ,
fpm_getparm() pointer clear .

Valuep address 가 가 (2byte 10byte char)

NOTE: FAX parameter vpm_getparm() .

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
```

```

#include <faxlib.h>

int dev;

unsigned short value;

extern int errno;

/* Clear value. */
value = 0;

/*
 * Open device using fpm_open( ). Obtain SCT FAX device
 * handle in dev.
 */
.
.
/*
 * FPM_TXCODING parameter uses 2 bytes. Pass the address of
 * the variable value (unsigned short) to fpm_getparm( ).
 */
if (fpm_getparm(dev, FPM_TXCODING, (void *)&value) == -1) {
    /* Error processing. */
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}

printf("Number of retries was %ld\n", value);

```

Errors

가 error code list Appendix D .

fpm_initstat() sets the initial fax state

Name:	int fpm_initstat(dev, state)		
Inputs:	int dev	·FAX channel device handle	
	Int state	·Initial fax state	
Returns:	, 0		
	, -1		
Includes:	srllib.h		
	dxxplib.h		
	faxlib.h		
Category:	Set Initial State		
Mode:	synchronous		

Description

fpm_initstat() fax . CALLER(DF_TX) application
CALLED(DF_RX_ application channel fax
가 .

NOTE: fpm_initstat() fax SCbus local
, SCbus fax fax SCbus routing
function fpm_sendfax(), rcvfax(), rcvfax2() .
SCbus routing fpm_initstat() , fax local .

Parameter Description	
dev:	channel open FAX channel device handle
state:	fax state , .
Value	Description
DF_RX	CALLED application (receive state)
DF_TX	CALLER application (transmit state)

Details

ITU T.30 protocol , CALLER Transmitter ,


```

.
/*
* Set the initial FAX state to be RECEIVER. */
if (fpm_initstat(dev,DF_RX) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
/* Issue a fpm_rcvfax( ). */
.

```

Errors

가 error code list Appendix D .

opens a FAX channel or board device

fpm_open()

Name: int fpm_open(namep, mode)

Inputs: char *namep ·Pointer to device name to open
 int mode ·Reserved for future use

Returns: , valid SCT device handle 가 0
 , -1

Includes: srllib.h
 dxxplib.h
 faxlib.h

Category: Resource Management

Mode: synchronous

Description

fpm_open() FAX channel board device open , FAX channel board device
 SCT device handle .

NOTE: open FAX channel board device FAX channel device가 close
 FAX API FAX channel/board device handle
 .

FAX device vpm_open() handle application device
 .

FAX device Process .

NOTE: FAX device handle SCT define .
 NT system file descriptor가 ,
 windows 2000 command .

 process child process handle child
process device가 open .

Parameter Description

namep: FAX channel board device NULL
Terminated ASCII string pointer. device name
.
name field value form 가 .
Board device: vpmBn
Channel device: vpmBnCm
Where:
n system board
m board channel
Boards and channels are numbered from one.

mode: parameter NULL
.

Cautions

FAX Device fpm_open() .

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
extern int errno;
main()
{
int dev; /* Fax channel device handle. */
/* Open the Voice channel resource (device) using vpm_open( ). */
.
.
/* Open the FAX channel resource (device). */
if ((dev = fpm_open("vpmB1C1", NULL)) == -1) {
/* Error opening device. */
printf("Error opening channel, errno = %d\n", errno);
exit(1);
}
```

```

.
.
/* FAX transfers (send/receive) calling FAX API functions using dev. */
.
.
}

```

Errors

가 ?1 , errno check

EINVAL	Invalid Argument
EBADF	Invalid file descriptor
EINTR	A signal was caught
EIO	Error opening Windows 95 or Windows 2000 device driver

See Also

·fpm_close()
 ·vpm_open() (Voice Software Reference)

fpm_rcvfax() receives fax data

Name:	int fpm_rcvfax(dev, faxname, rcvflag)
Inputs:	int dev ·FAX channel device handle
	char *faxname ·receive document file name
	unsigned long rcvflag ·Mode flag
Returns:	, 0 (asynchronous mode invocation)
	, -1 (asynchronous mode invocation)
Includes:	srllib.h
	dxxplib.h
	faxlib.h
Category:	Receive Fax
Mode:	synchronous/asynchronous

Description

fpm_rcvfax() open channel device fax data , TIFF/F format
file raw fax data .

NOTE: raw fax data fpm_rcvfax() , file
.

fax data encoding scheme modified Huffman ,
modified modifier Read, Modified modified Huffman .

fpm_rcvfax() .

fpm_rcvfax() fax data TIFF/F file , Phase D command
, fpm_rcvfax() application program .

Parameter Description			
dev:	channel open	FAX channel	channel device
	handle		
Faxname:	fax data가	file	
Rcvflag:	rcv flag	OR bit mask	.

- fax data가 file format type
- operation mode
- Phase B event
- Phase D event
-
- fax data sresolution
- I/O function

rcvflag bit mask 가 .

File Format bit:

Value	Description
DF_TIFF	TIFF/F structured formatted fax data
DF_RAW	Raw, unformatted fax data
Mode bit:	
Value	Description
EV_SYNC	Synchronous mode operation
EV_ASYNC	Asynchronous mode operation

Phase B, Phase D event

Value	Description
DF_PHASEB	Phase B event generation enable
DF_PHASED	Phase D event generation enable

Maximum receive width bits:

Value	Description
DF_1728MAX	Maximum receive width: 1728 pixels
DF_2048MAX	Maximum receive width: 2048 pixels
DF_2432MAX	Maximum receive width: 2432 pixels (default)

Fax data resolution(default= resolution)

Value	Description
DF_RXRESLO	low resolution incoming fax data
DF_RXRESHI	high resolution incoming fax data

Enable user-defined I/O bit (fpm_rcvfax2() only)

Value	Description
IO_UID	fpm_rcvfax2() I/O

Details

fpm_rcvfax() fax Receiver .

·TIFF/F file

·raw file

·

· width

·Phase B, Phase D event, Voice Request

· data resolution

·user-defined I/O

NOTE: faxname , TIFF/F raw file , . Raw

file multi page fax data fax page file

.

Issuing fpm_rcvfax()

CALLER application CALLED application fax connection ,

CALLER application TRANSMITTER , CALLED application Receiver

.

Error Handling

Synchronous Mode

· fpm_rcvfax() 0 , error
?1 . (3.4 FAX library Error handling)

Asynchronous Mode

· fpm_rcvfax() 0 , error
?1 .
· Error Standard Runtime Library event TFX_FAXERROR
· Event가 , Standard attribute function ATDV_LASTERR()
error code . Standard Attribute function ATDV_ERRMSGP()
error string . SCT Define FAX error code
Appendix D .
· fpm_rcvfax() 가 TFX_FAXRECV SCT Standard Runtime
library event가 .

NOTES: 1. Phase E T.30 fax protocol 가 error .
2. EDX_SYSTEM error code System error error
error code .

Status

FAX Extended Attribute .

NOTE: fax fax CS_RECV FAX . FAX
FAX
extended Attribute FPMX_STATE() .

Phase D FAX Extended Attribute FPMX_PHDCMD()
FPMX_PHDRPY() .

Stopping a Fax Reception

fpm_stopch() fax .

Storing Incoming Fax Data

fax data . fax data가 file , file
format(TIFF/F, raw), encoding scheme, resolution , fax data

NOTE: fax encoding scheme Modified Huffman coding 가 ,
FPMCH_RXCODING parameter coding .

TIFF/F file Fax Data

fax data fax document TIFF/F file

FAX Data Stored in a Raw File

fax data raw file , file 가 fax data 가
fax data TIFF/F format .

fax data 1page 1 file .

fax data Modified Huffman encoding .
-Modified Huffman : EOL sequence byte align .

Specifying the Resolution to Store Incoming Fax Data

Rcvflag argument resolution , fax data
resolution data . receiver Transmitter
Phase B , FPMX_RESLN() Resolution

Specifying the Encoding Scheme to Store Incoming Fax Data

FPM encoding scheme Modified Huffman , Modified

Modified Read encoding scheme , FPMCH_RXCODING
parameter reserved .

fax data line encoding scheme , FPMX_CODING()
Phase B .

Synchronous/Asynchronous Mode Operation

rcvflag 가 , fpm_rcvfax() , error가
return . Rcvflag 가 , ,

application . Fx_rcvfax() 0
, error ?1 . ,

application , device fax data . 가
application device Voice/fax .

fpm_rcvfax() Standard Runtime library event
, event .

Event	Description
TFX_FAXERROR	Error in processing
TFX_FAXRECV	Successful completion of fpm_rcvfax()

NOTE: event handling information Voice Software Reference .

Enable Phase B Event Generation

rcvflag argument Phase B event generation bit(DF_PHASEB)가 ,
fpm_rcvfax() 가 fax data ITU T.30 protocol Phase B가
, TFX_PHASEB event가 . TFX_PHASEB event가 , application
FAX Extended Attribute , .

FPMX_BSTAT() Phase B
FPMX_CODING() data encoding scheme
FPMX_SPEED() data baud rate

FPMX_STATE() fax channel device

NOTE: fpm_rcvfax(), fpm_rcvfax2() 가 , Phase B event

sr_enbhdr() , event handler가 .

Enable Phase D Event Generation

rcvflag argument Phase D event generation bit(DF_PHASED)가 ,
fpm_rcvfax() 가 fax data ITU T.30 protocol Phase D가
, TFX_PHASED event가 . Phase D event page ,
page . page , fpm_rcvfax() ,
TFX_FAXRECV event가 . TFX_PHASED event TFX_FAXRECV
event가 , FAX Extended Attribute .

FPMX_PHDCMD() Phase D command
FPMX_PHDRPY() Phase D reply
FPMX_WIDTH() page width
FPMX_RESLN() page resolution
FPMX_SCANLINES() scanline
FPMX_BADSCANLINES() bad scanline
FPMX_SPEED() data baud rate
FPMX_STATE() fax channel device
FPMX_TRCOUNT) byte

Phase D event page page , fax session application
monitor .

NOTES: 1. Phase D event Generation , page Phase D event
generation , page TFX_FAXRECV event가
, fpm_rcvfax() .

