The Fast Lane to EIB

The EIB Handbook Series

The EIB System
for
Home & Building Electronics

Marc Goossens - EIBA s.c. - Brussels, Belgium
February 1998

The EIB Association - Brussels, 1998

Contents

Preface

EIB concentrates unequivocally on home and/or building management. This focus permits it to deal with all tasks and challenges within this domain thoroughly and efficiently.

The European Installation Bus (EIB) is an open, comprehensive system which covers all aspects of Building Automation. It is managed by the neutral EIB Association.

Though standardised *Bus Access Unit* (BAU) building blocks are available from several vendors, EIB is in the 0^{th} approximation a *specification*, not an implementation (like a chip or a transceiver). This means EIB is *open*: EIB may be implemented by anyone, on any chip or processor platform chosen - both as proprietary implementation for individual products, as well as for OEM BAU s. Conformity tests are defined, and EIB Certification is open to all members of the Association.

Why in 0^{th} approximation ? Because EIB embeds the protocol in an encompassing Home and Building Electronics *System*, with standardised system components (such as the BAU s), network management and interworking standards, with a vendor-neutral tools and programming interfaces for PC s, training for electrical contractors, certifications schemes etc.

Network Topology

EIB is a fully peer-to-peer network, which accommodates up to 65 536 devices. The logical topology allows 256 devices on one *line*. As shown in Fig. 1, 15 lines may be grouped together with a *main line* into an *area*. An entire domain is formed by 15 areas together with a *backbone line*. On open media, nearby domains are logically separated with a 16-bit SystemID. Without the addresses reserved for couplers, (255 x 16) x 15 + 255 = 61 455 end devices may be joined by an EIB network. Installation restrictions may depend on implementation (medium, transceiver types, power supply capacity) and environmental (electromagnetic noise,   ) factors. Installation and product guidelines should be taken into account.

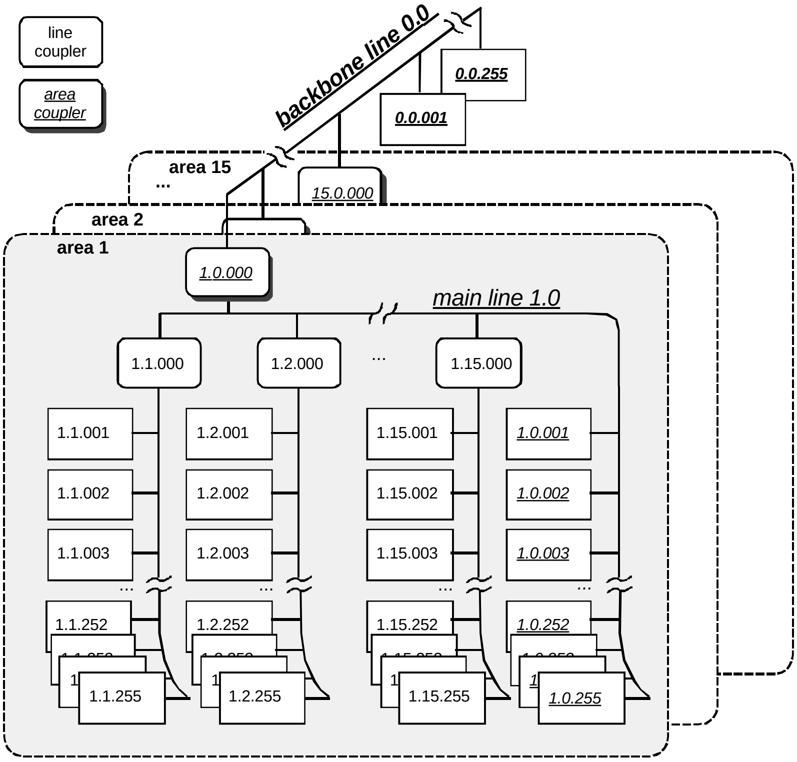

Fig. 1 The logical topology of EIB

Couplers connect lines or segments, e.g. within the Twisted Pair (TP) medium, or different media; their functionality may be (some combination of) repeater, bridge, router, package filter (for traffic optimisation), firewall protection etc. EIB defines various standard coupler profiles.

