

# **term**

---

A terminal program for Amiga computers

ARexx-interface and ARexx-commands explained  
24 September 1995

**by Olaf Barthel**

---

# 1 Changes

Previous ‘term’ releases would use a different ARexx host interface implementation. In order to conform to Commodore-endorsed user interface style guidelines it was redesigned from scratch for version 3.0. The design and implementation of the ARexx host interface was suggested by the *Amiga User Interface Style Guide* and strongly influenced by Martin Taillefer’s *TurboText* ARexx host interface.

Not a simple command has ‘survived’ the revision, the new implementation is no longer compatible with its predecessors, so existing ARexx scripts will have to be adapted or even entirely rewritten.

‘term’ no longer distinguishes explicitly between asynchronous and synchronous commands (i.e. commands which force the main program to wait and commands which need not bother the main program as the ARexx handler process is able to execute them). As of this writing it is safe to assume that almost any command will be processed by the main program, exceptions are noted.

## 2 term and ARexx

This document describes the ARexx(tm)<sup>1</sup> commands supported by 'term'. This is not intended to be an introduction to the language itself. Rexx was developed by Mike F. Cowlshaw on an IBM/SP system and ported to the Amiga by William S. Hawes.

ARexx (or Amiga Rexx) is a commercial product which is included with the AmigaDOS 2.0 Enhancer Package. If you need a good introduction and description of the language, try to get a hold of the book *The REXX Language A Practical Approach to Programming* by M.F. Cowlshaw, available from Prentice-Hall International, Inc.

The section entitled Section 2.1 [Command execution], page 3 gives a brief introduction how to write and run ARexx commands. For more information refer to the Release 2 Users Manual *Using the System Software*.

By default 'term' opens an ARexx host by the name of TERM (accessible via address term). If more than a single 'term' process is running on your machine, the name of the host will be adapted to the number of the program (i.e. the first program will use TERM, the second one will use TERM.1, the third one TERM.2, etc.). The default name can be overridden by invoking the program with certain parameters (see main program documentation). The name of the host is displayed in the status window (see main program documentation).

### 2.1 Command execution

In order to invoke any command supported by 'term' one usually has to address the host explicitly:

```
/* Address the 'term' host. */  
  
ADDRESS term  
  
/* Invoke the 'beepscreen' command. */  
  
BEEPSCREEN
```

However, if an ARexx script is invoked directly by the 'term' main program, the script will by default address the main program it was invoked by.

---

<sup>1</sup> ARexx is a registered trademark of Wishful Thinking Development Corp.

```
/* The following command will fail to send the file 'ram:foobar' as the colon
 * in the path name will cause an error:
 */
```

```
SENDFILE ram:foobar
```

```
/* Here is how to do it correctly: */
```

```
SENDFILE 'ram:foobar'
```

```
/* The following command will fail to send the file 'foo bar' as the
 * file name is treated as two single files:
 */
```

```
SENDFILE foo bar
```

```
/* The next line will still fail to send the file 'foo bar'
 * as the ARexx parser will split the argument into two
 * parameters.
 */
```

```
SENDFILE 'foo bar'
```

```
/* Here is how to do it correctly: */
```

```
SENDFILE '"foo bar"'
```

```
/* The following command will not transmit the string 'Hello sailor'
 * across the serial line as the single words will be capitalized,
 * they will be transmitted as 'HELLO SAILOR':
 */
```

```
SEND Hello sailor
```

```
/* Here is how to do it correctly: */
```

```
SEND 'Hello sailor'
```

### 3 Stopping a command

Programs and commands sometimes fail to do what the user is expecting them to do which makes it necessary to bring program/command execution to a stop. A common ARexx script to call no external functions or host commands one can be halted in the following ways:

1. Executing the `HI` command (located in the `'SYS:rexxc'` drawer) from Shell. This command will attempt stop *all* currently running ARexx scripts.
2. If the ARexx script to be executed runs in an environment to sport an output window, activate the window and press the `Control + C` keys. A break signal will be sent to the ARexx script, causing it to stop as soon as possible.

With host environments such as `'term'` it may not always be possible to abort a command using the simple measures described above. As for `'term'` any command to wait (such as the Section 4.33 [`READ`], page 33, Section 4.13 [`DELAY`], page 19 or Section 4.59 [`WAIT`], page 51 commands) can be aborted by sending `'term'` itself a break signal in the following fashion:

1. If the `'term'` program is still attached to a Shell output window, activate the window and press the `Control + D` keys.
2. If the `'term'` program was invoked from a Shell but is no longer attached to it, enter `status` command from Shell, remember the number printed, then enter `break <number>` with `<number>` being the number returned by the `'status'` command.
3. Press the hotkey combination configured in the program hotkey settings (see main program documentation). The default is `Right Shift + Left Shift + Escape`. This will cause a break signal to be sent to the `'term'` program.

## 4 Commands

The commands supported by ‘term’ are listed in a table of the following form:

Format :

The command name with its possible calling parameters. In this table parameters are enclosed in brackets and braces, separated by commas and vertical bars; *do not type these special characters along with the parameters!*:

< > (Angle brackets)

Angle brackets enclose parameters whose contents **must not** be omitted in order to make the command work properly.

[ ] (Square brackets)

Square brackets enclose optional parameters.

{ } (Curly braces)

Curly braces enclose items which can be repeated a number of times, such as file name lists.

| (Vertical bar)

Vertical bars separate alternative, mutually exclusive options.

, (Comma)

Commas separate multiple applicable options.

Template:

The command template, similar to the command templates employed by AmigaDOS Shell commands. Possible templates are:

<Parameter>/A

The parameter must **always** be included in order to get accepted.

