



## Elzet80

There are many competing fieldbusses and BITBUS is not the one that has the support of the big business. Here is why ELZET 80 uses BITBUS and encourages its customers to follow:

### We do not need an i/o bus!

Our philosophy is to distribute decision points as far as possible into the field. Contrary to a centralized point of view, our general approach is to have dedicated controllers all over the place that do not rely on a master to maintain their function. We then network these controllers on a level where they merely exchange requests and answers. Speed requirements are moderate at this level, resulting in low cabling cost and large area coverage.

High speed networked i/o (so called sensor-actuator-busses) is mandatory for the way conventional programmable controllers (PLC/SPS) solve a problem. When using real programming languages, it's much simpler - and much safer - to isolate problems and put their solution to where the problem is, i.e. put the regulator close to the unit that needs to be controlled. BITBUS is optimized for distributed control and the mCAT kernel supports this idea with its dedication to message passing.

A sensor-actuator-bus makes sense one level below in the system hierarchy, but it must be cheap enough to connect single sensors (while the currently favored busses use i/o-boxes). For this reason, we will support AS-i for low level i/o in the future. AS-i carries supply voltage and information on two wires and there are many cheap switches and initiators available. Insulation displacement connectors make wiring very easy with this bus.

### Where is competition?

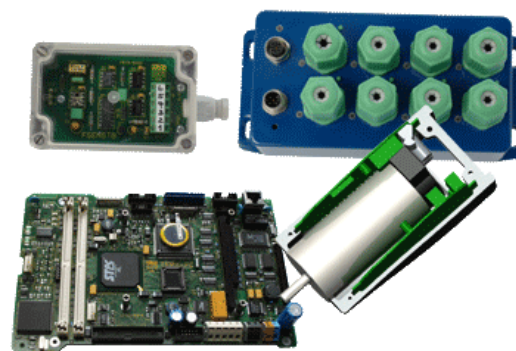
Profibus has been a second german attempt to standardize a fieldbus after PDV bus (DIN19241) failed. Being loaded with functions, it proved difficult for developers to support more than a marginal subset of the features. The concepts of object directories and application domains are not easily understood; master passing might have its advantages for esoteric applications but it makes bus handling complex and the timing indetermined. The main momentum behind Profibus has (for good reasons?) meanwhile moved to Profibus-DP, a mere i/o-bus, renaming the old Profibus to Profibus-FMS. There are some other national fieldbusses like the french FIP that have a certain user base but then there is the argument of international acceptance that lets us favor BITBUS.

### Technically better!

BBITBUS can cover large areas: 300m at high speed, 1.2km at 62.5kBit/s. Unlike others, BITBUS can use repeaters to multiply cable lengths.

BITBUS can easily be isolated. Standard optocouplers (VDE0884 available) are used and there is no line length penalty like with busses based on bitwise arbitration.

BITBUS uses SDLC, a message based protocol with automatic error detection. Standard serial controllers like the 85C30 do the SDLC



protocol handling in hardware while character based protocols (simple asynchronous transmission) are "unsafe at any speed". BITBUS uses a single twisted pair plus ground for easy cabling (compare this to fiber optics). The RS485 standard is widely recognized for its noise immunity. BITBUS uses NRZI encoding: clock is transmitted with the data and a polarity needs not to be obeyed with the BITBUS wire pair.

A faulty station or one without energy does not block the bus (compare most i/o busses).

## **BITBUS is simple!**

Easy cabling and the simple master/slave architecture opens up BITBUS to people that don't want to get tangled in a network but rather apply it to their tasks. The emphasis of BITBUS is not on fancy features but on basic functionality and reliability. Concepts and use of BITBUS can easily be presented to your service and sales force resulting in high general acceptance.

## **These advantages have led to an increasing use of BITBUS by ELZET 80 customers and to more and more ELZET 80 products that can be networked by BITBUS.**

Intel's withdrawal of the first generation i8044 single chip BITBUS controller used in many of ELZET 80's early BITBUS products forced us to look for alternatives. As BITBUS is an open protocol, it proved relatively easy to implement the protocol using a modern TLCS900 16-Bit microcontroller and the Z16C32 IUSC serial controller with SDLC capability. This combo has many advantages over the i8044: the processing power of a new 16-Bit controller, no limit on message lengths, low power CMOS technology.

There is more choice to implement BITBUS, starting with the 8044-successor i80C152 over MC68302 to the MPC860 Power PC. We like the TLCS900, however, for its price-/performance-ratio and Z80 upward compatibility.

BITBUS is dedicated to distributed control, not distributed i/o, and therefore expects a real time kernel as application layer in each slave. Fortunately Volker Goller wrote a modern real time kernel for the TLCS900. His message based mCAT, now in its second generation, conforms perfectly to the requirements of BITBUS and adds state-of-the-art support for the real-time C-programmer.

## **BITBUS applications are:**

- Plant data control  
Exchange of machine status, operation times, number of parts produced.  
Download of product specifications.
- Building automation  
Send and get room or floor status from host to slaves. Control lighting and climate
- Transport and conveyor belt control Detect product markings and shift switches accordingly.
- Data logging for quality control  
Report product data (weight, tolerance etc.) to host for statistical analysis.
- Inventory booking terminals  
Control parts going in and out of storage.
- Distributed staff information  
Make order data, quality status and messages available on graphic/character LCD throughout a factory.
- Data loggers  
Collect process or environmental data at a slave, store and display locally.  
Make current or concentrated stored data available to the host.
- Motion control  
Using BITBUS based servo amplifiers, move robots and control tooling machines.

BITBUS is not well suited as i/o-bus, i.e. to connect single sensors or valves.

