An Inexpensive High Speed Modem for the Universal Serial Bus (USB)

Thomas Sailer, HB9JNX/AE4WA,
and
Johannes Kneip, DG3RBU

August 3, 1999

Abstract

This article describes a simple and inexpensive modem intended to link end users at 76.8kBit/s to the high speed backbone network. The modem can be connected to standard PC's using the Universal Serial Bus (USB).

1 Introduction

Last year, we presented a modem design for the Enhanced Parallel Port (EPP). Three years ago, when the basic design decisions for that project were made, the Enhanced Parallel Port was the only ubiquitous high speed interface available on PC's and Laptops.

In the mean time, a new high speed interface emerged: the Universal Serial Bus (USB). The new interface offers several advantages over the Enhanced Parallel Port:

- high transmission speed, 12MBit/s
- serial transmission, i.e. thinner and cheaper cables
- easier port sharing, up to 127 devices per port by using hubs
- rigorous specification

Furthermore, the necessary infrastructure like operating system drivers for popular operating systems like Windows98, Linux, FreeBSD and NetBSD is either in place or emerging at a rapid pace, and USB accessories like hubs can be bought at every computer store.

Therefore, we decided to develop a modem similar to the EPP design, but with an USB interface.

2 Design Goals

Our goal was to provide a flexible design that supports a wide variety of applications without sacrificing cost effectiveness.

Our design supports the following modes of operation:

1. FSK (G3RUH compatible) fkm9.6 kBit/s up to about 300 kBit/s

2. 1200 Baud AFSK

3. External Modem using standard Modem Disconnect interface [4]

4. Audio IO

1 covers the same applications as the EPP modem designs [7, 8]. 2 makes this design attractive for 1200 Baud users by offering a smooth upgrade path to higher speeds. 3 allows it to accomodate modems for nonstandard modulation formats, and 4 together with suitable application software offers the same features as a standard FM transceiver, but software controllable.

If the Modem Disconnect port is not used to connect to a modem, it may be used as eight general purpose PIO pins.

The circuit also has a port to control the operation of a T7F [10] transceiver. The T7F is a popular synthesizer 9600 Baud data radio design. An RSSI ADC makes the signal strength accessible to the PC, four digital lines allow the channel selection, one digital line selects 12.5kHz or 25kHz channel spacing, and a 1200 Baud asynchronous UART allows the channel memory to be programmed.

3 Design Considerations

Figure 1 shows the prototype circuit diagram. There will likely be changes for the production unit, so do not expect the final drivers to work with that circuit.

In order to shorten the development cycle, we decided to base the new design on the successful EPP design [7, 8]. On first sight, the circuit diagrams of both modems look very similar, but there are some important differences, as explained below.

