```
NOTE: This is an unmaintained driver.  It is not guaranteed to work due to
changes made in the tty layer in 2.6.  If you wish to take over maintenance of
this driver, contact Michael Warfield <mhw@wittsend.com>.

Changelog:
----------
11-01-2001:	Original Document

10-29-2004:	Minor misspelling & format fix, update status of driver.
		James Nelson <james4765@gmail.com>

Computone Intelliport II/Plus Multiport Serial Driver
-----------------------------------------------------

Release Notes For Linux Kernel 2.2 and higher.
These notes are for the drivers which have already been integrated into the
kernel and have been tested on Linux kernels 2.0, 2.2, 2.3, and 2.4.

Version: 1.2.14
Date: 11/01/2001
Historical Author: Andrew Manison <amanison@america.net>
Primary Author: Doug McNash
Support: support@computone.com
Fixes and Updates: Mike Warfield <mhw@wittsend.com>

This file assumes that you are using the Computone drivers which are
integrated into the kernel sources.  For updating the drivers or installing
drivers into kernels which do not already have Computone drivers, please
refer to the instructions in the README.computone file in the driver patch.


1. INTRODUCTION

This driver supports the entire family of Intelliport II/Plus controllers
with the exception of the MicroChannel controllers.  It does not support
products previous to the Intelliport II.

This driver was developed on the v2.0.x Linux tree and has been tested up
to v2.4.14; it will probably not work with earlier v1.X kernels,.


2. QUICK INSTALLATION

Hardware - If you have an ISA card, find a free interrupt and io port.
		   List those in use with `cat /proc/interrupts` and
		   `cat /proc/ioports`.  Set the card dip switches to a free
		   address.  You may need to configure your BIOS to reserve an
		   irq for an ISA card.  PCI and EISA parameters are set
		   automagically.  Insert card into computer with the power off
		   before or after drivers installation.

	Note the hardware address from the Computone ISA cards installed into
		the system.  These are required for editing ip2.c or editing
		/etc/modprobe.conf, or for specification on the modprobe
		command line.
```

```
        Note that the /etc/modules.conf should be used for older (pre-2.6)
                kernels.

Software -

Module installation:

a) Determine free irq/address to use if any (configure BIOS if need be)
b) Run "make config" or "make menuconfig" or "make xconfig"
   Select (m) module for CONFIG_COMPUTONE under character
   devices.  CONFIG_PCI and CONFIG_MODULES also may need to be set.
c) Set address on ISA cards then:
   edit /usr/src/linux/drivers/char/ip2.c if needed
        or
   edit /etc/modprobe.conf if needed (module).
        or both to match this setting.
d) Run "make modules"
e) Run "make modules_install"
f) Run "/sbin/depmod -a"
g) install driver using `modprobe ip2 <options>` (options listed below)
h) run ip2mkdev (either the script below or the binary version)


Kernel installation:

a) Determine free irq/address to use if any (configure BIOS if need be)
b) Run "make config" or "make menuconfig" or "make xconfig"
   Select (y) kernel for CONFIG_COMPUTONE under character
   devices.  CONFIG_PCI may need to be set if you have PCI bus.
c) Set address on ISA cards then:
           edit /usr/src/linux/drivers/char/ip2.c
           (Optional - may be specified on kernel command line now)
d) Run "make zImage" or whatever target you prefer.
e) mv /usr/src/linux/arch/i386/boot/zImage to /boot.
f) Add new config for this kernel into /etc/lilo.conf, run "lilo"
        or copy to a floppy disk and boot from that floppy disk.
g) Reboot using this kernel
h) run ip2mkdev (either the script below or the binary version)

Kernel command line options:

When compiling the driver into the kernel, io and irq may be
compiled into the driver by editing ip2.c and setting the values for
io and irq in the appropriate array.  An alternative is to specify
a command line parameter to the kernel at boot up.

        ip2=io0,irq0,io1,irq1,io2,irq2,io3,irq3

Note that this order is very different from the specifications for the
modload parameters which have separate IRQ and IO specifiers.

The io port also selects PCI (1) and EISA (2) boards.

        io=0    No board
        io=1    PCI board
        io=2    EISA board
        else    ISA board io address
```

