DCC Command Stati

Digital model control with the ARM7

Patrick Smout

Electronics is making more and more inroads into the domain of model trains. Trains are now controlled with digital codes, and in many cases the entire system can be operated from a computer. In this article we present a design for the device that forms the heart of a digitally controlled model railway: the DCC Command Station. The computing power in this design is provided by a high-performance ARM7 processor.

One of the key elements of a modern digitally controlled model railway is the control unit that produces the digital signals for controlling the locomotives. This device also has several secondary functions, such as communicating with a PC.

This article describes a control unit (called a 'Command Station' in model railway jargon) that employs the DCC standard and is built around a powerful ARM7 processor, which is also used on the *Elektor* 'ARMee' board.
First a brief introduction to the topic.
With conventional analogue control of a model railway system, the speed of the locomotives is adjusted by varying the voltage on the track. The simplest arrangement consists of a transformer and a potentiometer, possibly complemented by a rectifier. More advanced systems use pulse-width modulated (PWM) signals. The principal advantage of this approach is improved running characteristics. This is quite nice, but even with these systems it is very difficult to control other functions in a locomotive, such as sound effects or a smoke generator.
If a decoder is fitted in the locomotive and the transformer is replaced by an electronic control unit with a track booster, we enter the realm of digitally controlled model railways.

The electronic controller that makes all this possible is called a 'Command Station'. The Command Station described here is suitable for systems that employ what is called the 'DCC standard'. DCC is not the only method for digital control of model railways, but it is one of the few methods that is not tied to a specific manufacturer.
The advantages of digital train control can be summarised as follows:

- Dramatically improved locomotive running characteristics, especially at low speeds and on uphill grades.
- Individually switchable auxiliary functions in locomotives, such as sound effects, smoke generators, remote decoupling, and lighting (including when the train is standing still).
- Several trains can travel on the track or in the same block at the same time.
- A computer can be added for easy automation of control and protection functions.

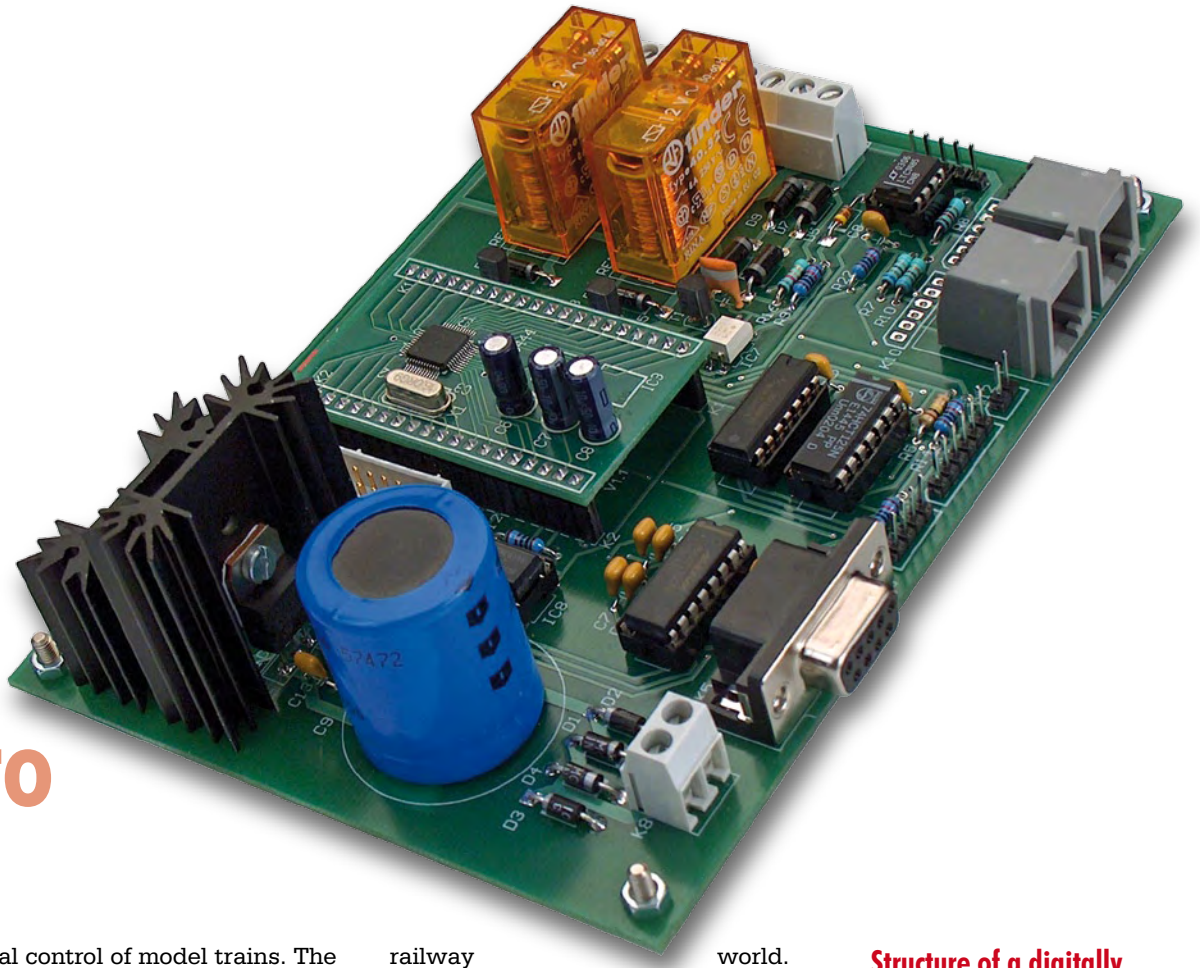
An additional benefit of the system described here is:

- The standard is widely used, so products from different suppliers can be used 'mix and match' in the same system.

