**New Method of Electronic Control of Model Railway Track**

Jan Horáček

Faculty of Informatics,

Masaryk University

Botanická 68a, 602 00 Brno

CZECH REPUBLIC

Jiří Rybička

Department of Informatics,

Faculty of Business and Economics, Mendel University in Brno,

Zemědělská 1, 613 00 Brno,

CZECH REPUBLIC

Robert Čížek

Department of Informatics,

Faculty of Business and Economics, Mendel University in Brno,

Zemědělská 1, 613 00 Brno,

CZECH REPUBLIC

*Abstract:* Digital control of a model railway track requires control of the movement of vehicles and their functions (sounds, lighting) and control of track-side elements (switches, signals, occupancy detectors, etc.). While the control of the vehicles is given by a standard whose functionality is perfectly adequate for normal operation, the control of track-side elements can be handled by more ways. For small domestic tracks, commercial solutions are available from several companies, but for extensive (large) tracks there is no such concept and the appropriate solution is usually made tailored to the application (club tracks, simulators for training of railway operators). Track in the Railway Vehicles Control Laboratory of the Faculty of Business and Economics of Mendel University in Brno can be categorized as large; therefore, a specific system is necessary. However, the existing one is not suitable for several fundamental reasons. Therefore, a modernized system has been developed. New protocol for RS485 bus has been designed, hardware and firmware of a new GPIO slave module MTB-UNI v4 has been developed, new RS485 master module MTB-USB has been developed and other two new hardware modules including firmware for their MCUs have been developed. A computer application MTB Daemon and a library have been developed to access the bus. The result is a system that meets the relatively demanding requirements and enables further development and use in the laboratory.

*Keywords:* model railway, electronic control, communication protocol, MTB, interlocking, Railway Vehicles Control Laboratory

1 Introduction

The Railway Vehicles Control Laboratory is a specialized workplace with two model railways covering an area of over 200 m2. Far from being a toy for children, the model railway now is a large computer-controlled electronic complex. The basic purpose of the laboratory is to be used in teaching process, primarily as a teaching instrument for program control of technological units and consequently programming in general. It is also used for thesis work on various technical aspects of operation (automatic train diversion, shuttle operation of motorized passenger trains, etc.). The track represents a digitally controlled technological unit, whose design has been adapted to the standards used by the Model Railway Club Brno I (MRC Brno I), in whose cooperation the laboratory was created 10 years ago and nowadays is operated.

2 Current state

Digital control of a relatively large track requires approaches that are different from the controlling of a small “home” tracks. The basic criteria that a system for managing larger tracks must meet are derived by Horáček [1, p. 13]. After some modification for the needs of the laboratory taken into account, the criteria can be summarized in the following partially interrelated points:

1. Ability to support hundreds of track-side elements (signals, switches, etc.) with several types of electric interfaces.
2. Sustainability over time – the concept should be usable over several decades, i.e., built on the generic components that are expected to be available for a long time in their current or compatible form.
3. Extensibility of supported functions – new types of modules could be added in future.
4. Acceptable financial demandingness.
5. Two-way communication with the control computer, ability to detect the correct functionality of modules.
6. Independence from proprietary enterprise solutions (for sustainability over time, but also for acceptable financial requirements).
7. Ability to operate in a way that mimics a real railway.

Although various electronic control components are available on the market, they are mostly intended for the general consumer, i.e. amateurs working in domestic conditions, and therefore usually do not meet some of the criteria defined above.

Let us briefly mention in practice frequent solutions.

2.1 Digital Command Control

The Digital Command Control (*DCC*) [2] is probably the most widely used system for digital control of model railways. There are no studies to confirm this, but it can be inferred from the number of components we see on the market. DCC is an open standard created by the National Model Railroad Association (nmra.org). DCC unifies controlling of both track-side and mobile units.

