

Re: DTMF dead?

Source: <http://sci.tech--archive.net/Archive/sci.electronics.design/2007-07/msg01589.html>

- *From:* "slebetman@xxxxxxxx" <slebetman@xxxxxxxx>
 - *Date:* Wed, 11 Jul 2007 19:38:42 -0700
-

On Jul 12, 2:08 am, n...@xxxxxxxx (Nico Coesel) wrote:

Rich Grise <r...@xxxxxxxx> wrote:

On Tue, 10 Jul 2007 22:56:39 +0000, Nico Coesel wrote:

"Joel Kolstad" <JKolstad71HatesS...@xxxxxxxx> wrote:

"Nico Coesel" <n...@xxxxxxxx> wrote
in message
news:469287fd.52824567@xxxxxxxxxxxxxxxxxxxx

That is a very valid
argument. But there many
reasons (wiring,
connectors, transformers,
code size, etc) why ethernet
doesn't fit in
this application.

You can still use a PoE controller IC even if
you don't have "real" Ethernet
anywhere... they're really just your standard
switching converter ICs with a
simple "protocol" thrown on top to let the
controller "handshake" with the
power provider system.

But that still seems like an overkill for the project at hand.
There
is an ethernet connected device acting as a bus master. Using
internet
technology also comes with security issues so SSL or a
similar
encrypted tunneling technique is required.

Re: DTMF dead?

A proprietary bus is not easy to hack or interface (lets say it takes more than a laptop with ethereal) and thus requires less security measures. SSL alone takes about 90kB of flash and some processing power.

Earlier in the thread, you were talking about DTMF. Now you're all the way to SSL? What are you trying to accomplish?

I'm (co) working on a system that is primarily based on ethernet / IP protocol. An antire system can consist of several thousand units (all kinds of devices) in total. Some of the units need to be really cheap and small to be economically viable.

For a particular type of unit there is a severe space constraint. So there is a trade-off to take ethernet all the way to the smallest units or place a master ethernet<->proprietary converter unit at a convenient location and feed the smaller units with a proprietary protocol.

Having to use SSL or not is just one of the trade-offs to choose between standard ethernet or going proprietary.

How about doing both! Implement a proprietary ethernet protocol. You can then re-use your ethernet network equipment and simplify cabling etc. The protocol can be as simple as insisting that your raw data packet fits in a single ethernet frame. Use MAC addresses directly as your node address and your embedded side doesn't even need a "proper" protocol stack. Something like Microchip's ENC28J60 is nice since it handles a lot of stuff for you.

The down side of this is that on the host side on the PC you need to either write a protocol driver for your proprietary protocol or write raw ethernet frames directly from your application — both are not so trivial (at least not for me).

.