<Option>/K

The option's **keyword** must be given.

<Option>/S

This option works as a **switch**. If this option keyword is included the switch is on, else it is off.

<Option>/N

A **numeric** parameter is expected.

<Option>/M

**Multiple** parameters are accepted.

<Text>/F

The text must be the **final** parameter on the command line.

## 4.2 The ADDITEM command

### Format :

ADDITEM [To] <Upload | Download | Dial | Wait> [Before | After] [Command <Command for trap list>] [Response <Response text>] [Phone <Entry number, name or wildcard pattern>] [Name <Name>]

### Template:

TO/A,BEFORE/S,AFTER/S,RESPONSE/K,COMMAND/K,PHONE/K/F,NAME/K/F

### Purpose:

Inserts an item (a name, a phone number, text, etc.) before or after the currently selected list item.

### Specifications:

‘term’ maintains a number of lists, these are:

#### Upload list

The list of files to be uploaded.

#### Download list

The list of files the program has downloaded.

#### Dial list

The list of phone numbers or phone book entries to be dialed.

#### Wait list

The list of texts the Section 4.59 [WAIT], page 51 command is to wait for.

New items can be added to the list with the ADDITEM command. The upload list expects the names of files the Section 4.52 [SENDFILE], page 47 command is to transfer. It makes little sense to add files names to the download list as the ‘term’ main program maintains it and adds the names of files received to it, but it is still possible. The wait list expects text lines the Section 4.59 [WAIT], page 51 command will look for in the terminal input stream. A wait list entry added using the RESPONSE keyword will if found in the input data stream cause the response text to be immediately sent to the remote. *Note: a wait list entry to make use of the RESPONSE keyword will be handled by the Section 4.59 [WAIT], page 51 command, the ARexx script will not notice if this list entry was found or not.*

The dial list accepts a number of different parameters:

#### Phonebook entry numbers

These are passed using the Phone parameter which should be a numeric value as it is used as an index to pick the corresponding entry from the phone book.

#### Phonebook entry names

These are also passed using the Phone parameter which can be a proper name or a wildcard pattern.

**Purpose:**

Sets the serial line transfer speed

**Specifications:**

Sets the serial line transfers speed to some defined value. The rate parameter passed in will be matched against all valid baud rates supported by 'term', the closest value will be used.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Change the serial transfer speed to 2400 bps. */  
  
BAUD 2400
```

## 4.4 The BEEPSCREEN command

**Format:**

BEEPSCREEN

**Template:**

,

**Purpose:**

'Beeps' the terminal screen.

**Specifications:**

Invokes a bell signal, as configured in the program settings.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Invoke a bell signal. */  
  
BEEPSCREEN
```

## 4.5 The CALLMENU command

**Warning:**

In case user was to select a file and aborted the selection.

**Example:**

```
/* Open a named capture file. */

CAPTURE TO file NAME 'ram:capture file'

/* Close the capture file, ask the user for a file name. */

CLOSE FILE
CAPTURE TO file

/* Capture to the printer. */

CAPTURE TO printer
```

## 4.7 The CLEAR command

**Format:**

CLEAR [From] <Upload | Download | Dial | Wait | Buffer> [Force]

**Template:**

FROM/A,FORCE/S

**Purpose:**

Clears the contents of a global list or the text buffer.

**Specifications:**

This command serves to clear the contents of the lists to be maintained using the Section 4.2 [ADDITEM], page 11, Section 4.36 [REMITTEM], page 36, Section 4.49 [SELECTITEM], page 45, etc. commands and to purge the contents of the text buffer. In the latter case the program will prompt for confirmation in case the buffer still holds any lines. This confirmation can be suppressed by using the `Force` parameter.

**Result:**

-

**Warning:**

In case the user did not confirm to clear the buffer.

**Example:**

```
/* Clear the wait list. */

CLEAR FROM wait

/* Clear the buffer, ask for a confirmation. */
```

Result:

-

Warning:

-

Example:

```
/* Terminate both file and printer capture. */  
  
CLOSE ALL
```

## 4.10 The CLOSEDEVICE command

Format:

CLOSEDEVICE

Template:

,

Purpose:

Release the current serial device driver

Specifications:

Frees the serial device driver for use with other applications. The driver can be reopened (or a different device driver can be selected) using the Section 4.24 [OPENDEVICE], page 27 command.

Result:

-

Warning:

-

Example:

```
/* Release the serial device driver, all serial I/O  
 * will be halted.  
 */  
  
CLOSEDEVICE
```

## 4.11 The CLOSEREQUESTER command

Format:

CLOSEREQUESTER

```

/* Iconify the program. */

DEACTIVATE

```

### 4.13 The DELAY command

**Format:**

```

DELAY [MIC | MICROSECONDS <Number>] [[SEC | SECONDS] <Number>] [MIN | MINUTES
<Number>] [QUIET]

```

**Template:**

```

MIC=MICROSECONDS/K/N,SEC=SECONDS/N,MIN=MINUTES/K/N,QUIET/S

```

**Purpose:**

Delays program execution for a couple of microseconds, seconds and minutes.

**Specifications:**

Will cause both the program to make the call and the application to wait for a certain period of time. Unless the QUIET option is in effect will process and display data received from the serial line.

**Result:**

-

**Warning:**

If command was aborted before the timeout had elapsed.