A brief history of DCC

Despite what you might think, DCC (which stands for 'Digital Command Control') is not new. The history of DCC goes back to the late 1980s. At that time a working group of the National Model Railroad Association (NMRA), an umbrella organisation of model railway enthusiasts in the USA, was looking for a standard that could be used

on micro



for digital control of model trains. The commercially available systems at that time were proprietary and mutually incompatible.
After an extensive study of the available systems and a few diversions, in the early 1990s the NMRA decided on a system developed by Lenz, a German company. Reliable data transfer and interference-free operation were naturally the most important criteria for their choice. In addition, the new standard had to be able to evolve with the available technology and satisfy new requirements arising in the model railway world.
The requirements and expectations that the new standard must fulfil were specified by the DCC working group of the NMRA in documents called 'Standards' and 'Recommended Practices' (RPs). The electrical properties that the signals must fulfil, as well as the form and content of the transferred data, are defined precisely. Among other things, the recommended practices define the requirements that must be met by the digital boosters that supply power to the model railway.

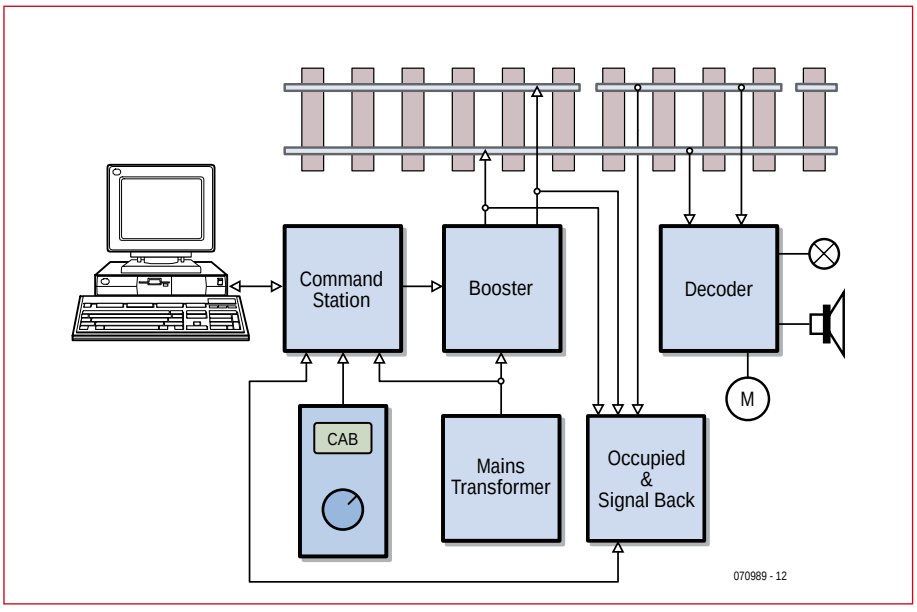


Figure 1. Block diagram of a model railway system with digital control.

