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

Software User's Guide (Linux)

Libip1553

IPACK 1553 module

Libip1553

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

Revised 05/31/2020

Product Description

Libip1553 user library supports the Dynamic Engineering IP-1553 module. This library interfaces with the ipack-core (Open Source ported from 3.5 kernel) via the ipack_gen(eric) driver. Thus, this kernel module serves as a gasket between the user-libraries and the ipack-core. The Dynamic Engineering PciNIP driver is a bus/carrier driver supporting all our released carrier/bridge cards interfacing with the ipack-core. Further, the IP1553 package utilizes the Holt API Library for controlling the 62203 1553 chips resident on the module. Holt API SW has been modified to access chip internals (registers and memory) via the Dynamic Engineering libip1553 library. Holt application demo software has also been modified to utilize libip1553 library as well. For information related to Holt software APIs, please see HI-6130-API Library Software Manual. **Keith, insert link here.**

Software Description

As described in the PciNIP SW manual, the ipack-core and de_PciNIP kernel modules must be built and installed prior to utilization of any other IPACK components including those described within this document. Please see that manual for details WRT building and installing these modules.

Libip1553 API descriptions

```
/*****************************************************************
*  libip_reset_dev
*
*  Reset 1553 chip/device
*
*  Parameters:
*  enbl_int - Enable interrupts for this device.
*
*  Returns:
*  0 upon success, < 0 upon failure
*/
int ip_1553_reset (ipack_handle_t module, unsigned dev_num,
                                                    int enbl_ints);
```

```
/*****************************************************************
*  libip_1553_init
*
*  Initialize library. This function must be invoked prior to
*  utilizing any of the following access routines.  This function
*  returns a list of IP-1553 modules either containing the first
*  module found, or all modules.
*
*  Parameters:
*  find_all     - (0=find first, 1=find all)
*  modules      - pointer to an array of size
*                 (libipack:MAX_IP_MODULES)
*                  if find_all is true.  Otherwise, an array of a
*                  single element is sufficient.
*
*  Returns:
*  Number of modules upon success, < 0 upon failure
*/
int libip_1553_init (int find_all, ipack_handle_t *modules);
```

```
/*****************************************************************
*  libip_1553_exit
*
*  Exit/shutdown library. This function should be invoked upon
*  application termination.
*
*  Parameters:
*  N/A, void
*
*  Returns:
*  void
*/
void libip_1553_exit (void);


/*****************************************************************
*  ip_1553_wrDev
*
*  Write 1553 Chip
*
*  Parameters:
*  handle   -   Handle returned in module list (lib_1553_init)
*               specifying which IP-1553.
*  dev      -   0 or 1
*  region   -   IP_1553_REG or IP_1553_MEM
*  offset   -   register or memory offset
*  count    -   Only valid for mem writes, number of 16 bit
*               words.
*  *vals    -   16 bit value(s) to be written
*
*  Returns:
*  0 upon success, < 0 upon failure
*/
int ip_1553_wrDev (ipack_handle_t handle, unsigned dev,
    ip_1553_region_t region, uint8_t count,
    uint32_t offset,uint16_t* vals);
```

```
/*****************************************************************
*  ip_1553_rdDev
*
*  Read 1553 Chip
*
*  Parameters:
*  handle   -   Handle returned in module list (lib_1553_init) *
*               specifying which IP-1553.
*  dev      -   0 or 1
*  region   -   IP_1553_REG or IP_1553_MEM
*  count      - Only valid for mem reads, number of 16 bit words.
*  offset   -   register or memory offset
*  vals     -   16 bit value read
*
*  Returns:
*  0 upon success, < 0 upon failure
*/
int ip_1553_rdDev (ipack_handle_t handle, unsigned dev,
    ip_1553_region_t region, uint8_t count,
    uint32_t offset, uint16_t *vals);
```

```
/*****************************************************************
*  ip_1553_rmwDev
*
*  Read/modify/write 1553 Chip, only valid for register space
*
*  Parameters:
*  handle   -   Handle returned in module list (lib_1553_init)
*               specifying which IP-1553.
*  dev      -   0 or 1
*  offset   -   register offset
*  mask     -   Mask specifying bits of interest
*  val      -   16 bit value to write, unmasked 16 bit value read
*               is returned.
*
*  Returns:
*  0 upon success, < 0 upon failure
*/
int ip_1553_rmwDev (ipack_handle_t handle, unsigned dev,
                 uint32_t offset, uint16_t mask, uint16_t *val);
```

```
/*****************************************************************
*
*  ip_1553_await_int
*
*  Await 1553 interrupts.
*
*  Parameters:
*  handle       -   Handle returned in module list
*                  (libip_1553_init) specifying which
*                   IP-1553 module.
*  dev          -   Device/chip (0 or 1)
*  int_stat     -   Interrupt status registers
*                   (IP_1553_IES1, IP_1553_IES2)
*  q_ptr        -   Queue pointer (IP_1553_QPTR)
*  timeout      -   Timeout awaiting interrupt in msec.
*
*  Special considerations:
*
*  Returns:
*  0 upon success, < 0 upon failure.
*/
int ip_1553_await_int (ipack_handle_t handle, unsigned dev,
          uint16_t int_stat[2], uint16_t* q_ptr, long timeout);
```

