Terminal Node Controllers

Towards The Next Generation ?

A possible basic architecture for future TNC designs.

Darryl Smith, BE, VK2TDS
POBox 169
Ingleburn NSW 2565
Australia
VK2TDS@OzEmail.Com.AU
+61298295414(H)
+61 2 9268 7890 (Fax)
+61 4 1292 9634 (Mobile)

ABSTRACT

This paper describes work into a new generation of hardware for Terminal Node Controllers (TNC's). This development has been done under Linux on IBM compatible hardware, but is easily transferable to a more traditional microprocessor based TNC design.

INTRODUCTION

Have a look about my shack and you will find an amazing number of TNC's. There is an original TNC l, a couple of home built TNC-2's in various states of repair, another TNC-2 board with no parts and a KPC-9612 dual port TNC.

With the exception of the KPC-9612, they are all single port designs. The KPC-9612 is a dual port design, but as far as I can tell it implements the serial ports using software rather than hardware limiting further expansion.

These TNC's are also limited in their bit rates. I doubt that any of the designs could exceed 19.2 kBit/sec.

Higher speed equipment is needed if amateur radio is to survive. This paper describes my work into assisting those designing high speed radios and modems.

'What works is what matters' and through my use of a development board, I have proved that this solution will work.

My work will not suit those working on really high speed links but is ideal for applications up to 100 kBps.

TNC-TNG MAILING LIST

Much of the initial work on this project was done as a result of discussions on the TNC-TNG mailing list at LANTZ.COM.

All the participants have gone out of their way to help in this project, exemplifying the spirit of Ham Radio. Some of the suggestions made are included here whist others have been discounted for various reasons.

Early on I made the decision that more intelligent hardware was required if we were to proceed. Using IC's designed 2 years ago would not help with 10 year old TNC designs.

MOTOROLLA 68360

The 68360 communications controller from Motorolla is one IC that initially seemed most suitable for a new TNC design. With a 32 bit CPU, 4 sync/async ports, 2 more async ports, and a programming port it is quite suitable for use within a TNC. With the 68360EN, one of the sync/async ports becomes a full Ethernet port, with very little internal hardware required.

The 68360 contains almost 2Kbytes of memory for pointers to packet data structures necessitating little host intervention during packet reception.

However, several things led me away from using this IC. Firstly, development boards were too expensive, with all boards I could find costing over US$2,000. In addition, low quantities of this chip were approaching the US$lOO mark, without guaranteed supply.

CIRRUS CD-2341

Once these problems became obvious, one participant of TNC-TNG suggested the use of the Cirrus Logic CD-2400 series of chips. When I received the data books from Cirrus Logic I was sold.

Cirrus call this IC family 'Advanced Multi-Protocol Communications Controllers', and they are truly advanced. The chip has the following capabilities

- Four full duplex ports each running up to 134kBits/second, with two independent bit rate generators per port.
- Support for async, async-HDLC along with synchronous HDLC & SDLC on all channels
- 32 bit address, 16 bit data double buffered DMA controller for each transmitter and receiver.
- NRZ, NRZI and Manchester coding.
- a Digital PLL on each receiver and two timers per channel
- PPP, SLIP and MNP-4
- Intel and Motorolla hardware and software compatibility.
- 16 byte receive and transmit FIFO's

To top off that list they are also quite cheap at less than US$25.00 in small quantities.

Hardware

If this chip were used along with a microprocessor, designing and manufacturing a TNC with 4 or 8 radio ports, each capable of high speed operation, would not be a complex task. A TNC could literally consist of a microcontroller, memory, CD-2431 and maybe an Ethernet chip.

Development hardware is however scarce for this chip. After searching for months I finally found a suitable board. It is hoped that this paper spurns some activity in designing a development board even more suited to amateur radio use.

THE CRONYX-SIGMA 22 DEVELOPMENT BOARD

The development board I eventually chose was the Cronyx Sigma-22 at only US$340 plus postage. This is not cheap, but I feel that the board is well worth it.

Although documentation for the card is lacking the technical support that I received was remarkable. Unfortunately, although all the normal manuals were supplied with the card, they were all in Russian. I understand that Cronyx is working on an English translation.