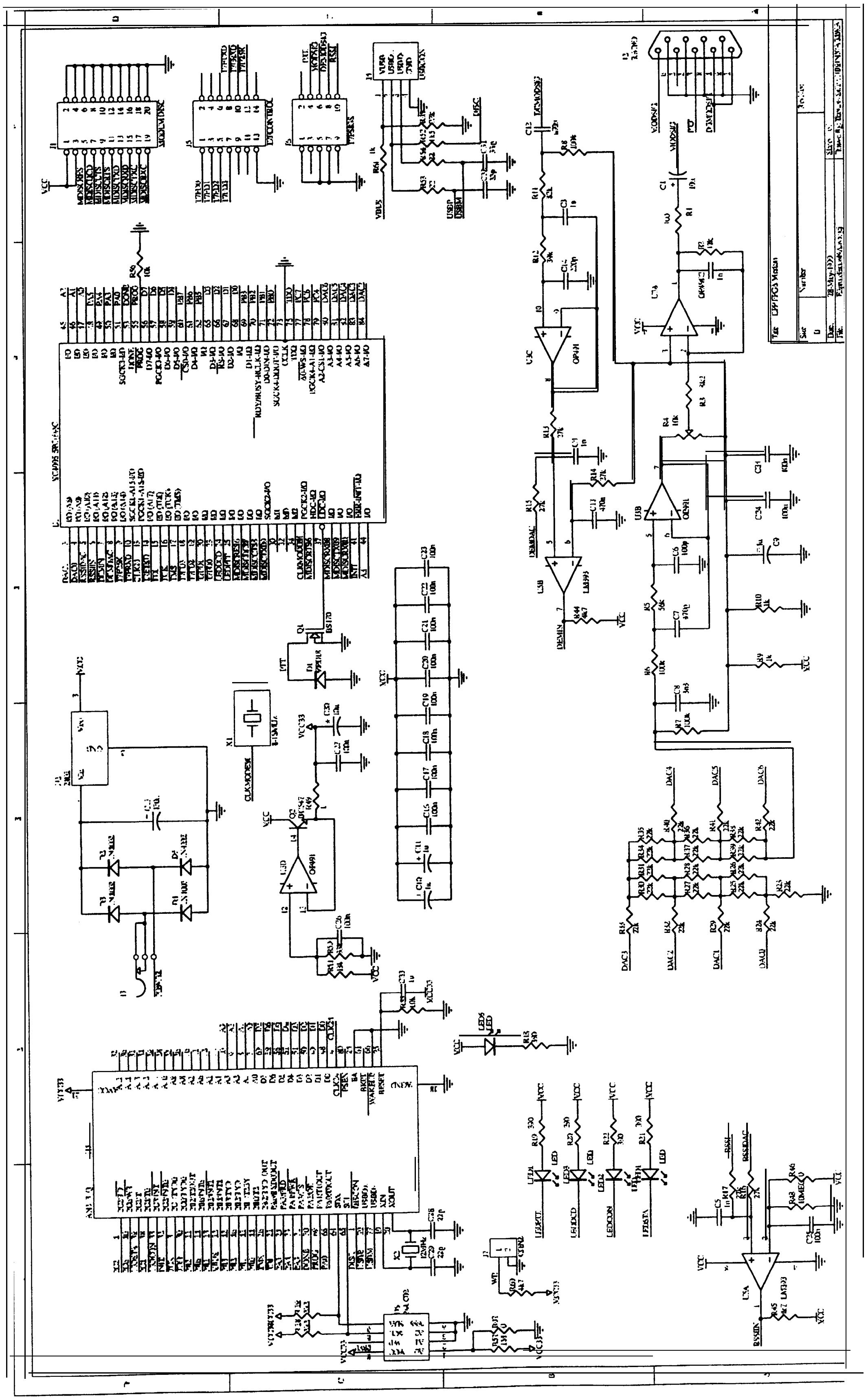
3.1 Microcontroller selection

The USB bus specification requires the devices to be able to respond to relatively complicated inquiries. This necessitates a microcontroller on the device and makes the implementation of an USB controller on a FPGA[1] infeasible. In April 1999, we therefore surveyed the market for suitable microcontrollers with built-in full speed (12 MBit/s) USB interface engines (table 1).

The AMD 186CC looks ideal for our purpose on first sight. It is however quite expensive, and since it doesn't contain any memory on chip, requires external FLASH and SRAM memories, making it even

“‘Field Programmable Gate Array”: programmable logic devices [1]

Figure 1: Prototype Circuit Diagram – see text



Manufacturer	Device	CPU core	features	problems
AMD	Am186CC	8086	HDLC	$
AnchorChips	AN2131	8051	8kB SRAM, no nonvolatile - -	slow microcontroller core
Cypress	7C64213, 7C64313	proprietary	8kB EPROM 256B SRAM, 1kB FIFO SRAM, master DMA into FIFO SRAM	not yet available, simple microcontroller
Infineon	SAB-C541U	8051	8kB ROM, 256B SRAM	very slow microcontroller core, not enough SRAM
Motorola	MPC823	PowerPC	HDLC, ethernet	$$, BGA package

Table 1: April 1999 market survey for microcontrollers with USB interface engine

more expensive. It looks however promising for a small node application with three radio channels and one USB channel. We already have a prototype design for that.

The AnchorChips part has 8k of on-chip SRAM. Its distinguishing feature however is that the complete firmware for the CPU can be downloaded from the USB bus. It doesn't require any nonvolatile program storage. Its disadvantage however is the relatively slow CPU core. It's faster than the original 8051, but access to external data and USB data is inefficient in the 8051 architecture. But overall the AnchorChips part is still a good choice for our purpose.

