DYNAMIC ENGINEERING
150 DuBois St. Suite C Santa Cruz CA 95060
831-457-8891 **Fax** 831-457-4793
http://www.dyneng.com
sales@dyneng.com
Est. 1988

Software User's Guide (Linux)

PMC-BISERIAL-III BAE9

Eight-Channel UART Interface
PMC Module
Revision B1

DRAFT

PMC-BiSerial-III BAE9
Eight-Channel
PMC Base UART Interface

Dynamic Engineering
150 DuBois St Suite C
Santa Cruz, CA 95060
831-457-8891
831-457-4793 FAX

Revised 3/18/13

Table of Contents

Product Description

The PMC compatible PMC-BiSerial-III has 8 I/0 channels each populated with two RS-485 transceivers. Channels may be operated in half or full-duplex modes. Hardware flow control (RTS/CTS) may be enabled and can be operated in pulse or level modes. Hardware implements a programmable RS-485 “delay_rts_before_send”, that is, a delay after RTS detection, before transmission begins. Further, repeated transmission can be enabled which occurs at a periodic rate, or upon RTS assertion and then at a periodic rate. The BiSerial-III currently supports baud rates of 10Mbps to 2.5 Mbps.

For a detailed description of the hardware including register definitions, see HW User Manual, PMC-BiSerial-III, revision B1.

Software Description

The PMC-BiSerial-III driver utilizes the standard Linux serial layer implemented in the generic serial core. Almost all application interaction with the driver occurs via this layer. The only required custom ioctl is DE_SET_TERM which allows the application to enable/disable RTS input signal termination. Other custom ioctls are provided to read and write channel registers which are available to aid debug, but must be used with caution as undesired interactions with nominal driver operation may be effected.

Installation

1) Create the /dev/ttyDExx devices by either editing your /etc/device_table and adding the following line assuming the default major/minor device numbers do not conflict with other devices already present:
   /det/ttyDE c 640 0 5 204 5 0 1 8 // Assuming 1 I/O card
   This will create the specified devices upon boot, otherwise you may create the devices manually upon every reboot via the MKDEV command.
2) Copy de_BiSerBae9.c and de_BiSerBae9.h to your module build directory. Invoke the system “make”
3) Copy the resulting de_BiSerBae9.ko module to the target platform.
4) Execute insmod de_BiSerBae9.ko

Application Programming model

As previously mentioned, standard serial interfaces/methods are utilized for configuring and retrieving status from the I/O channels. Standard open, read,

write operations are of course supported as well. Ports by default are initialized to full-duplex, 2.5 Mbps baud rate, no parity, one stop bit, flow control disabled.

Channel setup/control is accomplished via the standard method **tcsetattr**: termios.c_cflag applicable fields:

CSTOPB	enable 2 stop bits
PARENB	enable parity generation
CMSPAR	enable “stick” parity (If PAODD set, parity bit always 1, otherwise0)
PARODD	parity is odd

termios.c_iflag applicable fields:

INPCK	Enable parity checking
IGNPAR	Don’t count parity/framing errors
ITERM	Dynamic Engineering extension, enable receiver shunt termination

Baud rate is set via standard method cfsetspeed due to non-standard custom speeds. **tcsetattr** must be invoked after speed is set so the baud rate setting is applied.

RS485 control is accomplished via the standard serial ioctl **TIOCSRS485**: serial_rs485.flag applicable fields:

DE_SER_RTS_ENBL	Enable RTS (overloads SER_RS485_ENABLED)
DE_SER_RTS_IN_POL	Polarity of input RTS is active high, else low (same as standard def., SER_RS485_RTS_ON_SEND, renamed for clarity)
DE_SER_RTS_OUT_POL	Polarity of output RTS is active high, else low (same as standard def. SER_RS485_RTS_AFTER_SEND, renamed for clarity)
DE_SER_RTS_SEL	Select input signal to be utilized for RTS, 4 bit field with valid values 0-9. Dynamic Engineering extension.

The following indices apply to the serial_rs485.padding[] array which is being utilized to set RTS signal monitoring parameters, all parameters are specified in units of .100 usec based upon PLL clock B frequency of 10 Mhz. This is subject to change if a custom/new PLL file is loaded.

DE_SER_MIN_LOW_IDX	Minimum low bit period

DE_SER_MAX_LOW_IDX Maximum low bit period
DE_SER_MIN_HI_IDX Minimum high bit period
DE_SER_MAX_HI_IDX Maximum high bit period

Port statistics can be retrieved by invoking the standard serial ioctl **TIOCGICOUNT**. This includes Dynamic Engineering overloads/extensions for RTS monitoring counts. Upon return from the ioctl, the structure should be cast as de_uart_icount_t, definition is a follows:

```
typedef struct de_uart_icount {
    /* Count of excessive high-level durations  */
    unsigned long    rts_hi_long;
    /* Count of insufficient high-level durations */
    unsigned long    rts_hi_short;
    /* Count of excessive low-level durations    */
    /* Count of excessive low-level durations   */
    unsigned long    rts_low_long;
    unsigned long    rts_low_short;
    /* Remaining fields retain standard definitions */
    unsigned long    rx;
    unsigned long    tx;
    unsigned long    overrun;
    unsigned long    parity;
    unsigned long    brk;            /* N/A */
    unsigned long    brk_overrun;  /* N/A */
} de_uart_icount_t;
```

Sample application

A sample application, de_BiSerApp.c is provided to demonstrate configuration, ioctl invocation, and execution of various modes. See de_BiSerApp.c for details.

1) Compile the sample application for your platform, the output executable for this example is dyn_app. The input file name is de_BiSerApp.c
2) The sample application is designed to be invoked from separate processes such as 2 different rsh sessions.
3) The sample app assumes either a Dynamic Engineering full or half duplex test fixture is attached.

Usage is as follows:

dyn_app r(reader)|w(riter) f(ull duplex)|h(alf duplex) c(optional continuous)
Note: Reader most be invoked first, two processes are utilized to demonstrate SMP “safeness”

Warranty and Repair

Please refer to the warranty page on our website for the current warranty offered and options.

http://www.dyneng.com/warranty.html

Service Policy

Before returning a product for repair, verify as well as possible that the suspected unit is at fault. Then call the Customer Service Department for a RETURN MATERIAL AUTHORIZATION (RMA) number. Carefully package the unit, in the original shipping carton if this is available, and ship prepaid and insured with the RMA number clearly written on the outside of the package. Include a return address and the telephone number of a technical contact. For out-of-warranty repairs, a purchase order for repair charges must accompany the return. Dynamic Engineering will not be responsible for damages due to improper packaging of returned items. For service on Dynamic Engineering Products not purchased directly from Dynamic Engineering contact your reseller. Products returned to Dynamic Engineering for repair by other than the original customer will be treated as out-of-warranty.

Out of Warranty Repairs

Out of warranty repairs will be billed on a material and labor basis. The current minimum repair charge is $150. Customer approval will be obtained before repairing any item if the repair charges will exceed one half of the quantity one list price for that unit. Return transportation and insurance will be billed as part of the repair and is in addition to the minimum charge.

For Service Contact:

Customer Service Department
Dynamic Engineering
150 DuBois St. Suite C
Santa Cruz, CA 95060
831-457-8891
831-457-4793 fax
InterNet Address support@dyneng.com