You only need to specify the boards which are present.

```
        Examples:

                2 PCI boards:

                        ip2=1,0,1,0

                1 ISA board at 0x310 irq 5:

                        ip2=0x310,5
```

This can be added to and "append" option in lilo.conf similar to this:

```
        append="ip2=1,0,1,0"
```

3. INSTALLATION

Previously, the driver sources were packaged with a set of patch files
to update the character drivers' makefile and configuration file, and other
kernel source files. A build script (ip2build) was included which applies
the patches if needed, and build any utilities needed.
What you receive may be a single patch file in conventional kernel
patch format build script. That form can also be applied by
running patch -p1 < ThePatchFile. Otherwise run ip2build.

The driver can be installed as a module (recommended) or built into the
kernel. This is selected as for other drivers through the `make config`
command from the root of the Linux source tree. If the driver is built
into the kernel you will need to edit the file ip2.c to match the boards
you are installing. See that file for instructions. If the driver is
installed as a module the configuration can also be specified on the
modprobe command line as follows:

```
        modprobe ip2 irq=irq1,irq2,irq3,irq4 io=addr1,addr2,addr3,addr4
```

where irqnum is one of the valid Intelliport II interrupts (3,4,5,7,10,11,
12,15) and addr1-4 are the base addresses for up to four controllers. If
the irqs are not specified the driver uses the default in ip2.c (which
selects polled mode). If no base addresses are specified the defaults in
ip2.c are used. If you are autoloading the driver module with kerneld or
kmod the base addresses and interrupt number must also be set in ip2.c
and recompile or just insert and options line in /etc/modprobe.conf or both.
The options line is equivalent to the command line and takes precedence over
what is in ip2.c.

/etc/modprobe.conf sample:
```
        options ip2 io=1,0x328 irq=1,10
        alias char-major-71 ip2
        alias char-major-72 ip2
        alias char-major-73 ip2
```

The equivalent in ip2.c:

```
static int io[IP2_MAX_BOARDS]= { 1, 0x328, 0, 0 };
static int irq[IP2_MAX_BOARDS] = { 1, 10, -1, -1 };
```

The equivalent for the kernel command line (in lilo.conf):

```
        append="ip2=1,1,0x328,10"
```

Note: Both io and irq should be updated to reflect YOUR system. An "io"
        address of 1 or 2 indicates a PCI or EISA card in the board table.
        The PCI or EISA irq will be assigned automatically.

Specifying an invalid or in-use irq will default the driver into
running in polled mode for that card. If all irq entries are 0 then
all cards will operate in polled mode.

If you select the driver as part of the kernel run :

```
        make zlilo (or whatever you do to create a bootable kernel)
```

If you selected a module run :

```
        make modules && make modules_install
```

The utility ip2mkdev (see 5 and 7 below) creates all the device nodes
required by the driver. For a device to be created it must be configured
in the driver and the board must be installed. Only devices corresponding
to real IntelliPort II ports are created. With multiple boards and expansion
boxes this will leave gaps in the sequence of device names. ip2mkdev uses
Linux tty naming conventions: ttyF0 - ttyF255 for normal devices, and
cuf0 - cuf255 for callout devices.

4. USING THE DRIVERS

As noted above, the driver implements the ports in accordance with Linux
conventions, and the devices should be interchangeable with the standard
serial devices. (This is a key point for problem reporting: please make
sure that what you are trying do works on the ttySx/cuax ports first; then
tell us what went wrong with the ip2 ports!)