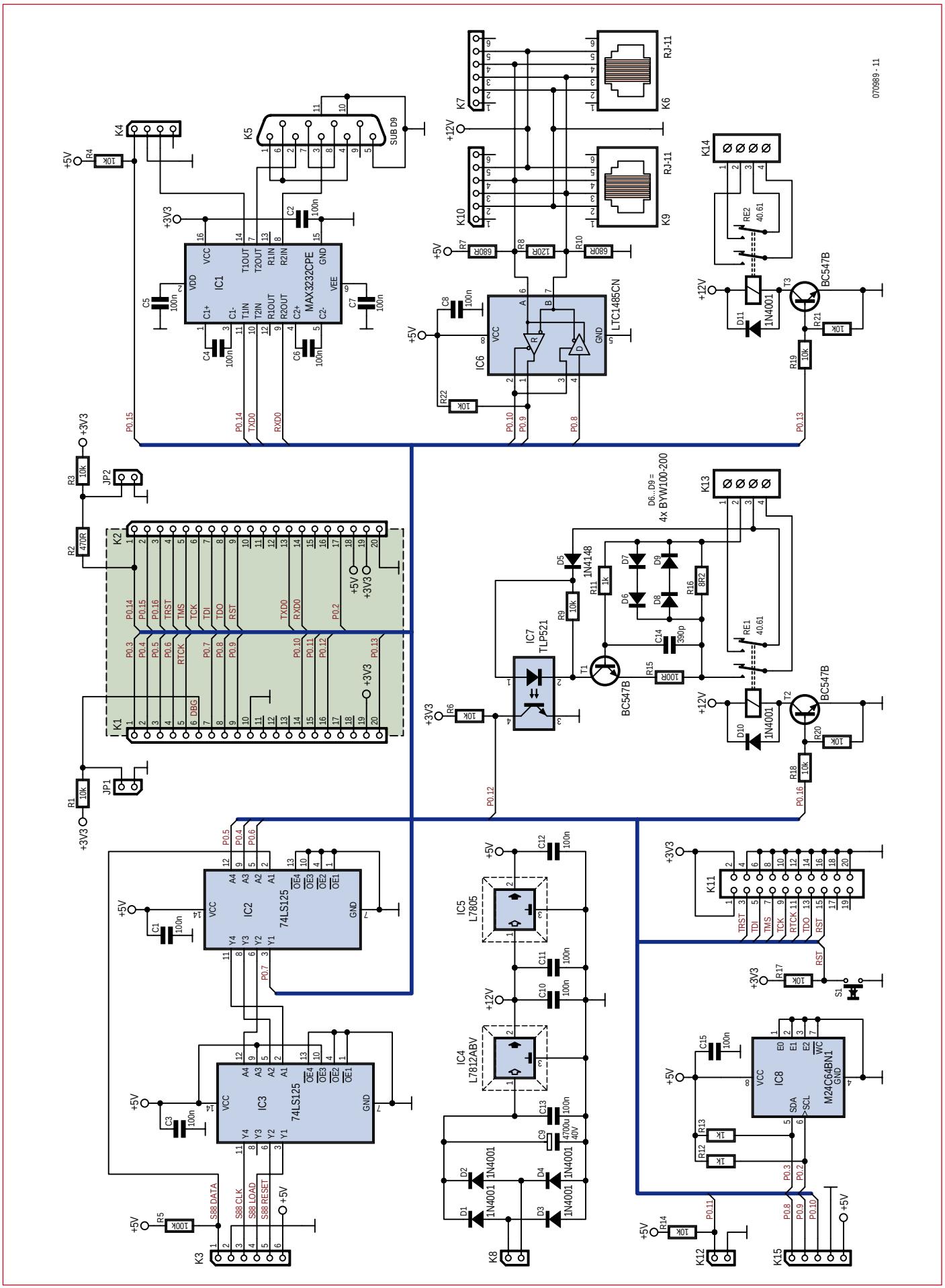
Structure of a digitally controlled model railway

A digitally controlled model railway consists of the following functional blocks (see **Figure 1**):

- **Transformer**. Every system needs a source of power.

- **Decoder**. This processes the signals that originate from the Command Station and are placed on the track by the booster. A decoder can be fitted in a locomotive, but it can also be used in a fixed location for controlling components such as turnouts and signals (in the latter case, it is called an accessory decoder). A remarkable aspect of these decoders is that they receive power and digital signals via the same leads, so there is no need for separate power leads. As each decoder has a unique, configurable address, it is easy to connect several decoders to the same track. Each decoder only responds to the information intended for it.

- **Occupancy feedback detectors**. These are used to keep track of which parts of the track are occupied or free. For this purpose, the track is divided into a number of small sections, with one rail of each section electrically insulated from the adjacent sections. The insulated rail of each section is connected to an occupancy feedback circuit. In its simplest form, this is a current detector. If a section is occu-


070889 - 11
IC1 MAX3232CPE
IC2 74LS125
IC3 74LS125
IC4 L7812ABV
IC5 L7805
IC6 LTC1485CN
IC7 TLP521
IC8 M24C64BN1
T1 BC547B
T2 BC547B
T3 BC547B
D6..D9 = 4x BYW100-200
K1
K2
K3
K4
K5 SUB D9
K6 RJ-11
K7
K8
K9 RJ-11
K10
K11
K12
K13
K14
K15
JP1
JP2
S1
RE1 40.61
RE2 40.61
+5V
+3V3
+12V
P0.15
P0.14 TXD0
RXD0
P0.10
P0.9
P0.8
P0.13
P0.12
P0.16
P0.7
P0.5
P0.4
P0.6
P0.3
P0.2
P0.11
TRST
TDI
TMS
TCK
RTCK
TDO
RST
S88 DATA
S88 CLK
S88 LOAD
S88 RESET