Drivers are available from Cronyx for Windows, Linux and BSD/386 Unix. The software supplied does is not suitable for amateur use. For this reason, I have rewritten much of the Linux driver. The changes I have made are centred in adding support for timer delays and passing full frames to other networking layers without processing.

*

The Sigma-22 has two external serial ports at RS-232 level and two more on the internal connector at TTL levels. Appendix A documents the external pinouts for each port, whilst Appendix B shows the internal pinout of the otherwise undocumented internal connector. Like the rest of the board thought has been put into the pinouts of this connector. Should you connect the plug around the wrong way, the only consequence will be that ports 2 and 3 will be swapped. (Ports 0 and 1 are the external ports).

This internal connector is designed and positioned so that it is possible to have a daughter board inside the computer with a couple of 1200 bps modems with PTT hardware, leaving RS-232 level outputs on the other ports for higher speed hardware.

When my Sigma board, the 40 pin header was missing. It was quite simple to insert a 40 pin header from my junk box.

All the signals are at TTL level so interfacing is simple. If connecting to the external connectors is required at TTL levels, level converters are required.

HARDWARE

Essential Pins

Today there are three main physical interfaces used to connect to equipment such as modems. They are TTL, RS-232 and RS-422/RS-485. Although I prefer working with TTL I can see that some users will prefer one of the other signal levels.

A watchdog timer for PTT lines is essential in the TNC, as is a way to disable it, without opening the back of the controller. In a previous design I implemented this feature by allowing two pins in the connector to be shorted disabling the watchdog.

I feel that RTS/CTS, as well as DCD and data should be available with RS-232 levels, allowing 1200 bps voice modems to be connected without any glue logic.

Clock Signals

After dealing with the normal array of SCC chips, the CL-CD853 1 is a nice change. Gone are the days when bit rate generators were scarce and much anguish took place over the hardware configuration of clocks.

Clock inputs and outputs are available for each transmitter and receiver on the chip, with programmable divisors. Thus modems with a 32* receive clock interface with the same ease as a modem requiring a 1 * or 16* clock.

Unlike 8530 designs no external divider is required to loop the transmit clock into the receiver, as the divider is available inside the chip.

Transmit Clocks

The only glaring problem I could find with the CL-CD243 1 is the lack of a transmit clock at a rate other than unity. This limits the use of modems such as the K9NG 9600 bps modem which require a clock sixteen times the bit rate (Unless you have purchased a modem such as the TAPR 9600 bps unit and 'Clock Option').

If a full design were produced using this chip it may be desirable to use add an LSI bit rate generator. However in most cases the clock recovery in the IC will be good enough so that external DPLL's ,Which is the main use for these clocks anyway, are not required.

The receiver can accept input clocks at any integer multiple of the input clock. Appendix C contains an evaluation of where present modem designs would need to be modified to work with this chip.

SOFTWARE

To check the viability of using a Cirrus communications controller in a production design I have prototyped the software under Linux. The software is very basic and implements a simple TNC. At the time of writing, not even the timers were implemented, but transmitting and receiving data works flawlessly.

Why KISS?

The best way to learn what I did in software is to use the source - both the original and my modified version. Unless you are familiar with the Linux SKB buffer system I suggest you look at Alan Cox's excellent article[l] in the Linux Journal.

The Cronyx Sigma Linux drivers are based on the same underlying structure as Linux in that it uses sk_buf's rather than mbuf's common to BSD and NOS programmers. Sk_buf 's are based on the presumption that memory is cheap, and the overhead of buffer assignment inside an Interrupt Service Routine is expensive. Therefore s k_bu f's are assigned to a size slightly larger than the average packet size, and. can be chained

Th driver however assumes that each s k_bu f contains only one packet meaning that if the pre-assigned buffer is smaller than the incoming packet then that packet is lost.

This is not a problem with the transmitter as an s k buf 's can be allocated large enough. In practice since the AX-25 protocol required to be used in many countries the maximum packet size is known.

Transmit Delays

In packet radio there are several delays required for correct operation. They are TxDelay, TxTail and SlotTime. Luckily none of these delays overlap allowing a single programmable timer to be used on each channel.