Basic properties: main element of the system is the *command station* transmitting DCC signal to control train movement and track accessories. The signal was designed as unidirectional, to obtain status of the elements in the track a second bus, called *feedback bus*, is used. The bus is controlled by the command station, which inquiries feedback modules for data. The command station can be connected to a computer over another communication bus – *throttle bus* (e.g. LocoNET, XpressNET). In the current state-of-the-art deployments, neither vehicle nor track-side decoders acknowledge receiving of the DCC command to the command station, which may lead to a failure (non-delivery of the command or failure to execute the command for various reasons). Schematic of DCC track control is shown in Figure 1.



Fig. 1: Typical schematic of a DCC system. Modified according to [3]

There is no standard for how the feedback bus works, therefore various possibilities have been developed: S88 [4], RSbus [5] or LocoNET [6]. The latter is a licensed bus created by Digitrax and its full use is subject to licensing fees. However, it supports two-way communication, allowing direct commanding of track devices without the need for DCC signal.

Typical limitation of the DCC system is a relatively small maximum number of feedback modules. Extensive railways usually address this issue by dividing the track into sufficiently small isolated areas and operating independent modules in each area (applied, for example, to very large track in the Railway Kingdom in Prague, presented on professional excursion in 2015).

2.2 BiDiB

BiDiB (BiDirectional Bus) is an open bus designed by a community of model railroaders [7]. It is based on RS485, several types of modules exist, communication protocol is available online. However, BiDiB is just a protocol specification, no schematics nor PCB designs nor firmware are available by BiDiB. BiDiB fulfills basically all the criteria defined above.

2.3 MTB

The MTB v2 system (Model Train Bus) [8] is currently used in the laboratory. It allows to control hundreds of track-side elements.

Control of the whole track is divided into two different parts:

* Vehicle control – not addressed in the MTB system. For example, the DCC system with extended loco addresses can be used.
* Track-side (accessory) control – here, on the other hand, the MTB concept departs from the commercial concepts (the DCC system in commercial solutions typically controls accessories as well, as already mentioned in the relevant section).

Separating the accessory control from vehicle control provides a number of advantages. Single communication bus *MTBbus* is used for both commanding track-side equipment and obtaining information from it, which allows for simpler (cheaper) design, own design brings high extensibility, fast bug fixing, independence on commercial manufacturers and other benefits connected with in-house solution.

Own design is not beneficial for vehicle control, because in this area, open protocols exists [2], thus risk of vendor-lock is minimized and price of the components is quite low due to competitive environment on the market.

It is then possible to build software support over both parts to allow to mimic the behavior of the signaling equipment on a real railway, which is a significant advantage for the teaching purposes of the laboratory.

2.3 Technical description

Complex scheme incorporating both DCC and MTB systems is presented in Fig. 2.

MTB system consists of:

1. MTBbus protocol specification. MTBbus is RS485 max. 115200 Bd single-master bus.
2. MTB-USB module, which creates interface between MTBbus and PC (via USB). It is master on MTBbus. It performs all real-time operations on MTBbus.
3. Specification of protocol between MTB-USB and PC.
4. MTB modules. There are various MTB v2 modules: MTB-UNI (universal – 16 digital inputs with IR point detectors support, 16 open-collector outputs), MTB-UNIm (same as MTB-UNI, but without IR support), MTB-TTL (16 digital inputs, 16 TTL outputs), MTB-REG (8 powerful analog outputs), MTB-POT (8 analog inputs).
5. Software support in PC (libraries for connecting to MTB-USB etc.).

More comprehensive specification of MTB v2 is available [9], unfortunately in Czech language only.



Fig. 2: Track control architecture in RVCL before modernization

2.3.1 Shortcomings of the current MTB system

As complexly illustrated in the Fig. 2, the system is capable of controlling large tracks. The diagram describes an architecture of railway in MRC Brno I and in the Railway Vehicles Control Laboratory of the FBE MENDELU. This allows interchange of hardware, software and know-how between the Club and the faculty.

