Console

The Console module implements an I/O multiplexer for multiple processes. It is the default destination for *stdin*, *stdout* and *stderr* for processes which use the standard ‘C’ runtime library initialisation flag. The process handles input from a designated serial device and from IP connections using the telnet protocol. By default, output is identified by prefixing lines with the name of the generating process, and line-by-line input must be prefixed with the destination process name. The process supports a single ‘bound’ process which receives unprefixed input, possibly one character at a time. Most of the complexity of the process comes from handling output strings that span multiple lines, or incomplete lines. In both cases data are buffered internally so that each line from a given process has the correct prefix applied.

Process Information

Prototype Name	Console
Process Priority	below driver-level
Process Name	should correspond with the project definition of the Console process defined by the CLIB_CONSOLE_PROCESS option. The default is "console".

Module Options

CONSOLE_SERIAL_PORT	The ROME URL string of the designated serial interface. The default value is "serial:0" which is the standard polled-mode I/O port. Either this process must exist, or the *telnet* process must exist (or both).

Process Operation

The module has a queue handler and main process. The main process tries to open streams to the serial interface and to the *telnet* process. If both fail the process calls *rome_fatal*. Otherwise requests for data are sent downstream (*FETMBLK* to the serial file and *GETMBLK* to the *telnet* file). The process accepts the messages defined in the *Standard* messageset for dataflows into and out of the process and the *Console* messageset with the following processing:

CLOSE	messages are handled in the main process, where the stream is unbound if it was the current bound stream and any outstanding *F/GETMBLK* messages on the file are returned to the sender.

CONSOLE	messages are handled in the main process. The various flags are used to set options in the data structure associated with the stream, with the exception of the *bind* flag, which is used to the set the global (single) bound process variable.
FETMBLK	messages are added to the file's queue of pending input requests in the queue handler. Replies are handled in the main process where they contain single characters of input from the serial interface. Depending on the mode, lines are either buffered, with backspace interpretation, or passed up immediately as replied to *F/GETMBLK* messages.
FLUSH	messages are handled in the main process, and force any characters held in the file's line buffer to be send to the destination file. This is necessary, for example, to generate prompts without trailing newlines.
GETMBLK	messages are added to the file's queue of pending input requests in the queue handler. Replies to *GETMBLK* messages arrive from the *telnet* process and are handled in the main process. Messages with errors cause the file to be closed and re-opened. If this is the first message from this connection, the console process prompts for the "logon:" string and takes the next line of input as the validation. Otherwise input is buffered until a full line is received and this is passed up as a reply to a *F/GETMBLK* message.
NEWMBLK	messages are passed down towards the current output device from the queue handler, after adding sufficient bytes to allow for the process prefix to be added. Replies are handled in the main process, where the process name is added to the start of the buffer before passing the message upwards towards the application.
OPEN	messages are replied to from within the queue handler. It is essential that any process providing console functionality (i.e. being set as the value of *CLIB_CONSOLE_PROCESS*) handles *OPEN* messages in its queue handler. Otherwise, processes on the IP stack, which use stdio, will create a message deadlock. A new data structure is allocated for each open file and a pointer to it is returned as the *dest_context* field in the reply.
OUTMBLK	messages are handled in the same way as *PUTMBLK* messages below. Replies are either passed upwards or freed locally, depending on the origin of the message.
PUTMBLK	messages are sent directly to the *telnet* process if that is the current output stream, Otherwise the message is scanned to see if it is a single line of output terminated by a newline. If so, and there is no pending output it is sent to the serial code. If not, the output is broken into individual lines which are sent one-by-one to the serial process. Downstream replies are passed up towards the application.
RETMBLK	messages that are not for immediate buffers have their buffers linked into a buffer free chain and the messages are replied to from within the queue handler. Downstream replies are passed up towards the application.

Shared Library Macros and Routines

The following routine formats the message in the *CONSOLE* messageset and sends it to the destination process.

console_set_options

void *console_set_options*(
  FILE **fp*,
  uint *echo*,
  uint *prefix*,
  uint *raw*,
  uint *bind*)

The *console_set_options* routine sets the options for the file *fp* which should be open to the console process (or another process capable of interpreting the *ROME_M_CONSOLE* message). Each of the four options can take one of four values: *CONSOLE_ASIS* leaves the option unchanged, *CONSOLE_SET* sets the option, *CONSOLE_UNSET* unsets the option and *CONSOLE_DEFAULT* resets the options to its default value.

The *echo* parameter, if *UNSET*, disables local character echo in the console process. This option is only effective for the bound path. When *SET*, characters are echoed as they are received; the backspace and delete control codes are interpreted locally, and discarded unless the path is in raw mode.

The *prefix* parameter, if *UNSET*, disables the default action of adding the process name to all output lines. This option is available for all paths, whether bound or not.

The *raw* parameter, if *SET*, enables character-by-character input on a bound path (only). In this mode, any *FETMBLK* or *GETMBLK* request will be satisfied by exactly one character. All the control characters are passed along unchanged.

The *bind* parameter, if *SET*, binds the console to this path. In this mode, no process prefix is decoded in the input direction, and any input is automatically passed up on this stream. This option also disables any of the ‘@’ commands decoded by the console. It does not bypass the telnet ‘logon’ mechanism.