2. fpm_rcvfax() 가 , sr_enbhdr()
event handler가 .

Selectable Receive Width

Rcvflag fax data width . DF_2432MAX (default)

fax data 1728, 2048, 2432 pixel . Transmitter Phase B

data width . fax

machine

page Maximum width scale .

Select Preferred Maximum Receive Baud Rate

Fax data 가 baud rate . FPMCH_RXBAUDRATE

parameter baud rate .

, Phase B , Phase B event

FPMX_SPEED .

Bad Scan Line Replacement – Incoming Fax Data

line scan line error . MH encoding fax data

, data stream scan line check . Scan line

error가 , Bad line replacement(BLR) bad scan line 가

scan line . page bad

scan line FPMX_BADSCANLINES() .

User-defined I/O functions

Application standard I/O function lseek(), read(), write() 가 I/O

function . User I/O function application

seek, read, write pointer 가 DF_UIO structure .

, application fpm_setuio() , DF_UIO structure .

Fax library user-defined seek, read, write pointer ,

, Standard I/O seek, write argument

.

User defined I/O fax data fpm_rcvfax2() 가

IO_UIO bit가 . , fax library fax data가 I/O

device user defined seek, write . I/O function

file descriptor fpm_rcvfax2() argument fd .

Cautions

None.

Example

1,2 fpm_rcvfax() TIFF/F raw format .
 programming code channel thread multi-
 thread application . thread operation
 channel .

3 fpm_rcvfax()가 programming
 code , program multi thread multi-thread application
 , thread channel channel
 operation . Programming mode SRL function Voice
 Software Reference .

Example 1 – Receive Fax Data into TIFF/F File Format – Synchronous

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxlib.h>
#include <faxlib.h>

int voxdev; /* Voice channel device handle. */
int dev; /* Fax channel device handle. */
extern int errno;

unsigned long rcvflag = DF_NOPOLL|DF_TIFF|EV_SYNC;
unsigned short value;

/*
 * Open the channel using vpm_open( ) to obtain the SCT
 * VOICE channel device handle in voxdev. Use voxdev for
 * all Voice API calls.
 */

if ((voxdev = vpm_open("vpmB1C1", NULL)) == -1) {
    /* Error opening device. */
    printf("Error opening channel, errno = %d\n", errno);
    exit(1);
}
```

```

}
/*
 * Open the channel using fpm_open( ) to obtain the FAX
 * channel device handle in dev. Use dev for all Fax API
 * calls.
 */
if ((dev = fpm_open("vpmB1C1", NULL)) == -1) {
    /* Error opening device. */
    printf("Error opening channel, errno = %d\n", errno);
    exit(1);
}
.
/*
 * Set channel on-hook using vpm_sethook( ) in synchronous
 * mode.
 */
.
/*
 * Wait for 1 ring and go off-hook using vpm_wtring( ).
 */
.
rcvflag |= DF_RXRESLO;
value = DF_MH;
if (fpm_setparm(dev,FPMCH_RXCODING,(void *)&value) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
/* Set initial state of FAX channel to RECEIVER. */
if (fpm_initstat(dev,DF_RX) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
}

```

```

/*
 * Receive the fax data into "myfax.tif" file - synchronous
 * mode.
 */
if((fpm_rcvfax(dev,"myfax.tif",rcvflag)) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),
    ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
    /* Application specific error handling. */
    .
}

```

Example 2 – Receive Fax Data into Raw File – Synchronous

```

#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int count = 0;
char faxname[30];

int voxdev; /* Voice channel device handle. */
int dev; /* Fax channel device handle. */
extern int errno;

unsigned long rcvflag = DF_RAW|EV_SYNC;
unsigned short value;

/*
 * Open the channel using vpm_open( ) to obtain the SCT
 * VOICE channel device handle in voxdev. Use voxdev for
 * all Voice API calls.
 */
if ((voxdev = vpm_open("vpmB1C1", NULL)) == -1) {
    /* Error opening device. */
    printf("Error opening channel, errno = %d\n", errno);
}

```



```

        exit(1);
    }
    /*
    * Open the channel using fpm_open( ) to obtain the FAX
    * channel device handle in dev. Use dev for all Fax API
    * calls.
    */
    if ((dev = fpm_open("vpmB1C1", NULL)) == -1) {
        /* Error opening device. */
        printf("Error opening channel, errno = %d\n", errno);
        exit(1);
    }
    .
    /*
    * Set channel on-hook using vpm_sethook( ) in synchronous
    * mode.
    */
    .
    /*
    * Wait for 1 ring and go off-hook using vpm_wtring( ).
    */
    .
    rcvflag |= DF_RXRESLO;
    value = DF_MH;
    if (fpm_setparm(dev,FPMCH_RXCODING,(void *)&value) == -1) {
        printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
        if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
    }
    .
    /* Set initial state of the FAX channel to RECEIVER. */
    if (fpm_initstat(dev,DF_RX) == -1) {
        printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
        if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
    }

```

```

    }
}
do {
    /* Receive each page into a separate file until the application
    * receives a DFS_EOP Phase D status value. fpm_rcvfax( ) is
    * being used in synchronous mode.
    */
    .
    /*
    * Generate a file name in faxname, for example, rcv_pg0.raw,
    rcv_pg1.raw, etc.
    */
    .
    if(fpm_rcvfax(dev,faxname,rcvflag) == -1)
        printf("Error - %s (error code %d)\n",ATDV_ERRMSGP(dev),
            ATDV_LASTERR(dev));
        if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
            printf("errno = %d\n", errno);
        }
        printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
        /* Application specific error handling. */
        .
    }
} while(FPMX_PHDCMD(dev) != DFS_EOP);
/* Show results. */
printf("Fax received: %ld pages\n",FPMX_PGXFER(dev));
/*
* Note: The encoding scheme of the received RAW data is specified
* in the variable 'value' used for setting the FPMCH_RXCODING
* parameter. If these RAW files have to be transmitted, the same
* encoding scheme value will have to be specified in the DF_IOTT
* entry.
*/
.

```

Example 3 – fpm_rcvfax() using asynchronous programming mode

```
#include <stdio.h>
```

```

#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
#define MAXCHANS 24
extern int errno;
int catchall( );
int recv_fax( );
/* Error routine - print error information. */
void print_err(dev)
int dev;
{
    printf("Error - %s (error code %ld)\n",ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev)==EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    return;
}
/*
* main( ): Opens all channels and enables handler for
* asynchronous operation. Channels go on-hook and wait for
* rings. On receiving rings, the channel goes off-hook and
* receives a fax.
*/
main( )
{
    int chan;
    char * chnamep;
    int mode = SR_POLLMODE;
    int voxdev; /* Voice channel device handle. */
    int faxdev; /* Fax channel device handle. */
    for (chan=0; chan < MAXCHANS; chan++) {
        /*
        * Set chnamep to the channel device name, e.g.,
        * vpmB1C1, vpmB1C2, etc.
        * Open the channel using vpm_open( ) so that voxdev

```

```

    * has the VOICE channel device handle.

    * Open the channel using fpm_open( ) so that faxdev

    * has the FAX channel device handle.

    */

    .

/*

    * Place channel on-hook by calling vpm_sethook( ) with

    * its mode field set to EV_ASYNC (asynchronous).

    */

    .
}

/*

    * All channels have been opened and a sethook function

    * issued to place the channels on-hook. Use sr_waitevt( )

    * to wait for completion events.

    * On receiving any completion event, control is transferred

    * to the catchall( ) handler function.

    */

while(sr_waitevt(-1)) {
    catchall( );
}

.
}

/* Event handler. */

/*

    * This routine is called when sr_waitevt( ) receives an event.

    * Maintain a state machine for every channel and issue the

    * appropriate function depending on the next action to be

    * performed on the channel, e.g., the application may wish

    * to wait for rings after a on-hook completion event and

    * start receiving a fax as soon as rings are received.

    */

int catchall( )
{
    int dev = sr_getevtddev( );
    char * fnamep;

```

```

/* Determine the event. */
switch(sr_getevtttype( )) {
    case TDX_SETHOOK:
        /*
         * If channel has gone off-hook, start receiving the
         * fax.
         */
        if (ATDX_HOOKST(dev) == DX_OFFHOOK) {
            /*
             * Set the fax state of the channel to DF_RX using
             * fpm_initstat( ).
             */
            .
            /*
             * Set up fnamep to point to TIFF/F file name.
             * Start receiving the fax.
             */
            if (fpm_rcvfax(dev, fnamep, DF_TIFF|DF_NOPOLL|EV_ASYNC) == -1) {
                print_err(dev);
                printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
                /* Application specific error handling here. */
                .
            }
        } else {
            /*
             * Channel is on-hook. State machine dependent
             * action.
             */
            .
        }
        break;
    case TDX_CST:
        /* Handle rings received event. */
        .
        .
        break;
}

```

```

    case TFX_FAXRECV:
        /* The document has been successfully received. */
        printf("Received %ld pages at speed %ld, resln %ld,width %ld\n",
            FPMX_PGXFER(dev), FPMX_SPEED(dev),FPMX_RESLN(dev), FPMX_WIDTH(dev));
        .
        break;
    case TFX_FAXERROR:
        /* Error during the fax session. */
        print_err(dev);
        printf("Phase E status %d\n", FPMX_ESTAT(dev));
        /* Application specific error handling. */
        .
        break;
    default:
        .
        break;
} /* End of switch. */
return(0);
}

```

Errors

error code Appendix D .

See Also

- FPMX_BADSCANLINES()
- FPMX_BSTAT()
- FPMX_CHTYPE()
- FPMX_CODING()
- FPMX_ESTAT()
- FPMX_PGXFER()
- FPMX_PHDCMD()
- FPMX_PHDRPY()
- FPMX_RESLN()
- FPMX_SCANLINES()
- FPMX_SPEED()
- FPMX_STATE()

- FPMX_TERMMSK()
- FPMX_TRCOUNT()
- FPMX_WIDTH()
- fpm_getDCS()
- fpm_getDIS()
- fpm_getNSF()
- fpm_rcvfax2()

receives fax data (file descriptor argument) fpm_rcvfax2()

Name:	int fpm_rcvfax2(dev, fd, rcvflag)		
Inputs:	char dev		·FAX channel device handle (to receive fax data)
	int fd		·file descriptor
	unsigned long rcvflag		·Mode Flag
Returns:	, 0 (on invocation in asynchronous mode)		
	, -1 (on invocation in asynchronous mode)		
Includes:	srllib.h		
	dxxplib.h		
	faxlib.h		
Category:	Receive Fax		
Mode:	synchronous/asynchronous		

Description

fpm_rcvfax2() open channel device fax data , TIFF/F format
file raw, unstructured, unformatted fax data .

NOTES: 1. fpm_rcvfax2() fpm_rcvfax() file file
,
file descriptor argument .
2. user-defined I/O , fpm_rcvfax2()
가
.
3. fpm_rcvfax2() argument file descriptor가 , user-
defined I/O 가 가 fpm_rcvfax() .

Parameter Description			
dev:	channel open	FAX channel	channel device
	handle		.
fd:	file descriptor		
rcvflag:	OR bit mask (fpm_rcvfax())		

Cautions

1. application `open` , file descriptor
`fpm_rcvfax2()` .
2. FAX library fax error가 file descriptor close
. Application receive file close .

Example

