New8n1

Iain Barclay

COLLABORATORS

	TITLE : New8n1		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Iain Barclay	October 23, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

New8n1

1.1 8n1.device © 1996-97 Iain Barclay

```
              8n1.device - By Iain Barclay
   was by L.Lucius

A replacement for the standard serial.device
 and based on v34serial.device.




          ~Distribution~~
              What you can do with this

          ~Disclaimer~~~~
              Blah Blah Blah

          ~Background~~~~
              Why I decided to take on this task

          ~Installation~~
              Just follow these simple instructions

          ~Compatibility~
              Differences between 8n1.device & serial.device

          ~Thanks       ~
              For the bug reports and suggestions

          ~Feedback/Bugs~
              All reports good or bad are needed

          ~NComm Note~~~~
              A note to NComm 3.0.? users

           000-040 notes
              Notes on the different CPU versions

           Speed notes
              Notes on the different baud rates
```

```
                 Recompilation
                    Notes on re-compiling the device
  ~Change~Log~~~~    A list of all changes made to 8n1.device
```

1.2 Distribution

You may use or misuse this program in any way you like.

1.3 Disclaimer

I will not be held responsible for ANY loss incurred by this program.

1.4 Background

When "v34serial.device" showed up on Aminet, I was excited, because all I ever used was 8N1 and RTS/CTS. That was it. I didn't need parity or XON/XOFF or breaks and thought that if the device didn't have to worry about all that then it would be faster.

Well, I used it for awhile, but every now and then it would cause GURUs, so I went back to "artser.device" and forgot about it.

Until a little while ago, when I found out that I would be getting a SLIP connection to the net.

I wanted something faster than "artser.device" and with less overhead. So I went to debugging "v34serial.device" and wound up rewriting the whole thing.

The end result is a minimal serial device replacement that tries to keep system overhead at a reasonable level.

And i started coding this because the original 8n1.device crashed my system sometimes (and i wanted to make it faster).

    Iain

1.5 Installation

```
                Simply copy 8n1.device to the DEVS: directory and tell your ↩
                   communications
software the new name.
                NComm users read this!
                If your communications software in unable to accept the name of ↩
                   the serial
```

```
device to use, get the file on Aminet called SerPat20.lha in the hard/drivr
directory.  It will give you this capability.

        !!!IMPORTANT!!!

If you have two or more programs that share the serial port, make sure you
change ALL programs.  Otherwise, you will probably get a visit from some guy
called GURU.  (Thanks Juan!)

Also a note for the thick:

device should be called
    DEVS:8n1.device
and NOT!
    DEVS:serial.device
do NOT rename it!!!
```

1.6 Compatibility

```
8n1.device should be compatible with the "serial.device" as long as the
following options are the only ones used: B-)

    8 data bits
    No parity
    1 stop bit
    RTS/CTS handshaking
    No handshaking
    EOFMODE

It also supports sending breaks, but not receiving them.

If you need something else, let me know and I'll see what I can do.
```

1.7 Thanks for the bug reports and suggestions

```
8n1.device was originaly writen & released by

L.Lucius
E-mail at: llucius@millcomm.com

He did all the hard work :)

Special note for '*Art'
 Thanks for your great patch. I can't find your email address to
 ask permission to include your patch in this archive so i hope you
 don't mind, let me know.

Thanks to the following people for their help, reports, testing, and patience:

   William Crawford IV         Mans Engman    Ayan George
   Paul Harrison        Harold H. Ipolyi     Mathias Juvall
   Alpay Kasal          George Kourkoutas    Laura
   John Millington        Bryan Myers    Robert N. Olson
   Greg Olstad          Ernest Otte    Koen Peetermans
   Juan Ramirez         Michal L. Rybarski   George Sanderson
   Orlando Santiago       Peter S. Struijk     Tinic Urou
   Ashok Vadekar        Ronald van Eijck     Alexander Wild
   Dwight Zenzano       Stimpson       Andrew Bennett
   Frank Wille          Oliver Jeannet     Mark Knibbs
   Jens Rosenboom       Jorma Oksanen
```

Let me know if I've missed anyone.

1.8 Feedback/Bugs

I have included the source in hopes that some other programmers can make suggestions. It's always better to have more than 1 eye (literally B^) looking at the code.

If you have problems or feedback I would be very glad to hear about it:

```
KNOWN BUGS
    I have been told MOD players don't get on with 8n1
    but have not had this problem myself.

Bug reports should include the following data
    The Version of 8n1.device you were using (V37.37, 030)
    The software you were using (AmiTCP, Miami, Term etc..)
    The Version of the Software you used (Miami V2.1)
    The Patches you have installed. (MCP V1.30Beta16 ...)
    The baud rate you were using.
    If you have fastram, Is it 60ns fast ram?
    The CPU type (68000, 68010, 68020, 68030, 68040, 68060, PowerUP ...)
    The OS version (2.04, 2.05, 3.0, 3.1)
    The Gfx on your system. (ECS, AGA, CyberGfx ....)
    The Type of machine you have (A500, A600 ...)
    The Type and speed of your modem. (USR 14.4k ...)

Iain 'Ook' Barclay               /                  /
E-mail at: iainb@thenet.co.uk /   Team *AMIGA*   /
IRC: Ook    DALNET  #AmIRC     /      /
IRC note: Try Weekends 6-12pm (GMT)
```

1.9 Note to NComm 3.0.? users

```
NOTE: I have not included an NComm version as i think NComm is sh**e
    get TERM it's much better.
    You can easily recompile for NComm if you wish.
Iain
```

I spent over 2 weeks trying to track down a bug that I (and everyone else) thought was a bug in 8n1. I mean it made sense that 8n1 was the problem since ALL the other drivers worked, right? Well...