Unfortunately, however, the accessory control part (MTB part) currently carries some very fundamental problems:

1. Outdated component base – the last update of the system was made in 2007 and some of the components used are currently unavailable. Installed electronics manufactured in the past and used on track has no direct replacement. This makes it impossible to expand the track with additional railways and stations, makes it impossible to replace broken parts of the electronics and to use electronics for other (e.g. diagnostic) purposes.
2. Unresolved licensing conditions – the authors of the system did not provide the source code for the firmware, electrical schematics nor PCB designs in a source form. Thus, essentially no updates can be made, bugs cannot be fixed, new features cannot be added.
3. Technical capabilities of the processors – current MTB system is based on *AT89C2051* MCUs with 2 kB of flash memory, 128 B SRAM. This capacity is insufficient for more complex applications. The firmware cannot be extended and the processors lack some key peripherals, e.g. EEPROM.

To enable development of the laboratory it is absolutely necessary to modernize or replace the MTB system (the main purpose of the Internal Grant Agency project PEF\_TP\_2020004 on Faculty of Business and Economic MENDELU). For this phase, the following general requirements have been defined resulting from the facts already mentioned:

1. The new system must be reasonably backward compatible with the current track control software.
2. The new system shall be sustainable for a sufficiently long time (estimated at 20 years).
3. The solution must not be costly.
4. Communication must include acknowledgement of controller actions; the correct functionality of the modules must be detected.
5. New functions and requirements defined in the future must be implementable, i.e. the system must be sufficiently extensible.

It is recommended to develop the new system as an open solution, i.e., hardware components as open hardware, all software as open source.

3 MTB v4

Based on the current situation described in the previous chapter, we decided to improve MTB v2 system and design its new version *MTB v4*. We chose this solution, because we perceive the advantages of in-house solution as significant. Furthermore, own design allows students to understand the model railway controlling better, it allows to experiment with it, develop alternatives and simply transform student's ideas into reality.

We chose not to use BiDiB, because the cost of deployment of the system. BiDiB modules in general are relatively cheap, however the cost of replacing all the modules and wiring in the existing railway is unacceptable. MTB v2 must be simply upgradable to MTB v4.

Now we describe design of MTB v4 system.[[1]](#footnote-1)

2.4 Interfaces

On one side, MTB communicates with the track-side equipment (points, signals, sections, lights, etc.) – this must have been and was preserved. On the other side, MTB communicates with computer. We keep using USB for this interface, because we find no disadvantages of it. MTB v4 will be connected to interlocking software hJOP [10] (currently used in laboratory). It connects to the track via a dynamically linked library with a defined API. New design must support multi-master control, so that the individual elements can be controlled solely by the computer, but with multiple applications. This design allows to connect e.g. both train interlocking hardware and signals hardware to a single MTB module, which is then controlled by two different applications. This solution requires less hardware and wiring, which is highly desirable.

Until now, two types of outputs have been used for communication with the track-side hardware – digital binary (for all two-state outputs) and S-COM [11] for transmission of signal aspects. In addition to these two, the new system adds the so-called oscillatory output (digital signal with frequency in small number of Hz) for indication purposes and for special devices for train set uncoupling (uncouplers in the tracks).[[2]](#footnote-2)

2.5 Modules

It was conceptually decided not to support analog inputs and outputs or pulse-width modulated outputs in MTB v4, because they are not needed in connected peripherals. In case they will be needed in future, new modules will be developed.

MTB v2 uses three variants of universal IO modules – UNI, UNIm and TTL – which differ in their support for infrared point sensors and in type of outputs (TTL vs open-collector). To simplify and cheapen the solution, only one universal mass-producible module shall be developed. To connect specific peripherals, expansion modules can be used.

2.6 Other improvements

In line with some of the above framework requirements, the new system should be able to detect modules that are added on the bus at runtime (not just at system start-up), as opposed to the current state. A problem in the same category is enabling easy detection of a malfunctioning module (e.g. during a power failure, disconnection of connectors etc.). The innovation against the current state in this context appears to be the consistent implementation of command acknowledgement, which will enable the above functions.