```
/*
 * The principal difference between fpm_rcvfax( ) and
 * fpm_rcvfax2( ) is that the application must open the
 * receive file and pass the file descriptor to the
 * fpm_rcvfax2( ) function instead of the receive file name.
 * Example 1 from the function reference for fpm_rcvfax( ) has
 * been modified for use with fpm_rcvfax2( ) and included
 * below. The other examples in fpm_rcvfax( ) can be modified
 * similarly.
 */
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int voxdev; /* Voice channel device handle. */
int dev; /* Fax channel device handle. */
extern int errno;

int rcvfd;

/*
 * Open the channel using vpm_open( ) and obtain the SCT
 * VOICE channel device handle in voxdev.
 */
.
/*
 * Open the channel using fpm_open( ) and obtain the SCT
 * FAX channel device handle in dev.
 */
.
```

```

/*
Set channel on-hook using vpm_sethook( ) in synchronous
* mode.
*/
.
/*
* Wait for 1 ring and go off-hook using vpm_wtring( ).
*/
.
.
rcvflag |= DF_RXRESLO;
value = DF_MH;
if (fpm_setparm(dev,FPMCH_RXCODING,(void *)&value) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
/*
* Set the fax state of the channel to DF_RX using
* fpm_initstat( ).
*/
.
/*
* Open the file "myfax.tif" in preparation for receiving a
* fax. Use vpm_fileopen( ) to open the file.
*/
if ((rcvfd = vpm_fileopen("myfax.tif", O_BINARY|O_WRONLY|O_CREAT|O_TRUNC,
                        0666)) == -1) {
    /* Error opening file. */
    printf("Error opening receive file, errno = %d\n", errno);
    .
}
/*
* Receive the fax data into "myfax.tif" file - synchronous
* mode.

```

```

*/
if((fpm_rcvfax2(dev,rcvfd,rcvflag)) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
    /* Application specific error handling. */
    .
}
/* Close the received file. */
if (vpm_fileclose(rcvfd) == -1) {
    /* Error closing file. */
    printf("Error closing receive file, errno = %d\n", errno);
    .
}

```

See Also

·fpm_rcvfax()

transmits fax data

fpm_sendfax()

Name: int fpm_sendfax(dev, iotp, sndflag)

Inputs: char dev ·FAX channel device handle
DF_IOTT *iotp ·fax transfer table pointer
unsigned long sndflag ·Mode Flag

Returns: , 0 (on invocation in asynchronous mode)
, -1 (on invocation in asynchronous mode)

Includes: srllib.h
dxxplib.h
faxlib.h

Category: Transmit Fax

Mode: synchronous/asynchronous

Description

fpm_sendfax() DF_IOTT transfer table data structure fax data

.

Parameter Description			
dev:	channel open handle .	FAX channel	channel device
iotp:	fax data	DF_IOTT table entry	pointer
sndflag:	sndflag field	field	OR bit mask .
	· operation mode (synchronous/asynchronous)		
	·Phase B event	generation	enable
	·Phase D event	generation	enable
	·remote station	voice request	enable
	·remote station	voice request	enable
	·low, high resolution	DF_IOTT entry	enable
	·subaddress	enable	
	sndflag	가	.
	Mode bit:		

Value	Description
EV_SYNC	Synchronous mode operation
EV_ASYNC	Asynchronous mode operation

Phase B, Phase D and Voice Request enable bits (set one or more of the following, default = disabled):

Value	Description
DF_PHASEB	Enable Phase B event generation
DF_PHASED	Enable Phase D event generation

Fax transmission resolution (default = transmit at resolution specified in file (TIFF/F) or DF_IOTT (Raw))

Value	Description
DF_TXRESLO	Transmit all DF_IOTT entries at low (coarse) resolution
DF_TXRESHI	Transmit all DF_IOTT entries at high (fine) resolution

Detail

```
fpm_sendfax()          TRANSMITTER application
    ·linked host array  DF_IOTT entry          fax document
    Receiver
    ·
    · operation
    ·Phase B event generation
    ·Phase event generation
    ·DF_IOTT entry      data    low    high resolution
```

Specifying Fax Data for Transmission

