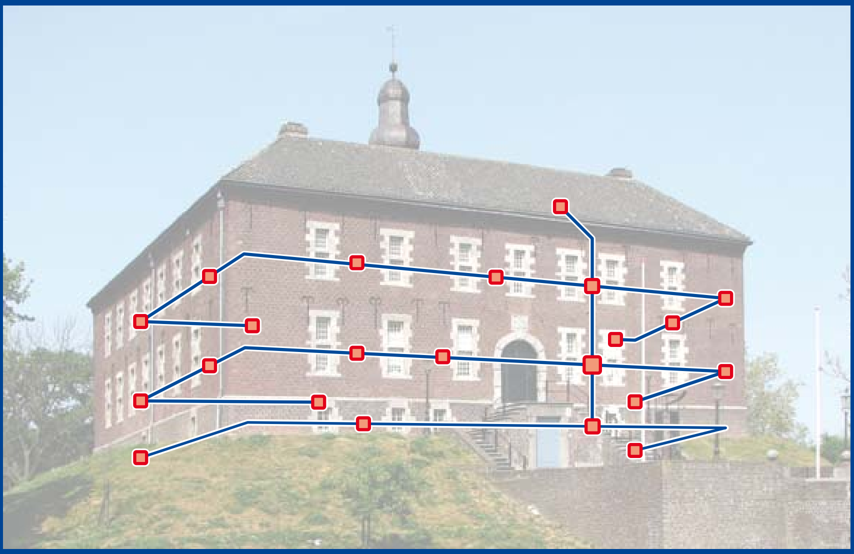
Here comes the Bus!



By Jens Nickel

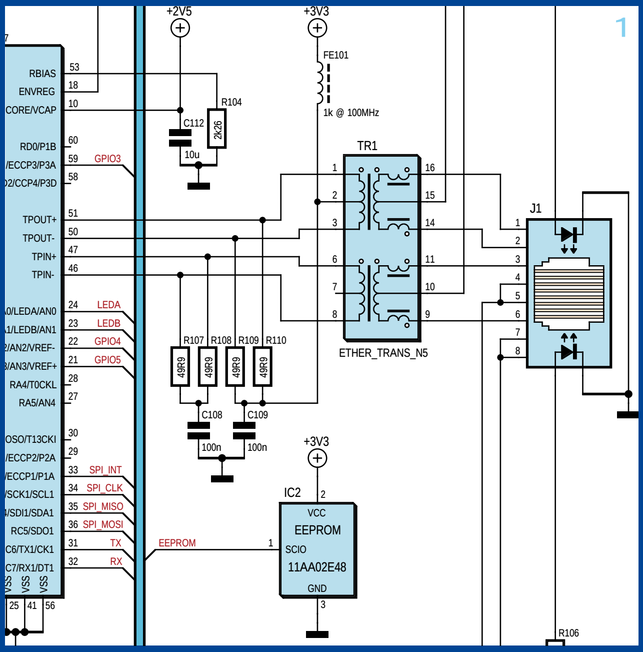
The best things in life may or may not be free (in the *Elektor* labs we know a song about that), but sometimes there just isn’t time to get round to them. Lots of good ideas come out of our editorial planning meetings, but many just end up languishing in the bottom of a drawer never to see the light of day.
‘E-Labs Inside’ to the rescue! As the chief lab correspondent, I often find myself looking around at what my colleagues are doing. Sometimes this can be a source of ideas, and on this occasion I was inspired to dust off an old project that had never come to fruition, and use it as an example to show how a concept is developed from scribbles on the back of an envelope to reality.

After a bit of thinking I narrowed the choice down to two large-scale project ideas, and went to consult a couple of colleagues: Luc and Chris from the lab and Clemens from the international editorial team, who has already seen several large projects through to print début.
The first idea was a general-purpose microcontroller base board, into which a variety of processor boards could be plugged. Perhaps we could include AVRs, PICs and 8051-series processors? The second idea was to develop our own bus system for interconnecting sensors, actuators, instruments and anything else we could think of.