**Example:**

```

/* Wait for five seconds. */

DELAY 5

/* Wait for one second and seven microseconds. */

DELAY MIC 7 SEC 5

```

### 4.14 The DIAL command

**Format:**

```

DIAL [WAIT | SYNC] [[Num] <Phone number>]

```

**Template:**

```

WAIT=SYNC/S,NUM/F

```

DUPLEX [Full | Half | Echo]

Template:

FULL/S,HALF=ECHO/S

Purpose:

Sets the serial line duplex mode.

Specifications:

Sets the serial line duplex mode, this can be either full duplex or half duplex (local echo).

Result:

-

Warning:

-

Example:

```
/* Enable local terminal echo. */
DUPLEX ECHO
```

## 4.16 The EXECTOOL command

Format:

EXECTOOL [Console] [Async] [Port] [Command] <File name>

Template:

CONSOLE/S,ASYNC/S,PORT/S,COMMAND/A/F

Purpose:

Executes a program.

Specifications:

Will load and execute an AmigaDOS program. The `Console` parameter will cause an output file or window to be opened, the `Async` parameter will cause the command to return as soon as the execution process has been launched. The `Port` parameter will cause the current ARexx host port name to be passed to the tool on the command line.

Result:

-

Warning:

-

Example:

```
/* Launch the 'Dir' command. */
EXECTOOL CONSOLE COMMAND 'dir c:'
```

## Specifications:

Obtains information on an object, if possible will store it in the `result` variable. If a `stem` or simple variable name is given using the `Stem` or `Var` parameters will store the information in this variable. If no `Field` parameter is given, will store the entire object contents which requires that the `Stem` parameter is given. For a list of valid attributes see the section entitled Section 5.23 [Attributes], page 74.

## Result:

Returns information either in `result` variable or in the supplied `Stem` or `Var` variable.

## Warning:

-

## Example:

```
/* Enable command results. */
OPTIONS RESULTS

/* Query the name of the ARexx host we are communicating with. */
GETATTR OBJECT term FIELD arexx

/* Output the result. */
SAY result

/* Same as above, but using a different syntax. */
GETATTR term.arexx
SAY result

/* Store the entire contents of the phone book in a
 * stem variable.
 */
GETATTR phonebook STEM book

/* Output the phonebook contents. */
SAY 'phone book contains' book.count 'entries'

DO i = 0 TO book.count - 1
  SAY 'entry #' i

  SAY 'name      :' book.i.name
  SAY 'number    :' book.i.number
  SAY 'comment   :' book.i.commenttext
  SAY 'username:' book.i.username
END i
```

**Purpose:**

Cause 'term' to go into online state.

**Specifications:**

After this command is processed 'term' will immediately go into online state. If the carrier signal is to be checked and no signal is present 'term' will drop into offline state right away.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Go into online state. */  
  
GOONLINE
```

## 4.21 The HANGUP command

**Format:**

HANGUP

**Template:**

,

**Purpose:**

Hang up the serial line.

**Specifications:**

Hangs up the serial line, executes logoff and cleanup operations.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Hang up the line, whether the program is online or not. */  
  
HANGUP
```

## 4.22 The HELP command

**Specifications:**

This command reads the contents of a disk file and stores the information either in the configuration, the phone book or the text buffer. If text is read into the text buffer it will be appended to the existing text. If no file name is given will prompt the user to select a file.

**Result:**

-

**Warning:**

If user was requested to select a file and cancelled the selection.

**Example:**

```
/* Load the configuration from a file. */
OPEN NAME 'ram:term.prefs' TO configuration
/* Add text to the text buffer. */
OPEN TO buffer
```

## 4.24 The OPENDEVICE command

**Format:**

OPENDEVICE [Name <Device name>] [Unit <Number>]

**Template:**

NAME/K,UNIT/K/N

**Purpose:**

(Re-)Opens the serial device driver.

**Specifications:**

(Re-)Opens the previously released (see Section 4.10 [CLOSEDEVICE], page 17 command) device driver or a different device driver/unit if the corresponding Device and Unit parameters are given.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Release the serial device driver. */
CLOSEDEVICE
```

Result:

-

Warning:

-

Example:

```
/* Set the parity mode. */
PARITY NONE
```

## 4.27 The PASTECLIP command

Format:

PASTECLIP [Unit <Number>]

Template:

UNIT/K/N

Purpose:

Feed the contents of the clipboard into the input stream.

Specifications:

Feeds the contents of the clipboard into the input stream. Obtains the data either from the given clipboard unit or from the default unit configured in the program settings.

Result:

-

Warning:

If clipboard does not contain any text.

Example:

```
/* Paste the contents of clipboard #2. */
PASTECLIP UNIT 2
```

## 4.28 The PRINT command

Format:

PRINT [From] <Screentext | Clipboard | Buffer | Dial | Wait | Upload | Download> [TO <File name>] [Serial | Modem | Screen | Terminal | User | Comment | Size | Date | Attr]

Template:

FROM/A,TO/K,SERIAL/S,MODEM/S,SCREEN/S,TERMINAL/S,USER/S,COMMENT/S,SIZE/S,DATE/S,

**Purpose:**

Turns serial I/O processing on or off.

**Specifications:**

Usually, the ‘term’ main program processes incoming data from the serial line, i.e. text is displayed in the terminal window or data transfers are started. This can interfere with custom I/O processing, such as done by an ARexx program which wants to receive and process all incoming serial data, without getting interrupted by the main program. For an application example see Section 4.59 [WAIT], page 51.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Turn off I/O processing. */  
  
PROCESSIO OFF
```

## 4.30 The PROTOCOL command

**Format:**

PROTOCOL [None | RTSCTS | RTSCTSDTR]

**Template:**

NONE/S,RTSCTS/S,RTSCTSDTR/S

**Purpose:**

Sets the serial line handshaking protocol.