```
fax data DF_IOTT structure . (6.3.1. DF_IOTT Fax Transmit Data
Descriptor ). DF_IOTT structure fax data source(
```

, TIFF/F file raw file) . DF_IOTT structure linked list array
 fpm_sendfax() page fax document .
 DF_IOTT structure parameter data가 check . Fx_sendfax()
 argument DF_IOTT array linked list pointer .

Setting Values in the DF_IOTT Structure for fpm_sendfax()

Fax data DF_IOTT entry value default FAX function fpm_setiott()
 , value .

Specifying the Data Type

TIFF/F, raw data .
 ·io_datatype field가 DF_TIFF , TIFF/F file .
 ·io_datatype field가 DF_RAW , TIFF/F file .

Connecting DF_IOTT Entries

io_type DF_IOTT table entry가 OR filed .

Fax encoding scheme Modified Huffman 가 ,
 MMR .

DF_IOTT entry io_type , .
 ·IO_LINK , io_nextp DF_IOTT structure pointer 가
 .
 ·IO_EOT가 , DF_IOTT structure .
 . , IO_EOT IO_LINK가 , next entry contiguous
 .

NOTE: io_type entry가 IO_CONT

·FAX library fpm_sendfax()가 , DF_IOTT chain io_prevp
 field . DF_IOTT entry io_prevp NULL

Sending Data from a Device or Memory

, io_type field data가 disk data(IO_DEV)

memory(IO_MEM) data

NOTE: io_type IO_MEM raw data (DF_RAW)

Sending TIFF/F Files

DF_IOTT structure page ,

·io_firstpg page .

page 1 .

·io_pgcount page . io_pgcount가 -1 , file

page가 .

NOTE: io_firstpg가 0 , io_pgcount page .

Page가 0,1,2 TIFF/F file , io_firstpg 0 , io_pgcount 3

TIFF/F file fax library file TIFF/F tags fax data

encoding scheme . DF_IOTT io_coding field .

encoding scheme MH coding .

TIFF/F file fpm_sendfax() , .

· TIFF/F tags가 (TIFF/F tag

Appendix A) (ATDV_LASTERR , EFX_BADTIF가 ,

data .)

error가 EFX_BADTIF , FPMX_TFNOTAG() ,

TIFF/F tag .

·file TIFF/F file header 가 (EFX_BADTFHDR

ATDV_LASTERR .)

· TIFF/F tag value가 (ATDV_LASTERR ,

EFX_BADTIF가 , data .)

NOTE: EFX_BADTAG가 , FPMX_TFBADTAG() FAX Extended Attribute

TIFF/F tag value TIFF/F tag .

·tag value page number 가 , EFX_BADPAGE error가
 ATDV_LASTERR() .

· page가 TIFF/F file , EFX_NOPAGE error가
 ATDV_LASTERR() .

Sending Specific Pages in a TIFF/F File

TIFF/F file , io_pgcount io_firstpg .
 io_pgcount page , io_firstpg page
 .

Sending Raw Files

Raw file fax data fax data format 가 .
 Raw data 가 width, resolution, encoding scheme .
 io_width value fax data width io_width value .

io_resIn field fax data resolution 가 .

io_coding field raw file encoding scheme .

io_coding value data encoding scheme , 가 encoding
 scheme MH MMR . 가 .
 DF_MH Modified Huffman
 DF_MMR Modified Modified Read

io_offset data 가 , file/memory byte . io_offset
 0 file/memory data .

io_length value byte 가 .
 ·io_type IO_DEV , io_length가 -1 , data file .
 ·io_type IO_MEM , io_length buffer byte 가 .

io_type IO_MEM , io_bufferp raw image data
 pointer 가 .

Specifying the Encoding Scheme for Transmitting Fax Data

FAX data	encoding scheme	Modified Huffman	, MMR
version	.	fax coding	Phase B
FPMX_CODING		, FPMCH_TXCODING	parameter
.			

Issuing fpm_sendfax()

CALLER application	CALLED application	fax connection	,
CALLER application	TRANSMITTER	, CALLED application	Receiver
.			

fpm_sendfax() Issued by the TRANSMITTER

Transmitter application	fpm_sendfax()	가	,	DF_IOTT
entry	fax data	.	Receiver가 fax	call
disconnect	, ATDV_LASTERR	EFX_DISCONNECT	.	
NOTES: 1.	error가	?1	.	
	, TFX_FAXERROR event가	.	,	
	application	error code	check	.

Error Handling

Synchronous Mode

·fpm_sendfax()	0	, error
-1	.	

Asynchronous Mode

·fpm_sendfax()	0	,	error
-1	.		
.	error	SCT Standard Runtime library event	
(TFX_FAXERROR)	.	event가	, SCT
error code	ATDV_LASTERR()	.	Standard Attribute
function	ATDV_ERRMSG()	error	string pointer

. SCT FAX error code Appendix D .

·fpm_sendfax()가 , TFX_FAXSEND SCT Standard

Runtime library event가 .

NOTES: 1. Phase E (FPMX_ESTAT()) T.30 fax protocol 가
error information .

2. System error EDX_SYSTEM error code , error code
errno error code .

structure fax data가 DF_IOTT structure entry check

ATDV_LASTERR() error code .

·EFX_BADPAGE error code가 , page number tag

·EFX_BADPAGE EFX_BADTIF error code가 , DF_IOTT structure
page page offset FPMX_BADPAGE()

·EFX_BADIOTT error code가 , DF_IOTT structure data 가
DF_IOTT structure data

·EFX_COMPAT error code가 , DF_IOTT structure error
condition Transmitter Receiver

. DF_IOTT data . FPMX_BADIOTT fax
extended Attribute error DF_IOTT structure pointer

·error code가 EFX_DISCONNECT , FPMX_ESTAT() disconnect가

Status

FAX Extended Attribute function .

NOTE: fax channel device CS_SENDFAX . Channel device
FPMX_STATE()

가 , Phase D FPMX_PHDCMD()

FPMX_PHDRPY .
DF_IOTT entry pointer FAX Extended Attribute
FPMX_LASTIOTT() .

Stopping a Fax Transmission

fax fpm_stopch() .

Synchronous/Asynchronous Mode Operation

rcvflag 가 , fpm_rcvfax() , error가
return . Rcvflag 가 , ,
application . Fx_rcvfax() 0
, error ?1 . ,
application , device fax data . 가
application device Voice/fax .
fpm_rcvfax() Standard Runtime library event
, event .

Event	Description
TFX_FAXERROR	Error in processing
TFX_FAXRECV	Successful completion of fpm_rcvfax()

NOTE: event handling information Voice Software Reference .

Enable Phase B Event Generation

rcvflag argument Phase B event generation bit(DF_PHASEB)가 ,
fpm_rcvfax() 가 fax data ITU T.30 protocol Phase B가
, TFX_PHASEB event가 . TFX_PHASEB event가 , application
FAX Extended Attribute , .

FPMX_BSTAT()	Phase B
FPMX_CODING()	data encoding scheme
FPMX_SPEED()	data baud rate
FPMX_STATE()	fax channel device

NOTE: fpm_rcvfax(), fpm_rcvfax2() 가 , Phase B event
 sr_enbhdIr() , event handler가 .

Enable Phase D Event Generation

rcvflag argument Phase D event generation bit(DF_PHASED)가 ,
 fpm_rcvfax() 가 fax data ITU T.30 protocol Phase D가
 , TFX_PHASED event가 . Phase D event page
 , page . page , fpm_rcvfax()
 , TFX_FAXRECV event가 . TFX_PHASED event
 TFX_FAXRECV event가 , FAX Extended Attribute .

FPMX_PHDCMD()	Phase D command
FPMX_PHDRPY()	Phase D reply
FPMX_WIDTH()	page width
FPMX_RESLN()	page resolution
FPMX_SCANLINES()	scanline
FPMX_BADSCANLINES()	bad scanline
FPMX_SPEED()	data baud rate
FPMX_STATE()	fax channel device
FPMX_TRCOUNT)	byte

Phase D event page page , fax session application
 monitor .

NOTES: 1. Phase D event Generation , page Phase D event
 generation , page TFX_FAXRECV event가
 , fpm_rcvfax() .
 2. fpm_rcvfax() 가 , sr_enbhdIr()
 event handler가 .

Select Resolution for Entire fpm_sendfax() Transmission

sndflag argument resolution (DF_TXRESLO, DF_TXRESHI)가 ,
fpm_sendfax() fax data resolution sndflag
resolution .

Fax DF_IOTT structure resolution 가 Raw TIFF/F
. sndflag DF_TXRESHI DF_TXRESLO bit ,
DF_IOTT entry data resolution .
, DF_IOTT array low resolution Raw file , , high
resolution TIFF/F file , Phase D Raw file Low
resolution , high resolution TIFF/F file Raw resolution
.

Bad Scan Line Replacement – Transmit Data

MH encoding fax data data stream , scan line
pixel count scan line . (, image가 scan line error
, data bad scan line .)

scan line error가 , Bad Scan line Replacement(BLR) bad scan line
pixel count 가 scan line .

bad scan line FPMX_BADSCANLINES() .

Selectable Baud Rate – Transmit Data

baud rate fpm_sendfax() ,
fpm_setparm() . Fx_setparm()
FPMCH_TXBAUDRATE parameter default baud rate
.

User-defined I/O functions

Application standard I/O function lseek(), read(), write() 가 I/O
function . User I/O function application

seek, read, write pointer 가 DF_UIO structure .
 , application fpm_setuio() , DF_UIO structure .
 Fax library user-defined seek, read, write pointer ,
 , Standard I/O seek, write argument
 .

User defined I/O fax data fpm_rcvfax2() 가
 IO_UIO bit가 , fax library fax data가 I/O
 device user defined seek, write . I/O function
 file descriptor fpm_rcvfax2() argument fd .

Cautions

fpm_sendfax DF_IOTT structure global static
 . Fx_sendfax()가 DF_IOTT structure .
 DF_IOTT structure fax가 .
 NOTE: , fax library fpm_sendfax가 fax
 application , DF_IOTT structure entry access
 . process channel process
 DF_IOTT structure 가 .

DF_IOTT structure io_type field structure
 IO_EOT .

Example

Example 1: array DF_IOTT structure 가 fpm_sendfax()
 Example 2: fpm_sendfax(). programming code
 program multi thread multi thread application .
 thread channel channel operaion
 . Programming code SRL Voice Software Reference
 .

Example 1 – Array Based DF_IOTT – Synchronous

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
```

```

#include <dxxilib.h>
#include <faxlib.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include <io.h>

#define NUMDOC 3

/* Need a DF_IOTT entry for each document to send. */
DF_IOTT iott[NUMDOC];

int rawfd1, rawfd2;

int voxdev; /* Voice channel device handle. */
int dev; /* Fax channel device handle. */

extern int errno;

unsigned short value;

.
/*
* Open the channel using vpm_open( ) to obtain the SCT
* VOICE channel device handle in voxdev.
*/
if ((voxdev = vpm_open("vpmB1C1", NULL)) == -1) {
    /* Error opening device. */
    printf("Error opening channel, errno = %d\n", errno);
    exit(1);
}

/*
* Open the channel using fpm_open( ) to obtain the FAX
* channel device handle in dev.
*/
if ((dev = fpm_open("vpmB1C1", NULL)) == -1) {
    /* Error opening device. */
    printf("Error opening channel, errno = %d\n", errno);
    exit(1);
}

.
/*
* Take channel offhook using vpm_sethook( ) and perform

```

```

* outbound dial using vpm_dial( ). Use voxdev as
* channel device handle for Voice API functions.
*/
.
/* Required -- Set initial state of FAX channel to TRANSMITTER. */
if (fpm_initstat(dev,DF_TX) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
.
/* Open raw and TIFF/F files to transmit. */
rawfd = vpm_fileopen("coversht.raw",O_RDONLY|O_BINARY, NULL);
fpm_setiott(&iott[0],rawfd,DF_RAW,0);
iott[0].io_firstpg = 2L;
iott[0].io_pgcount = 2L;

rawfd = vpm_fileopen("coversht2.raw",O_RDONLY|O_BINARY, NULL);
fpm_setiott(&iott[1],rawfd,DF_RAW,0);
iott[1].io_type |= IO_EOT;
iott[1].io_firstpg = 2L;
iott[1].io_pgcount = 2L;

/*
* Set the fax state of the channel to DF_TX using
* fpm_initstat( ).
*/
/* Send all fax data now - synchronous mode. */
if (fpm_sendfax(dev,iott,EV_SYNC) == -1) {
    printf("Error code: %ld Error message: %s\n",ATDV_LASTERR(dev),
        ATDV_ERRMSGP(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    printf("Phase E status: %ld\n", FPMX_ESTAT(dev));
}

```



```

        /* Further error processing - application specific. */
        .
    }

```

Example 2 – fpm_sendfax() – Asynchronous programming mode

```

#include <stdio.h>
#include <errno.h>
#include <string.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include <io.h>
#define MAXCHANS 12
/* Global variables. */
extern int errno;
int catchall( );
int fax_send( );
/* Error routine - print error information. */
void print_err(dev)
int dev;
{
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    return;
}
/*
 * main( ): Opens all channels and enables handler for
 * asynchronous operation. Channels go off-hook, dial the
 * appropriate number and send the fax document.
 */
main( )

```

```

{
int chan;

int voxdev; /* Voice channel device handle. */
int faxdev; /* Fax channel device handle. */
char * chnamep;
int mode = SR_POLLMODE;
for (chan=0; chan < MAXCHANS; chan++) {
/*
* Set chnamep to the channel device name, e.g.,
* vpmB1C1, vpmB1C2, etc.
* Open the channel using vpm_open( ) such that voxdev
* has the VOICE channel device handle.
* Open the channel using fpm_open( ) such that faxdev
* has the FAX channel device handle.
*/
.
/*
* Place channel on-hook by calling vpm_sethook( ) with
* its mode field set to EV_ASYNC (asynchronous).
*/
}
/*
* All channels have been opened and a sethook function
* issued to place the channels on-hook. Use sr_waitevt( )
* to wait for completion events. On receiving any
* completion event, control is transferred to the
* catchall( ) handler function.
*/
.
.
}
/* Event handler. */
/*
* This routine gets called when sr_waitevt( ) receives any event.
* Maintain a state machine for every channel and issue the
* appropriate function depending on the next action to be

```

```

* performed on the channel, e.g., the application may wish
* to perform an outbound dial after receiving an offhook
* completion event.
*/

int catchall( )
{
    int dev;
    char * fnamep;
    long phdcmd, phdrpy;
    dev = sr_getevtdev( );
    /* Determine the event. */
    switch(sr_getevtttype( )) {
        case TDX_SETHOOK:
            .
            .
            break;
        case TDX_DIAL:
            /* Dial complete. */
            .
            .
            /*
            * Connection has been established with remote
            * receiver. Prepare to send fax. Call fax_send( ) -
            * fnamep is the name of the file (TIFF/F) containing
            * the document to be sent.
            */
            if (fax_send(dev, fnamep,DF_TIFF) == -1) {
                /*
                * Application specific error handling here;
                * fax_send( ) prints out error information.
                */
                .
                .
            }
            break;
        case TFX_FAXSEND:

```

```

/* The document has been successfully sent. */
printf("Sent %ld pages at speed %ld, resln %ld,width %ld\n", FPMX_PGXFER(dev),
        FPMX_SPEED(dev),FPMX_RESLN(dev), FPMX_WIDTH(dev));

/* Set channel on-hook; fax session completed. */
.
.
break;

case TFX_FAXERROR:
/* Error during the fax session. */
print_err(dev);
printf("Phase E status %ld\n", FPMX_ESTAT(dev));
/* Application specific error handling. */
.
.
break;

default:
.
.
break;

} /* End of switch. */
return(0);
}

/*
* This routine is called from the catchall( ) event handler
* after an outbound dial has successfully completed and a
* fax document has to be sent. The fax_send( ) routine will
* perform the necessary initialization of the DF_IOTT
* structure and call fpm_sendfax( ) to send the document.
*/

int fax_send(dev, filenamep, datatype)
int dev;
char * filenamep;
int datatype;
{
int fhandle;
/*

```

```

* Set the Local ID using fpm_setparm( ) and set the
* initial state of the channel to be a transmitter
* (DF_TX) using fpm_initstat( ).
*/
.
/*
* Set up the DF_IOTT structure to send the required
* document.
*/
if((fhandle = vpm_fileopen(filenamep, O_RDONLY|O_BINARY, NULL))==-1) {
printf("Unable to open send file %s\n",filenamep);
return(-1);
}
fpm_setiott(&iott, fhandle, datatype, 0);
iott.io_type |= IO_EOT;
/*
* Set the fax state of the channel to DF_TX using
* fpm_initstat( ).
*/
.
if (fpm_sendfax(dev, &iott, EV_ASYNC) == -1) {
    printf("Error issuing sendfax\n");
    print_err(dev);
    .
    return(-1);
}
return(0);
}

```

NOTES: 1. Example 1 , DF_IOTT entry io_type ,
DF_IOTT entry DF_IOTT array . Array entry
. (DF_IOTT structure가
io_type field IO_CONT)
2. Example 1 , io_type field fpm_setiott .
3. Example 1 DF_IOTT entry , io_type field IO_EOT가
DF_IOTT entry .

Errors

error code Appendix D .

See Also

- FPMX_BADIOTT()
- FPMX_BADPAGE()
- FPMX_BADSCANLINES()
- FPMX_BSTAT()
- FPMX_CHTYPE()
- FPMX_CODING()
- FPMX_ESTAT()
- FPMX_LASTIOTT()
- FPMX_PGXFER()
- FPMX_PHDCMD()
- FPMX_PHDRPY()
- FPMX_RESLN()
- FPMX_SCANLINES()
- FPMX_SPEED()
- FPMX_STATE()
- FPMX_TERMMSK()
- FPMX_TFBADTAG()
- FPMX_TFNOTAG()
- FPMX_TFPGBASE()
- FPMX_TRCOUNT()
- FPMX_WIDTH()
- fpm_getDCS()
- fpm_getDIS()
- fpm_getNSF()
- fpm_senddraw() (Convenience Function Reference, Chapter 5)
- fpm_sendtiff() (Convenience Function Reference, Chapter 5)
- fpm_setiott()
- fpm_setuio()

sets up default DF_IOTT structure values

fpm_setiott()

Name:	int fpm_setiott(iotp, fhandle, dtype, cont)		
Inputs:	DF_IOTT *iotp	·DF_IOTT	pointer
	int fhandle	·file	descriptor
	unsigned short dtype	·fax data	type
	unsigned short cont	·Phase D	continuation value
Returns:	None		
Includes:	srllib.h		
	dxxplib.h		
	faxlib.h		
Category:	Initialize DF_IOTT		
Mode:	synchronous		

Description

fpm_setiott()

dtype

fax data

DF_IOTT structure

· DF_IOTT structure

6.3.1

DF_IOTT -Fax Transmit

Data Description

·

NOTE : DF_IOTT field value

DF_IOTT

structure

·

Parameter Description	
iotp:	DF_IOTT structure pointer
fhandle:	file descriptor
dtype:	data type. dtype 가 .
Value	Description
DF_TIFF	TIFF/F structured formatted fax data
DF_RAW	Raw, unformatted fax data
Phase D continuation value. Cont 가 .	

Details

Contiguous Entries

fpm_setiott() default file fax data , DF_IOTT entry
.(fpm_setiott source code fpm_sendfax reference
example .)

entry가 io_nextp io_prevp NULL .

Linked Entries

DF_IOTT entry가 entry , fpm_setiott entry
, io_type field IO_LINK OR masking , io_nextp DF_IOTT
entry 가 .

TIFF/F File Entry Defaults

dtype DF_TIFF , descriptor TIFF/F data .
io_firstpg = 0L;
io_pgcount = -1L;

NOTE: TIFF/F file entry default DF_IOTT structure page
. Reference source
code .

Raw File Entry Defaults

dtype DF_RAW , descriptor format raw fax data .
io_offset = 0L;
io_length = -1L;
io_width = DF_WID1728;
io_resln = DF_RES1;
io_coding = DF_MH;

NOTE: raw file entry default DF_IOTT structure raw file
.

Cautions

None.

Example

fpm_setiott() , fpm_sendfax() .

Source Code for fpm_setiott()

```
void fpm_setiott(iotp,fhandle,dtype,cont)
DF_IOTT *iotp;
int fhandle;
unsigned short dtype;
unsigned short cont;
{
/* Data in file, next entry contiguous. */
iotp->io_type = IO_DEV;
iotp->io_fhandle = fhandle;
iotp->io_nextp = (DF_IOTT *)NULL;
iotp->io_prevp = (DF_IOTT *)NULL;
iotp->io_datatype = dtype;
iotp->io_phdcont = cont;
switch (dtype) {
    /* For TIFF/F, set up firstpg and pgcount to send all pages. */
    case DF_TIFF:
        iotp->io_firstpg = 0L;
        break;
    /*
    * For raw file, set up to send complete file at default width and
    * resolution.
    */
    case DF_RAW:
        iotp->io_offset = 0L;
        iotp->io_length = -1L;
        iotp->io_width = DF_WID1728;
        iotp->io_resln = DF_RES1;
        iotp->io_coding = DF_MH;
        break;
```

```
        default:
            break;
    }
    return;
}
```

Example

None.

See Also

- FPMX_BADIOTT()
- FPMX_BADPAGE()
- FPMX_LASTIOTT()
- FPMX_TFBADTAG()
- FPMX_TFNOTAG()
- FPMX_TFPGBASE()
- fpm_sendfax()
- fpm_senddraw() (Convenience Function Reference, Chapter 5)
- fpm_sendtiff() (Convenience Function Reference, Chapter 5)

fpm_setparm() sets the FAX parameter

Name:	int fpm_setparm(dev, parm, valuep)		
Inputs:	int dev	·FAX channel device handle	
	Unsigned long parm	·Parameter to set	
	Void *valuep	·Pointer to parameter value	
Returns:	, 0		
	, -1		
Includes:	srllib.h		
	dxxplib.h		
	faxlib.h		
Category:	Configuration		
Mode:	synchronous		

Description				
fpm_setparm()	parm	fax parameter	dev	channel device
.				

Parameter Description				
dev:	channel	open	FAX channel	channel
	device handle .			
parm:	FAX Parameter	(alphabet	
	parameter	.)		
valuep:	parm value가	void*	cast	location pointer
FPMCH_MYFAXNO	Default: NULL			
	Local identification – null terminated 10 character			
FPMCH_SOURCEFAXNO	Remote identification – null terminated 10 character			
FPMCH_RXBAUDRATE	Byes: 2			
	Default: DF_MAXBAUD			
	fax data	baud rate.		

DF_MAXBAUD : byte
 DF_9600BAUD : 9600baud
 DF_7200BAUD : 7200baud
 DF_4800BAUD : 4800baud

FPMCH_RXCODING Bytes: 2
 Default: DF_MH
 fax data encoding scheme

DF_MH Modified Huffman
 DF_MMR future use

FPMCH_TXBAUDRATE Bytes: 2
 Default: Maximum baud rate:
 maximum baud rate

DF_MAXBAUD: maximum baud rate
 DF_9600BAUD : 9600baud
 DF_7200BAUD : 7200baud
 DF_4800BAUD : 4800baud
 DF_2400BAUD : 2400baud

FPMCH_TXCODING Bytes: 2
 Default: DF_MMR
 Fax encoding scheme

DF_MH Modified Huffman
 DF_MR Modified Read
 DF_MMR Modified Modified Read

Details

Maximum value string , string . string
 maximum value , string blank .

FPMCH_RXBAUDRATE parameter fax baud rate .

FPMCH _RXCODING parameter fax data가 TIFF/F RAW
encoding scheme .

FPMCH _TXCODING parameter fax encoding scheme 가
.

Cautions

set parameter (void *) variable cast
.

NOTE: FAX parameter vpm_setparm() .