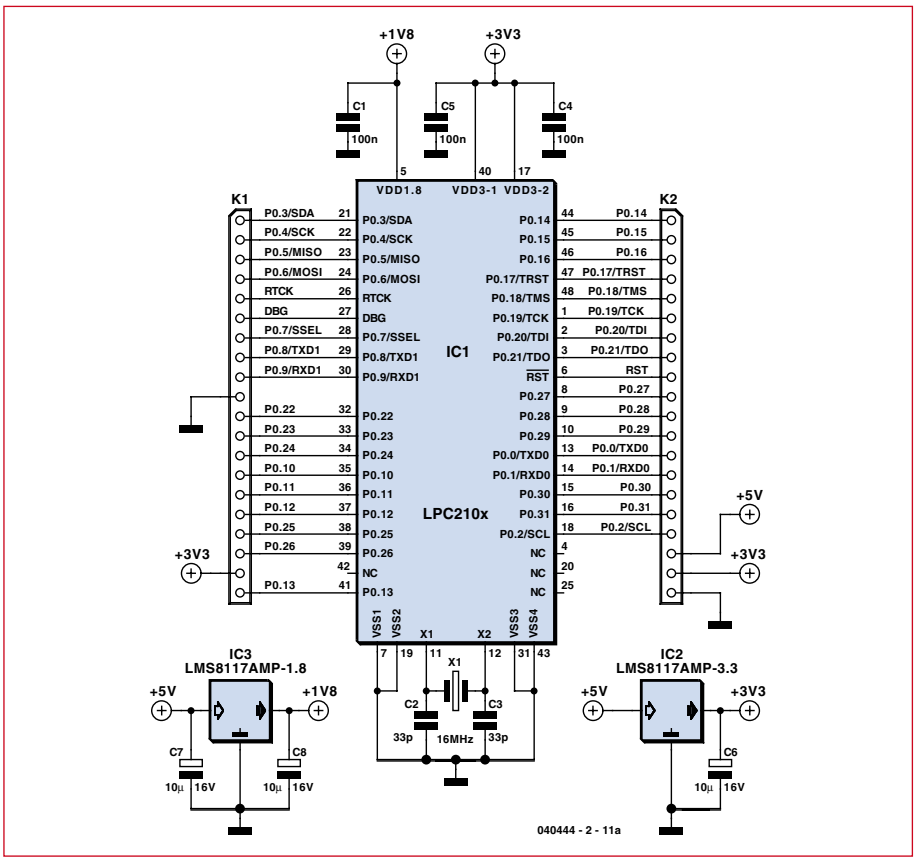


Figure 2b. Schematic diagram of the ARM module.

pied, there will be a measurable current consumption (even if it is relatively small). The measured signal is usually sent to the Command Station in the form of a binary signal.

● **Hand controller** (also called a cab controller, after the driver's cab of a locomotive). The hand controller replaces the traditional rotary knob on the transformer, and it also provides some extra functions, such as switching lights and sound effects on and off. Each controller can control its own locomotive, but it is also possible to operate several locomotives with a single controller.

● **PC**. Although a PC is not absolutely necessary, including a computer in the system adds a lot of extra scope to digital model railroading. In the simplest case, it can be used to display the track occupancy data. Naturally, a lot more is possible. There are various software packages available, both commercial and free, that can handle the entire control and protection functions of a railway. This certainly doesn't have to spoil the fun of a model railway – while you are piloting a goods train through the marshalling yard just like a real engine driver, the PC ensures that no disasters occur in other parts of the system.

● **Booster**. As a respectable model railway system can consume a considerably amount of current, one or more boosters are used to supply power to the track. If several boosters are used, they are all driven from the same Command Station. In this case, each booster supplies power to a portion of the track system.

● **Command Station**: the heart of the system. Its most important task is generating the digital signals that control the locomotives and all other digital components in the railway system. In addition, the Command Station looks after communication with one or more hand controllers, which are usually connected to a bus. The bus used here is called XpressNET, which is a polled single-master EIA 485 bus. A link to the occupancy feedback detectors is also essential. A separate bus can be used for this purpose, but in

Figure 2a. Schematic diagram of the DCC Command Station. The ARM module is fitted onto connectors K1 and K2.

DCC: what and how

A DCC signal consists of an AC voltage in which the digital information is encoded in the lengths of a series of pulses.

The half-bit time for a '1' bit is 58 µs, so the transmission time for a '1' bit is 116 µs.

The half-bit time for a '0' bit is at least 100 µs. In principle, the two half-bit times for a '0' bit should be the same to avoid producing a DC potential on the track. However, the standard allows the stretching of '0' bits in order to intentionally superimpose a DC voltage on the traction voltage. This can be used to operate an analogue locomotive without a decoder on the track at the same time. This possibility is not utilised in the Command Station described here, and the half-bit time for a logic '0' is set to a fixed value of 116 µs. The transmission time for a '0' bit is thus 232 µs with this Command Station.

Messages are sent in the form of data packets. The structure of a standard packet can be described briefly as follows:

- Preamble: a series of at least 14 '1' bits preceding the start of a new packet.
- Packet Start Bit: a '0' bit that marks the end of the preamble and the start of the address.
- Address Byte: a series of 8 bits that indicates the address (or part of the address) of the decoder for which the data is intended.
- Data Byte Start Bit: a '0' bit that marks the start of the data.
- Data Bytes: several groups of 8 data bits with '0' bits as separators between the groups. This part of the package contains the supplementary address, instructions, data, and a checksum.
- Packet End Bit: a '1' bit that marks the end of the packet.