I will be implementing these delays using one of the general timers on each channel in the CD-243 1. According to the documentation, general timers on the CD2341 can be set under 1 mSec and still maintain accuracy. For simplicity I will be setting the timer base unit to one mSec in software, with all timers being a multiple of this.

Using a 1 mSec granularity allows for a timer of over a minute without loss in accuracy or additional hardware or software, which is fast enough for even the slowest HF radios.[1]

Testing

Whilst testing I like to remove as many variables as possible, as well as cause as little impact to those around me[2]. The first variable to remove was the radio interface, including modems and radios.

The KPC-9612 has an unusual feature on the 9600 bps radio port; it has digital input and output signals. These could be directly connected to the TTL signals on the Sigma board. OK, this is not entirely correct! The signals on the KPC-9612 are in fact scrambled[3] like all digital signals should be going to a data radio. However none of the documentation in the KPC manual mentioned this.

I was going to say that the KPC-9612 enabled me to ensure that everything was working before touching a radio. However it did not turn out that way. In the end I needed to start up my TNC-1 as a source of packets.[4]

The only thing that I could find that was lacking in the TNC-1 was that I was unable to ignore CRC errors. This is essential when trying to debug new hardware and software. I eventually saved the incoming kiss stream received from the TNC through the Cirrus chip, and replayed this back to the TNC to get the transmit running for the first time.

I noted something else whilst attempting to make sense of the data coming through the cirrus chip from the KPC. I had for some reason believed that the call signs in an AX.25 frame would be in ASCII. They are not.

This will not be a surprise to those who have read and understand the protocol definition. All call signs in AX.25 are standard ASCII upper case characters, BUT they are then bit shifted up by one bit.[5]

CONCLUSION

Through my work with the Cronyx board and the Cirrus CL-CD243 1, I have proved that this chip is suitable for use in amateur radio. I have already demonstrated that writing software to support this IC is much easier than many other chips. I have also shown that CPU requirements are reduced considerably with the CD243 l's in built DMA controller. However potential users should note that clock outputs are limited to the set bit rate of the channel

[1] I Hope ☺

[2] Whilst I was completing my University Thesis I noticed that a friend, who was also working on his thesis, was transmitting one packet every l-2 seconds for a few hours. Suspecting a bug, I rang my friend and had to leave a message on his answering machine. He got back to me the next week, when he returned from a weekend away. In the meantime he had transmitted too many packets to count as his software was not quite as bug free as he thought.

[3] See 'Wireless Digital Communications' from TAPR for an explanation of scrambling.

[4] I should thank the designers of the TNC-1 for their forethought in adding an internal modem connector to the TNC-1. It was a work of genius, even if there are one or two features missing.

[5] I know this is not a revelation to many, but many Amateurs such as myself will fall into this trap.

In the past, most serial controllers used for amateur radio have been based on the ubiquitous 8530. This was understandable with so much software available for it.

I hope that through this project and paper that I have opened new doors to those developing for Amateur Radio.

Source Code

The source code as well as full documentation is available at
`HTTP://www.ozemail.com.au/~vk2tds`
Some of the software is available from Cronyx at their FTP site.

The software that is supplied with the Cronyx Sigma-22 card implements Frame Relay, Cisco and PPP packets.

Future Software and Hardware

This project has proved that a design based on the Cirrus CD-243 1 is quite viable with following areas of investigation suggested.. .

- Design of a microprocessor TNC - ideally including an Ethernet port.
- Transfer the code from Linux/Unix to a microprocessor architecture. Actually rewrite all the code.
- Use PPP rather than KISS for connections between the PC and the TNC.
- Add the complete AX.25 and maybe TCP/IP stack to the drivers.
- Maybe even implement a WWW server for configuration!
- Generate a 16 times transmit clock. This could then be implemented in a PAL or GAL.

Acknowledgments

I would like to thank all those who assisted me with this project. In particular I need to thank the following.

- Terry Behan, VK2TLU
- Craig Small, VK2XLZ
  `(csmall@gonzo.triode.net.au)`
- Cirrus Logic
- Serge Vakulenko of Cronyx.
  `(Vak@cronyx.ru)`
- TAPR and the designers of the TNC-1 and TNC-2.
- David Kelly, N4HHE,
  `(dkelly@hiwaay.net)`
- All the participants on the TNC-TNG Mailing list.