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int dev;

unsigned short value;

extern int errno;

char *coname = "ABCDE Company";

/*
 * Open device using fpm_open( ). Obtain SCT fax device
 * handle in dev.
 */
.

value = DF_MH;

if (fpm_setparm(dev, FPM_TXCODING,(void *)&value) == -1) {
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev),ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
}
```

}

Errors

가 error code list Appendix D .

NOTE: FAX hardware channel parameter
, ATDV_LASTERR() EFX_UNSUPPORTED error code .

See Also

·fpm_getparm()

Name:	int fpm_setuio(df_uio)
Inputs:	DF_UIO df_uio .DF_UIO structure
Returns:	none
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Miscellaneous
Mode:	synchronous

Description

fpm_setuio() I/O function , standard I/O function read, write, lseek SCT FAX library . network device data access .

Parameter	Description
df_uio:	DF_UIO structure

application DF_UIO structure , read, write, lseek address . , application fpm_setuio() , I/O function .

Fax I/O function mode , fax library fpm_setuio() I/O standard I/O function . I/O read(), write(), lseek() .

Fax fpm_setuio() , write .

Fax fpm_setuio() , read .

Cautions

None.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxlib.h>
#include <faxlib.h>

DF_UIO userio;

/* User read function (note: same arguments as read( )) */
int user_read(filedes, buf, size)
int filedes;
char * buf;
unsigned size;
{
/* Application specific read( ) function. */
.
.
}

/* User write function (Note: Same arguments as write( )). */
int user_write(filedes, buf, size)
int filedes;
char * buf;
unsigned nbyte;
{
/* Application specific read( ) function. */
.
.
}

/* User lseek function (Note: Same arguments as lseek( )). */
long user_lseek(filedes, offset, whence)
int filedes;
long offset;
int whence;
{
```



```

/* Application specific lseek( ) function. */

.
.
}
main( )
{
.
.
userio.u_read = user_read;
userio.u_write = user_write;
userio.u_seek = user_lseek;
/* Register these functions with the FAX library. */
fpm_setuio(userio);
.
.
}

```

Error

None.

fpm_stopch() forces termination of a fax transmission or reception

Name:	int fpm_stopch(dev, mode)
Inputs:	int dev ·Valid SCT FAX channel device handle Unsigned short mode ·Synchronous/asynchronous mode bit map
Returns:	, 0 , -1
Includes:	srllib.h dxxplib.h faxlib.h
Category:	Resource Management
Mode:	synchronous/asynchronous

Description

fpm_stopch() FAX channel device fax
FAX channel busy idle

Parameter	Description
dev:	channel open valid FAX channel device handle
Mode:	EV_SYNC ? synchronous mode operation EV_ASYNC ? asynchronous mode operation

Details

, fax fpm_stopch()

Fax phase , fpm_stopch() FAX channel device ,
fax , fax session T.30 protocol Phase E .
send receive -1 , ATDV_LASTERR() ,
EFX_DISCONNECT , FPMX_ESTAT EFX_ABORTCMD
.
NOTE: fpm_stopch()가 , fax T.30 protocol Phase
, fpm_stopch()가 .

Synchronous Mode Operation

`fpm_stopch()` 가 , `fpm_stopch()` FAX channel device
가 idle 가 .

Asynchronous Mode Operation

, fax channel device stop
 .

Even handler `fpm_stopch()` , `fpm_stopch()`
 .

Killing a Process Running a Synchronous or Asynchronous FAX Application

T.30 fax protocol , process
가 kill exit signal handler . signal
handler fax call `fpm_stopch()` .

Cautions

1. FAX send receive 가 `fpm_stopch()` .
2. signal handler `fpm_stopch()`가 , mode `EV_ASYNC` .

Example

```
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>
main( )
{
    int dev; /* Fax channel device handle.*/
    /* Open the FAX channel device. */
    if ((dev = fpm_open("vpmB1C1", NULL)) == -1) {
        /* Error opening device. */
        printf("Error opening channel, errno = %d\n", errno);
        exit(1);
    }
}
```

```

}

/* Use the FAX channel device to send or receive faxes. */

.

.

/*

* Issue a stop to force the termination of the fax session
* if necessary.
*/

if (fpm_stopch(dev, EV_ASYNC) == -1) {
    /* Error stopping device. */
    printf("Error stopping channel\n");
    printf("Error - %s (error code %d)\n", ATDV_ERRMSGP(dev), ATDV_LASTERR(dev));
    if (ATDV_LASTERR(dev) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    exit(1);
}

.
}

```

Errors

가 ?1 , ATDV_LASTERR() ATDV_ERRMSGP()
, error가 .

EDX_BADPARAM	Invalid FAX parameter
EDX_SYSTEM	System error ? check errno

See Also

- FPMX_TERMMSK()
- fpm_rcvfax()
- fpm_rcvfax2()
- fpm_sendfax()
- fpm_stopch() (refer to the Voice Software Reference)

5. FAX Convenience Function Reference

FAX Convenience function		reference	
FAX convenience	faxconv.c	file	.
NOTE: convenience			.
	, faxconv.c file		.

send a single page of raw fax data fpm_sendraw()

Name: int fpm_sendraw(faxname, width, resln, phdcont)

Inputs:

- char * faxname · Raw file
- unsigned short width · data
- unsigned char resln · data resolution
- unsigned char phdcont ·Phasd D continuation value

Returns:

- , 0
- , -1

Includes:

- srllib.h
- dxxplib.h
- faxlib.h

Category: Transmit Fax

Mode: synchronous

Description

fpm_sendraw() parameter , width resolution 가 page raw
fax data . Fx_sendraw() convenience , faxconv.c 가
·

Parameter	Description								
faxname:	send raw file ·								
width:	carriage width . Width 가 .								
	<table><tr><th>Value</th><th>Description</th></tr><tr><td>DF_WID1728</td><td>line 1728 pixel</td></tr><tr><td>DF_WID2048</td><td>line 2048 pixel</td></tr><tr><td>DF_WID2432</td><td>line 2432 pixel</td></tr></table>	Value	Description	DF_WID1728	line 1728 pixel	DF_WID2048	line 2048 pixel	DF_WID2432	line 2432 pixel
Value	Description								
DF_WID1728	line 1728 pixel								
DF_WID2048	line 2048 pixel								
DF_WID2432	line 2432 pixel								
resln:	data resolution , resolution 가 .								
	<table><tr><th>Value</th><th>Description</th></tr></table>	Value	Description						
Value	Description								

DF_RESLO	Low (coarse) resolution (98 lpi)
DF_RESLO	Low (coarse) resolution (98 lpi)

fpm_senddraw()	fpm_sendfax()	. fpm_sendfax()	reference
fpm_senddraw()	raw file error handling		Phase D
continuation value		.	

Cautions

fpm_senddraw	:	
·vox handle	SCT handle	vpm_open()
channel open	.	
Devhandle	fax device handle	fpm_open
channel open	.	
· Scubas가	, convenience	routing
.		

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int voxhandle; /* Voice channel device handle. */
int devhandle; /* Fax channel device handle. */
extern int errno;

/*
 * Open the channel using vpm_open( ) to obtain the SCT
 * VOICE device handle in voxhandle.
 * Open the channel using fpm_open( ) to obtain the FAX channel
 * device handle in devhandle.
 */
/*
 * Take channel offhook using vpm_sethook( ) and perform outbound
 * dial using vpm_dial( ).
```

```

*/
.
.
/*
* Set the fax state of the channel to DF_TX using
* fpm_initstat( ).
*/
.
.
/*
* Transmit raw document at page width 1728 pixels per line
* at low (coarse) vertical resolution and disconnect when
* finished.
*/
if (fpm_sendraw("document.raw",DF_WID1728,DF_RESLO,0) == -1 {
    printf("Error code: %ld Error message: %s\n",ATDV_LASTERR(devhandle),
        ATDV_ERRMSGP(devhandle));
    if (ATDV_LASTERR(devhandle) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    printf("Phase E status: %ld\n", FPMX_ESTAT(devhandle));
}

```

Source Code for fpm_sendraw()

```

/*
* NOTE: devhandle is a global variable of type int. Prior to
* calling fpm_sendraw( ), the channel is opened using
* fpm_open( ) to obtain the SCT FAX channel device handle in
* devhandle.
*/
DF_IOTT iott;
int fpm_sendraw(faxname,width,resln,phdcont)
char * faxname;
unsigned short width;
unsigned char resln;
unsigned char phdcont

```



```

{
    int erc;

    /* Open the file as read-only. */
    if ((iott.io_fhandle = vpm_fileopen(faxname,O_RDONLY|O_BINARY,0)) == -1) {
        return(-1);
    }

    /*
     * Set up the DF_IOTT structure as the default and then
     change the necessary fields.
     */
    fpm_setiott(&iott,iott.io_fhandle,DF_RAW,phdcont);
    iott.io_type |= IO_EOT;
    iott.io_width = width;
    iott.io_resln = resln;
    erc = fpm_sendfax(devhandle,&iott, EV_SYNC)
    vpm_fileclose(iott.fhandle);
    return(erc);
}

```

Errors

error code list Appendix D .

See Also

- FPMX_TERMMSK() (Main Library Function Reference, Chapter 4)
- fpm_sendfax() (Main Library Function Reference, Chapter 4)
- fpm_setiott() (Main Library Function Reference, Chapter 4)

send pages of a single TIFF/F file fpm_sendtiff()

Name: int fpm_sendtiff(faxname, firstpg, pgcount, phdcont)

Inputs: char * faxname · TIFF/F file pointer

 unsigned long first pg ·

 unsigned long pgcount · consecutive page

 unsigned short phdcont ·Phase D continuation value

Returns: , 0

 , -1

Includes: srllib.h

 dxxplib.h

 faxlib.h

Category: Transmit Fax

Mode: synchronous

 Description

fpm_sendtiff()	TIFF/F file	width	resolution	TIFF/F file
application		·	Fx_sendtiff()	faxconv.c
convenience	·			

Parameter	Description
faxname:	send TIFF/F file 가 pointer
firstpg:	page document page
pgcount::	page
phdcont:	0
fpm_sendtiff()	fpm_sendfax() · fpm_sendfax() reference
fpm_sendtiff	TIFF/F file error handling , Phase D continuation value
	·

 Cautions

fpm_sendtiff :

·vox handle SCT handle vpm_open()

channel open ·

Devhandle fax device handle fpm_open
channel open .

.SCbus가 , convenience routing
.

Example

```
#include <stdio.h>
#include <errno.h>
#include <srllib.h>
#include <dxxplib.h>
#include <faxlib.h>

int voxhandle; /* Voice channel device handle. */
int devhandle; /* Fax channel device handle. */
extern int errno;

/*
 * Open the channel using vpm_open( ) to obtain the SCT
 * VOICE device handle in voxhandle.
 * Open the channel using fpm_open( ) to obtain the FAX channel
 * device handle in devhandle.
 */

/*
 * Take channel offhook using vpm_sethook( ) and perform outbound
 * dial using vpm_dial( ).
 */

.
.
/*
 * Set the fax state of the channel to DF_TX using
 * fpm_initstat( ).
 */