Example:
An order for setting the direction of travel and the speed (in 28 individual steps) is constructed as follows:

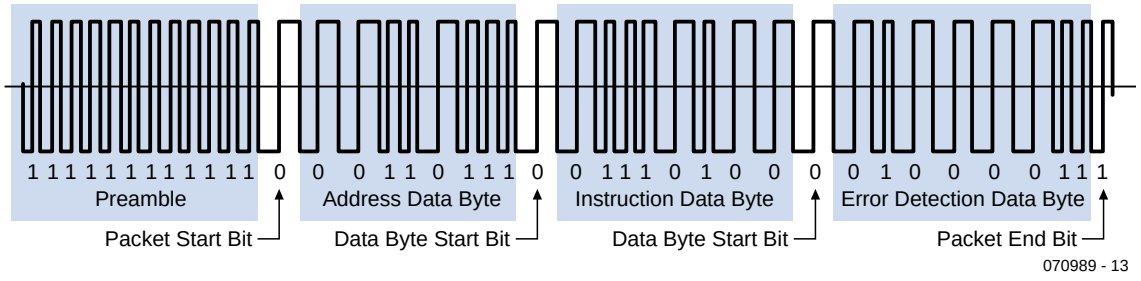
11111111111111	0	AAAAAAAA	0	01DSSSSS	0	EEEEEEEE	1
Preamble		Address		Instruction		Checksum	

AAAAAAAA	Address of the target locomotive for the instruction
D	Direction of travel (1 = forward)
SSSSS	Speed
EEEEEEEE	Checksum (XOR of the previous bytes excluding the preamble – in this case [AAAAAAAA] ^ [01DSSSSS])

A bit stream for setting locomotive number 83 to forward motion with a speed of 0 thus takes the form:

11111111111111	0	01010011	0	01100000	0	00110011	1

The signal waveform of an order could be:



A complete description of the standard, including a summary of all the instructions, is available at the NMRA website (www.nmra.org) on the DCC working group activities page

practice these devices sometimes share the cab bus. We decided to use a separate bus for this in our design: the S88 bus.

Finally, there is the link to the computer, which in this case is an RS232 port. Thanks to the use of an open protocol that is supported by a variety of software packages, expansion is problem-free.

Schematic diagram

The circuit shown in **Figure 2** is built around an LPC2106. This is a powerful 16/32-bit NXP microcontroller based on an ARM7TDMI-S processor core with a maximum clock rate of 60 MHz. The microcontroller has 128 kB of flash EEPROM and 64 kB of RAM, which are more than adequate for quite a few projects. There are also two 16C550-compatible serial ports, two versatile 16-bit timers, an I²C interface, and an SPI interface. A maximum of 32 I/O lines are available for connection to the outside world.

The microcontroller is shown in the middle of the schematic diagram in the form of a separate daughterboard, which some readers will certainly find familiar. This board (o/n 040444-1) was

used earlier in another *Elektor* design: the 'ARMee' project published in April 2005. Practically all important signals are fed out via two connectors (K1 and K2). The board is powered from +5 V. The 1.8-V and 3.3-V supply voltages required by the microcontroller are generated on-board. The 3.3-V supply is also made available via the connectors for I/O devices that require a lower supply voltage.
Unfortunately, the LPC2106 does not have an internal EEPROM. As quite a few things must be stored in non-volatile memory in the Command Station, we remedied this shortcoming by using a small external EEPROM (IC8, 8 k × 8). Fortunately, the link to the microcontroller is quite simple because we can take advantage of the microcontroller's standard I²C interface. As there is only one device on the I²C bus, addressing is unnecessary, so the E0, E1 and E2 address pins of the EEPROM are tied to ground. R12 and R13 are pull-up resistors, which are necessary for proper bus operation.

An RS232 interface is provided for communication with the computer. Half of a MAX3232 powered from the 3.3-V rail provides level shifting for the Transmit and Receive lines of UART 0. Using UART 0 for the computer link also provides a convenient way to download new software via the built-in boot loader. The pin assignments of K5 correspond to a DCE (such as a modem), so a standard one-to-one cable (available in every computer shop) can be used for the connection between the control unit and the computer.

The other half of the MAX3232 provides the level shifting necessary for driving an external track booster. This signal is fed out via K4. This connector also has a pin for a short-circuit feedback signal. If the signal on pin 1 is pulled to ground (by the booster), the control unit will disable the output.
The connectors for the hand controllers, which are driven via XpressNET, are located at the right. The hardware is a standard EIA 485 port implementation. Driver IC8 (LTC1485) provides the signal shaping. Resistor R8 ensures that the line is terminated properly. Although this is not especially critical at the data rate used here, it's better to be safe than sorry. Resistors R7 and R10 hold the Transmit and Receive lines in a defined state when they are not being driven by a device on the bus. This prevents the receivers from picking up invalid data due to interference coupled into the floating bus lines. RJ-12 6P/4C modular connectors or plug-and-socket connecters can be used to connect the hand controllers to the network.

There are two relays at the bottom middle. The contacts of relay RE2 can be used to connect the traction voltage from the booster to the main line, while the contacts of RE1 can be used to connect the booster output to a separate programming track. This relay is only engaged if the Command Station receives special programming instructions from the programming software. When this happens, relay RE2 is released. This ensures that programming instructions for a locomotive on the programming track are not sent to other locomotives on the main track as well.