Unlike the EPP modem, the microcontroller already has on chip SRAM and the USB bus is designed for low latencies, an additional SRAM for buffering is not necessary.

3.2 Power Supply Considerations

USB delivers 5V over the standard four conductor wire. However, the specification allows huge tolerances to accomodate the resistive loss in the cabling which makes it impossible to directly power 5V devices without a DC/DC converter. It is however possible to feed a low drop 3.3V regulator.

Most USB controllers are 3.3V devices, the AnchorChips part including. It features 5V tolerant IO pins, too.

Since powering the analog circuitry with only 3.3V would result in output voltage swings too low for many transceivers in use, we decided to power the analog part with 5V, and to simplify the interface of the analog part to the FPGA we used a 5V FPGA. In fact, the analog circuitry is the same as in the EPP modem design [7, 8].

The USB interface may supply up to 500mA[2], which is approximately 2.5 Watts. Typical 9600 Bauds data radios have around 7 Watts of HF output power, therefore requiring about twice the supply power. So it is impossible to feed the radio from the USB port, and therefore at least the radio requires an external

[2] only if the device is connected to a self powered hub

power supply, typically 12V DC. Given that it does not make sense to use an expensive DC/DC converter to power the modem from the USB bus, so we decided to require an external supply for the modem too.

3.3 Firmware Download

In order to achieve the flexibility of the design outlined in the design goals section, the modem design employs two programmable devices, namely the Field Programmable Gate Array (FPGA) and the USB microcontroller. Apart from a small I^2C-EEPROM there is no nonvolatile storage in the modem. The EEPROM only stores USB vendor and product ID codes to make the design distinguishable from other AnchorChips designs.

To make the modem operational, the firmware for both programmable IC's needs to be downloaded from the host PC. The first step for the host PC is to download special code for the microcontroller whose only purpose it is to accept data from the USB bus and download it to the FPGA. Once the FPGA is configured, the host PC stops the microcontroller again and downloads the firmware for the particular operating mode requested.

The microcontroller then simulates the unplugging and replugging of the modem from the USB bus. This forces the operating system to reread the USB interface tables from the modem again. This allows the modem to have different USB requirements than the default ones provided by AnchorChips, and it also allows the modem to supply another product ID to force the operating system to load another driver.

3.4 The FSK Mode

Unlike the EPP modem design which encodes and decodes HDLC in the driver running on the host PC, the USB modem exchanges raw packets with the PC. Therefore, the modem has to implement the HDLC encoding and decoding process.

HDLC processing can be classified into two categories: Bit oriented tasks like (un)stuffing, CRC calculation and flag detection, and memory management oriented tasks, like storing a frame until the CRC is received and then determining whether to keep it or not. It turns out that the former can be done very easily on the FPGA, while FPGA's are quite unsuited to the latter.

The microcontroller communicates with the FPGA mainly through a register interface connected to its address and data bus. The interface consists of 16 8Bit registers aliasing over the whole external part of the XDATA address space of the CPU.

Small FIFOs decouple the HDLC encoder/decoder and the microcontroller. The FIFOs store eight bits of data and a one or two bit tag word indicating if the data is to be considered as "payload" data or "meta" data.

The transmit FIFO may be accessed at four addresses, which determines the two tag bits. The tag bits are listed in table 2.

In the receive case, the tag is only one bit wide and distinguishes between payload and meta data bytes. A meta data byte is inserted when a HDLC flag or abort is received. The data byte then contains indications whether a flag or abort was received, and in the former case whether the CRC was correct or not and the number of residual bits. Since processing a HDLC flag is an expensive operation (the old packet has to

