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-VI-UART

8-Channel UART Interface

PMC-BiSerial-VI UART

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

©2017 by Dynamic Engineering.
Other trademarks and registered trademarks are owned by their respective manufactures.
Revised 08/17/2017

This document contains information of proprietary interest to Dynamic Engineering. It has been supplied in confidence and the recipient, by accepting this material, agrees that the subject matter will not be copied or reproduced, in whole or in part, nor its contents revealed in any manner or to any person except to meet the purpose for which it was delivered.

Dynamic Engineering has made every effort to ensure that this manual is accurate and complete. Still, the company reserves the right to make improvements or changes in the product described in this document at any time and without notice. Furthermore, Dynamic Engineering assumes no liability arising out of the application or use of the device described herein.

The electronic equipment described herein generates, uses, and can radiate radio frequency energy. Operation of this equipment in a residential area is likely to cause radio interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

Dynamic Engineering's products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of Dynamic Engineering.

Connection of incompatible hardware is likely to cause serious damage.

Product Description

The PMC-BiSerial-VI PMC is an eight channel, full duplex UART interface card supporting various modes of operation. All channels are supported with their own DMA engines

For a detailed description of the hardware including register definitions, see HW User Manual, PmcBis6UartHwManA8.

Software Description

The driver supports full duplex operation on all 8 channels.

A default configuration is applied when ports are opened for the first time. These default settings are defined in the driver header file, de_BiSerUart.h. The default I/O port config setting is named de_default_pt_config. The default config parameters can be customized for a particular application, and the driver recompiled. This may eliminate the need for invoking the config ioctl.

Applicable I/O configuration parameters include blocking timeout, baud-rate, mode, parity, flow control, inter-char timer (utilized for packet modes), and various UART options (data size, stop bits, and terminations). Blocking timeout provides a mechanism to timeout on blocking operations.

Default I/O configuration is as follows: Blocking timeout on reads = 5 sec. (if opened as blocking), 115200 baud-rate, packed mode of operation, even parity, flow control enabled (CTS/RTS), auto compute inter-char timer based upon baud-rate, 8 bit data, 1 stop bit, terminate CTS and Rx signals.

The version of this driver is v1.0.2. The driver has been validated on an i7 Ubuntu server running 3.8.0-44 kernel (64 bit) SMP and embedded target P2020 running 3.0.48 kernel (32 bit) SMP.

Modes of operation

The HW and SW support 5 modes of operation on a port by port basis, all modes accept (writes) and return (reads) a packed byte stream. Please note I/O limitations between ports populating different platform types (little endian to/from big endian). If required for specific customer applications, these limitations can be addressed/resolved for an additional fee.

- Upacked

  Prepends or strips 3 fill bytes for each data byte, max frame size = 255 bytes. Size does not have to be a multiple of 4 bytes. I/O between big/little endian platforms not supported.

- Packed

  Max frame size = 1020 bytes, size must be a multiple of 4 bytes

- Packet

  Packed data, max frame size = 1020 bytes, size does not have to be a multiple of 4 bytes, however for non-aligned receive packets least significant bytes are filled with zeros to force alignment. Non-aligned (not a multiple of 4 bytes) I/O between big/little endian platform not supported.

- Alternate Packet

  Prepends/strips control byte for every 3 bytes of data max frame size = 765 bytes. Does not have to be a multiple of 4 bytes, and received packet will contain no fill bytes. This mode is not supported on big endian platforms.

- Test

  Raw mode of operation supporting test.

When operating in either of the packet modes, a read will return the next available packed irrespective of size. Thus, reads should be issued with a size of DE_MAX_FRAME. Please see HW manual for further discussion of advantages/disadvantages of each mode.

PLL Programming

The PLL can be programmed with a custom PLL file generated by the Cypress CyberClocks tool. PLL frequencies of up to 64 Mhz are supported. The PLL must be programmed prior to specifying the PLL as the clock for baud-rate generation (e.g. port_cfg.br_clk_src = 1) during port configuration. PLL clocks A-D are assigned as follows: CLKA - ports 0 & 1, CLKB - ports 2 & 3, CLKC - ports 4 & 5, CLKD – ports 6 & 7. Code for reading jed files generated by Cypress tool can be found in the application de_IoctlApp.c. The application may be executed to load the PLL file, or the code may be ported to a customer application.

