rt_com

Jochen Küpper $Date:ff1999/07/31ff1ff:20:4ffff$

1 Device Driver Support

$Id: rt_com.lyx,v 1.1 1999/07/28 11:19:47 jochen Exp $

1.1 rt_com - serial port driver

author: Jochen Küpper <jochen@uni-duesseldorf.de>

last changed: $Date: 1999/07/28 11:19:47 $

This manual is intended to describe `rt_com` version 0.3, but is not fully updated to match that version yet. The interface should be right, though.

1.1.1 Introduction

This chapter of the manual describes the `rt_com` kernel module. That module provides a reasonable easy software interface (driver) to the standard serial ports of the PCs. 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.

1.1.2 Installation

The rt_com package should contain the source code (`rt_com.h`, `rt_com.c`, `rt_comP.h`), makefiles (`Makefile.am`, `Makefile.standalone`), some informational files (`COPYING`, `License`, `README`) and this documentation in various formats. The master file is `rt_com.lyx`, there are also html and Postscript versions of it available.

The module works with RT-Linux v1 and v2. In order to run it on an v1 system (Linux kernel 2.0.x) you need to define `RTLINUX_V1` at compile time.

1.1.3 Interface functions

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 paragraph 1.1.5 (rt_com_table)), 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.

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 paragraph 1.1.5 (rt_com_table)), buf is the memory address of the data to write to the port, cnt is the number of bytes that shall be written.

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 paragraph 1.1.5 (rt_com_table)), 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.

1.1.4 Internals

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 1.1.5 (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 beeing loaded.

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

1.1.5 Data Structures

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.

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 1.1.4 (init_module)) the type and a copy of the IER register. Moreover it contains two FIFOs as defined by the `rt_buf_struc` (see paragraph 1.1.5 (rt_buf_struct)), one for reading from the port and one for writing to it, respectively.

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

1.1.6 Availability

The package is available at the *rt_ com homepage* `<http://www-public.rz.uni-duesseldorf.de/~jochen/computer/software/rt_com/>` and the *RT-Linux homepage* `<http://www.rtlinux.org>`.

1.1.7 Bugs

Please report bugs to *Jochen Küpper* <mailto:jochen@uni-duesseldorf.de> and the *RT-Linux maiing list* <mailto:rtl@rtlinux.cs.nmt.edu>.

There are no known bugs right now.

1.1.8 Acknowledgment

Thanks to Jens Michaelsen for providing his code.