

# ELEKTOR ELECTRONICS

DECEMBER 2003  
£5.20

THE ELECTRONICS & COMPUTER MAGAZINE

[www.elektor-electronics.co.uk](http://www.elektor-electronics.co.uk)

**WINTER  
HOLIDAYS  
ISSUE**

**EXTRA:  
MORE THAN 40  
SMALL PROJECTS!**

**LED Christmas  
Decoration**

**P87LPC76x  
Programmer**

**Project  
Timekeeper**

**Experimental  
DRM Receiver**

**WIRELESS  
RS232  
LINK**

**using licence-  
free SRDs**



# Wireless RS232 Link

## Using licence-exempt Short Range Devices (SRDs)

Design by D. Langwald, P. Groppe and B. vom Berg

As we show in this article, off the shelf radio modules called Short Range Device (SRDs) with an integrated microcontroller are here, making home construction of a wireless RS232 link a reality for DIY constructors.

Wireless transmission of 'information', be it digital data, music or video, is sure to find wide application areas. Today, those of you with an active interest in 'over-the-air' technology have a large number of options to choose from, ranging from transmission of infrared light pulses (remote controls, IrDA) right up to more challenging fields like RF applications called DECT, Bluetooth, Wireless LAN and so on. For 'large' Windows computers you can buy ready-made modules which

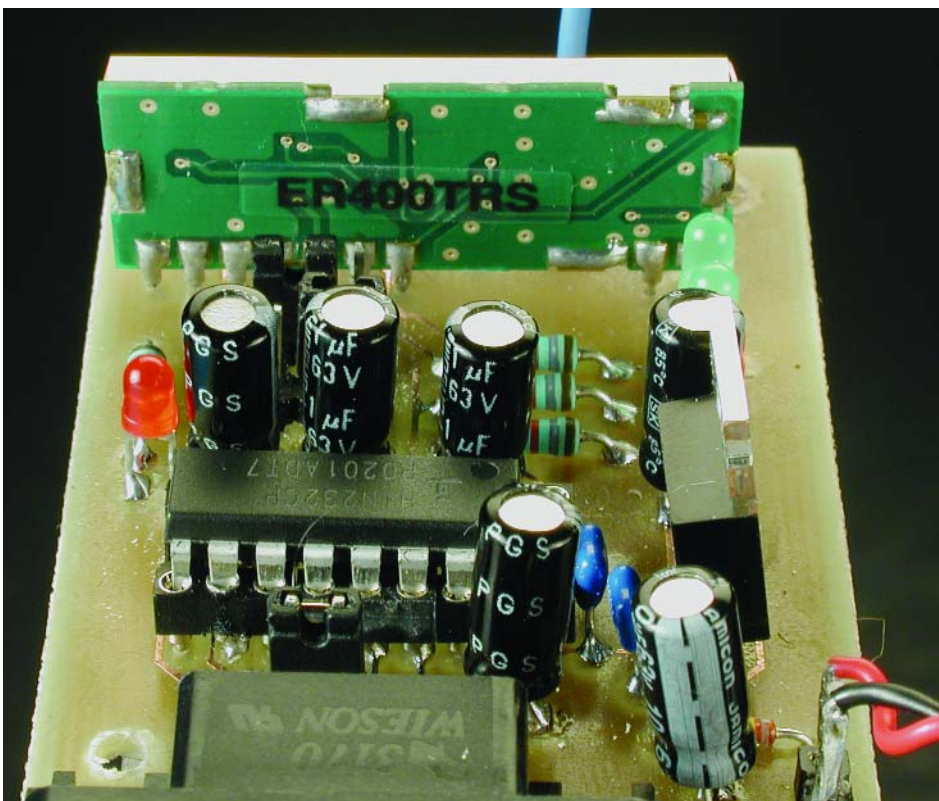
are not only easy to add to existing equipment, but also certified and complete with a manufacturer's product warranty, not forgetting a suitable driver for the software link.

The alternative road is a bit more exacting. For example, if an automation component or a measurement system (sporting just about any microcontroller) is to be upgraded with a radio link, a custom solution

involving RF components is sure to demand a lot of expertise, time and money, mostly due to three aspects:

- Developing an RF front end and transmitter for the relevant frequency range, observing the legal limits for bandwidth, spurious radiation and output power.
- Certification or type approval of the RF parts by an appointed national or international regulating authority, and paying the bill for the paperwork.
- The design of a suitable data transmission protocol to make sure that interference due to external noise and interference, moving objects, reflections, indirect paths etc., can be overcome by the system itself — the requirements in this respect are much higher than with, say, an RS232 link using the traditional cable.