**Specifications:**

Sets the serial line handshaking protocol. See the main program documentation for details.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Set the handshaking protocol. */  
  
PROTOCOL NONE
```

Warning:

If user did not confirm termination.

Example:

```
/* Try to terminate the program, ask for confirmation. */
QUIT

/* If no confirmation was given terminate by force. */
IF rc ~= 0 THEN QUIT FORCE
```

### 4.33 The READ command

Format:

READ [Num <Number of bytes>] [CR] [Noecho] [Verbatim] [[Prompt] <Prompt text>]

Template:

NUM/K/N,CR/S,NOECHO/S,VERBATIM/S,PROMPT/K/F

Purpose:

Reads a number of bytes or a string from the serial line.

Specifications:

If Num parameter is given will read a number of bytes from the serial line (*note: only a maximum of 65,536 bytes can be read*). The command will return when the read request has been satisfied, the timeout (settable using the Section 4.57 [TIMEOUT], page 50 command) has elapsed or the command was aborted.

If the CR parameter is given will handle simple line editing functions (Backspace, Control-X) and return a string as soon as the Carriage return key is pressed, the timeout (settable using the Section 4.57 [TIMEOUT], page 50 command) has elapsed or the command is aborted.

The Noecho parameter will cause ‘term’ not to echo typed characters back to the remote. *Note that in order to see any input on the local side the remote is to echo the characters typed back.*

If present, the Prompt text will be sent across the serial line, much the same as if it had been sent using the Section 4.50 [SEND], page 46 command.

This command pays attention to the current character translation table for incoming characters. If the characters are to be read without any changes made one has to use the Verbatim parameter.

Result:

The string read.

**Purpose:**

Receive one or more files using the XPR protocol.

**Specifications:**

Receives one or more files using the currently configured XPR protocol. The Mode parameter determines the file transfer mode (either plain ASCII, Text mode or binary file mode), if omitted the file(s) will be received in binary mode. Some file transfer protocols do not require any file names to be given as they have their own means to determine the names of the files to be received. However, a file name parameter can be given. If omitted the file transfer protocol will prompt the user for a file name if necessary.

The names of all files received are placed on the download list for processing. The list will be cleared before the transfer is started.

**Result:**

-

**Warning:**

If user cancelled file selection.

**Example:**

```
/* Start to receive a file in text mode. */
RECEIVEFILE MODE text
```

## 4.35 The REDIAL command

**Format:**

REDIAL

**Template:**

,

**Purpose:**

Redials the numbers remaining in the currently configured dialing list.

**Specifications:**

Redials the numbers which still remain in the dialing list built either by the phone book or by the Section 4.14 [DIAL], page 19 command. *Note that this command will return as soon as the dialing process is initiated.*

**Result:**

-

**Warning:**

If dialing list is empty.

```
REQUESTFILE [Title <Title text>] [Path <Path name>] [File <File name>] [Pattern <Wildcard
pattern>] [Multi] [Stem | Name <Variable name>]
```

Template:

```
TITLE/K,PATH/K,FILE/K,PATTERN/K,MULTI/S,STEM=NAME/K
```

Purpose:

Requests one or more file names from the user.

Specifications:

Requests one or more file names from the user. Will present a file requester with given title text and preset path, file name and pattern values. If only a single file name is to be requested will place the result in the `result` variable. The `Multi` parameter allows multiple files to be selected, the number of files selected and the file names will be placed in the variable specified using the `Stem` parameter.

Result:

The name of the file selected will be placed in the `result` variable. If multiple file were selected, will place the following information in the supplied stem variable:

```
<Variable name>.COUNT
```

The number of files selected.

```
<Variable name>.0 through <Variable name>.n-1
```

The file names selected.

Warning:

If user cancelled selection.

Example:

```
/* Enable command results. */

OPTIONS RESULTS

/* Request a single file name from the user. */

REQUESTFILE TITLE '"select a file"'

/* Output the result. */

IF rc ~= 0 THEN
    SAY 'no file was selected'
ELSE
    SAY result

/* Request several files. */

REQUESTFILE TITLE 'select several files' MULTI STEM names

/* Output the result. */
```

**Purpose:**

Requests a numeric value from the user

**Specifications:**

Requests a numeric value from the user, will display the provided prompt text or a default text and present the provided default number, so user can simply hit return to accept the defaults.

**Result:**

The number the has entered.

**Warning:**

If user cancelled requester.

**Example:**

```
/* Enable command results. */

OPTIONS RESULTS

/* Requester a single number. */

REQUESTNUMBER DEFAULT 42 PROMPT 'Enter the answer'

/* Output the result. */

IF rc ~= 0 THEN
    SAY 'no number was entered'
ELSE
    SAY result
```

## 4.40 The REQUESTRESPONSE command

**Format:**

REQUESTRESPONSE [Title <Title text>] [Options <Options string>] [Prompt] <Prompt text>

**Template:**

TITLE/K,OPTIONS/K,PROMPT/A/F

**Purpose:**

Request a response from user.

**Specifications:**

Requests a response from the user, uses provided title and prompt text and a number of options. If no options are specified will use Yes | No as the defaults.

**Result:**

For Options passed as Yes | Perhaps | No will return 1 for Yes, 2 for Perhaps and return a warning for No.