Higher speeds can be obtained using the setserial utility which remaps
38,400 bps (extb) to 57,600 bps, 115,200 bps, or a custom speed.
Intelliport II installations using the PowerPort expansion module can
use the custom speed setting to select the highest speeds: 153,600 bps,
230,400 bps, 307,200 bps, 460,800bps and 921,600 bps. The base for
custom baud rate configuration is fixed at 921,600 for cards/expansion
modules with ST654's and 115200 for those with Cirrus CD1400's. This
corresponds to the maximum bit rates those chips are capable.
For example if the baud base is 921600 and the baud divisor is 18 then
the custom rate is 921600/18 = 51200 bps. See the setserial man page for
complete details. Of course if stty accepts the higher rates now you can
use that as well as the standard ioctls().

5. ip2mkdev and assorted utilities...

Several utilities, including the source for a binary ip2mkdev utility are
available under .../drivers/char/ip2. These can be build by changing to
that directory and typing "make" after the kernel has be built. If you do
not wish to compile the binary utilities, the shell script below can be
cut out and run as "ip2mkdev" to create the necessary device files. To
use the ip2mkdev script, you must have procfs enabled and the proc file
system mounted on /proc.

6. NOTES

This is a release version of the driver, but it is impossible to test it
in all configurations of Linux. If there is any anomalous behaviour that
does not match the standard serial port's behaviour please let us know.

7. ip2mkdev shell script

Previously, this script was simply attached here. It is now attached as a
shar archive to make it easier to extract the script from the documentation.
To create the ip2mkdev shell script change to a convenient directory (/tmp
works just fine) and run the following command:

```
	unshar Documentation/serial/computone.txt
		(This file)
```

