Intelligent Digipeating using DIGI_NED on Obsolete PCs

Henk de Groot, PElDNN
Schopenhauerstraat 10,
7323 MC Apeldoom, The Netherlands
peldnn@amsat.org

Abstract

In Europe the roll-out of APRS is well under way right now. However, we do not have a stable APRS infrastructure. This paper describes how we use old and obsolete PCs, together with packet hardware commonly used in western Europe, to augment the APRS infrastructure.

Key words

DIG1 NED, digipeater, intelligent, PC, DOS, Linui

Introduction

At the end of 1999 we experienced the re-birth of APRS in Europe. Introduction of APRS had been tried before, but it never really took off. Now it is different. All packet and RTTY bulletins published by different clubs have some feature articles on APRS. Also PWGN, the Dutch Packet Workgroup, which previously supported the “traditional” packet network, now runs articles about APRS in their magazine. At every hamfest in the last few months there has been a demonstration of APRS.

The situation concerning packet in Europe is quite different from the US. Here we have a reasonably efficient packet network, all connected by radios I must add. Bulletins and private messages are almost guaranteed to arrive. Most of the nodes, BBSs and DX clusters do not use TNCs. The TNCs that are used are almost always fitted with NORD><LINK’s “TheFirmware” and operate in KISS mode. FlexNet TNCs use a special 6Pack EPROM to give better control over the radio.

The stations that do not use TNCs (and that 1s the majority) use special SCC cards fitted with Zilog’s Z85x30 chips. Connected to these cards are a variety of modems, from TMS3105based lk2 modems, G3RUH 9k6, Cadams and DF9IC (both German modems) to real high-speed modems (from 19k2 through 2 Mbps) for interlinks. All nodes in this area support at least 9600 bps FSK. Besides that, German stations use RMNCs which are FlexNet nodes using dedicated hardware.

The end users - the individual stations - do not use TNCs either (or if they do, they only use KISS mode). Most of them use cheap modems like BayCorn’s lk2 modem (and clones), Par96, PicPar and IV3NWV’s YAM modem (this one I use myself). Power users also use SCC cards for their private station.

Finally we in Europe, at least here in the Netherlands, do not have the same emergency tasks as in the US. Western Europe is densely populated, with an excellent communications infrastructure almost everywhere. Furthermore, we are not bothered by earthquakes or tornadoes like in the US. This means that APRS is really a “fun” project, and the “emergency” part of it is much less important,.

This is the background against which our involvement in APRS started.

How we started with APRS

About one year ago Remko (PEIMEW, a nearby ham) and I had some thoughts about having weather measurement stations all over the country, collecting and exchanging WX telemetry. At that time it was only an outline plan, and life went on. Later, when we started to look at APRS, we realized that this was a suitable exchange mechanism for just this kind of data. If APRS were used we would also have the clients to display the weather data, which is a huge advantage.

For this to work there have to be enough digipeaters, so when one disappears and another appears the network remains usable, as long as there are at least a critical number of digipeaters on the air.

Our first objective was to create a tight closed network. For this we needed very cheap digipeaters that run reliably. Given the history in Europe, the most logical way to do this was to use PCs, with cheap and obsolete 80286.class PCs being more than adequate. Since many people have migrated to 9k6 over recent years, there are now many BayCom lk2 modems which are no longer used, and which could be applied again for APRS. Also for cross-band work SCC cards should be supported as well as the good old KISS modem.

We need the cross-band functionality. In Europe the major APRS frequency is 144.800 MHz. In the Netherlands novices are not allowed to use digital modes on 2m. They are allowed to use digital modes on 70cm however. So cross-band functionality between 2m and 70cm is needed to let novices join in the fun.

After evaluation of existing digipeaters we found none would do what we wanted; the digipeater should be “intelligent”, conform to the APRS specification and be stable. Besides that, we also wanted source code so we can add the features we want, like support for home-brew weather measurement equipment.

The plan

This is how the plan was born to build a new digipeater. The major requirements for this new digipeater have already been mentioned. In addition, the digipeater should be completely configurable and should not have hard-coded APRS knowledge. All configuration parameters should be soft-coded in an initialization file. This is important because every station is different. There was one program, StealthDigi, that already worked that way. StealthDigi itself was however not reliable and could not do what we wanted our digipeater to do, but the basic idea was okay and was adopted for our project.

To keep the access to the hardware simple and not to reinvent the wheel, we decided to base the digipeater on TFPCX. TFPCX is a popular program in Europe and is used as software TNC for the cheap and simple modems, for SCC cards and for KISS TNCs. Except for sound-cards, the use of TFPCX would cover all popular packet hardware.

That is how I started to write the program. After some experiments, however, I found that TFPCX could not be used. An application on top of TFPCX is either the originator of a packet or the receiver, but not some station in between - there is no access to the digipeater list to allow application-controlled digipeating. So I took TFPCX apart and reused only the lowest layer. Advantage of this was that this part of the code was covered by the General Public License, whereas the rest of the code was copyrighted by ALAS - a more restricted public license.

The low-level TFPCX part became a simple resident program that sends and receives raw AX.25 frames. On transmission the driver adds the CRC and transmits it to whatever hardware is specified. On reception the driver verifies the

CRC and offers the frame to the application. The driver still supports all the hardware that TFPCX supported, and just like with TFPCX, 8 ports can be used simultaneously.

The application

After solving access to the hardware, the digipeater application was built. The name of the digipeater is “DIG1 NED”: “DIGI” because it is a digipeater and “NED” for “Nederland” which means “The Netherlands” in Dutch - showing some pride in our country.. .