```

/* Enable command results. */

OPTIONS RESULTS

/* Request a secret string. */

REQUESTSTRING SECRET DEFAULT '"hello sailor!"' ...
...PROMPT 'Enter secret message'

/* Output the result. */

IF rc ~= 0 THEN
    SAY 'no text was entered'
ELSE
    SAY result

```

## 4.42 The RESETSCREEN command

Format:

RESETSCREEN

Template:

,

Purpose:

Resets the terminal screen to defaults.

Specifications:

Resets the terminal screen to defaults, this includes clearing the screen, moving the cursor to the home position and resetting text, text rendering styles and colours.

Result:

-

Warning:

-

Example:

```

/* Reset the terminal screen. */

RESETSCREEN

```

## 4.43 The RESETSTYLES command

Format:

## 4.45 The RESETTIMER command

Format:

RESETTIMER

Template:

,

Purpose:

Reset the online timer.

Specifications:

The online timer is reset to 00:00:00, regardless whether ‘term’ is currently online or not.

Result:

-

Warning:

-

Example:

```
/* Reset the online timer. */
```

```
RESETTIMER
```

## 4.46 The RX command

Format:

RX [Console] [Async] [Command] <Command name>

Template:

CONSOLE/S,ASYNC/S,COMMAND/A/F

Purpose:

Invokes an ARexx macro file.

Specifications:

Invokes an ARexx macro file, if `Console` argument specified opens a console output window, else uses ‘NIL:’, if `Async` argument specified executes the macro asynchronously.

Result:

-

Warning:

-

Example:

**Specifications:**

Saves data to a disk file, will prompt for a filename to save to if none is provided. Will save either parts of the program configuration or the phone book contents (`Phonebook` parameter), the contents of the terminal screen as plain ASCII text (`Screentext` parameter) or the contents of the terminal screen as an IFF-ILBM-file (`Screenimage` parameter).

**Result:**

-

**Warning:**

If user cancels save operation.

**Example:**

```
/* Save the program configuration to a file. */
SAVEAS NAME 'ram:term.prefs' FROM configuration
```

## 4.49 The SELECTITEM command

**Format:**

`SELECTITEM [Name <Name>] [From] <Upload | Download | Dial | Wait> [Next | Prev | Previous | Top | Bottom]`

**Template:**

`NAME/K,FROM/A,NEXT/S,PREV=PREVIOUS/S,TOP/S,BOTTOM/S`

**Purpose:**

Select an item from a list.

**Specifications:**

Selects an item from a list, returns the item name in the `result` variable. The `Top` parameter will select the first list item, `Bottom` the last item. The `Previous` parameter will select the previous list item, `Next` the next successive item. Instead of using a positioning parameter, it is also possible to use a wildcard pattern or name with the `Name` parameter. The first list item to match the name will be selected.

*Note: cannot be used with the dial list.*

**Result:**

Returns the list item in the `result` variable.

**Warning:**

If end of list reached.

**Example:**

```
/* Enable command results. */
OPTIONS RESULTS
```

## 4.51 The SENDBREAK command

Format:

SENDBREAK

Template:

,

Purpose:

Send a break signal across the serial line.

Specifications:

Send a break signal across the serial line.

Result:

-

Warning:

-

Example:

```
/* Send a break signal. */
```

```
SENDBREAK
```

## 4.52 The SENDFILE command

Format:

SENDFILE [Mode <ASCII | Text | Binary>] [Names] { File names }

Template:

MODE/K,NAMES/M

Purpose:

Transfers files using the currently selected file transfer protocol.

Specifications:

Transfers one or more files using the currently configured XPR protocol. The Mode parameter determines the file transfer mode (either plain ASCII, Text mode or binary file mode), if omitted the file(s) will be sent in binary mode. Some file transfer protocols do not require any file names to be given as they have their own means to determine the names of the files to be sent. However, a file name parameter can be given. If omitted the file transfer protocol will prompt the user for a file name if necessary. Several file names can be given if necessary, they will be transferred along with the file names stored in the upload list. The file transfer process will remove any files successfully transferred from the upload list, leaving only those behind which were not to be transferred correctly.

Example:

```
/* Set the transfer speed. */  
SETATTR serialprefs baudrate 2400
```

## 4.54 The SPEAK command

Format:

SPEAK [Text] <Text>

Template:

TEXT/A/F

Purpose:

Speaks the provided text using the Amiga speech synthesizer.

Specifications:

Speaks the provided text using the Amiga speech synthesizer, requires that speech support is enabled.

Result:

-

Warning:

-

Example:

```
/* Say something sensible. */  
SPEAK 'something sensible'
```

## 4.55 The STOPBITS command

Format:

STOPBITS [0 | 1]

Template:

0/S,1/S

Purpose:

Sets the serial line stop bits.

Specifications:

Sets the serial line stop bits.

**Purpose:**

Sets the serial read timeout.

**Specifications:**

Sets the timeout the Section 4.59 [WAIT], page 51 and Section 4.33 [READ], page 33 commands will wait until they exit.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Set the read timeout. */  
  
TIMEOUT SEC 5
```

## 4.58 The TRAP command

**Format:**

TRAP <On|Off>

**Template:**

ON/S,OFF/S

**Purpose:**

Turns the trap list processing on or off.

**Specifications:**

This command tells the main program whether it should process entries of the trap list when filtering input or not.

**Result:**

-

**Warning:**

-

**Example:**

```
/* Ignore the trap list. */  
  
TRAP OFF
```

## 4.59 The WAIT command

```

/* Output the result. */

IF rc ~= 0 THEN
    SAY 'no text was received'
ELSE
    SAY result

```

## 4.60 The WINDOW command

Format:

```

WINDOW [Names] { <Buffer | Review | Packet | Fastmacros | Status | Main | UploadQueue> }
[Open | Close] [Activate] [Min | Max] [Front | Back] [Top | Bottom | Up | Down]

```

Template:

```

NAMES/A/M,OPEN/S,CLOSE/S,ACTIVATE/S,MIN/S,MAX/S,FRONT/S,BACK/S, TOP/S,BOTTOM/S,UP/

```

Purpose:

Manipulates the aspects of a window.