In addition I would like to thank all my friends at Pacific Power for their assistance.

REFERENCES

1. Kernel Komer: Network Buffers And Memory Management by Alan Cox, Issue 29, The Linux Journal, September 1996.[6]

2. The KISS TNC by Mike Chepponis, K3MC and Phil Kam, KA9Q

3. Multi-Drop KISS operation by Karl Medcalf, WK5M, 10th CNC, p 109-l 11

4. Linux Kernel Hacker's Guide. Now available at http://www.redhat.com

5. Computer Networking Conference Proceedings, 1984-l 996, ARRL & TAPR.

6. Cirrus Logic Data Book Cl-CD243 1, Version 3.0, August 1996, Cirrus Logic.[7]

[6] This article can be found on the Internet at http://redhat.com.au
[7] NOTE: If you are writing software for the CD-243 1 you should also specifically ask for the errata sheets from Cirrus Logic.

7. HAPN-2 by John Vanden Berg, VE3DW, CNC-11, Pp.98-105

8. PI Card, Dave Perry, VE3IFB, CNC-10, Pp.121-124

9. A prototype TNC-3 Design Approach, CNC-12, Pp.1 1-15

10. A proposal for a standard digital interface, Jeffrey Austen, A9 JA, CNC- 13, Pp. l-4

Appendix A

HDB-26	V.35	RS-232
1	T$_x$Da	
2	T$_x$Db	
3	R$_x$Da	
4	GND	GND
5	R$_x$C$_{in}$a	
6	select	select
7	T$_x$C$_{out}$a	
8	GND	GND
9	T$_x$C$_{out}$b	
10	RTS	RTS
11	GND	GND
12		T$_x$D
13	GND	GND
14		R$_x$D
15	R$_x$C$_{in}$b	
16	CD	CD
17		T$_x$C$_{in}$
18	GND	GND
19	DTR	DTR
20		T$_x$C$_{out}$
21	CTS	CTS
22	T$_x$C$_{in}$a	
23	T$_x$C$_{in}$b	
24	R$_x$Db	
25	DSR	DSR
26		R$_x$C$_{in}$

Appendix B

GND	1 2	RTS$_2$
+12 V	3 4	R$_x$D$_2$
+5 V	56	T$_x$C$_{lk}$In$_2$
GND	7 8	R$_x$C$_{lk}$In$_2$
-12 V	9 10	T$_X$D$_2$
GND	11 12	
	1 3 1 4	T$_x$C$_{lk}$Out$_2$
CD$_3$	15 16	
DTR$_3$	17 18	CTS$_2$
DSR$_3$	19 20	
	21 22	DSR$_2$
CTS$_3$	23 24	DTR$_2$
	25 26	CD$_2$
T$_x$C$_{lk}$Out$_3$	27 28	
	29 30	GND
T$_x$D$_3$	31 32	-12V
R$_x$C$_{lk}$In$_3$	33 34	GND
T$_x$C$_{lk}$In$_3$	35 36	+5 V
R$_x$D$_3$	37 38	+12 V
RTS$_3$	39 40	GND

Pin outs of the Cronyx internal feature connector.

Appendix C

KD2BD PacSat Modem (1200)	Requires 16* clock although this is divided to a 1* clock anyway.
TAPR/K9NG 9600	16 or 32 times. Used for clock recovery. It should be possible use the IC for clock recovery.
HAPN-4800 and derivatives	No clock signals required.
TCM-3 105	No clock signals required
7 109 World Modem Chip	No clock signals required

CONTACTS

CRONYX LTD (Russia)

Tel. (7-095)939-23-23
EMAIL: info@cronyx.ru
WWW: http://www.conyx.ru

Many people will however find it easiest talking to their USA agents.

TETRA GROUP

USA Agent for Cronyx

Joseph Kulinets
(Joseph_Kulinets@compuserve.com)
Tel: (203) 968-9158
fax: (203) 968-94 18

Cirrus Logic(USA)

3 100 West Warren Ave
Freemont, CA, 94538
Tel: (510) 623 8300
Fax: (510) 252 6020