We soon came to the conclusion that ‘ElektorBus’ would be the better choice for illustrating the development process in the ‘E-Labs Inside’ series, and so the ‘Elektor Controller Platform’ went back in the drawer until next time.
The ElektorBus should, we decided, be able to carry rapidly-sampled sensor measurements as well as control signals, and be suitable for use in a small home automation system. We would be able to add new types of nodes gradually, the whole thing would be easy to build and, of course, the software would be open source down to its last bit.

A few details were nailed down at the first brainstorming session with Clemens and Chris. We wanted to keep things as simple as we could, and so for the lowest protocol layer (the purely electrical specification) we decided to use an established standard and readily-available components. Clemens suggested using Ethernet, and I proposed a bus based on the RS-485 standard. Either would be able to work with cable runs of tens of metres, enough for wiring up the elegant castle that is the *Elektor* nerve centre.

There are fairly cheap microcontrollers available with a built-in 10Base-T Ethernet transceiver, giving a data rate of up to 10 Mbit/s (see for example the ‘NetWorker’ project in the last edition [1]; circuit diagram snippet in **Figure 1**). RS-485 buses can in principle achieve similar data rates. With a 10 Mbit/s link we have plenty of bandwidth: even enough to send high-quality sampled audio data in real time.
It was clear, however, that not every network node would need to be capable of top-speed performance. To make the project interesting to as many readers as possible, we decided that the components for a minimal node, perhaps for connecting up a simple sensor, shouldn’t come to more than fifteen euros. That seemed feasible, as RS-485 transceivers (such as the LTC1535, or the SN65HVD08P, which Clemens had used in the InterSceptre project [2]) are available for about five pounds. And, if we make the simple bus node run at ordinary UART data rates up to say 115.2 kbit/s, we can use an ordinary AVR microcontroller for its brains.



RS-485 thus looked like the ideal option, and, as many readers will know, is the electrical standard on which many diverse bus systems are based. But still Ethernet, with its highly-developed and sophisticated protocol stack and, significantly, enormous open-source code base, seemed too attractive to discard.

Clemens had an idea: why not use both? Previously we described a USB-to-RS-485 adaptor that uses a standard Ethernet cable with four twisted pairs and RJ45 plugs at each end to carry RS-485 signals (**Figure 2**) [3]. RS-485 needs just two data wires, and 10Base-T Ethernet uses just two of the four pairs in the cable. Can we carry RS-485 and Ethernet in the same cable? Could we even use the last remaining pair to supply power to the nodes?

There is always a catch, and in this case it was the fact that modern Ethernet networks are arranged in a star topology. This means that if we want to wire up a few sensors and actuators in a single room, we end up with a spaghetti of cables. Many home automation networks have a hierarchical structure where Ethernet is used between the individual rooms of the house, but the wiring between the nodes within each room uses a two-wire bus.

Furthermore, as Chris pointed out, our combi-cable would be incompatible with existing (mostly 100Base-T) Ethernet equipment. Often in modern devices the spare pairs in the Ethernet cable are grounded (see for example **Figure 3**) [4,5]. Clemens started thinking about how to make best use of ready-made cables in practice. For example, it is not possible to thread them through a cable gland, so it is not easy to see how to make a waterproof node for use outside or in the bathroom. This problem hadn't even occurred to me, so it was a good thing that there were three heads brainstorming rather than one!

We decided to leave out of our first specification any details that we had not settled on. We would definitely use standard Ethernet cable with four twisted pairs, with one pair for our bus and one pair for 12 V and ground. (Clemens thought 12 V was the best choice for most applications, as it gives some headroom when powering 9 V or 5 V equipment.) The remaining pairs we would reserve for future expansion, or perhaps an enterprising reader might come up with a clever use for them. Of course anyone making such 'unauthorised' modifications would have to forfeit the right to use our neat ElektorBus logo on their devices...