This difference in the intended uses of the two relays is the reason for the additional circuitry around relay RE1. This is a simple current detector that senses the acknowledgment pulses generated by the decoder during a programming session. An acknowledgement pulse is generated by causing the current consumption to rise to at least 60 mA for a defined time (approximately 6 ms), for example by briefly switching on the motor. The presence (or absence) of these acknowledgement pulses enables the control unit to put together the response from the decoder. The decoder current flows through R16 during the programming session. This resistor thus determines the sensitivity of the detector. Four fast diodes (two pairs of diodes connected head to tail, since the track booster supplies an AC voltage) limit the voltage across the sense resistor to twice their forward voltage drop. Two diodes connected in series are used here to ensure that T1 can be driven into conduction at a suitable level. R11 and C4 form a low-pass filter that suppresses noise pulses. The LED in optocoupler IC7 is driven by the signal on the collector of T1. The collector current, and thus the current through the LED, is determined by resistor R15. The optocoupler (which provides electrical isolation from the traction voltage) supplies the ack pulses to the microcontroller.

Connector K11 is used for the JTAG port. S1 is reset button. If you want to develop your own software for this

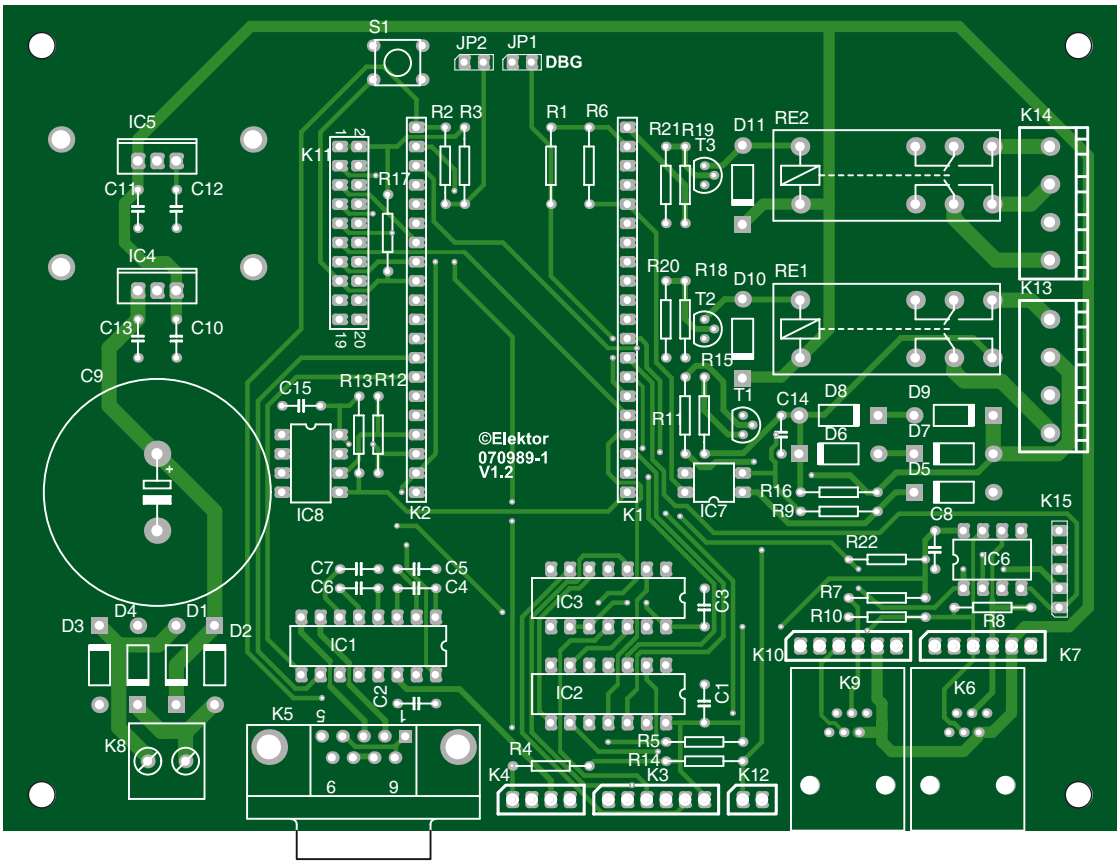


Figure 3. The PCB is designed with a spacious layout to make assembly easy.

project and you have the right tools, you can use this port for downloading and debugging the program code. It is not used during normal operation in and around a model railway.

The circuit is powered by a transformer with a secondary voltage of 13 to 14 V connected to K8. A transformer rated at 16 to 20 VA is more than adequate.

A standard model train transformer is not especially suitable for this purpose, since it has a slightly higher output voltage (14–16 V_{AC}), which will cause IC4 to become rather warm. The transformer voltage is full-wave rectified by a set of four diodes, and the DC voltage is smoothed by C9. This voltage is then reduced in two stages, first to +12 V and then to +5 V. The 12-V supply voltage is used directly to power the two relays and the hand controllers connected to the XpressNET bus. The +5-V supply voltage powers the microcontroller and its peripheral logic. Four ceramic capacitors provide effective suppression of undesirable oscillations.