.
.
/*
 * Send 2 pages starting at page number 4, disconnect when
```

```

* finished.
*/
if (fpm_sendtiff("document.tif",4L,2L,0) == -1) {
    /* Process error. */
    printf("Error - %s (error code %d)\n",ATDV_ERRMSGP(devhandle),
        ATDV_LASTERR(devhandle));
    if (ATDV_LASTERR(devhandle) == EDX_SYSTEM) {
        printf("errno = %d\n", errno);
    }
    printf("Phase E status: %ld\n", FPMX_ESTAT(devhandle));
/* Application specific error handling. */
.
.
}

```

Source Code for fpm_senddraw()

```

/*
* NOTE: devhandle is a global variable of type int. Prior to
* calling fpm_sendtiff( ), the channel is opened using
* fpm_open( ) to obtain the SCT FAX channel device handle in
* devhandle.
*/
DF_IOTT iott;
int fpm_sendtiff(faxname,firstpg,pgcount,phdcont)
char * faxname;
unsigned long firstpg;
unsigned long pgcount;
unsigned short phdcont
{
    int erc;
/* Open the file as read-only. */
if ((iott.io_fhandle = vpm_fileopen(faxname,O_RDONLY|O_BINARY,0)) == -1) {
    return(-1);
}
/*
* Set up the DF_IOTT structure as the default and then

```

```

* change the necessary fields.
*/

fpm_setiott(&iott,iott.io_fhandle,DF_TIFF,phdcont);
iott.io_type |= IO_EOT;
iott.io_firstpg = firstpg;
iott.io_pgcount = pgcount;
erc = fpm_sendfax(devhandle,&iott, EV_SYNC)
vpm_fileclose(iott.io_fhandle);
return(erc);
}

```

Errors

error code list Appendix D .

See Also

- FPMX_TERMMSK() (Main Library Function Reference, Chapter 4)
- fpm_sendfax() (Main Library Function Reference, Chapter 4)
- fpm_setiott() (Main Library Function Reference, Chapter 4)

DF_IOTT	·FAX Transfer Table Structure
DF_UIO	·User -definable I/O structure
DF_DCS	·Digital Command Signal Information Structure
DF_DIS	·Digital Identification Signal Information Structure

application , DF_IOTT structure DF_UIO structure
access .

6.3. FAX Library Data Structures Reference

FAX library data structure faxlib.h . program install ,
install .

6.3.1. DF_IOTT – Fax Transmit Data Description

DF_IOTT structure fax data .

```
typedef struct df_iott DF_IOTT;
struct df_iott {
    unsigned long io_offset; /* Starting page number or offset */
    unsigned long io_length; /* Number of pages or length of data */
    char *io_bufferp; /* Memory transfer start buffer location */
    DF_IOTT *io_prevp; /* (Optional) Pointer to previous DF_IOTT */
    DF_IOTT *io_nextp; /* Pointer to next DF_IOTT entry (for linked list) */
    void *io_datap; /* Pointer to additional data associated */
    /* with io_datatype */
    int io_fhandle; /* File descriptor */
    unsigned short io_type; /* Entry type (file, memory; linked, contiguous, */
    /* last structure; select user-defined I/O */
    /* functions for transmit) */
    unsigned short io_datatype; /* Type of data to transmit */
    unsigned short io_phdcont; /* Phase D continuation value to send */
    unsigned short io_width; /* Width of image (raw) */
    unsigned char io_resln; /* Vertical resolution of image (raw) */
    unsigned char io_coding; /* Encoding of stored data (raw) */
    unsigned char rfu[2]; /* Reserved for future use */
};
```

```
        Define      DF_IOTT structure .
#define io_firstpg  io_offset
#define io_pgcount  io_length
```

DF_IOTT Structure Field Values

Field	Possible Values	Description
io_offset	0	Raw files: data offset byte (0 = no offset)
io_firstpg	0	Byte offset file(or memory) to start TIFF/F file only: page number (decimal value). page (page 0 .)
io_length	>0 -1	Raw files: byte raw file
io_pgcount	>0 -1	TIFF/F files: page page page
io_bufferp		buffer
io_prevp		DF_IOTT pointer (Optional)
io_nextp		DF_IOTT pointer (for linked list)
io_datap		io_datatype 가 data pointer (cast as void *)
io_fhandle		File descriptor
io_type	IO_DEV IO_MEM IO_CONT IO_LINK IO_EOT IO_UIO	file (raw data only) DF_IOTT가 memory . (default) DF_IOTT가 entry link . DF_IOTT entry . fax I/O .
io_datatype	DF_TIFF	TIFF/F file data

	DF_RAW	Raw file data
io_phdcont	not use	
io_width		Raw files (width of image):
	DF_WID1728	1728 pixels per line
	DF_WID2048	2048 pixels per line
	DF_WID2432	2432 pixels per line
io_resln		Raw files (resolution of image):
	DF_RESHI	High (fine) vertical resolution (196)
	DF_RESLO	Low (coarse) vertical resolution (98)
		io_coding Raw files:
		raw data encoding scheme
	DF_MH	Modified Huffman - one -dimensional encoding
	DF_MMR	
rfu[2]		

DF_IOTT structure document sources fax .

Application fax DF_IOTT structure array linked list

. Send 가 , DF_IOTT structure structure가

parameter 가 check .

Setting Values for DF_IOTT Fields

df_iott.io_type OR bit masked .

·fax data가 file(IO_DEV) memory (IO_MEM raw

data .)

·structure가 array (IO_CONT) linked list (IO_LINK)

NOTE: df_iott.io_type IO_LINK , df_iott.io_nextp field

entry DF_IOTT entry .

df_iott.io_type IO_DEV , .

·df_iott.io_datatype DF_TIFF

- data source TIFF/F file .

- df_iott.io_firstpg (df_iott.io_offset) page number

. (page number 0 .)

- df_iott.io_pgcount (df_iott.io_length) page .
- df_iott.io_width image width .
- df_iott.io_resln fax data vertical resolution .
- df_iott.io_coding raw data encoding scheme .

df_iott.io_type IO_MEM .

- df_iott.io_datatype DF_RAW
- data source raw data .
- df_iott.io_bufferp data 가 memory buffer pointer
가 .
- df_iott.io_offset dadta byte .
- df_iott.io_length byte .
- df_iott.io_width image width .
- df_iott.io_resln vertical resolution .
- df_iott.io_coding raw data encoding scheme .

6.3.2. DF_UIO – User-Defined I/O

fpm_setuio() DF_UIO structure I/O function read, write, lseek
pointer .

```
typedef struct df_uio {
    int (*u_read)( ); /* User defined replacement for read( ) */
    int (*u_write)( ); /* User defined replacement for write( ) */
    long (*u_seek)( ); /* User defined replacement for lseek( ) */
};
```

u_read field read() pointer .

u_write field write() pointer .

u_seek field lseek() pointer .

Fax library read, write, seek pointer

NOTE: I/O I/O ,
return type .

fpm_sendfax() I/O: application I/O
, fpm_sendfax() DF_IOTT structure biotype field IO_UIO
OR bit mask . fax library가 seek, read
DF_IOTT structure . file
descriptor DF_IOTT structure io_fhandle .

NOTE: fpm_sendfax() DF_IOTT structure array ,
DF_IOTT structure io_type field가 IO_UIO bit
masking . DF_IOTT structure I/O
가 .

fpm_rcvfax2() User I/O: fax ,
fpm_rcvfax2() 가 . rcvflag IO_UIO bit OR
masking . , fax library I/O device fax
data .

NOTE: I/O file descriptor fpm_rcvfax2()
fd .

6.3.3. DF_DCS – Digital Command Signal Information

DF_DCS structure fpm_getDCS() 가 , T.30 Digital command signal
.