Media

As indicated, EIB Medium Access Control is highly optimised for each medium individually. Available implementations further optimise for a combination of transceiver performance and cost. EIB.IR (*Infrared)* is currently being developed. EIB.MMS will extend EIB with dedicated MultiMedia Services.

EIB.*TP* - Twisted Pair

On EIB TP (Twisted Pair), bit-level collision detection with dominant logical 0 ensures that in case of collision, the transmission always succeeds for one of the communication partners. The resulting elimination of re-transmissions further enhances the performance of EIB TP. Together with EIB s powerful group addressing, EIB TP1 Collision Avoidance caters for extreme efficiency with reaction times 100 ms for two simultaneous transmissions. Fast polling allows up to 14 devices to be polled for 1 byte status-information within 50 ms. A physical TP segment may be up to 1 000 m long.

EIB.*PL* - Powerline

EIB PL (Powerline) uses a novel Spread Frequency Shift Keying modulation technique. With a corresponding numerical matched filter, the available BAU s guarantee adequate communication for group addressing to work reliably on PL. Medium access is controlled via a preamble sequence, with randomised re-transmission slots.

Maximum distance between 2 devices (without repeater): 600 m. (Communication is influenced by electromagnetic pollution conditions in the installation.)

EIB.*RF* - Radio Frequency

EIB RF (Radio Frequency) lines are physically separated by a different carrier frequency. In free field conditions, the transmission distance is about 300 m. Retransmission ensures that large volumes can also be covered inside the building. Retransmitter functionality is optimally distributed among the installed devices by the system itself.

EIB.*net* - Automation Networking

The EIB.*net* specification realizes EIB on all media with a logical link layer according to ISO/IEC 802-2, including 10 Mbit Ethernet. Not limited to high-speed backbones, EIB.*net* also allows management or automation level devices to be directly connected.

With EIB.*net i ,* EIB.*net* becomes routable across existing office and building networks - or even remotely through the Internet - employing the Internet Protocol (IP).

The EIB OSI Communication Protocol

Fig. 2 shows how the EIB communication stack is structured according to the OSI 7-layer model. This is also reflected in the frame structure shown in Fig. 3. The physical layer and link layer obviously depend on the characteristics of the physical medium. For medium access control, EIB prescribes Carrier Sense Multiple Access (CSMA) with Optimised Collision Avoidance. As explained in section 2, EIB may be termed opportunistic here, in that the precise mechanism may be highly optimised for the particular medium. The Destination Address Flag (DAF) distinguishes between Group and Device Oriented telegrams.

Through the Network Protocol Control Information (NPCI), the Network Layer controls the hop count; for devices other than routers or bridges, it is trivial. The Transport Layer manages logical communication relationships, which can be:

1. one-to-many connectionless (group multicast)
2. one-to-all connectionless (broadcast)
3. one-to-one connectionless
4. one-to-one connection-oriented

It provides the mapping between addresses and an abstract internal representation, the Communication_Reference_ID (cr_id).