Installation

1) Copy de_BiSerUart.c and de_BiSerUart.h to your module build directory. Invoke the system “make.” A makefile for this module has been included in the release tar-ball.
2) Copy the resulting de_BiSerUart.ko module to the target platform/directory.
3) Copy the startup script bnm to the target.
4) Invoke the script (./bnm), it will create the devices required by the driver and performs an insmod of the module. You may invoke this script from the systems rc.local file as well.

Application Programming model

After a port is opened, it may be configured for the desired mode of operation via the DE_CONFIG_PT ioctl. Both blocking and non-blocking modes of operation are supported. This behavior is set via the standard file flags upon open.

Please see de_BiSerUart.h for details of the parameters for this and other supported ioctls.

Sample application

Three sample applications de_IoApp.c, de_IoAppS.c, de_IoctlApp.c are provided to demonstrate configuration, ioctl invocation, and I/O in the supported modes. Various modes of operation and options maybe validated/demonstrated by changing port configuration parameters in the application and recompiling.

de_IoApp.c is a board to board test. It requires two boards to be installed in the platform and connected via a board-to-board test fixture. A minimum of two instances must be invoked, first the reader, then the writer within 5 seconds. The applications run asynchronously to one another. Port 0 is connected to port 8, port 1 to port 9, and so on via test fixture.

1) If utilizing custom PLL file, modify line 102 to reference custom jed file:
   if (xlate (“your_file_name.jed”, pll_cfg.data))
   Compile Ioctl App:
   gcc -Wall -o dyn_ioctl de_IoctlApp.c
   Invoke ioctl app for each board:
   dyn_ioctl 0 1 // Any port on 1st board.
   dyn_ioctl 8 1 // Any port on 2nd board.

2) Compile de_IoApp for your platform.
gcc -DMODE=1 -Wall –o dyn_io de_IoApp.c
See de_BiSerUart.h for mode definitions (de_mode_t)
The app should compile without warnings, it is assumed de_BiSerUart.h is resident in the same directory as the application for this example.

de_IoAppS.c is a single board test. Ports are looped back to themselves externally via single board test fixture. The application first writes to the specified port, and then reads received data. Data integrity is then validated.

1) Execute step 1 from above if necessary.
2) Compile de_IoAppS for your platform.
gcc -DMODE=1 -Wall –o dyn_ioS de_IoAppS.c
See de_BiSerUart.h for mode definitions (de_mode_t)
The app should compile without warnings, it is assumed de_BiSerUart.h is resident in the same directory as the application for this example.

Invocation parameters

I/O application invocation is as follows:

dyn_io - 2 board test

./dyn_io 1 0 baud-rate frame_len num_iterations //(reader, port 0, board 1)
./dyn_io 0 8 baud-rate frame_len num_iterations //(writer, port 8, board 2)

The first parameter specifies reader/writer. The second parameter is port number, third parameter is baud-rate. Frame length is specified in bytes. Data is validated upon reception. Application will execute for num_iterations, or until terminated due to an error or interrupted via <CTRL-C>.

dyn_ioS - single board test

./dyn_ioS 0 baud-rate frame_len num_iterations //(port 0, board 1)

The first parameter specifies port. The second parameter is baud-rate followed by frame length in bytes. Data is validated upon reception. Application will executedfor num_iterations, or until terminated due to an error or interrupted via <CTRL-C>.

Support Contract

Dynamic Drivers are provided AS-IS and sometimes our clients need a little help. Please refer to the support contract page on our website for options about getting help with your driver use and SW development.

http://www.dyneng.com/TechnicalSupportFromDE.pdf

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

Software support contracts are available to update, add features, change for different revisions of OS etc. Please contact Dynamic Engineering for these options.

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