```
typedef struct {
    char dcs_data[10]; /* DCS information in LSB format */
} DF_DCS;
```

NOTE: DCS ITU Publication Procedures for Document Facsimile
Transmission in the General Switched Telephone Network, Recommendation
T.30 .

6.3.4. DF_DIS – Digital Identification Signal Information

Appendix A

TIFF/F Tags and Values

Input to the Library (from Disk Storage)

Library	TIFF/F file	tag	가	TIFF/F file	. Page
number tag	,	tag	.	tag	default
check	. TIFF/F tag field	TIFF/F tag	subset	.	

Field	Valid Values	Description
BitsPerSample	1	one bit per sample
Compression*	3	ITU T.4: Group 3 encoding
Field	Valid Values	Description
	4	ITU T.6: Group 4 encoding - MMR
FillOrder*	1	most significant bit first
	2	least significant bit first
ImageWidth*	1728	number of pixels per line
	2048	number of pixels per line
	2432	number of pixels per line
PageNumber	x/x	page number/number of pages
SamplesPerPixel	1	one sample per pixel
StripByteCounts*	x	byte count in image (as appropriate)
StripOffsets*	x	byte offset of image (as appropriate)
T4Options	x/x	1-Dimensional - End Of Line not padded/padded (MH encoding)
T6Options	x	Group 4 - MMR encoding
YResolution*	< or =150	coarse (normal)
	>150	fine
page number tag가	, FAX library	page . Page number
0	1	.

NOTE: page numbering 0 , TIFF/F .

T4Option T6Option tag가 , FAX library .

Field	Valid Values	Description
T4Options		1-Dimensional
	0	End Of Line: not padded
	4	End Of Line: padded
T6Options	0	Group 4 - MMR encoding
NOTE:	, RowsPerStrip	Image Length . StripOffsets
StripByteCounts tag		TIFF/F file . Image
multiple strips		.

Output from the Library (to Disk Storage)

Library TIFF/F file 가 tag .

Field	Valid Values	Description
BadFaxLines	x	bad scan lines
BitsPerSample	1	one bit per sample
CleanFaxData	0	bad scan line data
		Bad Scan line Replacement(BLR) Modified
		Huffman encoded scan line clear Fax data
	tag 0	pixel count
	가 .	scan line image
		. Application data
		error , Bad Fax Lines
		, image .
Compression	3	ITU T.4 encoding: Group 3 encoding - MH
	4	ITU T.6 encoding: Group 4 encoding ? MMR
DateTime	YYYY:MM:DD HH:MM:SS	
FillOrder	2	least significant bit first

ImageWidth	1728	number of pixels per line
Field	Valid Values	Description
	2048	number of pixels per line
	2432	number of pixels per line
ImageLength	x	number of scan lines in image
NewSubFileType	2	single page in multi-page image
Orientation	1	1st row = top left, 1st column = top left
PageNumber	x/x	page number/number of pages
PhotometricInterpretation	0	gray scale/bilevel: zero = white
ResolutionUnit	2	inch
RowsPerStrip	x	number of scan lines in image
SamplesPerPixel	1	one sample per pixel
Software	SCT	TIFF/F Library version x.xx
StripByteCounts	x	byte count in image (as appropriate)
StripOffsets	x	byte offset of image (as appropriate)
T4Options	0	1-Dimensional (MH) - End Of Line not padded
T6Options	0	Note: Replaces T4Options if MMR
XResolution	204	number of pixels per resolution unit X
YResolution	196	fine
	98	coarse (normal) (number of pixels per resolution unit Y)

Appendix B

FAX Phase D Status values

Phase D value fax 가 .

Phase D value TFX_PHASED event가 , fax
page , FAX Extended Attribute .

FPMX_PHDCMD Phase D command . Phase D command fax session
Phase Receiver .

Phase D Command (from TRANSMITTER to RECEIVER)

Value	Description
DFS_EOP	End Of Procedure - fax session ; Phase E ; fax call .
DFS_MPS	Multi-page Signal ? fax document page ; fax document format fax document format ; Phase C
DFS_EOM	End Of Message ? fax document page ; Phase B ; fax parameter .
FPMX_PHDRPY()	Receiver Phase D reply . Phase D Reply quality 가 .

Phase D Reply (from RECEIVER to TRANSMITTER)

Value	Description
DFS_RTN	fax page가 . .
DFS_RTP	fax가 . Baud rate가 .
DFS_PIP	Procedure interrupt positive (operator intervention request)
DFS_PIN	Procedure interrupt negative (operator intervention request)

Appendix C

FAX Phase E Status values

Phase E fax error 가 . Phase E
value FPMX_ESTAT() FAX Extended Attribute .

Phase E Status Values Returned to the TRANSMITTER

Value	Description
EFX_BADDCSTX	Received bad response to DCS (Digital Command Signal) or training
EFX_BADPGTX	Received a DCN (Disconnect) from remote after sending a page
EFX_COMMERRTX	Transmit communication error
EFX_GOTDCNTX	Received a DCN (Disconnect) while waiting for a DIS (Digital Identification Signal)
EFX_INVALIDMMRTX	Invalid input MMR data
EFX_INVALIDRSPTX	Invalid response after sending a page
EFX_NODISTX	Received other than DIS (Digital Identification Signal) while waiting for DIS
EFX_NOFINERECTX	Remote cannot receive fine resolution
EFX_NOISETX	Too much noise at 2400 bps
EFX_NOWIDTHTX	Remote cannot receive at this width
EFX_NXTCMDTX	Timed out waiting for next send_page command from driver
EFX_PHBDEADTX	Received no response to DCS (Digital Command Signal), training or TCF (Training Check)
EFX_PHDDEADTX	No response after sending a page
EFX_RXCOMP	Remote site is not receive compatible
EFX_T1EXPTX	Timed out while waiting for a message
EFX_COMMERRRX	Receiver communication error
EFX_DCNDATARX	Unexpected DCN (Disconnect) while waiting for fax data
EFX_DCNFAXRX	Unexpected DCN (Disconnect) while waiting for EOM (End Of Message), EOP (End Of Procedure) or MPS (Multi-page Signal)

Value	Description
EFX_DCNNORTNRX	DCN (Disconnect) after requested retransmission
EFX_DCNPHDRX	Unexpected DCN (Disconnect) after EOM (End Of Message) or MPS (Multi-page Signal) sequence
EFX_DCNRRDRX	Unexpected DCN (Disconnect) after RR/RNR sequence
EFX_GOTDCSRX	DCS (Digital Command Signal) received while waiting for DTC
EFX_INVALIDCMDRX	Unexpected command after page received
EFX_NOCARRIERRX	Lost carrier during fax receive
EFX_NOEOLRX	Timed out while waiting for EOL (End Of Line)
EFX_NOFAXRX	Timed out while waiting for first line
EFX_NXTCMDRX	Timed out waiting for next receive page command
EFX_PNSUCRX	High speed training success not returned by modem during receive
EFX_T1EXPRX	Timed out while waiting for a message
EFX_T2EXPDCNRX	Timed out while waiting for DCN (Disconnect)
EFX_T2EXPDRX	Timed out while waiting for Phase D
EFX_T2EXPFAXRX	Timed out while waiting for fax page
EFX_T2EXPMPSTRX	Timed out while waiting for next fax page
EFX_T2EXPRRRX	Timer T2 expired waiting for RR command
EFX_T2EXPRX	Timed out waiting for NSS (Non-standard set-up), DCS (Digital Command Signal) or MCF (Message Confirmation)
EFX_TXCOMP	Remote site is not transmit compatible
EFX_WHYDCNRX	Received unexpected DCN (Disconnect) while waiting for DCS (Digital Command Signal) / DIS (Digital Identification Signal)

General Phase E Status Values Returned to RECEIVER or TRANSMITTER

Value	Description
EFX_ABORTCMD	Command stopped by stop_fax firmware command
EFX_BUSYCHN	Request to start fax while channel is currently busy
EFX_CEDTONE	Remote CED (Called Station Identification) tone exceeds 5 seconds
Value	Description
EFX_CHIPNORESP	FAX modem is not responding
EFX_HDLCCARR	Excessive HDLC carrier
EFX_OPINTFAIL	Operator intervention failed

Appendix D

FAX Error Codes

	SCT defined	FAX error code	.	error code	SCT Standard
Runtime library	ATDV_LASTERR()				. Error
string		ATDV_ERRMSGP()			.

Value	Description
EFX_BADIOTT	Invalid data in DF_IOTT entry
EFX_BADPAGE	TIFF/F page number tag missing
EFX_BADPARAM	Invalid value for fax parameter
EFX_BADPHASE	Unexpected Phase transition (internal)
EFX_BADSTATE	Invalid initial state value specified
EFX_BADTAG	Incorrect values for TIFF/F tags
EFX_BADTIF	Incorrect TIFF/F format (no data sent)
EFX_BADTFHDR	Bad TIFF/F header - incorrect values in fields
EFX_CMDDATA	Last command contained invalid data
EFX_COMPAT	I/O file type, image width and resolution not compatible with the RECEIVER's hardware
EFX_DISCONNECT	Fax call disconnected by other station
EFX_DRVERROR	Error occurred in driver (internal)
EFX_FWERROR	Firmware error
EFX_INVALIDARG	Illegal argument to function
EFX_INVALIDFUNC	Illegal call to function
EFX_LIBERROR	Error in library state machine (internal)
EFX_MAXCHAN	Maximum channel capacity reached
EFX_NODATA	Data requested is not available (NSF, DIS, DCS)
EFX_NOFAX	No fax capability on this device
EFX_NOMEM	Cannot allocate memory for more pages
EFX_NOPAGE	TIFF/F page not found
EFX_NOPOLL	Poll not accepted
EFX_NOSTATE	Initial state value not set
EFX_NOTIDLE	Channel is not idle

Value	Description
EFX_NOTIMESLOT	No time slot assigned
EFX_NSFBUF	Buffer length supplied to fpm_getNSF() <2 bytes
EFX_RDFWER	Error reading firmware version
EFX_RETRYDCN	Disconnected after specified retries
EFX_UNSUPPORTED	Unsupported feature

NOTE: Phase E value fax session error 가

Appendix E

FAX Event Codes

FAX event code .

NOTE: , Phase B Phase D event event handler가

.

Value	Description
TFX_FAXERROR	Error
TFX_FAXRECV	FAX reception complete
TFX_FAXSEND	FAX send complete
TFX_PHASEB	Phase B event
TFX_PHASED	Phase D event

Glossary

asynchronous function program task가

CCITT (International Telegraph and Telephone consultative Committee)

See ITU.

configuration file application 가 format ASCII file

delimiting fax page fax page group

application application page file

extended attributes input parameter 가 ,

class FAX Extended Attribute resource

fax session fax session T.30 protocol Phase A Phase E

Group 3 CCITT , 1980 , 1984 1988

facsimile device T.4 . Group 3 digital fax 8.5 "x11 "(A4) page

9,600bps PSTN 15-30ch .

Group 4 CCITT facsimile device T.6 . Public Data Network

PSTN , data ECM .

ID fax . ITU T.30 recommendation , local ID 2.0 character

data + 가 ,

code, , 가 .

ITU (formerly CCITT) (International Telegraph Union) An international organization

that recommends communication standards

Modified Huffman code Group3 fax device fax fax data

one-dimensional encoding scheme . , text 가

8.5 " (=1728 bits) 가 , Modified Huffman Code 1728

bit 17-bit code word . 가 , 0

1728 raw length 92 binary code group .

Modified Modified Read code (MMR) An optional, Group 4 facsimile two-dimensional digital encoding scheme with improved transfer speed over Modified Read encoding.

Modified Read code Modified Huffman encoding 가 speed

가 Group 3 two-dimensional digital encoding scheme

normal fax transmission fax document가 CALLER application CALLED

application
 pel blank white 가 picture Element. facsimile single point
 Phase A Fax call setup phase
 Phase B fax call Pre -message phase. station , RECEIVER
 TRANSMITTER , parameter
 Phase C fax call message
 Phase D data data가
 post-message procedure
 Phase E Fax call release Phase. Disconnect
 Pixel grey information level picture Element. facsimile single
 point
 PSTN Public Switched Telephone Network
 SCbus multiple data line (Signal
 Computing Bus) Third generation TDM (Time Division Multiplexed) resource
 sharing bus
 Synchronous function device program block

 TIFF/F Tagged Image File Format Class F. TIFF is a tag based general purpose raster
 format used to exchange image data between application programs. Class F
 indicates specific format information for fax applications.
 Time out network , event