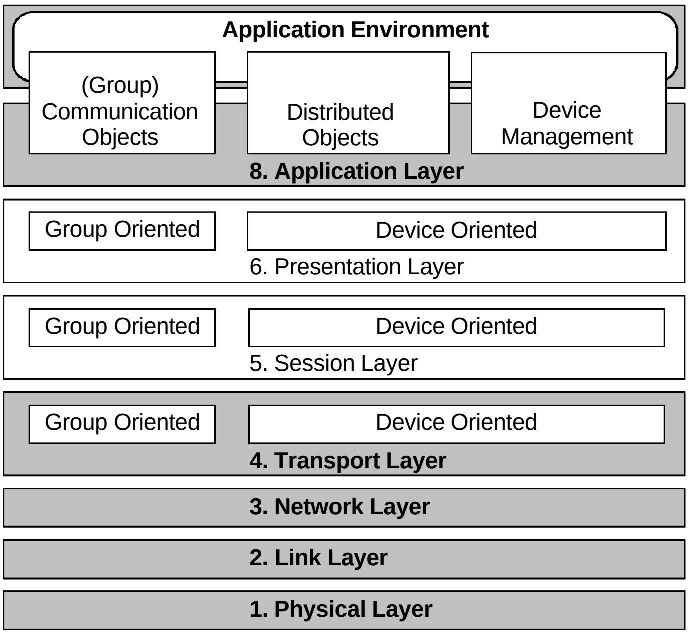

Fig. 2 EIB Communication according to the OSI reference model

All services are mapped transparently across Session and Presentation Layer, which are reserved. Application Layer implements the API for client/server management of EIB networks (see section 6.1). The Group Application Layer deals with the assignment of a group cr_id to a local instance of a Group Communication Object (or shared variable), for receiving (one-to-n) and for sending (one-to-one). For convenience, both Group Communication Objects and Distributed Objects are encapsulated by the EIB User Layer, which takes over the Application Layer nitty-gritty from the application. The User Layer acts also as default management server application.

octet 0	1	2	3	4	5	6		7	8	..	N - 1	N ≤ 22
Control Field	Source Address		Destination Address		DAF; NPCI; length	TP CI	AP CI	data /AP CI	data			Check Octet

Fig. 3 EIB PDU frame structure (long frames allow N < 255).

EIB Protocol Data Unit (PDU) frames may carry application data formats of up to 14 bytes (extension to 230 is currently under consideration). (See also section 6.)

In the next section, we will discover the central importance of the dedicated group-oriented facilities of the EIB Operating System.

EIB Network Management and Addressing

Network Management

To manage network resources (e.g. when configuring an installation), EIB uses a combination of broadcast and point-to-point communication.

Via broadcast (optionally using a device s unique serial number), each device in the installation is assigned a unique Physical Address, which is used from then on for further point-to-point communication.

- A connection (optionally with access authorisation) may be built up, for example to download the complete applet binary image of an application program.
- Connectionless access is possible to EIB Distributed Objects through *device . object . property* addressing, as a native EIB management-level mechanism for status visualisation and control.
- A dedicated master-slave *fast polling* mode ensures live- and status check of critical subsystems.

Group Addressing for Run-time Efficiency

EIB supports full multicast (group) addressing. *Full* means that:

1. EIB is not limited to grouping devices: each device may publish several variables (known as (Group) Communication Objects) individually, which can be grouped independently from one another into network-wide shared variables. As a bonus, properties of Distributed Objects may be published as shared variables as well.
2. As explained above in the description of the group-oriented EIB communication stack, a shared variable can be fully read/write bi-directional. In this way, all devices can also send unsolicited multicast frames.
3. EIB makes a 16 bit address space available for these shared variables. Even with the limitation of some implementations to 15 bits, this signifies that one installation may have up to 32k shared variables (or group addresses), each with any number of local instances.

The resulting scope and efficiency makes group address communication the preferred runtime mode for autonomous EIB field level communication.

In this maybe slightly unexpected way, EIB goes some distance towards reducing the need for redundant automation hierarchy levels (and bandwidth!) through appropriate addressing and device modelling schemes.

Multi-client / Multi-server Management of the OO EIB Network

An EIB installation may be seen as a collection of distributed resources, which can be managed across the network. To this end, each EIB device implements a server which provides control over local resources (including hosting services for external CPU or memory resources accessed via the serial Physical External Interface (PEI, see 7.1.2). A series of APCI s render these services accessible to remote clients. Through the introduction of EIB Distributed Objects, the network resources actually become Object Oriented (OO).

Management clients typically access the network either for control or for (initial) configuration services. EIBA implements a complete suite of vendor-neutral, standard PC-based configuration tools, which manage loadable applets, as described in 8. Hand-held devices are also available. Network-based (typically DIN-rail mounted) clients permit Interactive Self-configuration (*Easy Installation*) of (sub)-systems.