An interesting alternative to the above is available in the form of cheap, off the shelf, type approved and multi-purpose radio modules operating in one of the allocated ISM band (Industrial / Scientific / Medical) and having a UART interface. In the case of the module used in our project, the 'firmware' for the data transmission process is available from the SRD manufacturer. Admittedly, controlling the SRD in software using the available firmware is not child's play, but it's not rocket science either!



Within the boundaries of the ISM bands (for example the pan-European 433.05 – 434.79 MHz band) it is possible (in many, but not all countries) to employ, licence-free, a type-approved SRD with an ERP (effective radiated power) not exceeding 10 mW.

As a matter of course, the manufacturer (but also the end user) of SRDs has to ensure that the product is compliant with rather strict regulations. In the Europe, for example, the standards for unlicensed use of SRDs in the 433/434 MHz band are laid down in EN300-220-3 and EN301489-3. Those of you interested in the legal/technical aspects of SRD type approval standards are referred to the Low Power Radio Association (LPRRA) which excels in supplying information on SRDs at all levels as well as in publishing a magazine and a website, the latter being posted at [www.lpra.org](http://www.lpra.org).

Most SRD-based radio systems employed in remote controls convey data, telemetry or alarm information across relatively small distances (usually limited to a few hundred metres 'line of sight' — LOS). In some ISM bands, the transmission of audio and video signals is allowed, too. In everyday life, SRDs are used in appliances like vehicle central locking systems, outdoor wireless thermometers, cordless mice and keyboards for the PC, wireless headphones, security cameras, and so on. The internationally used abbreviation for such radio systems used to be LPD (low-power devices) but that seems to have evolved into the more modern SRD (short range device).

The UK-based firm Low Power Radio Systems (LPRS) has a number of 'embedded-software' SRDs in its impressive range of licence-exempt radio modules. One of these, the ER400TRS 'Easy Radio' transceiver, is used in our project — its main specifications and internal organisation are given in the inset.

### Little more than the module

The circuit diagram of a complete wireless data transmission system as shown in **Figure 1** proves that no special RF experience is required to

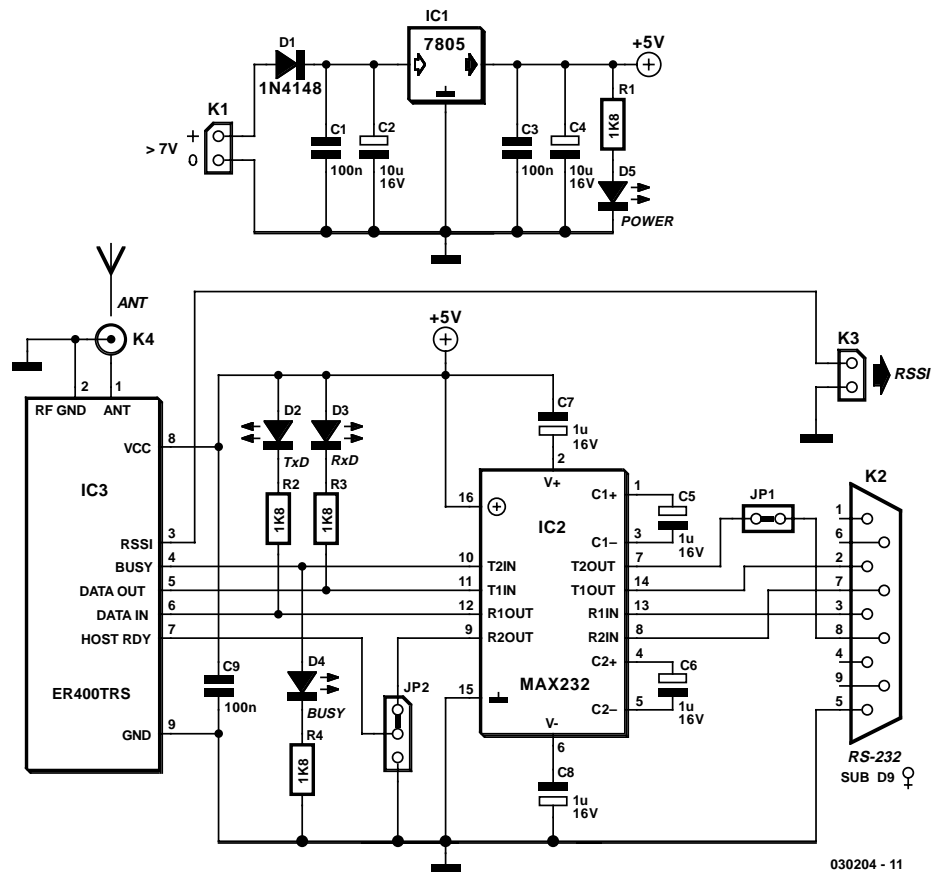


Figure 1. The external circuitry around the radio module comprises just a few parts, really.