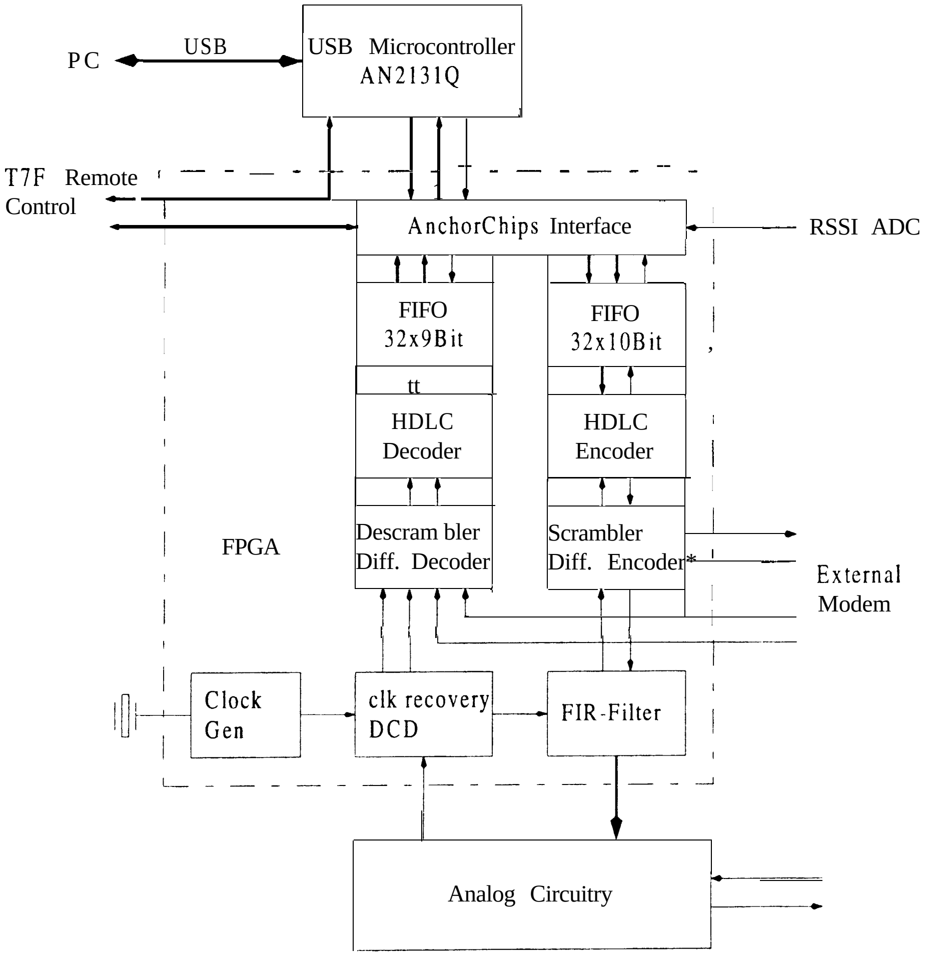


Figure 2: Diagram of the FSK mode

be either discarded or queued, and memory for a new one has to be allocated), the receive FIFO collapses multiple consecutive flags into a single flag.

The modem part of the FPGA is the same as in the EPP modem adapter, and a description may be found in [7, 8].

3.5 The AFSK Mode

Implementing FSK modems with FPGA's has been popular for quite some time, so there exist "competing" designs, most notably the YAM [11]. The flexibility of the FPGA's spurred interest in realising an AFSK modem with the same hardware, and YAM got its AFSK modem firmware about a year ago.

But because the analog circuitry of these modems was designed for the FSK case, the performance of the AFSK demodulators implemented on these modems is pretty unsatisfactory. The next section shows how a simple modification to the analog circuitry boosts AFSK performance tremendously.

Tag (Address)	Purpose
00 FSKTXDATA	Transmit payload data byte. The byte is fed through stuffing and CRC generator.
0 1 FSKTXCRC	Transmit 8 CRC bits. Data byte supplied is don't care.
1 0 FSKTXRAW	Transmit 8 raw bits, i.e. without feeding through stuffing and CRC generator. Can be used to transmit flags.
1 1 FSKTXRAWCLR	Like 10, additionally initializes the CRC register to all ones.