For easy identification of a module in the track (individual modules are positioned on the underside of the track at the locations where the corresponding equipment is placed) it should be possible to switch on an indication LED on it from the control computer.

Updating module firmware should be possible directly over the MTBbus. Considering the number of modules on a large track (higher tenths or hundreds), it is very difficult to update the firmware manually. Using more advanced processors, this operation is achievable and allows updating of a firmware on a module by module basis across the entire track. This entails the elimination of errors and omissions, including the elimination of time delays in manual updates.

2.7 MTBbus v4

The bus is designed from scratch. This is due to a number of significant changes compared to the current state, as well as the need to resolve licensing issues.

Hardware-wise, the RS485 standard is retained. The supported communication speeds remain three – 38 400 Bd, 57 600 Bd and 115 200 Bd. Lower speed is not necessary; at maximum speed, there are approximately 10 scans of each module per second with 100 modules on the bus (real packet length considered), which is enough for safety functions.

The communication is based on periodic polling all modules on the bus, each module responds with a message. This confirms that the module is alive. Modules that are active are polled more frequently than inactive modules to reduce latency. However, inactive modules must be included in the queries sometimes as well, so that it can be detected when new modules have been connected. Some ideas how to create the communication protocol can be also found in work of Lascano and Clyde [13].

Message of MTBbus v4 consists of the following parts:

* Module address – 1 B, provides an address space for 255 modules, which is sufficient (address 0 is broadcast).
* Message length – as opposed to the MTB v2 situation where the maximum length could be 7 B, the length of the data part has been increased substantially, up to 120 B, which allows sending in addition to regular messages, firmware for module updates.
* Message code – a byte defining the meaning of the message.
* Message data part – up to 120 B of data.
* Checksum – compared to the MTB v2, where only XOR of the message is calculated, MTB v4 calculates a CRC-16 checksum instead. For longer message length in MTB v4, this type of checksum is more appropriate. This solution is similar to ModBus [14] industrial bus.

The specification[[3]](#footnote-3) of MTBbus v4 is two-layered – the protocol defines messages, but the content of some messages is different for each MTB module type. The data format of inputs and outputs, the configuration format of the module, addressing and memory organization for firmware updates are defined for each module type separately. This is advantageous in view of possible further development, as the protocol can still remain the same, although other types of modules with different characteristics may be added in the future without the need to modify a firmware of MTB-USB nor firmware of any older module.

Particular attention in the design is paid to firmware updates of MTB modules. The firmware update procedure is described in detail online in the document <https://github.com> /kmzbrnoI/mtbbus-protocol/blob/master/workflows.md. The update can be done while the bus is fully operational, which is a great advantage especially if a module with a firmware of different original version is added or replaced.

The flash memory in MTB modules can be overwritten from the bootloader. The update protocol reboots the processor into this program. Because it is unsafe to update the bootloader over the MTBbus, it must be a small and well tested piece of firmware that will not need to be reprogrammed and will be loaded into the processor only once during manufacture by programming the module directly.

The changes in the design concept of the track control electronics are outlined in Figure 3. The implementation is described in [1, chap. 5] in detail, here we briefly mention key parts.



Fig. 3: A new concept of track control electronics – MTB v4 architecture

2.8 MTB-UNI v4

Instead of the several types of MTB modules, only one type was designed with three freely adjustable options on each output pin: binary, S-COM and oscillation mode. The *MTB-UNI v4[[4]](#footnote-4)* universal module is built on modern components, but with the expectation of long-term availability. MTB-UNI v4 is based on ATmega128A MCU [15], firmware is programmed in C.

The configuration of the module is permanently stored in the MCU. It is necessary for example to store the communication speed of the bus and convenient for the defined state of the outputs immediately after the power supply is switched on. However maintaining an authoritative configuration is handled by the computer, where all storage, backup and versioning of configurations can be handled elegantly and much more easily. See section 3 for more details.