You should now have a file ip2mkdev in your current working directory with
permissions set to execute. Running that script with then create the
necessary devices for the Computone boards, interfaces, and ports which
are present on you system at the time it is run.

```
#!/bin/sh
# This is a shell archive (produced by GNU sharutils 4.2.1).
# To extract the files from this archive, save it to some FILE, remove
# everything before the `!/bin/sh' line above, then type `sh FILE'.
#
# Made on 2001-10-29 10:32 EST by <mhw@alcove.wittsend.com>.
# Source directory was `/home2/src/tmp'.
#
# Existing files will *not* be overwritten unless `-c' is specified.
#
# This shar contains:
# length mode       name
# ------ ---------- ------------------------------------------
#   4251 -rwxr-xr-x ip2mkdev
#
save_IFS="${IFS}"
IFS="${IFS}:"
gettext_dir=FAILED
locale_dir=FAILED
first_param="$1"
for dir in $PATH
do
```

```
  if test "$gettext_dir" = FAILED && test -f $dir/gettext \
     && ($dir/gettext --version >/dev/null 2>&1)
  then
    set `$dir/gettext --version 2>&1`
    if test "$3" = GNU
    then
      gettext_dir=$dir
    fi
  fi
  if test "$locale_dir" = FAILED && test -f $dir/shar \
     && ($dir/shar --print-text-domain-dir >/dev/null 2>&1)
  then
    locale_dir=`$dir/shar --print-text-domain-dir`
  fi
done
IFS="$save_IFS"
if test "$locale_dir" = FAILED || test "$gettext_dir" = FAILED
then
  echo=echo
else
  TEXTDOMAINDIR=$locale_dir
  export TEXTDOMAINDIR
  TEXTDOMAIN=sharutils
  export TEXTDOMAIN
  echo="$gettext_dir/gettext -s"
fi
if touch -am -t 200112312359.59 $$.touch >/dev/null 2>&1 && test ! -f
200112312359.59 -a -f $$.touch; then
  shar_touch='touch -am -t $1$2$3$4$5$6.$7 "$8"'
elif touch -am 123123592001.59 $$.touch >/dev/null 2>&1 && test ! -f
123123592001.59 -a ! -f 123123592001.5 -a -f $$.touch; then
  shar_touch='touch -am $3$4$5$6$1$2.$7 "$8"'
elif touch -am 1231235901 $$.touch >/dev/null 2>&1 && test ! -f 1231235901 -a -f
$$.touch; then
  shar_touch='touch -am $3$4$5$6$2 "$8"'
else
  shar_touch=:
  echo
  $echo 'WARNING: not restoring timestamps.  Consider getting and'
  $echo "installing GNU \`touch', distributed in GNU File Utilities..."
  echo
fi
rm -f 200112312359.59 123123592001.59 123123592001.5 1231235901 $$.touch
#
if mkdir _sh17581; then
  $echo 'x -' 'creating lock directory'
else
  $echo 'failed to create lock directory'
  exit 1
fi
# ============= ip2mkdev ==============
if test -f 'ip2mkdev' && test "$first_param" != -c; then
  $echo 'x -' SKIPPING 'ip2mkdev' '(file already exists)'
else
  $echo 'x -' extracting 'ip2mkdev' '(text)'
  sed 's/^X//' << 'SHAR_EOF' > 'ip2mkdev' &&
#!/bin/sh -
#
#       ip2mkdev
#
#       Make or remove devices as needed for Computone Intelliport drivers
#
#       First rule!  If the dev file exists and you need it, don't mess
#       with it.  That prevents us from screwing up open ttys, ownership
#       and permissions on a running system!
#
#       This script will NOT remove devices that no longer exist if their
#       board or interface box has been removed.  If you want to get rid
#       of them, you can manually do an "rm -f /dev/ttyF* /dev/cuaf*"
#       before running this script.  Running this script will then recreate
#       all the valid devices.
#
#       Michael H. Warfield
#       /\/\|=mhw=|\/\/
#       mhw@wittsend.com
#
#       Updated 10/29/2000 for version 1.2.13 naming convention
#               under devfs.    /\/\|=mhw=|\/\/
#
#       Updated 03/09/2000 for devfs support in ip2 drivers. /\/\|=mhw=|\/\/
#
X
if test -d /dev/ip2 ; then
#       This is devfs mode...  We don't do anything except create symlinks
#       from the real devices to the old names!
X       cd /dev
X       echo "Creating symbolic links to devfs devices"
X       for i in `ls ip2` ; do
X               if test ! -L ip2$i ; then
X                       # Remove it incase it wasn't a symlink (old device)
X                       rm -f ip2$i
X                       ln -s ip2/$i ip2$i
X               fi
X       done
X       for i in `( cd tts ; ls F* )` ; do
X               if test ! -L tty$i ; then
X                       # Remove it incase it wasn't a symlink (old device)