Data Formats and Interworking

As we have seen, today s EIB frame may carry data formats of up to 14 bytes. The basic data formats specified by the EIB specification include:

- boolean (1bit)
- (un)signed short (16 bit)
- (un)signed long (32 bit)
- short float (16 bit)
- IEEE float (32 bit)
- date (24 bit)
- time (24 bit)

control (4 bit) etc. Identifiers are defined for nearly all physical values like temperature, length, speed, field strength, energy, power, etc.

Type information is used mainly at configuration time: it is not transmitted for better performance and to avoid imposing unnecessary restrictions on the combinations of devices.

Properties with these basic data types are grouped into Distributed Objects, accessible via the network. The EIB Interworking Standards (EIS) specify various specialised objects for all area s of building automation such as lighting (dimming control,), HVAC applications (room temperature control, boiler temperature control,), time and event management (schedule handler, event handler,).

Hosting and Interfacing Features of the EIB Operating System

Not only does the distributed EIB Operating System (OS) serve remote clients over the network. Of course it also puts its services as a communication server (section 3 and 4) and management server (section 5.3) at the disposal of *local* client applications.

Internal Applications

To an internal application, the BAU will moreover provide CPU and memory resources, timers etc.: the application runs in the BAU . Advanced implementations allow up to three asynchronous application threads.

Utility Library API

As part of the User abstraction Layer, EIB standardises a Utility (or User) Library API, which provides further infrastructure to the application. Included are user timers, debouncing, arithmetic, bit logic, message handling etc. Through the API, the application may also access external application hardware, as explained in the next section.

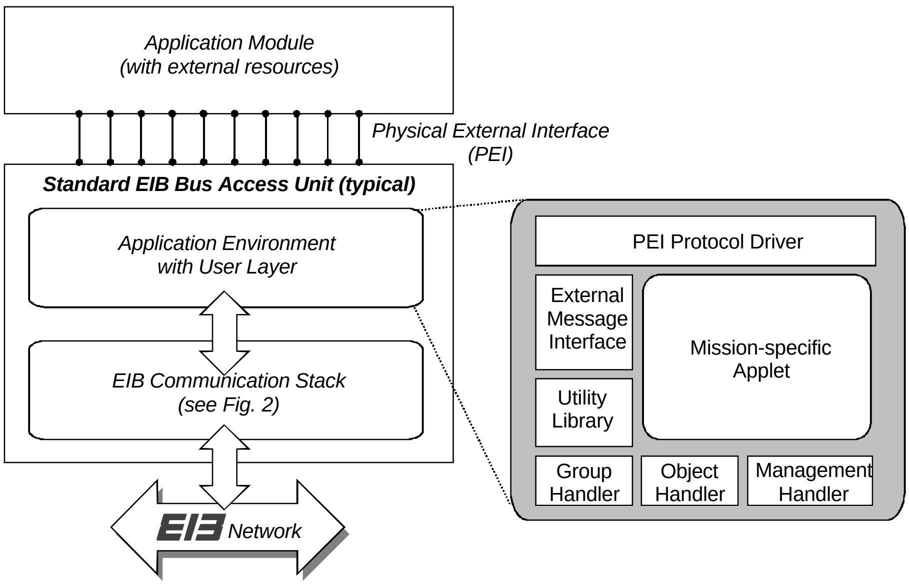

Fig. 4 PEI, User Layer and hosting features of typical EIB BAU

Hosting of External Application Hardware

Another unique feature of EIB streamlines the hosting features for application hardware or external resources: he standardised Physical External Interface (PEI). The PEI defines both electromechanical and software services for connecting an external application module to a BAU. Via a type resistor, the application module identifies its capabilities to the application; the BAU can deal with about 20 types (including binary, analog and serial I/O) and provides the application with corresponding services (via the API).

As an extra boon, the combination API / User Layer + PEI allows the combination loadable application + application module to be used *as is* on any physical medium. Particularly for flush-mounted devices, engineering overhead or adjustments to production are eliminated completely, and distribution logistics simplified accordingly.