MTB-UNI v4 PCB design is based on well-known parts which are commonly used in industry. The emphasis is placed on its protection against damage from wrong wiring. The damage is done by connecting too high or too low voltage to the board. In that case components may be burned. The protection is implemented with PTCs, thyristors and Zener diodes. The design is based on a crowbar circuit [16]. Big advantage in the design of the protection is its reversible cut-off – no fuses are burned.

MTB-UNI v4 firmware consists of a main firmware and a bootloader firmware. The bootloader is used for main firmware updating over MTBbus.

2.9 MTB-2-AVR

A cost-saving upgrade option was developed to keep the original MTB-UNI v2 modules, with the original processor being replaced with a new one (ATmega328P [17]), all other components, cabling and mechanical parts remain. As there is not a new processor available that matches the pinout of the original processor, an add-on *MTB-2-AVR* board[[5]](#footnote-5) was created to slide into the socket of the original processor. The add-on board carries the new processor in an SMD design with the necessary additional components. By simply replacing the processor in the original socket with the new board, a new MTB-UNI module is created.

MTB-2-AVR firmware consists of a main firmware and a bootloader firmware. The bootloader is used for main firmware updating over MTBbus.

2.10 IRdet

Infrared point sensors support had to be implemented in order to make the new solution deployable on the current and the new track where these sensors are used. An expansion board called *IRdet* was created to allow the connection of up to 8 IR sensors. Its outputs can be connected to the inputs of the MTB-UNI module. These outputs can also be used for other purposes (e.g. signaling in control panels).

2.11 MTB-USB v4

Implementation of the requirements forcing a different bus operation requires a new design of the *MTB-USB v4* module[[6]](#footnote-6). Here, backward compatibility with the original MTB-USB v2 board does not have to be strictly adhered to, as this board is only one throughout track and its complete replacement is neither technically nor financially too demanding.

PCB was designed in an open-source software KiCad [18], design is based on MCU STM32F103 [19], firmware is programmed in plain C. Firmware consists of a main firmware and of a bootloader, which allows updating the main firmware via USB without the need for special programmer (e.g. STlink [20]).

3 Software

Two computer applications were developed as part of the solution –*MTB Daemon*[[7]](#footnote-7) and a library *hJOP MTB Network Library*[[8]](#footnote-8)*.*

The requirement for multi-master control was solved in such a way that the MTB-USB v4 communicates with one application on the computer side – MTB Daemon –, but multiple control applications can connect do MTB Daemon via JSON REST API and command it. This functionality, among other things, allows students in the laboratory to program their own simple applications controlling the tracks (IP-B project No. 8.1.17 of 2018). The programming language of MTB Daemon is C++17 with the Qt framework, which allows for a multi-platform solution; compilation is possible for both Linux-type OSes, as well as Windows type OS. Configuration of MTB Daemon and MTB modules is stored in human-readable form in json file. The file is under version control system (git) separate for each deployment.

The hJOP MTB Network RCS Library interfaces the existing track control system hJOP and MTB Daemon. It is also developed in C++ as DLL library.

4 Discussion

MT v4 represents a solution for further development in the Railway Vehicles Control Laboratory. Expansions are planned to include additional station, including the locomotive depot, which will allow to increase the operational possibilities and to address other possible traffic situations, including various safety features and the implementation of the train timetable.

API of MTB Daemon allows students to create railroad-control applications without the need to struggle with hardware-related issues. Students can use programming languages independent on implementation of the hardware and software components of the track. This contributes to an effective teaching of programming in the laboratory. In addition, the space has been expanded for students of specialized courses and thesis writers who can create applications basing of MTB v4 and solve more challenging train control operation tasks.

API of MTB Daemon allows to control one bus with multiple control applications, which is quite unique on the market and useful especially for students.

