gm_fd is the file descriptor of an open input file.

gm_len contains the length of the message, but not including the message termination character.

gm_delim is the message termination character, such as 03.

gm_lncnt contains a count of the number of lines in the message. The calling program may use this variable, but should not change its value.

gm_nchar contains the number of characters in the buffer after a read has been completed. This variable should not be used or changed by the calling program.

gm_lptr contains the starting addresses of each line in the message.

gm_fptr is the address of a requested field in some specified line of the message. This variable is used primarily by the getfld() routine.

gm_bufp is the address of the message in gm_buf.

gm_bufe is a pointer to the next message in gm_buf. This variable should not be used or changed by the calling program.

gm_buf is the data buffer and is usually not written into by the calling program.

FILES

/usr/include/gtmhdr.h which contains the definitions for GMBUF, **GM_BUFSIZ**, **GM_MAX_LNS**, **GFR_FLD**, and **GFR_LN**.

LIBRARY

/lib/lib1.a

SEE ALSO

gtmsg(3L), cpyfld(3L)

DIAGNOSTICS

The error codes returned by this subroutine are:

GFR_FLD The argument field is out of range.
GFR_LN The argument line is out of range.

BUGS

NAME

getlin - get line of input data

SYNOPSIS

```
int getlin(indev)
int indev;

int getl2(indev)
int indev;
```

DESCRIPTION

This subroutine reads data from the file specified by the file descriptor given as an argument and breaks it into individual lines terminated by a new line character.

Indev is the input file descriptor of an opened file. The number of characters read will be returned as the value of the function. Global variables BUFP and LEN will contain the address of the input line, and the length of the line, respectively. Global variables BUFE and IBUF are used by the routine, and should not be used by the users program.

The global variables used are:

```
char IBUF[513];     /* internal data buffer */
char *BUFP;         /* user's data ptr */
char *BUFE;         /* pointer to next location in IBUF */
int LEN;            /* count of characters read */
```

Getl2(indev) is identical to getlin(indev), except that the global variables BUFP, LEN, BUFE, and IBUF in getlin(indev), are changed to BUFP2, LEN2, BUFE2, and IBUF2, respectively, in getl2(indev).

LIBRARY

/lib/lib1.a

SEE ALSO

gets(3)

NAME

getpat -- get pattern

SYNOPSIS

jsr r5,getpat;pname;buffer;otype

DESCRIPTION

Returns 1 in r0 for successful read, zero otherwise
Returns new buffer pointer in r1 (even location)

This subrotine is designed to read in patterns written on disk by the RC:PAT command. It attempts to open the file PNAME.P. If this fails it attempts to open the file /typexx/pname, where xx = otype(2chars). If this fails, the directory, /compat is searched. if all of the above fail, the directory /usr/pat is searched. if all four searches fail, the pattern is not to be found.

RESTRICTIONS

This command will not exist beyond SC5.

LIBRARY

/lib/lib1.a

SEE ALSO

ppgetpat

NAME

glberr -- global error routine

SYNOPSIS

```
glberr(etype,ecode,enum,emsg)
char *etype,*ecode,*enum,*emsg;
```

DESCRIPTION

This subroutine prints a system error message on the system teletype, causes the error message to be logged onto the system error file, and causes an appropriate audible alarm to be generated.

Glberr has four arguments, etype, ecode, enum, and emsg. A description of these arguments follows.

- etype — is the address of a string containing the severity or type associated with an error message.
- ecode — is the address of a string containing a three-character error code.
- enum — is the address of a string containing a three-character error number.
- emsg — is the address of a string containing the message associated with the error.

LIBRARY

/lib/lib1.a

SEE ALSO

fmterr(3L), sccerr(3L), shlerr(1L)

DIAGNOSTICS

BUGS

NAME

glogin - get login name

SYNOPSIS

```
glogin (usrpwd, size)
char *usrpwd;
int size;
```

DESCRIPTION

glogin determines the login name of the user and places this null terminated string in usrpwd, a buffer supplied in the calling routine. size is the size of the buffer usrpwd, which must be greater than, or equal to the size of a line in the /etc/passwd file, which is presently 80.

