

The Serial Port Driver of Real-Time Linux

J. Küpper

Heinrich Heine Universität, Institut für Physikalische Chemie I, Universitätsstraße 1,
D-40225 Düsseldorf, Deutschland

jochen@pc1.uni-duesseldorf.de

Abstract. This documentation describes the `rt_com` serial port driver for RT-Linux. The driver works with **NMT RT-Linux** v1 and v2, as well as **RTAI**. This manual is intended to describe `rt_com` version 0.5.

1. License

This document is free. You can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation either version 2 of the License, or (at your option) any later version. This document is distributed in the hope that it will be useful, but *without any warranty*. Without even the implied warranty of *merchantability* or *fitness for a particular purpose*. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this document. If not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge MA 02139, USA.

2. Copyright(s)

©1997-2000, Jochen Küpper. All rights reserved.

3. Typographic Conventions

The conventions used in this document are described in Table 1. For reasons of clarity, the `\rtlmarg` is not shown as a margin note within the table. Verbatim-like output can be set using the `\begin{rtlcode} ... \end{rtlcode}` environment (Daly et al. 2000).

4. Introduction

This manual describes the `rt_com` kernel module. The module provides a reasonable easy software interface to the standard serial ports of the PCs for NMT RT-Linux v1 and v2 and RTAI.

There are a small number of user functions that provide an interface to the port, as well as several functions internally used to communicate with the hardware.

Table 1.: Typographical Conventions for this Document

Markup	Usage	Effect
<code>\rtlin{<i>blue type-face</i>}</code>	user input	<i>blue type-face</i>
<code>\rtlout{<i>magenta sans-serif</i>}</code>	machine output	<i>magenta sans-serif</i>
<code>\rtlnormal{<i>black times-roman</i>}</code>	normal text (reset)	black times-roman
<code>\rtlmargin{<i>teal italic</i>}</code>	margin notes	<i>teal italic</i>

5. Availability

The primary site of this package is [rt.com homepage](#). It is also distributed with RT-Linux systems from [NMT](#) and [RTAI](#).

6. Installation

The `rt.com` package you obtained should contain the source code (`rt.com.h`, `rt.com.c`, `rt.comP.h`), the makefile (Makefile), some informational files (COPYING, License, README) and this documentation — the documentation master file is `rt.com.tex`, it is also available in Portable Document Format (PDF) `rt.com.pdf`. Moreover there are a few examples to test the module and to show you how to program it in the directory `test/`.

In order to run the module on a NMT-RT-Linux v1 system (Linux kernel 2.0.x) or on RTAI you need to define `RTLINUX_V1` or `RTAI`, respectively, at compile time. For this edit the Makefile and add the define to the `CFLAGS` variable.

To compile the module `cd` to the `rt.com` directory and do `make && make install`.

When you obtained this module with a RT-Linux distribution, see the distribution for installation instructions.

7. Interface functions

7.1. Setting up a serial port

This is to set up the port for use by your module by providing some initialization data. The function is declared as

```
void rt_com_setup( unsigned int com, unsigned baud, unsigned parity,
                  unsigned stopbits, unsigned wordlength )
```

where `com` is the entry number from the `rt_com_table` (see section 9.3.), `baud` is the Baud rate the port shall be operated at, `parity` determines the parity policy to use (possible values are `RT_COM_PARITY_EVEN`, `RT_COM_PARITY_NONE`, `RT_COM_PARITY_ODD` - these are defined in `rt.com.h`), `stopbits` and `wordlength` are self explanatory and take the immediate value these flags shall be set at.

7.2. Writing data to a port

To write data to a port you need to call the function `rt_com_write`, which is declared as

```
void rt_com_write( unsigned int com, char *buf, int cnt )
```

where `com` is the entry number from the `rt_com_table` (see section 9.3.), `buf` is the memory address of the data to write to the port, `cnt` is the number of bytes that shall be written.

7.3. Reading data from a port

To read data from a port you need to call the function `rt_com_read`, which is declared as

```
int rt_com_read( unsigned int com, char *buf, int cnt )
```

where `com` is the entry number from the `rt_com_table` (see section 9.3.), `buf` is the memory address the data read shall be put in, `cnt` is the maximum number of bytes that shall be read. The function returns the number of bytes that really have been read.

8. Internals

8.1. Loading the module into memory

When the module gets loaded it requests the port memory and registers the interrupt service routine (ISR) for each member of the `rt_com_table` (see paragraph 9.3. (`rt_com_table`)). Moreover it initializes all ports.

On success it reports the loading of the module, otherwise it releases all resources, reports the failure and exits without the module being loaded.

8.2. Removing the module

Before the module is removed from memory, the function `cleanup_module` frees all allocated resources.

9. Data Structures

9.1. `rt_buf_struct`

Structure to implement software FIFOs. Used for buffering of the data that needs to be written to the port and data read from hardware that needs to be read by the user. The FIFO size is given by the define `RT_COM_BUF_SIZ`; it has to be a power of two.

9.2. `rt_com_struct`

Defines the hardware parameter of one serial port. The members of this structure are a magic number (not used yet), the base rate of the port (115200 for standard ports), the port number, the interrupt number (IRQ) of the port, the flags set for this port, the ISR (see paragraph 8.1. (`init_module`)) the type and a copy of the IER register. Moreover it contains two FIFOs as defined by the `rt_buf_struct` (see paragraph 9.1. (`rt_buf_struct`)), one for reading from the port and one for writing to it, respectively.

9.3. `rt_com_table`

This array holds a `rt_com_struct` for each serial port to be handled by the module.

10. Bugs

Please report bugs to [Jochen Küpper](#) and the [RT-Linux mailing list](#).

There are no known bugs right now.

11. Document Revision History

07. January 2000, JK: Changed from `sgml` to `rtldoc`.

last changed: January 21, 2000, jochen

Acknowledgments. The `rt_com` package is based on code sent to the Real-Time Linux mailing list by Jens Michaelson in 1997. [Roberto Finazzi](#) contributed various extensions to `rt_com`, esp. hardware control, handshaking. Linux is a registered trade mark of Linus Torvalds.

References

Daly, P. N., Mahoney, T. J., and Küpper, J. 2000, ‘RTLDOC L^AT_EX 2_ε Template and Style File’ in *Real Time Linux Documentation Project*, **1**, P. N. Daly and J. Küpper, eds., Real Time Linux Community Press