Though optional, the PEI feature is exploited by certain standardised BAU s, called Bus Coupling Units, (see 10.3 below), which would not be possible without it.

Message Interface for Access to External Resources

For the serial PEI, EIB defines an *External Message Interface* (EMI). The EMI server allows both a local or remote client to access external CPU or memory resources.

Dual-processor Design, External Applications and Systems

Using the EMI the other way around, an external application can now utilise the local communication stack as EIB communication server. This permits dual-processor design of EIB devices. The EMI server may be implemented optionally by EIB Bus Access Units.

Actually, this feature can be used as a general mechanism for serial interfacing to external systems. Typical examples are the BCU-based EIB RS-232 interfaces, available from several manufacturers.

Tool Suites and Software Engineering Framework

EIB explicitly encompasses a methodology for Project Engineering, i.e. for linking a series of individual devices into a functioning installation [7]. This is embodied in the two vendor-independent EIB Tool Software (ETS) suites for Windows:

- With the ETS Developer s Edition, the manufacturer encapsulates the remotely loadable applets in a series of abstract representations, which hide all implementation details. The resulting Component Description can be exported.
- A project engineer or electrical contractor can import the Component Description into he ETS Project Edition. All device instances can be customised to the needs of the project and logically linked by assigning Group Addresses.

As a result, the effort is radically shifted away from project engineering.

ETS is built on top of a framework of software-engineering components for PC/Windows platforms, called the *EIB Tool Environment* (ETE). This set of API s forms part of the EIB standard. The ETE implementation is commercially available for 3rd party use.

Other System Features

EIB Home Management

EIB s native object oriented management-level features explained in 5.1 above are exploited by the EIB Home Management concept. At it s core is a commercially available PC (Windows) *Home Assistant* API framework for supervision and control of home networks. It extends EIB with an *easy installation* approach for white and brown goods, aimed at the consumer.

Developing Application Software for EIB

EIB is not bound to any particular processor or processor design. For a particular implementation, a plethora of commercially available tools such as assemblers, compilers and emulators can be put to use. These range from shareware to fully fledged environments. Certain EIB system providers offer an *Integrated Development Environment*, which allows development in ANSI C, powerful debugging and come with dedicated EIB programming infrastructure.

In turn, the Developer s Edition of ETS (see 9.1) features the necessary utilities to smoothly import the results into ETE. [2]

System Implementations allow Scaleable Access

An EIB solution developer may gain access to the EIB system by using standard EIB Bus Access Units (BAU s) with various levels of integration (scaleability). Alternatively, he/she may go for a proprietary (but compatible) implementation on any microprocessor chip.

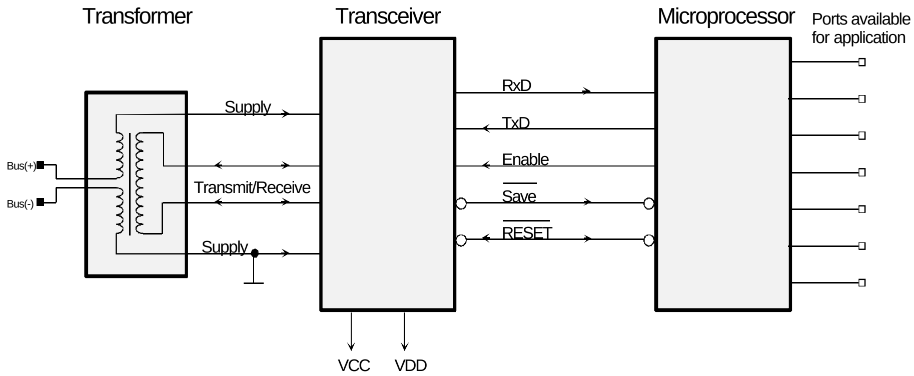

Fig. 5 Principle electrical schema of inductively coupled TP BCU or BIM

The *Bus Coupling Unit* (BCU) is the most complete BAU, with Medium Access, EIB OS firmware, application hosting resources (CPU, RAM, EEPROM,) and the full PEI and EMI. A BCU is housed in a compact, screened and *ready-to-install* package; construction shapes are aligned to the demands of practical installation, like DIN-rail of flush-mounting (Fig. 6). Application-compatible BCU s are available for all EIB media.