X                       rm -f tty$i
X                       ln -s tts/$i tty$i
X               fi
X       done
X       for i in `( cd cua ; ls F* )` ; do
X               DEVNUMBER=`expr $i : 'F\(.*\)'`
X               if test ! -L cuf$DEVNUMBER ; then
X                       # Remove it incase it wasn't a symlink (old device)
X                       rm -f cuf$DEVNUMBER
X                       ln -s cua/$i cuf$DEVNUMBER
X               fi
X       done
X       exit 0
fi
X
```

```
if test ! -f /proc/tty/drivers
then
X       echo "\
Unable to check driver status.
Make sure proc file system is mounted."
X
X       exit 255
fi
X
if test ! -f /proc/tty/driver/ip2
then
X       echo "\
Unable to locate ip2 proc file.
Attempting to load driver"
X
X       if /sbin/insmod ip2
X       then
X               if test ! -f /proc/tty/driver/ip2
X               then
X                       echo "\
Unable to locate ip2 proc file after loading driver.
Driver initialization failure or driver version error.
"
X               exit 255
X               fi
X       else
X               echo "Unable to load ip2 driver."
X               exit 255
X       fi
fi
X
# Ok...  So we got the driver loaded and we can locate the procfs files.
# Next we need our major numbers.
X
TTYMAJOR=`sed -e '/^ip2/!d' -e '/\/dev\/tt/!d' -e 's/.*tt[^     ]*[
]*\([0-9]*\)[   ]*.*/\1/' < /proc/tty/drivers`
CUAMAJOR=`sed -e '/^ip2/!d' -e '/\/dev\/cu/!d' -e 's/.*cu[^     ]*[
]*\([0-9]*\)[   ]*.*/\1/' < /proc/tty/drivers`
BRDMAJOR=`sed -e '/^Driver: /!d' -e 's/.*IMajor=\([0-9]*\)[     ]*.*/\1/' <
/proc/tty/driver/ip2`
X
echo "\
TTYMAJOR = $TTYMAJOR
CUAMAJOR = $CUAMAJOR
BRDMAJOR = $BRDMAJOR
"
X
# Ok...  Now we should know our major numbers, if appropriate...
# Now we need our boards and start the device loops.
X
grep '^Board [0-9]:' /proc/tty/driver/ip2 | while read token number type
alltherest
do
X       # The test for blank "type" will catch the stats lead-in lines
X       # if they exist in the file
X       if test "$type" = "vacant" -o "$type" = "Vacant" -o "$type" = ""
```

```
X	then
X		continue
X	fi
X
X	BOARDNO=`expr "$number" : '\([0-9]\):'`
X	PORTS=`expr "$alltherest" : '.*ports=\([0-9]*\)' | tr ',' ' '`
X	MINORS=`expr "$alltherest" : '.*minors=\([0-9,]*\)' | tr ',' ' '`
X
X	if test "$BOARDNO" = "" -o "$PORTS" = ""
X	then
#	This may be a bug.  We should at least get this much information
X		echo "Unable to process board line"
X		continue
X	fi
X
X	if test "$MINORS" = ""
X	then
#	Silently skip this one.  This board seems to have no boxes
X		continue
X	fi
X
X	echo "board $BOARDNO: $type ports = $PORTS; port numbers = $MINORS"
X
X	if test "$BRDMAJOR" != ""
X	then
X		BRDMINOR=`expr $BOARDNO \* 4`
X		STSMINOR=`expr $BRDMINOR + 1`
X		if test ! -c /dev/ip2ipl$BOARDNO ; then
X			mknod /dev/ip2ipl$BOARDNO c $BRDMAJOR $BRDMINOR
X		fi
X		if test ! -c /dev/ip2stat$BOARDNO ; then
X			mknod /dev/ip2stat$BOARDNO c $BRDMAJOR $STSMINOR
X		fi
X	fi
X
X	if test "$TTYMAJOR" != ""
X	then
X		PORTNO=$BOARDBASE
X
X		for PORTNO in $MINORS
X		do
X			if test ! -c /dev/ttyF$PORTNO ; then
X				# We got the hardware but no device - make it
X				mknod /dev/ttyF$PORTNO c $TTYMAJOR $PORTNO
X			fi
X		done
X	fi
X
X	if test "$CUAMAJOR" != ""
X	then
X		PORTNO=$BOARDBASE
X
X		for PORTNO in $MINORS
X		do
X			if test ! -c /dev/cuf$PORTNO ; then
X				# We got the hardware but no device - make it
```

```
X                               mknod /dev/cuf$PORTNO c $CUAMAJOR $PORTNO
X                       fi
X               done
X       fi
done
X
Xexit 0
SHAR_EOF
  (set 20 01 10 29 10 32 01 'ip2mkdev'; eval "$shar_touch") &&
  chmod 0755 'ip2mkdev' ||
  $echo 'restore of' 'ip2mkdev' 'failed'
  if ( md5sum --help 2>&1 | grep 'sage: md5sum \[' ) >/dev/null 2>&1 \
  && ( md5sum --version 2>&1 | grep -v 'textutils 1.12' ) >/dev/null; then
    md5sum -c << SHAR_EOF >/dev/null 2>&1 \
    || $echo 'ip2mkdev:' 'MD5 check failed'
cb5717134509f38bad9fde6b1f79b4a4  ip2mkdev
SHAR_EOF
  else
    shar_count="`LC_ALL= LC_CTYPE= LANG= wc -c < 'ip2mkdev'`"
    test 4251 -eq "$shar_count" ||
    $echo 'ip2mkdev:' 'original size' '4251,' 'current size' "$shar_count!"
  fi
fi
rm -fr _sh17581
exit 0
```