Table 2: HDLC encoder tags

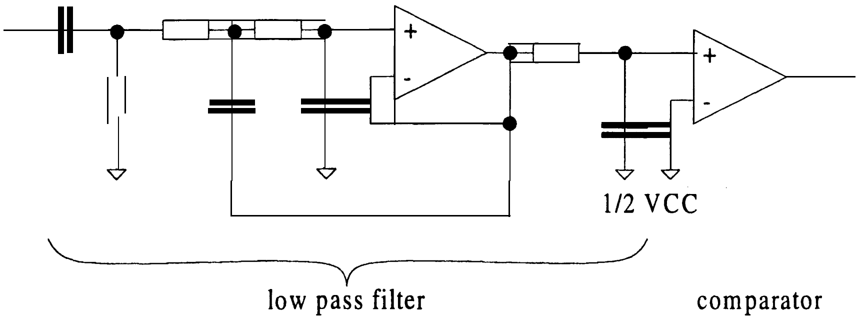


Figure 3: Standard FSK Modem Analog Circuitry

3.6 Modification of the analog circuitry for AFSK demodulation

Figure 3 shows a schematic circuit diagram of a standard analog circuitry of the receiver chain of a FSK modem. The comparator serves as the data slicer in the FSK case, but as a 1 bit A/D converter in the AFSK case. In literature, the signal to noise ratio of a B bit A/D converter is given by $(6B+1.25)$ dB. That formula however is based on the assumption that the quantisation error of two samples is uncorrelated, an assumption that is unrealistic for converters with a very low number of bits. This is also the reason why straight oversampling does not help.

So the AFSK demodulator has to work with a data stream that does not provide much information about the incoming signal. It therefore cannot make robust decisions.

3.6.1 $\Delta\Sigma$ A/D Converters

$\Delta\Sigma$ converters are very popular these days, even in quality audio A/D and D/A circuits, as they allow the construction of high quality converters with low precision analog components and some digital signal processing. Figure 4 shows a diagram of a $\Delta\Sigma$ A/D converter. The A/D and D/A converters do not need high precision, in fact they are often realised as 1 bit converters, as 1 bit converters cannot have a linearity error[3]. The only important aspect is that they are fast. The ratio of the sampling rate of the converters and

[3] it is always possible to perfectly fit a straight line through two points

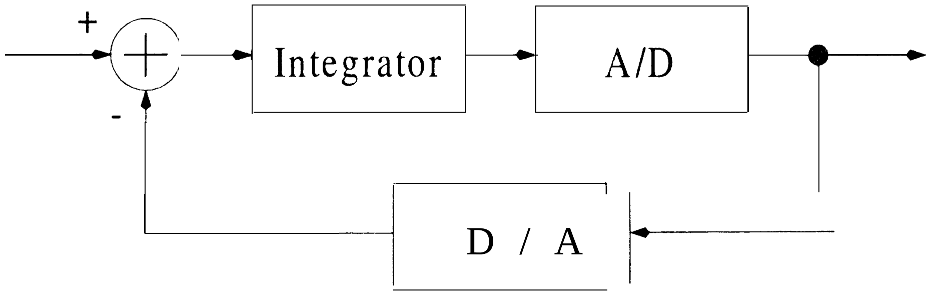


Figure 4: ΔΣ A/D converter diagram [12]

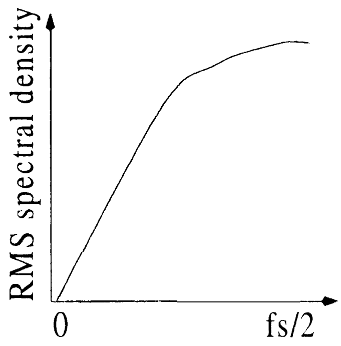


Figure 5: schematic plot of the quantisation noise

the highest signal frequency is called oversampling ratio.

The integrator's task is to change the spectrum of the quantisation noise and to decorrelate it. Instead of a flat noise spectrum, the integrator causes a reduction of the noise power at low frequencies and an increase in noise power at high frequencies. Figure 5 shows schematically the resulting quantisation noise spectrum. Since only low frequency signal components are of interest, a digital low pass filter removes the unwanted part of the spectrum and decimates the sampling rate to simplify the following circuits. The ΔΣ technique results in an improvement of 9dB or 1.5 bits per doubling of the oversampling ratio (using a first order loop [12]).

Comparing the diagrams of the ΔΣ A/D converter (figure 4) and the analog ciruitry of the FSK modem (figure 3) reveals that most components needed for a ΔΣ converter are already there. The comparator can serve as a 1 bit A/D converter, and the capacitor in front of the comparator can do the integrator's job. The only component missing is feedback part with the 1 bit D/A converter, which may be easily realised using a FPGA pin and a resistor (figure 6). In the FSK case, the FPGA pin may be programmed into high impedance state and therefore the resistor does not hurt.