Compared to systems available commercially, the designed and implemented solution has a great advantage in the openness of the hardware and software, thus the parts needed by different users can be used without restrictions. In addition, the cost is lower than the components available for conventional domestic tracks. Here a new field of application is emerging – hardware components have been developed, but adaptation to commercial software systems has not yet been addressed. Support for MTBbus could be implemented in e.g. JMRI model railroad control software \cite{jmri} as a separate software module, or integration with e.g. TrainController \cite{traincontroller} commercial model railroad control software could be implemented via changing PC interface of MTB-USB to the interface already supported by the TrainController software.

Compared to MTB v2 and other commercially available systems, MTB v4 brings higher level of safety -- all messages are acknowledged, module failure is detected etc. This creates a pressure on a safety of similar commercial components.

In the future, the possibility of supporting higher bus speeds may be considered. This will allow even shorter delay between change of input of MTB module and a reaction to the change in a computer. Another enhancement could be an automatic detection of the bus speed by MTB modules, important when connecting new modules or replacing broken modules on-the-fly. An interesting possibility could also be an extension to retransmit bus data over wireless connection to control parts of the track that are physically separated from the control computer due to spatial, architectural or other reasons. An example would be a railway consent console used in a foreign station at a modular tracks meeting, where each section of track is supplied by a different modeller but we need to operate the track as a whole.

Excluding already-mentioned latency and automatic bus speed detection, the last important limitation of MTB v4 is the maximum number of modules on one bus. From our point of view, 255 modules on one bus is enough, however some non-standard applications of MTBbus which need more modules could be imagined. Simply adding more bits for module address is not suitable, as mechanisms like discovering new modules could take long time and latency can grow considerably. Bus with bigger maximum number of modules would probably require a different media access control mechanism.

Conclusion

MTB v4 has been implemented and deployed on all tracks in the Railway Vehicles Control Laboratory as well as on all tracks in MRC Brno I. That includes 3 independent layouts with 103 MTB modules in total. Most of these are MTB-UNI v2 modules with MTB-2-AVR addon module. 7 MTB-UNI v4 modules have been in operation for 100+ hours each without any major problems. MTBbus v4 has been in operation for 300+ hours in total with only minor difficulties. These issues were solved with new firmware versions, where uploading a new firmware over MTBbus turned out to be very useful.

MTB v4 can also be used in other fields, where centralized computer control of several dislocated peripherals is used. Examples include home automatization, production lines controlling or remote monitoring. Generally, MTB is suitable for applications, where PLCs \cite{plc} are currently used.

Acknowledgements

The upgrade of the MTB system in the Railway Vehicles Control Laboratory was funded by the IGA team project No. PEF\_TP\_2020004 called “Innovation Electronic Control of Model Tracks”. A major contribution to modernization was made by the thesis of Jan Horáček [1].

References

[1] Horáček, J. Návrh a implementace nového protokolu sběrnice MTBbus. [Design and implementation of a new MTBbus protocol] Diploma thesis. Supervisor: Zdeněk Matěj. Masaryk University, Brno, 2021.

[2] NMRA. Electrical Standards for Digital Command Control. [online] Available at https://www.nmra.org/si­tes/default/files/standards/sandrp/pdf/s-9.1\_electrical\_standards\_for\_digital\_command\_control\_2021.pdf. Last rev. Apr 9, 2021

[3] Wikipedia. Digital Command Control [online] Available at <https://en.wikipedia.org/wiki/Digital_Command_Control>. 2021.

[4] OPENDCC. Transmission of S88 data over network cable [online]. 2021 [cit. 2021-04-26]. Available at https://www.opendcc.de/s88/s88\_n/s88-n\_e.html. 2021.

[5] Pras, A. RSbus Arduino Library [online]. [cit. 2021-04-26]. Available at https://github.com/aikopras/RSbus.

[6] DCC WIKI. LocoNet overview [online]. [cit. 2021-04-26]. Available at https://dccwiki.com/LocoNet.