Specifications:

Manipulates the aspects of a window. Not all windows will support all available commands.

The windows supported are:

Buffer

The text buffer window and screen. Supports the Open, Close, Activate and Front commands.

Review

The review window. Supports the Open, Close, Activate, Min, Max, Front, Back, Top, Bottom, Up, and Down commands.

Packet

The packet window. Supports the Open, Close, Activate, Min, Max, Front and Back commands.

Fastmacros

The fast! macro window. Supports the Open, Close, Activate, Min, Max, Front and Back commands.

Status

The status window. Supports the Open, Close, Activate, Front and Back commands.

Main

## 5 Attributes

Several of the application's internal variables are can be accessed and modified using the Section 4.18 [GETATTR], page 22 and Section 4.53 [SETATTR], page 48 commands. Information is available on the objects and their associated fields explained below. Each line consists of the object and field name and the type of the available data:

Numeric data

<Object>.<Field>

Numeric

The information is a numeric value.

Text data

<Object>.<Field>

Text

The information is a text string.

Boolean data

<Object>.<Field>

Boolean

The information is a boolean value and can be ON or OFF.

Mapped codes

<Object>.<Field>

<Value 1> ... <Value n>

The information can be one of the given values.

### 5.1 The TERM object (read-only)

TERM.VERSION

Text

The 'term' program revision.

TERM.SCREEN

Text

The name of the public screen the 'term' main window has been opened on.

TERM.SESSION.ONLINE

Boolean

Whether the program is currently online or not.

TERM.BUFFER.SIZE

Numeric

The size of the text buffer.

## 5.2 The PHONEBOOK object (read-only)

Available fields are:

PHONEBOOK.COUNT

Numeric

The number of entries in the phonebook. The single phonebook entries can be accessed as  
PHONEBOOK.0 . . . through PHONEBOOK.n-1 . . . .

PHONEBOOK.n.NAME

Text

PHONEBOOK.n.NUMBER

Text

PHONEBOOK.n.COMMENTTEXT

Text

PHONEBOOK.n.USERNAME

Text

PHONEBOOK.n.PASSWORDTEXT

Text

## 5.3 The SERIALPREFS object

Available fields are:

SERIALPREFS.BAUDRATE

Numeric

SERIALPREFS.BREAKLENGTH

Numeric

The break signal length in microseconds.

SERIALPREFS.BUFFERSIZE

Numeric

## 5.4 The MODEMPREFS object

Available fields are:

MODEMPREFS.MODEMINITTEXT

Text

MODEMPREFS.MODEMEXITTEXT

Text

MODEMPREFS.MODEMHANGUPTEXT

Text

MODEMPREFS.DIALPREFIXTEXT

Text

MODEMPREFS.DIALSUFFIXTEXT

Text

MODEMPREFS.CHARSENDDELAY

Numeric

MODEMPREFS.DIALMODE

PULSE TONE

MODEMPREFS.NOCARRIERTEXT

Text

MODEMPREFS.NODIALTONETEXT

Text

MODEMPREFS.CONNECTTEXT

Text

MODEMPREFS.VOICETEXT

Text

MODEMPREFS.RINGTEXT

Text

MODEMPREFS.BUSYTEXT

Text

MODEMPREFS.OKTEXT

Text

MODEMPREFS.ERRORTEXT

Text

MODEMPREFS.REDIALDELAY

Numeric

The redial delay in seconds

SCREENPREFS.MAKESCREENPUBLIC  
Boolean

SCREENPREFS.SHANGHAIWINDOWS  
Boolean

SCREENPREFS.BLINKING  
Boolean

SCREENPREFS.FASTERLAYOUT  
Boolean

SCREENPREFS.TITLEBAR  
Boolean

SCREENPREFS.STATUSLINEMODE  
DISABLED STANDARD COMPRESSED

SCREENPREFS.USEPUBSCREEN  
Boolean

SCREENPREFS.PUBSCREENNAME  
Text

SCREENPREFS.USEPENS  
Boolean

SCREENPREFS.WINDOWBORDER  
Boolean

SCREENPREFS.SPLITSTATUS  
Boolean

SCREENPREFS.ONLINEDISPLAY  
TIME COST BOTH

## 5.6 The TERMINALPREFS object

Available fields are:

TERMINALPREFS.BELLMODE  
NONE VISIBLE AUDIBLE BOTH SYSTEM

TERMINALPREFS.ALERTMODE  
NONE BELL SCREEN BOTH

TERMINALPREFS.EMULATIONMODE  
INTERNAL ATOMIC TTY EXTERNAL HEX

TERMINALPREFS.FONTMODE  
STANDARD IBM IBMRAW

EMULATIONPREFS . INSERTMODE

Boolean

EMULATIONPREFS . NEWLINEMODE

Boolean

EMULATIONPREFS . SCROLLMODE

JUMP SMOOTH

EMULATIONPREFS . DESTRUCTIVEBACKSPACE

OFF OVERSTRIKE OVERSTRIKESHIFT

EMULATIONPREFS . SWAPBSDELETE

Boolean

EMULATIONPREFS . PRINTERENABLED

Boolean

EMULATIONPREFS . ANSWERBACKTEXT

Text

EMULATIONPREFS . CLSRESETSCURSOR

Boolean

EMULATIONPREFS . NUMPADLOCKED

Boolean

EMULATIONPREFS . CURSORLOCKED

Boolean

EMULATIONPREFS . FONTLOCKED

Boolean

EMULATIONPREFS . WRAPLOCKED

Boolean

EMULATIONPREFS . STYLELOCKED

Boolean

EMULATIONPREFS . COLOURLOCKED

Boolean

EMULATIONPREFS . MAXPRESCROLL

Numeric

EMULATIONPREFS . MAXJUMP

Numeric

EMULATIONPREFS . USEPENS

Boolean

CAPTUREPREFS.CALLLOGFILENAME  
Text

CAPTUREPREFS.MAXBUFFERSIZE  
Numeric

CAPTUREPREFS.BUFFER  
Boolean

CAPTUREPREFS.BUFFERSAVEPATH  
Text

CAPTUREPREFS.CONNECTAUTOCAPTURE  
Boolean

CAPTUREPREFS.AUTOCAPTUREDATE  
NAME, INCLUDE

CAPTUREPREFS.CAPTUREFILTER  
Boolean

CAPTUREPREFS.CONVERTCHARACTERS  
Boolean

CAPTUREPREFS.CAPTUREPATH  
Text

CAPTUREPREFS.OPENBUFFERWINDOW  
TOP, END

CAPTUREPREFS.REMEMBERBUFFERWINDOW  
Boolean

CAPTUREPREFS.OPENBUFFERSCREEN  
TOP, END

CAPTUREPREFS.REMEMBERBUFFERSCREEN  
Boolean

CAPTUREPREFS.BUFFERSCREENPOSITION  
LEFT, MID, RIGHT

CAPTUREPREFS.BUFFERWIDTH  
Numeric

CAPTUREPREFS.SEARCHHISTORY  
Numeric

## 5.10 The COMMANDPREFS object

Available fields are:

MISCPREFS.REQUESTERTOP

Numeric

MISCPREFS.REQUESTERWIDTH

Numeric

MISCPREFS.REQUESTERHEIGHT

Numeric

## 5.12 The PATHPREFS object

Available fields are:

PATHPREFS.ASCIIUPLOADPATH

Text

PATHPREFS.ASCIIDOWNLOADPATH

Text

PATHPREFS.TEXTUPLOADPATH

Text

PATHPREFS.TEXTDOWNLOADPATH

Text

PATHPREFS.BINARYUPLOADPATH

Text

PATHPREFS.BINARYDOWNLOADPATH

Text

PATHPREFS.CONFIGPATH

Text

PATHPREFS.EDITORNAME

Text

PATHPREFS.HELPPFILENAME

Text

## 5.13 The TRANSFERPREFS object

Available fields are:

TRANSFERPREFS.DEFAULTPROTOCOL

Text

TRANSFERPREFS.SETARCHIVEDBIT  
Boolean

TRANSFERPREFS.COMMENTMODE  
IGNORE FILETYPE SOURCE

TRANSFERPREFS.TRANSFERICONS  
Boolean

TRANSFERPREFS.HIDEUPLOADICON  
Boolean

TRANSFERPREFS.TRANSFERPERFMETER  
Boolean

TRANSFERPREFS.DEFAULTTYPE  
XPR or PROGRAM

TRANSFERPREFS.DEFAULTSENDSIGNATURE  
Text

TRANSFERPREFS.DEFAULTRECEIVESIGNATURE  
Text

TRANSFERPREFS.ASCIIUPLOADTYPE  
XPR, PROGRAM, DEFAULT or INTERNAL

TRANSFERPREFS.ASCIIUPLOADSIGNATURE  
Text

TRANSFERPREFS.ASCIIDOWNLOADSIGNATURE  
Text

TRANSFERPREFS.ASCIIDOWNLOADTYPE  
XPR, PROGRAM, DEFAULT or INTERNAL

TRANSFERPREFS.ASCIIDOWNLOADSIGNATURE  
Text

TRANSFERPREFS.ASCIIDOWNLOADSIGNATURE  
Text

TRANSFERPREFS.TEXTUPLOADTYPE  
XPR, PROGRAM or DEFAULT

TRANSFERPREFS.TEXTUPLOADSIGNATURE  
Text

TRANSFERPREFS.TEXTDOWNLOADSIGNATURE  
Text

TRANSFERPREFS.TEXTDOWNLOADTYPE  
XPR, PROGRAM or DEFAULT

TRANSFERPREFS.TEXTDOWNLOADSIGNATURE  
Text

FUNCTIONKEYPREFS.SHIFT.n

Text

FUNCTIONKEYPREFS.ALT.n

Text

FUNCTIONKEYPREFS.CONTROL.n

Text

## 5.17 The CURSORKEYPREFS object

Available fields are:

CURSORKEYPREFS.UPTTEXT

Text

CURSORKEYPREFS.RIGHTTEXT

Text

CURSORKEYPREFS.DOWNTEXT

Text

CURSORKEYPREFS.LEFTTEXT

Text

CURSORKEYPREFS.SHIFT.UPTTEXT

Text

CURSORKEYPREFS.SHIFT.RIGHTTEXT

Text

CURSORKEYPREFS.SHIFT.DOWNTEXT

Text

CURSORKEYPREFS.SHIFT.LEFTTEXT

Text

CURSORKEYPREFS.ALT.UPTTEXT

Text

CURSORKEYPREFS.ALT.RIGHTTEXT

Text

CURSORKEYPREFS.ALT.DOWNTEXT

Text

CURSORKEYPREFS.ALT.LEFTTEXT

Text

CURSORKEYPREFS.CONTROL.UPTTEXT

Text

## 5.20 The SPEECHPREFS object

Available fields are:

SPEECHPREFS . RATE  
    Numeric  
SPEECHPREFS . PITCH  
    Numeric  
SPEECHPREFS . FREQUENCY  
    Numeric  
SPEECHPREFS . SEXMODE  
    MALE FEMALE  
SPEECHPREFS . VOLUME  
    Numeric  
SPEECHPREFS . SPEECH  
    Boolean

## 5.21 The SOUNDPREFS object

Available fields are:

SOUNDPREFS . BELLNAME  
    Text  
SOUNDPREFS . CONNECTNAME  
    Text  
SOUNDPREFS . DISCONNECTNAME  
    Text  
SOUNDPREFS . GOODTRANSFERNAME  
    Text  
SOUNDPREFS . BADTRANSFERNAME  
    Text  
SOUNDPREFS . RINGNAME  
    Text  
SOUNDPREFS . VOICENAME  
    Text  
SOUNDPREFS . ERRORNAME  
    Text

## 6 Wanted!

As of this writing only a single example ARexx script is included in the ‘term’ distribution (see the ‘Rexx’ drawer). However, it is desirable to include more sample scripts so more users will be able to take advantage of the ARexx interface.

If you wish to share your scripts with the ‘term’ user community, send them (including documentation) to:

Olaf Barthel  
Brabeckstrasse 35  
D-30559 Hannover  
Federal Republic of Germany  
Internet: [olsen@sourcery.han.de](mailto:olsen@sourcery.han.de)

# Index

,  
 , (Comma) ..... 9, 10

[  
 [ ] (Square brackets) ..... 9

{  
 { } (Curly braces) ..... 9

|  
 | (Vertical bar) ..... 9

<  
 <> (Angle brackets) ..... 9  
 <Option>/K ..... 9  
 <Option>/M ..... 9  
 <Option>/N ..... 9  
 <Option>/S ..... 9  
 <Parameter>/A ..... 9  
 <Text>/F ..... 9

## A

ACTIVATE ..... 10  
 ADDITEM ..... 11

## B

BAUD ..... 12  
 BEEPSCREEN ..... 13

## C

CALLMENU ..... 13  
 CAPTURE ..... 14  
 CAPTUREPREFS ..... 64  
 CLEAR ..... 15  
 CLEARSCREEN ..... 16  
 CLIPBOARDPREFS ..... 64  
 CLOSE ..... 16  
 CLOSEDEVICE ..... 17  
 CLOSEREQUESTER ..... 17

COMMANDPREFS ..... 65  
 CONSOLEPREFS ..... 74  
 CURSORKEYPREFS ..... 71

## D

DEACTIVATE ..... 18  
 DELAY ..... 19  
 DIAL ..... 19  
 Dial list ..... 11  
 Download list ..... 11  
 DUPLEX ..... 20

## E

EMULATIONPREFS ..... 62  
 Example: ..... 10  
 EXECTOOL ..... 21

## F

FASTMACROPREFS ..... 72  
 FAULT ..... 22  
 FILEPREFS ..... 74  
 Format: ..... 9  
 FUNCTIONKEYPREFS ..... 70

## G

GETATTR ..... 22  
 GETCLIP ..... 24  
 GOONLINE ..... 24

## H

HANGUP ..... 25  
 HELP ..... 25  
 HOTKEYPREFS ..... 72

## M

MISCPREFS ..... 66  
 MODEMPREFS ..... 59

# Table of Contents

<b>1</b>	<b>Changes</b>	<b>1</b>
<b>2</b>	<b>term and ARexx</b>	<b>3</b>
2.1	Command execution	3
<b>3</b>	<b>Stopping a command</b>	<b>7</b>
<b>4</b>	<b>Commands</b>	<b>9</b>
4.1	The ACTIVATE command	10
4.2	The ADDITEM command	11
4.3	The BAUD command	12
4.4	The BEEPSCREEN command	13
4.5	The CALLMENU command	13
4.6	The CAPTURE command	14
4.7	The CLEAR command	15
4.8	The CLEARSCREEN command	16
4.9	The CLOSE command	16
4.10	The CLOSEDEVICE command	17
4.11	The CLOSEREQUESTER command	17
4.12	The DEACTIVATE command	18
4.13	The DELAY command	19
4.14	The DIAL command	19
4.15	The DUPLEX command	20
4.16	The EXECTOOL command	21
4.17	The FAULT command	22
4.18	The GETATTR command	22
4.19	The GETCLIP command	24
4.20	The GOONLINE command	24
4.21	The HANGUP command	25
4.22	The HELP command	25
4.23	The OPEN command	26
4.24	The OPENDEVICE command	27
4.25	The OPENREQUESTER command	28
4.26	The PARITY command	28
4.27	The PASTECLIP command	29
4.28	The PRINT command	29
4.29	The PROCESSIO command	30

5.10	The COMMANDPREFS object .....	65
5.11	The MISCPREFS object .....	66
5.12	The PATHPREFS object .....	67
5.13	The TRANSFERPREFS object .....	67
5.14	The PROTOCOLPREFS object .....	70
5.15	The TRANSLATIONPREFS object .....	70
5.16	The FUNCTIONKEYPREFS object .....	70
5.17	The CURSORKEYPREFS object .....	71
5.18	The FASTMACROPREFS object .....	72
5.19	The HOTKEYPREFS object .....	72
5.20	The SPEECHPREFS object .....	73
5.21	The SOUNDPREFS object .....	73
5.22	The CONSOLEPREFS object .....	74
5.23	The FILEPREFS object .....	74
<b>6</b>	<b>Wanted! .....</b>	<b>75</b>
	<b>Index .....</b>	<b>77</b>