Using Stefan Pröls's excellent serlog.device and another program (and a lot of paper for debugging), I was able to determine that 8n1 was working the same as all the rest.

So, if it works the same then why the problem?

Well, the problem occurs when you try to upload using zmodem. One of the first things zmodem does is sync up with the other end. Part of this process checks to see if any characters have been transmitted by the other end. To do this, zmodem asks NComm how many characters are available and NComm tells zmodem that one character is ready to be read. But, there isn't and since zmodem doesn't know that, it reads a character and get's whatever happens to be at the beginning of NComms read buffer.

To further illustrate, here's the sequence of I/Os that artser goes through just prior to where the problem would occur:

```
     CMD_READ (     ): OK, 1 of 1 byte
       0000: 2A                 "*"

     SDCMD_QUERY (QUICK): OK
       status:   $0004
       buffered: 12

     CMD_READ (QUICK): OK, 12 of 12 byte
       0000: 18 42 30 39 30 30 30 30 30 30 30 30        ".B0900000000"

     CMD_READ (QUICK): OK, 1 of 1 byte
       0000: 61                 "a"

     SDCMD_QUERY (QUICK): OK
       status:   $0004
       buffered: 6

     CMD_READ (QUICK): OK, 6 of 6 byte
       0000: 38 37 63 0D 0A 11             "87c..."

     CMD_READ (     ): aborted, 0 of 1 byte
       0000:                  ""
```

And here's the sequence that 8n1 goes through:

```
     CMD_READ (QUICK): OK, 20 of 20 byte
       0000: 2A 18 42 30 39 30 30 30 30 30 30 30 30 61 38 37  "*.B0900000000a87"
       0010: 63 0D 0A 11              "c..."

     CMD_READ (     ): aborted, 0 of 1 byte
       0000:                  ""
```

As you can see they are slightly different, but the end result is the same.

However, looking at the second to last READ of each, you will notice that in artser's case the character at the beginning of the buffer is an "8" and in 8n1's case an "*" is at the beginning. This is the character that NComm wrongly returns to zmodem and since zmodem considers the "*" special, it will constantly tries to resync with the other end. Thus, the problem.

```
SO WHO CARES!  Just tell me how to get around it.

Well, I've included a special version of 8n1 for NComm users.  Simply rename
"8n1.device.ncomm" to "8n1.device" and place it in you're DEVS: directory.

It gets around the problem by placing a NULL (0x00) byte at the beginning of
the buffer when a READ is initiated.  This DOES NOT correct NComm's bug.  It
only attempts to work around it.
```

1.10 CPU notes

```
Note: All versions of this device work best with the VBR in FastRam.
      You can only put the VBR into fastram on 010+ cpus ;)
      These devices also assume and work best with
      all caches etc.. turned on.

    000:
  This device will work on any machine.

    010:
  Use the 000 version.

    020:
  Use the 030 version.

    030:
  This device will work on chip machines but works best with fast ram and
  a CopyMemQuicker patch installed.
  The 030 version IS the 020 version

    040:
  The 040 device neads fast ram and a CopyMemQuicker patch installed.
  Well anyone that has an 040 should have fast right.

    060:
  The 060 device neads fast ram and a CopyMemQuicker patch installed.

    PPC:
  Maybe someday. Depends on how the PowerUp boards work.

NB: If you have MCP or APatch installed then you don't nead the CopyMemQuicker  ↩
    patch
that comes with this archive.
```

1.11 Speed notes

```
These rates are a suggestion only as they assume you want to run other
software (AmIRC, IBrowse etc..) as well as the Terminal/tcp stack

    19200:
  Any machine can use this baud.
    38400:
  All machines from A500+ (with fast) to A4000 can use this baud
  with no errors.
  (dunno about bare A500+)

    57600:
  You nead an 020+ and some 60ns Fast Ram to use this baud.
  NB: ppl changing to 8n1 from artser or serial and suddenly finding lots of ↩
     errors
      during transfer should check that they have 60ns Fast Ram. If you don’t then ↩
           change
      down to 38400 baud this should solve your problems. This problem is caused
      by 8n1 being too fast for your ram ;)

    62400+:
  You should get an IO card for this sort of speed.
  (PortPlus is a very good option for a1200 owners @ 460k baud)
```

1.12 Re-compiling the device

```
  The code is known to compile with PhxAss and DevPac.
  There are several flags i use in the code to comtrol how it compiles these are ↩
      :

  NEWCODE
If set to 1 this will include the newest code i’ve added
(default 1)

  SETDSR
If set to 1 the DSR will ALWAYS be set
(default 0)

  NCOMM
If set to 1 the device will compile for use with NComm
(default 0)

  STATS
For debugging use
(default 0)

  FASTVBR
If set to 1 the device automatically puts the VBR into fastram if it can.
(default 1)

  DANGER
If set to 1 dangerous (but faster) code will be used, this could mean the device
does not work on your machine
(default 0)

  NEGCHECK
If set to one -1 size checking is included
(default 1)

  EOFCODE
```

```
If set to 1 the eofmode stuff is included
(default 1)
```