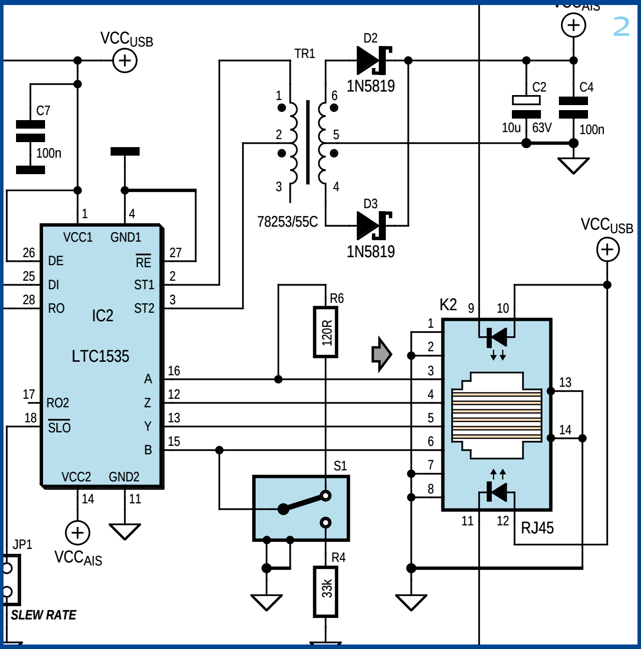
What bit rate should the RS-485 bus operate at? If we are using just one twisted pair then it is simplest if all communications happen at the same speed. We decided to take things gently and first define an 'ElektorBus low speed profile' with a standard bit rate of 9600 bit/s. This will allow us to use standard crystals and probably also experimental hardware and software that we already have. When that is working we will then venture on to define 'standard speed' (115.2 kbit/s) and perhaps also 'high speed' (around 5 Mbit/s) profiles.

To keep communications at different speeds separate we could later make use of the spare pairs in the cable. Clemens was also considering how he might arrange for different speed communications to run on a single conductor pair. The transmitter could transmit away, and the receiving node would have to determine from a preamble sequence whether the message was being sent at a speed it could read. Would the higher bit rates have to be multiples of the slowest, he wondered? It wasn't long before we were deep in grisly protocol details...

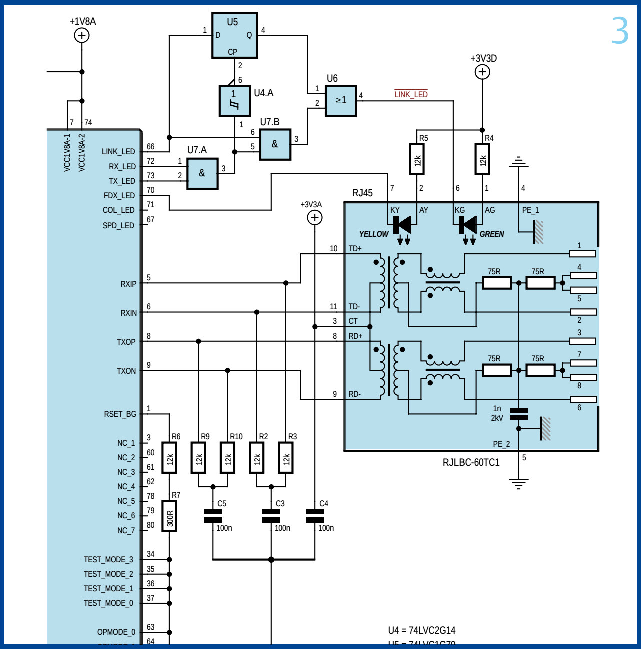
What do you think? You can help inspire the design: the ElektorBus team welcomes your ideas, improvements and thoughts. Send them by email to editor@elektor.com.

(100817)

[1] www.elektor.com/100552
[2] www.elektor.com/100174
[3] www.elektor.com/100372
[4] www.elektor.com/090607
[5] http://en.wikipedia.org/wiki/Ethernet_over_twisted_pair



2
VCCUSB
C7
100n
TR1
D2
1N5819
C2
10u 63V
C4
100n
D3
1N5819
78253/55C
VCCAIS
IC2
LTC1535
R6
120R
K2
RJ45
S1
R4
33k
JP1
SLEW RATE



3
+1V8A
U5
U4.A
U6
U7.A
U7.B
LINK_LED
+3V3D
+3V3A
RJ45
YELLOW
GREEN
75R
1n 2kV
RJLBC-60TC1
R5
R4
R6
R9
R10
R2
R3
R7
300R
C5
C3
C4
100n
U4 = 74LVC2G14