COMPONENTS LIST

Resistors

R1,R3,R4,R6,R9,R14,R17-R22 = 10kΩ
R2 = 470Ω
R5 = 100kΩ
R7,R10 = 680Ω
R8 = 120Ω
R11,R12,R13 = 1kΩ
R15 = 100Ω
R16 = 8Ω2

Capacitors

C1-C8,C10=C13,C15 = 100nF
C9 = 4700µF 40V radial
C14 = 390pF

Semiconductors

D1-D4,D10,D11 = 1N4001
D5 = 1N4148
D6-D9 = BYW100-200
T1,T2,T3 = BC547
IC1 = MAX3232CPE
IC2 = 74HCT125
IC3 = 74LS125
IC4 = 7812
IC5 = 7805
IC6 = LTC1485CN
IC7 = TLP521-1
IC8 = 24C64

Miscellaneous

S1 = pushbutton, 6mm, PCB mount
RE1,RE2 = 12-V relay, 2x changeover, e.g. Finder type 40.52
Heatsink, Fischer type SK104, 25mm, for IC5
Heatsink, Fischer type SK104, 35mm, for IC4
K1,K2 = 20-way SIL socket
K3,K7,K10 = 6-way SIL pinheader
K4 = 4-way SIL pinheader
K5 = 9-way sub-D socket (female), angled pins, PCB mounting
K8 = 2-way PCB terminal block, lead pitch 5mm
K6,K9 = 6P4C modular connector, PCB mount, e.g. Hirose TM5RE1-66
K11 = 20-way DIL pinheader
K12,JP1,JP2 = 2-way pinheader
K13,K14 = 4-way PCB terminal block, lead pitch 5mm
K15 = 5-way SIL pinheader
ARM CPU module (unprogrammed); Elektor SHOP item **040444-91**
PCB, Elektor SHOP item **070989-11**
Kit of parts incl. programmed ARM module; Elektor SHOP item **070989-71** (see www.elektor.com)

The circuitry around IC2 and IC3 is used to read the data from the S88 feedback detectors. The S88 bus is designed as a giant shift register composed of the S/R flip-flops of the individual feedback detectors, which register their 'occupied/free' status until they have been read out serially. Most of this work is done by software running in the microcontroller. The contribution of the hardware to this process is limited to buffering the control signals. IC3 buffers the outgoing signals (Clk, Load, and Reset), while IC2 buffers the incoming signal (Data). The serial data is clocked in at rate of 5 kHz. With the maximum number of detectors that can be connected to the bus (4096 inputs), this means that a full set of inputs can be read 10 times each second.
This all works as follows:

- Data supplied to the feedback detectors in parallel format is loaded from the S/R flip-flops into the shift register by setting the Load signal high and issuing a clock pulse on the Clk line.
- The S/R flip-flops are reset by setting Reset High, after which they are ready to register new data.
- The microcontroller then reads in the serial data by issuing a series of clock pulses on the Clk line.

These signals are output by a 6-way connector (K3). The +5-V supply voltage on this connector is used to power the connected feedback detectors.
Finally, there is one more connector (K15) at the bottom left of the drawing. It supports an alternative bus for the hand controllers and feedback detectors. Stay tuned for more information about this.

PCB assembly

The PCB layout for the Command Station is shown in **Figure 3**. Assembling the board is quite straightforward, as there are no components that require special attention. If you allow yourself sufficient time and work attentively, you are bound to achieve a good result. Fit the low-profile components first, followed by the high-profile components. As the final step, solder the electrolytic capacitor, IC4 and IC5 with their heat sinks, and relays RE1 and RE2.

It is advisable to fit IC1, IC2, IC3 and IC6 in sockets, since they provide the interfaces to the outside world. They bear the brunt of any inadvertent short circuit, and you can replace them easily if they are fitted in sockets.
The following components are optional:

- JP1 and K11 can be omitted if you do not intend to use the JTAG interface.

- For K7/K8 and K9/K10, fit only the connector or connectors that you personally prefer. All four of these connectors are wired in parallel.

Initial use and testing

Connect the transformer to K8 and switch on the mains voltage. Check the +5-V supply voltage (e.g. between pins 5 and 8 of IC6) and the +3.3-V supply voltage (e.g. between pins 15 and 16 of IC1).

The next step is to download the application code to the DCC Command Station. You can use a standard NXP tool for this (LPC2000 Flash ISP Utility). This tool enables you to download code to the processor via the PC serial port. The instructions for using this tool are described in detail in the supplementary information for this article on the *Elektor* website. You can also download the hex code for programming the processor from the website. Alternatively, if you buy the *Elektor* kit for this project you will receive a preprogrammed module.

When the Command Station is started up the first time, it reconfigures the non-volatile data stored in the EEPROM. This can take a while (around 20 seconds), so you can take a short break before continuing with the next step.

Now connect a pushbutton with a make contact (normally-open) between the two pins of K12. This is the Stop / Go / Cold Start button.
If you press this button, relay RE2 should engage so the digital signal for driving the booster is present on connector K4. Now the DCC Command Station is ready for normal operation.
If you press the button again, relay RE2 is released, and the digital signal is no longer routed to K4.