DIAGNOSTICS

0	if successful
-1	if an error is encountered

FILES

/etc/passwd

LIBRARY

/lib/lib1.a

NAME

gtgrp - Get a group name and group id.

SYNOPSIS

```
gtgrp (grpnm,grpid)
      char grpnm[]
      int *grpid
```

DESCRIPTION

This routine returns in grpnm the name of a group and in grpid the corresponding group id each time that it gets called until no more groups are defined in the system. If a group name and group id are found a 1 is returned. If no more group names are found a 0 is returned. A negative return indicates an error.

The routine keeps the GROUP FILE (see aparam.h) open for buffered reads until no more groups are found or until an error is detected. The routine may be called again if the list of groups is needed from the beginning again. At any point the routine may be reset by passing to it either argument equal to zero since this is considered an error and hence the GROUP FILE is closed and a -4 is returned.

This routine uses bropen(3L), bread(3L), and brclose(3L) in lib1 and atod(3L) in lib3.

FILES

/usr/include/aparam.h
/usr/include/bread.h

LIBRARY

/lib/lib1.a

SEE ALSO

bread(3), getgrent(3L)

DIAGNOSTICS

The values returned by this subroutine are:

1	A group has been loaded into grpnm and grpid.
0	No more groups exist.
-1	The GROUP FILE can not be opened.
-2	The GROUP FILE can not be read.
-3	The GROUP FILE has bad format.
-4	The arguments passed to the subroutine are in error.

NAME

gtline -- get a line of buffered input data

SYNOPSIS

```
#include <gtlhdr.h>

gtline(func,inbuf)
int func;
struct GLBUF *inbuf;
```

DESCRIPTION

This subroutine gets a line of buffered input data by reading data from a specified input file and breaking it into individual lines terminated by some termination character, such as 03, 012, etc. Inbuf is the address of a 523(10) byte buffer area whose format is:

```
struct GLBUF
  {  int gl_fd;
     int gl_len;
     int gl_delim;
     char *gl_bufp;
     char *gl_bufe;
     char gl_buf[gl_bufsz + 1];
  };
```

where gl_fd is the input file descriptor of an opened file.

gl_len contains the length of the input line, but not including the termination character.

gl_delim is the line termination character, such as 03, 012, etc.

gl_bufp is the address of the input line in gl_buf.

gl_bufe is a pointer to the next location in gl_buf. This variable should not be used or changed by the user's program.

gl_buf is the data buffer and should not be written into by the user's program.

gl_bufsz contains the value, 512.

The argument, func, should contain the value:

-1 if the structure variables gl_len, gl_bufp, and gl_bufe, are to be initialized for a new file,

0 if blank lines in the input file are not to be returned

to the calling program, and

1 if blank lines in the input file are to be returned to the calling program.

The user's program must perform the following sequence before using this subroutine:

```
<structure name>.gl_fd= <file descriptor>;
<structure name>.gl_delim= <termination character>;
gtline(-1,&<structure name>);
```

FILES

/usr/include/gtlhdr.h which contains the definitions for GLBUF and gl_bufsz.

LIBRARY

/lib/lib1.a

SEE ALSO

DIAGNOSTICS

The values returned by this subroutine are:

-1 for an error
0 for EOF
1 for initialization function
2 for line found

BUGS

NAME

gtmsg -- locate a message in an ASCII data buffer

SYNOPSIS

```
gtmsg(func,inbuf);
int func;
struct GMBUF *inbuf;
```

DESCRIPTION

Gtmsg locates a message by reading data from an input file into an ASCII data buffer and breaking it into individual messages terminated by some termination character, such as 03. The input file must be opened by the calling program prior to calling gtmsg. The ASCII data buffer, which is a structure of type GMBUF, is declared and allocated by the calling routine. The starting address of the message is returned in the structure variable, gm_bufp; the length of the message, excluding the message termination character, is returned in the structure variable, gm_len; and one of the following values is returned in r0. If an error is detected, gtmsg returns one of the values discussed below under **DIAGNOSTICS**; otherwise, gtmsg returns one of the following values:

GMR_EOF End of File detected.
GMR_INIT Initialization successfully performed.
GMR_MFND Valid message found.