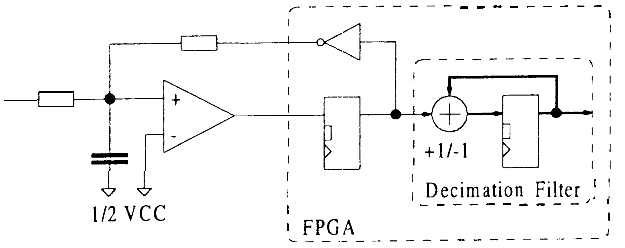


Figure 6: ΔΣ A/D Converter

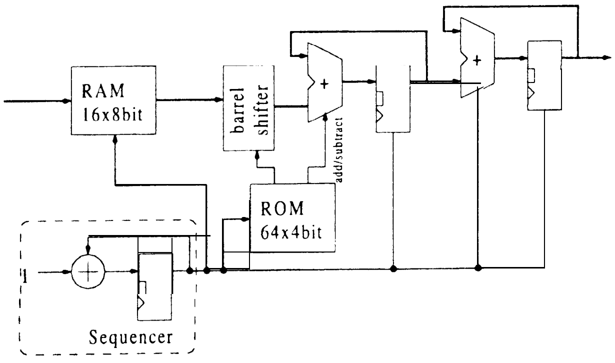


Figure 7: AFSK Demodulator

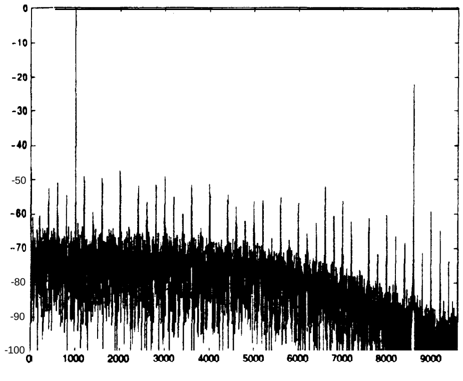


Figure 8: "Loopback spectrum"

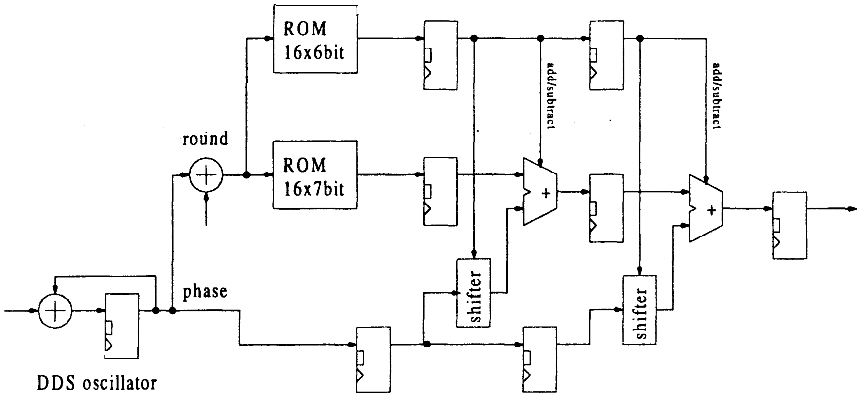


Figure 9: AFSK Modulator

3.6.2 Der ΔΣ-A/D-Wandler

Figure 6 shows a diagram of the A/D converter. The digital low pass decimation filter is realised as an "integrate and dump" or "boxcar filter". This filter has some strong sidelobes in the stop band, but these sidelobes are unproblematic and the filter is very easy to implement. Figure 8 shows the spectrum of the circuit in loopback configuration. The output signal of the A/D converter after the low pass filter was fed to the D/A converter in the transmit path. The circuit was excited with a 1kHz sine signal at the input, and the output was measured using a 16bit 48kSamples/s A/D converter. The signal was plotted using a 2^{16} point FFT.