EIB *Bus Interface Modules* (BIM) fulfil the same tasks of bus access and application host, as explained in the previous paragraph - but with only the electrical PEI (with EMI), and no EMC shielding or housing. This makes them ideal for tighter integration into the application-specific solution. A high-end C programmable TP BIM is available with 8-32 k EEPROM.

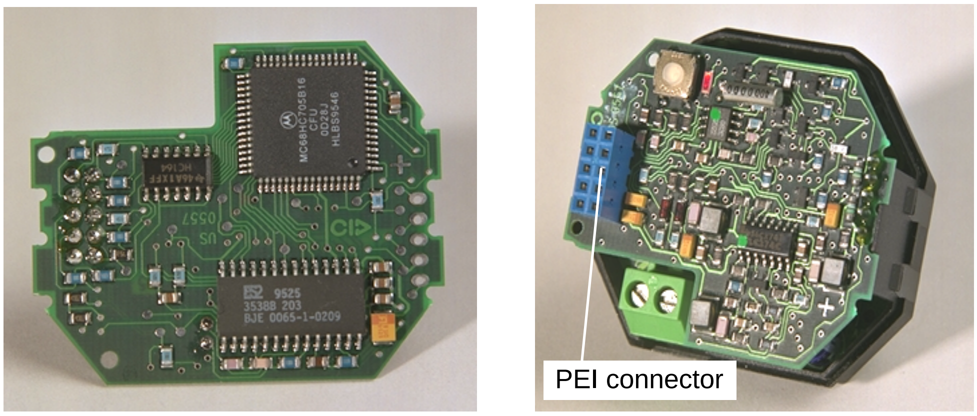

Fig. 6 The EIB Powernet BCU for flush-mounting

BAU building blocks may also be obtained as a *chip set*. To facilitate rapid 3rd party implementations of the EIB systems, the full source code can be obtained at non-discriminatory conditions.

Spectrum of Available Products

In the summer of 1997, roughly 50 manufacturers offer 2 500 commercial products, covering application domains such as heating control, energy management, security, time and event management, lighting control. Though compatible, EIB implementations are marketed under various brand names such as instabus, Tebis, i-bus EIB, Powernet, Home Electronic System, Domotik, ImmoCAD etc.

References

[1] The EIB Handbook Issue 2.21. EIBA 1996.

[2] The EIB Handbook Issue 3 - Vol. 2, Developer s Guide. EIBA 1997.

[3] Goossens, M.: A Survey of the EIB System. In: EIBA Proceedings 1997, p 49. EIBA 1997.

[4] Goossens, M.: Communication and Addressing on EIB. In: EIBA Proceedings 1997 p59. EIBA 1997.

[5] Goossens, M.: Object-based Distributed Application Design. In: Feldbutechnik in Forschung, Entwicklung und Anwendung (Dietrich, Ed.) p 152, Springer 1997.

[6] Goossens, M.: Easy Installation and Home Management. In: Feldbutechnik in Forschung, Entwicklung und Anwendung (Dietrich, Ed.) p 387, Springer 1997.

[7] Goossens, M.: Component-based Project Engineering. In: Feldbutechnik in Forschung, Entwicklung und Anwendung (Dietrich, Ed.) p 370, Springer 1997.

[8] Heite, Ch., Rosch R.: A Powerline Carrier Communication System with Matched Filter Receiver Technology. In: GMM Fachbericht 17 Microelektronik 97, VDE Verlag Berlin 1997.

[9] Heite, Ch., Zapp R.: Powernet - das neue EIB Medium. In: Elektrotechnik und Informationstechnik - ÖVE Verbandszeitschrift 5/97.