The application runs under DOS, because of the 80286 constraint. The system can easily be put onto a floppy, saving the need for a hard disk. Although all the digipeating rules are soft-coded in the initialization file, users subsequently came up with scenarios and problems that were not covered at first. So capabilities to support those new requirements have been added.

I chose to release the digipeater as Open Source. As I said in the introduction, APRS is a fun mode here in Europe, so as many people as possible should be able to enjoy it. But beyond that, we ran into the problem that no suitable digipeater code was available to realize our ultimate goal, to build weather stations. So if other groups have similar ideas they now have access to the DIG1 NED code to build whatever they want for their project. A nice spin-off would be that we also might enhance our own solution with third-party designs in the future.

The driver part of DIG1 NED, which was taken from TFPCX, is available as Open Source too. That means that when you want to build some DOS program that needs low-level access to AX.25 and should support a lot of hardware, you can take it. Since it is a separate TSR you can even bundle this GPL product with non-GPL stuff.

Linux

During development I had the need to test the digipeater with multiple ports. I had already played around with the Linux AX.25 tools and libraries, so I decided to try to run DIG1_NED under Linux. After some study of sockets programming and looking at the solutions used in other AX.25 utilities (especially net2kiss) it turned out to be rather simple to add Linux support. The rest of the code ran first time after I stubbed some DOS specific calls. In fact, from that moment on I d.eveloped the product under Linux. Before each new release I test DIG1_ NED under DOS, and up to now it has always worked first time. I can now test the digipeater under Linux using internal loop-back links. This way I can control the runtime environment and do not transmit bogus packets on the air.

So what do we have now

What we have accomplished now (and I say “we” because although I wrote it, I have incorporated much user feedback) is a state-of-the-art digipeater. Because the PC has so much more power than a TNC it can do more than any TNC on the market. And remember that the development started only in February 2000, less than 6 months of spare time. By the time you read this I’m sure more features will have been added.

Capabilities of the current version include:

- Low hardware demand; for DOS a 80286 PC is more than enough. I’m almost sure that it will even run on a 8088 - it needs a recompile for this. For Linux, any Linux-capable PC will do. I have not had the opportunity to test this on other non-PC Linux machines (may work nevertheless).
- DOS & Linux capable:
  - AX25 MAC driver for DOS (included)
  - Uses I%ux AX.25 kernel interfaces

- Rule driven:
  - The only APRS knowledge in the digipeater code is APRS message handling. Behavior for APRS digipeating is completely defined by the digipeating rules.
- Normal digipeating:
  - RELAY, TRACE, WIDE
  - <Fill in your own, it's all configurable!>
- Intelligent digipeating:
  - TRACEn N, WIDEn N
  - Digipeating on SSID -
  - <Fill in your own, it's all configurable!>
- Call substitution:
  - <Fully configurable>
- Extremely configurable:
  - <Everything>
- Drives up to 8 ports with different types of hardware:
  - BayCom lk2, PicPar, Par96
  - OptoPcSCC (PAOHZP), PElPET SCC, BayCom USCC, DRSI, FSCC
  - YAM 9k6 modem
  - 😐 🖐 💧 💧
  - BPQ-Ethernet
  - All Linux devices that are supported by the kernel.
- Digipeats frames with AX.25, IP, ARP and NETROM PID.
- Duplicate checking:
  - Remembers source call and CRC of the payload
  - <History time configurable>
- Mheard list:
  - Keeps call, heard date and port
  - Mheard list size configurable
  - Query on port number
  - Query on call
- Preemptive digipeating:
  - Takes packets out-of-order
  - Keeps or removes intermediate calls
  - <Fully configurable>
- Support for "local" ports:
  - To fill in "black spots", ports can be defined as "local"
- Message capability:
  - Some predefined built-in messages
  - Response to standard ?ping?, ?aprs?, ?aprst, ?aprsd and ?aprsm queries
  - Fully configurable roll-your-own queries, make TinyWebPages!
- Blocking of unwanted calls:
  - NOCALL, NOCALL, MYCALL
  - <Define your own, fully configurable!>
- Remote exit for remote maintenance:
  - Exit the digipeater and start a program for uploading new versions; we use NetCHL for this.
  - Remote reboot of system
  - Ability to switch off these remote functions.
- Logging:
  - Log digipeater actions for debugging
- Availability:
  - DOS-executable package
  - Sources for DOS and Linux (same source)
  - General Public License

Conclusion

Undoubtedly I have forgotten to mention some of the features - there is so much that has gone into this program in such a short time. With DIG1 NED you will have the most sophisticated APRS digipeater available today. If you disagree let me know and I will add the missing feature. DIG1 NED was initially stable for 1% months in a row, then it was interrupted by an upgrade. New versions with added functionlity have since been coming out about every two weeks or so.

DIG1 NED is delivered with documented sample files and manual.

Are there disadvantages? Yes. First of all you need a PC, and even if it is a small and cheap one, it will always be physically bigger than a TNC. Another disadvantage is the flexibility, which on one hand makes it possible to do what you want it to do but also makes understanding the complete product not so easy. The sample files will provide a good start. If you want to use the product’s specific features you have to understand what the digipeater does and how it fits in your local situation.

References

- APRS is a registered trademark of Robert Bruninga, WB4APR, reference: httr,://web.usna.navy.mil/-bruningdaprs.html
- DIG1 NED can be found on the Web on the VrzAiDxw pages: http://www.homepaAes.hetnet.nl/-remko
- APRS Protocol Specification version 1 .Ol : http://www.tapr.org/tapr/html/Faprswg.html