3.7 The Demodulator

The demodulator block (figure 7) mainly computes four FIR filters. Simulation showed that the algorithm is very insensitive to quantisation errors of the FIR filter coefficients, therefore the coefficients have been coded with a sign bit and an exponent, replacing the multiplier by a barrel shifter. Both filter output value pairs are combined by a sum-of-squares approximation and the resulting values compared to form the bit decision.

3.8 The Modulator

The mdoulator (figure 9) implements a sine generator. Because can only implement ROMs with very few address lines efficiently, the difficulty of this task lies in producing a good sine with only small tables. The tit approximates $\sin(x+\Delta x) \approx \sin x + \Delta x \cos x$.

3.9 Microcontroller Interface

Since the number of gates for the AFSK modem is significantly bigger than for the FSK modem and the data rates are very low, the HDLC encoder and decoder have been moved to the microcontroller firmware. The FPGA only provides a doubly buffered shift register for the raw bits.

4 The Analog IO mode

Occasionally, it may be useful to operate the data radio as a normal voice FM transceiver. It is therefore useful to provide a mode where the transceiver baseband signal is sampled using the techniques presented in the previous section and forwarded to the host PC. The host PC then adapts sampling rates and forwards the signal to an installed soundcard for playback on a speaker. The reverse direction is similar. USB supports this mode of operation by offering isochronous transfers, which provide guaranteed bandwidth and latency, but do not correct errors by retransmission.

5 State of the Project

Prototype hardware is working since end of may, and beta firmware is working since end of june. The current prototype was designed and built in a hurry to get hardware fast to be able to start the software/FPGA firmware development. Therefore we will have a second and hopefully last prototype generation soon.

A user mode application exists that is able to transmit and receive frames and can be (cross)compiled for Linux and Windows. Under Linux it uses a small driver that exports USB host controller driver functionality to user mode using an ioctl interface, and under Windows it uses a generic driver provided by AnchorChips which does a similar thing.

Host operating system drivers do not exist yet. Under Linux, the USB host controller driver API is still emerging and not fixed yet, and under Windows the FlexNet AX25 stack which we will be using is also undergoing some significant changes at the moment. We expect to have the drivers ready by fall 1999. The source code of the Linux driver will be available under the GNU GPL license in case anyone wants to port it to an operating system/AX25 stack we will not be directly supporting.