You can also use the pushbutton to perform a system reset. If the Command Station stops responding, even after the supply voltage is interrupted,

you can restore it to proper operation by holding the button depressed while switching on the power. This causes all non-volatile data stored in the Command Station to be reset to its default values. This process takes approximately 20 seconds.

Now you're ready to connect the Command Station to the track booster. You can use a DIY booster, such as the one described already in *Elektor* (EEDTs or EEDTs Pro Booster), or you can use a commercial model (such as the Lenz LV102). The basic setup requires two leads to be connected between the Command Station and the booster: 0 V from pin 2 of K4 and the DCC signal from pin 3. The short-circuit feedback signal can be connected to pin 1 if desired. When a short circuit occurs, this pin must be pulled to ground, which causes the Command Station to disable the digital signal.

Control software

Various programs are available for using the Command Station described here with a PC to control a model railway system. There is a very nice 'native Dutch' program available called 'Koploper', which is entirely free (www.pahasoft.nl). It is supported by an active forum where users can find answers to their questions (www.koploperforum.nl). We're sure these users won't mind the odd question in English.

You can use the 'PT' program (also a Dutch product) to program DCC decoders. This program also provides extensive functions for testing feedback detectors and accessory decoders. You'll find it at http://people.zeelandnet.nl/rosoft.

You can also use ADaPT (Advanced Driving and Programming Tool) from STP Software (www.stp-software.at) to program decoders. Although this software (available in English and German versions) is not free, it can also be used for locomotive management.

Of course, you can also take on the challenge of writing your own software. A sample program for Microsoft Visual Studio Express 2008 is available and can be downloaded free of charge from the project page on the Elektor website.

Let us and other readers know how you get on — a simple message in the *Elektor* forum does the trick.

(070989-I)

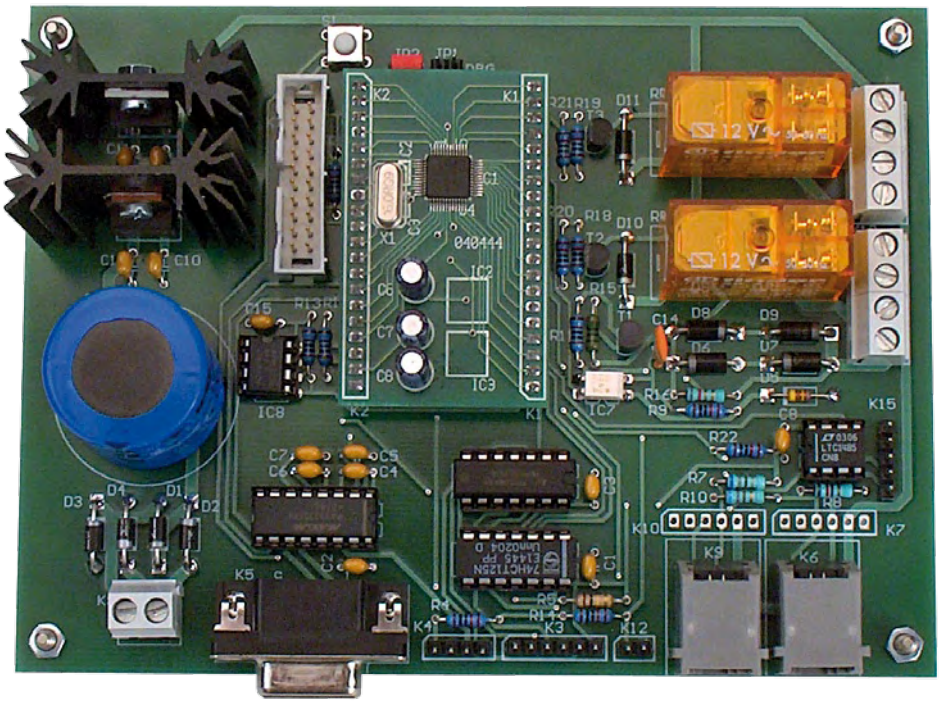


Figure 4. Fully assembled prototype with the ARMee module installed.

Web Links

DCC: www.nmra.org/standards
XpressNET: www.lenz.com
P50x: www.uhlenbrock.com
Koploper: www.pahasoft.nl
PT: http://people.zeelandnet.nl/rosoft
NXP: www.nxp.com

About the author

After the completion of his studies in Telecommunication & Microprocessors, Patrick Smout has been actively involved in electronics product development (hardware and software) since 1985. In recent years his focus has moved away form hands-on engineering, and he is now the coordinator of the development department.

His first electronic construction projects included the Elektor Junior Computer and all of its subsequently published extensions. These were complemented by DIY extensions, such as a graphics card.

His hobbies are model trains and (naturally) electronics and embedded software.

According to Patrick, the nicest thing about this profession is seeing how a design slowly comes to life, right from the incipient stage, and then watching it travel around the world as a full-fledged product.

E-mail: dcccommandstation@versateladsl.be