Installation

1) Install ipack and de_PCIeNIP kernel modules, see SW manual for the de_PciNIP.
2) Copy ipack_gen.c, ipack_gen.h (ipack_gen) to your module build directory. Invoke the system “make.” Alternatively a makefile for ipack_gen has been included for out of tree kernel module build. If this build method is utilized, cd to the build directory and invoke the script ./build_all. This script will invoke the Makefile to build ipack_gen.ko, compile/archive libipack,, libip1553, and Holt API code as well as building a test application (ip1553App).
3) Copy the resulting ipack.ko module to the target platform/directory.
4) Copy the startup script bnm to the target.
5) Invoke the script (./bnm), it will perform an insmod of ipack_gen and create the required device. The script may be invoked from the systems rc.local file as well.

Sample application

The application ip1553App.c demonstrates proper usage of library functions/operations for libipack, libip1553 and Holt API code. As previously mentioned, the Dynamic Engineering IP-1553 module is employed for demonstration purposes.

Invocation parameters (ip1553App)

The application can only be executed successfully in conjunction with a Dynamic Engineering Test fixture which connects BusA/B to the other device on the same module if a IP-1553-2 version. Otherwise, you will need two IP-1553-1 modules. **The IP-1553-2 version is recommended as it allows BC to be executing on 1 chip, and MT on the other.** IP-Debug-IO can be used or 1553-Ribbon-Triax adapter.

Application invocation is as follows:

ip1553App invocation:

Two instances of the application must be executed. One will execute BC tests, the other will invoke RT, RT/MT or MT. Assuming an IP-1553-2 version, invoke the app as follows in two separate terminal windows.

./ip1553App mod_num dev num(0|1)

./ip1553App 0 0
./ip1553App 0 1

or if 2 IP-1553-1 modules are installed
./ip1553App 0,0
./ip1553App 1,0

Upon invocation, a menu will be displayed in each window:

```
No external clock detected
Locked to IP and 1553 clock
PCI bus:slot:1:0::Man ID:model:design
id:rev:0x00001e:0x000b:0x0000:0x0004
IPACK bus:slot:0:0::Dips:0x00::speed is 32 MHz
```

```
*********************************************
   Holt/Dynamic Engineering HI-620x3 Demo Project
   Rev 1.1  Compiled: Jun  2 2020 05:39:14
*********************************************
 Press 'a' - BC Asynchronous Commands on Channel A...
 Press 'b' - BC Asynchronous Commands on Channel B...
 Press 'e' - BC Demo...
 Press 'f' - BC Multi Demo...
 Press 'c' - RT-MT ....
 Press 'r' - RT .......
 Press 'm' - MT .......
 Press 'w' - Reset 1553 chip ..
 Press 'd' - Display HI-620x3 Registers ..
 Press 's' - Select Clock Frequency ..
 Press 'v' - Toggle verbose BC output ..
 Press 'q' - quit
```

Most items are self-explanatory. One chip/device will execute BC functionality (items a,b,e, or f), the other will execute in RT, RT-MT, or MT (items c, r, or m) as previously mentioned RT, RT-MT or MT must be started first, followed by one of the BC modes.
RT/MT execute in polled fashion, though they could be implemented as interrupt driven. Some BC tests demonstrate interrupt driven status, see holt/bcdemo.c for more information.
RT/MT will output received messages/frames.

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
InterNet Address support@dyneng.com