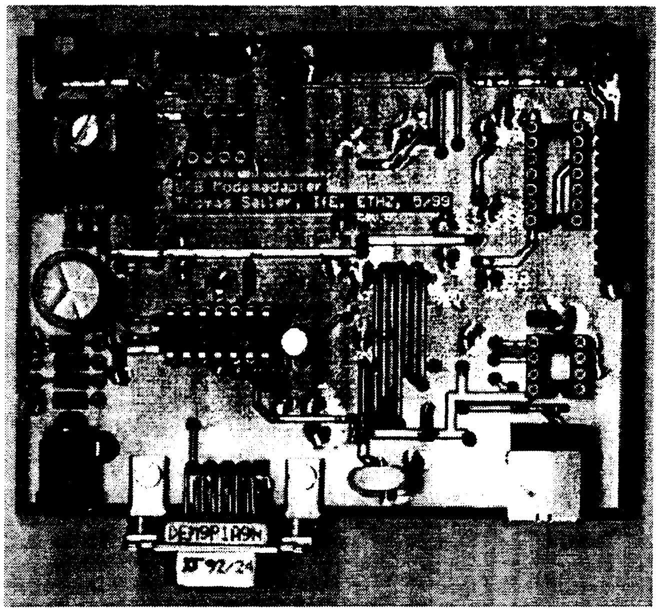


Figure 10: Top side of the prototype PCB

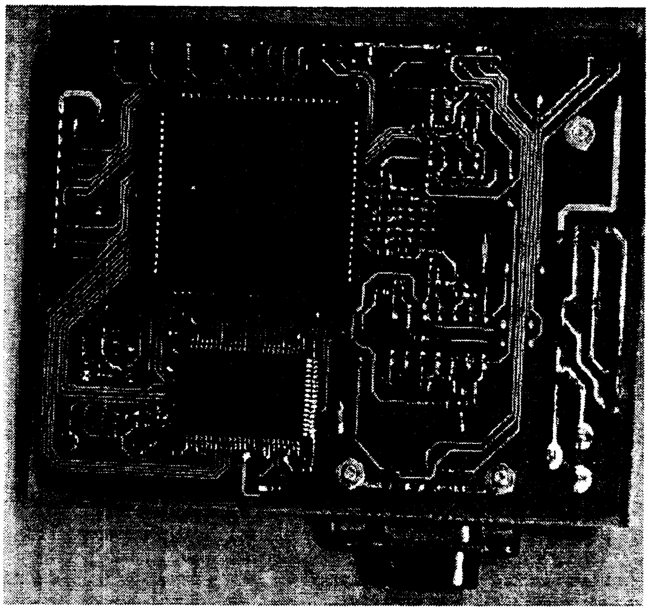


Figure 11: Bottom side of the prototype PCB

6 Conclusion

This paper presented a highly versatile amateur radio modem design that connects to a standard PC using the now ubiquitous Universal Serial Bus (USB).

The prototype was presented to the public at the Friedrichshafen, Germany Ham fair and generated significant interest from the amateur community.

The modem should be available soon from Baycom [5].

7 Outlook

Providing the interfaces to remotely controlling a transceiver is only the first step. We will be looking into combining modem and transceiver.

The hot spot of current amateur FSK transceivers is the modulator. Quartz oscillator transceivers often have deficiencies in the linearity of their voltage/frequency curve because they use a capacitance diode which has a very nonlinear voltage/capacitance curve. Synthesizer transceivers have a PLL with a loop filter whose cutoff frequency often lies in the middle of the signal band, requiring the injection of the modulation signal to the VCO and the reference oscillator. This method causes eye distortion even when the circuit was well trimmed.

Using a high bandwidth PLL and modulating it by modulating the divider of the prescaler using $\Delta\Sigma$ techniques looks like a simple yet promising technique to improve the modulator signal.

References

[1] Xilinx Corporation *The Programmable Logic Data Book*
`http://www.xilinx.com`

[2] Wolf-Henning Rech, DF9IC, Johannes Kneip, DG3RBU, Gunter Jost, DK7WJ, und Thomas Sailer, HB9JNX, *Ein Modemadapter für den EPP*, 41. Weinheimer UKW-Tagung, Skriptum, 1996

[3] Johannes Kneip, DG3RBU, *Das FSK+-Modem mit Echoduplex*, Adacom Magazin 10, 1997

[4] Wolf-Henning Rech, DF9IC, *Modernes FSK-Modem – kompatibel zum Standard nach G3RUH*, Adacom Magazin 2, 1991

[5] Baycom ◇ Hard- und Software GmbH, Bert-Brecht-Weg 28, D-30890 Barsinghausen, phone ++49 (5105) 585 050, fax ++49 (5105) 585 060, `http://www.baycom.org/`,
Email: `johannes@baycom.org`

[6] Martin Liebeck, DL2ZBN, und Alexander Kurpiers, DL8AAU,
Hochgeschwindigkeits-Packet-Radio – Baugruppen für das 70cm Band, Adacom Magazin 10, 1997

[7] Thomas Sailer, HB9JNX/AE4WA, and Johannes Kneip, DG3RBU, *An Inexpensive PC-Modem for 76.8kBit/s User Access*, 1998 ARRL and TAPR Digital Communications Conference, Chicago, Illinois

[8] Thomas Sailer, HB9JNX/AE4WA, und Johannes Kneip, DG3RBU, *Alternative Implementierung des EPP Adapters*, 17. Internationale Packet Radio Tagung, Darmstadt, 1998

[9] Thomas Sailer, HB9JNX/AE4WA, und Johannes Kneip, DG3RBU, *EPPFLEX – ein universeller Modemadapter für den EPP-Port*, 18. Internationale Packet Radio Tagung, Darmstadt, 1999

[10] Holger Eckardt, DF2FQ, *Handbuch zum 70cm-FM/FSK-Transceiver T7F*

[11] Nico Palermo, IV3NWV, *Yet Another 9k6 Modem*
`http://jupiter.web-hosting.com/~nicopal/yam/`

[12] James C Candy, and Gabor C Temes, *Oversampling Methods for A/D and D/A Conversion*, IEEE Circuits and System Society, 1992, ISBNO-87942-285-8