[7] BiDiB. Kurzzusammendfassung von BiDiB [online] [cit. 2021-04-26] Available at http://bidib.org/protokoll/intro.html.

[8] Báňa, V., Trávník, P. Komunikační protokol modulů MTB [Communication protocol of MTB modules]. Online. [cit. 2021-04-28]. Available at https://mtb.kmz-brno.cz/assets/pdf/mtb-protok20.pdf.

[9] Horáček, J. MTB, IO sběrnice pro řízení příslušenství modelových kolejišť [MTB, IO bus for model railway accessories control] [online] [cit. 2023-10-10] Available at https://mtb.kmz-brno.cz/cz/v2

[10] Horáček, J. hJOP: system RCS [online]. [cit. 2021-05-03]. Available at https://hjop.kmz-brno.cz/rcs.

[11] Báňa, V. 2021. Dekodér S-com pro světelná návěstidla ČSD [online] [cit. 2021-04-20] [The decoder S-com for signals on Czech Railways model] Available at https://www.mtb-model.com/elektro/s-com-nav.htm

[12] Galieva, Z. Automatizace objíždění souprav na modelovém kolejišti. [Automation of train detours on the model railway] Diploma thesis. Supervisor: Jiří Rybička. MENDELU. 2020.

[13] Lascano, J. E., Clyde, S. W. A Pattern Language for Application-level Communication Protocols. [online] In: The Eleventh International Conference on Software Engineering Advances. 2016. ISBN 978-1-61208-498-5. Available at https://www.academia.edu/31103209/A\_Pattern\_Language\_for\_Application\_level\_Communication\_Protocols

[14] Modbus Organization Inc. 2021. ModBus Application Protocol Specification [online]. [cit. 2021-05-04]. Available at https://www.modbus.org/docs/Modbus\_Application\_Protocol\_V1\_1b3.pdf.

[15] Microchip Technology. ATmega128A [online] 2023. Available at https://www.microchip.com/en-us/product/atmega128a

[16] Electronic notes. SCR Thyristor Crowbar: overvoltage protection circuit. [online] 2021. Available at https://www.electronics-notes.com/articles/analogue\_circuits/thyristor-scr-triac/overvoltage-protection-crowbar-circuit.php

[17] Microchip Technology. ATmega328P [online] 2023. Available at https://www.microchip.com/en-us/product/atmega328p

[18] Kicad. KiCad ESA – A Cross Platform and Open Source Electronics Design Automation Suite [online] 2023. Available at https://www.kicad.org/

[19] ST Microelectronics. STM32F103 [onli­ne]. 2023. Available at [https://www.st.com/en /microcontrollers-microprocessors /stm32f103.html](https://www.st.com/en%20/microcontrollers-microprocessors%20/stm32f103.html),

[20] ST Microelectronics. ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32 [online] 2023. Available at https://www.st.com/en/development-tools/st-link-v2.html

1. Full specification of MTB v4 is available at https://mtb.kmz-brno.cz/, in Czech language only. [↑](#footnote-ref-1)
2. The problem of train uncoupling was dealt with in detail in the work of Z. Galieva [12]. [↑](#footnote-ref-2)
3. Full specification is available online https://github.com/kmzbrnoI/mtbbus-protocol. [↑](#footnote-ref-3)
4. https://github.com/kmzbrnoI/mtb-uni-4-ele, https://github.com/kmzbrnoI/mtb-uni-4-fw [↑](#footnote-ref-4)
5. https://github.com/kmzbrnoI/mtb-2-avr-pcb, https://github.com/kmzbrnoI/mtb-2-avr-fw [↑](#footnote-ref-5)
6. https://github.com/kmzbrnoI/mtb-usb-4-pcb, https://github.com/kmzbrnoI/mtb-usb-4-fw [↑](#footnote-ref-6)
7. https://github.com/kmzbrnoI/mtb-daemon [↑](#footnote-ref-7)
8. https://github.com/kmzbrnoI/mtb-net-lib [↑](#footnote-ref-8)