The argument inbuf is the address of a data buffer whose format is:

```
struct GMBUF
  {     int am_fd;
        int gm_len;
        int gm_delim;
        int gm_lncnt;
        int gm_nchar;
        char *gm_lptr[GM_MAX_LNS];
        char *gm_fptr;
        char *gm_bufp;
        char *gm_bufe;
        char gm_buf[GM_BUFSIZ + 2];
  };
```

where

gm_fd is the file descriptor of an open input file.

gm_len contains the length of the message, but not including the message termination character.

gm_delim is the message termination character, such as 03.

gm_lncnt contains a count of the number of lines in the message. The calling program may use this variable, but

should not change its value.

gm_nchar contains the number of characters in the buffer after a read has been completed. This variable should not be used or changed by the calling program.

gm_lptr contains the starting addresses of each line in the message.

gm_fptr is the address of a requested field in some specified line of the message. This variable is used primarily by the getfld() routine.

gm_bufp is the address of the message in gm_buf.

gm_bufe is a pointer to the next message in gm_buf. This variable should not be used or changed by the calling program.

gm_buf is the data buffer and is usually not written into by the calling program.

The argument func should contain one of the following values:

GMF_INIT if the structure variables gm_len, gm_bufp, gm_bufe, gm_lncnt, and gm_nchar, are to be initialized for a new file,

GMF_NBMSG if blank messages in the input file are not to be returned to the calling program, and

GMF_BMSG if all messages, including blank messages, are to be returned to the calling program.

The calling program should perform the following sequence of instructions to initialize appropriate structure variables before calling this subroutine to extract messages:

```
<structure>.gm_fd= <file descriptor of input file>;
<structure>.gm_delim= <message termination character>;
gtmsg(GMF_INIT, &<structure>);
```

Once the above initialization has been performed, the calling routine can locate a message by performing one of the following instructions:

```
gtmsg(GMF_NBMSG, &<structure>);
gtmsg(GMF_BMSG, &<structure>);
```

FILES

/usr/include/gtmhdr.h which contains the definitions for **GMBUF**, **GM_BUFSIZ**, **GM_MAX_LNS**, **GMF_INIT**, **GMF_NBMSG**, **GMF_BMSG**, **GMR_IERR**, **GMR_BLNS**, **GMR_BMSG**, and **GMR_IOERR**.

LIBRARY

/lib/lib1.a

SEE ALSO

getfld(3L), cpyfld(3L)

DIAGNOSTICS

The error codes returned by this subroutine are:

GMR_IERR	Internal error.
GMR_BLNS	Number of lines in message > **GM_MAX_LNS**
GMR_BMSG	Message longer than **GM_BUFSIZ**.
GMR_IOERR	I/O error.

BUGS

NAME

gtofcnm - Get an office name.

SYNOPSIS

```
gtofcnm (ofcname)
     char ofcname[]
```

DESCRIPTION

This routine returns in ofcname the name of an office on the system each time it is called, until there are no more names left. The caller supplied ofcname array should be large enough to contain the largest ofcname (see aparam.h) followed by a null. When ofcname has been filled with a null terminated office name a 1 is returned. When there are no more office names a zero is returned and the subroutine is reset to give the list from the beginning if it is called again. When an error is found a negative number is returned and the subroutine is reset. This routine keeps OFCID_NAME file (see ofcid.h) open until no more office names are found or until it gets an error while reading it.

FILES

/usr/include/aparam.h /usr/include/ofcid.h

LIBRARY

/lib/lib1.a

DIAGNOSTICS

The values returned by this subroutine are:

```
 1 Office name was found and copied into ofcname.
 0 No more office names exits.
-1 Could not open the OFCID_NAME file.
-2 Could not read the OFCID_NAME file.
```