be able to build this interesting project. The four low-power (diagnostic) LEDs in the circuit indicate its main functionality:

- LED D1: supply voltage
- LED D2: data transfer: transmit
- LED D3: data transfer: receive
- LED D4: state of BUSY line

The ER400TRS is driven with TTL signals, which forces us employ a level translator (here, a MAX232) if the radio module is to be connected to the COM port of an ordinary PC. The MAX232 (IC2) is not required in a microcontroller system if you are absolutely sure the system UART operates with TTL levels only. The transmitted and received data are quite ordinary UART characters as they occur in a standard serial asynchronous (V24) interface like a COM port on a PC or a UART in a microcontroller system.

The ER400TRS offers two additional hardware handshake control signals, whose use is not mandatory:

#### BUSY (output)

This signal indicates that the module is processing commands (reception of data, error checking) and not capable of loading transmitter data. In RS232 parlance, this corresponds to the CTS (clear to send) signal.

The following states apply:

BUSY = 1: the ER400TRS is busy and no transmit data should be conveyed to it on penalty of being ignored (= lost).

BUSY = 0: The ER400TRS is ready to accept data and actually transmit them.

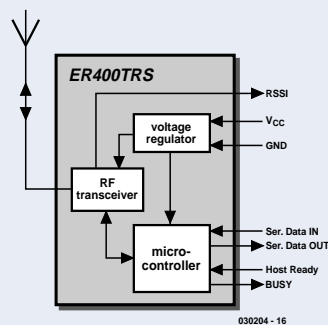
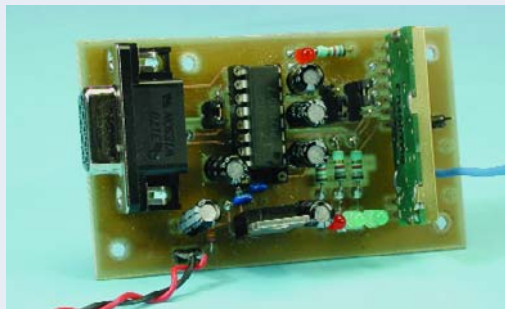
Jumper JP2 allows the BUSY signal to be applied to the RS232 converter and from there, to the host for processing by the PC software. Alternatively, the signal may be just visually signalled by LED D4. If this control signal is not actually processed by software or hardware, then provision must be made to ensure the ER400TRS is really ready to load and subsequently transmit data. In some cases, this means that delays following transmission and reception should be built into the software, and be strictly maintained until the next data block is sent.

Jumper JP3 enables the BUSY signal to be

## ER400TRS Quick Data

Datasheet at: <http://www.lprs.co.uk/main/viewdatasheet.php?datasheetref=112>

- Half-duplex FM transceiver
- 10 programmable channels within 433.25 MHz and 434.35 MHz (70 cms pan-European ISM allocation)
- 50-Ω antenna connection
- RF output power programmable in 10 steps between 1 mW and 10 mW
- Range 250 m line of sight (LOS) or 30 m in buildings
- Data speed on host link: 2.4 to 38.4 kbits/s, adjustable in five steps
- Fixed 19.2 kbits/s data speed on RF link
- Two hardware handshaking signals
- RSSI (received signal strength) analogue output
- Supply voltage 3.3 V to 5.5 V
- Current consumption at 5.5 V supply;
  - transmitting (10 mW): 23.0 mA
  - receiving: 17.0 mA
  - standby: 2.0 mA
- Dimensions: 37.5 x 14.4 mm



removed from the sub-D plug pin, which may be necessary to prevent signal conflicts or contention.

### Host Ready (input)

This signal tells the module whether or not the host is ready to load data from the receive buffer in the ER400TRS. Again, in RS232 terms this equates to an RTS (ready to send) signal, with the following logic states:

Host Ready = 0: Host CPU is ready to load and process the data that was just received. The module should respond by releasing the data to the host.

Host Ready = 1: Host CPU not ready to load and process the data that was just received. In this case the radio module should keep the data in its internal receive register, taking into account that the data are erased no sooner than 2.5 seconds after their reception. Within that period, the host must have fetched the data (i.e., pulled Host Ready to '0'), else they are lost.

When the host can be relied on to immediately fetch and process the received data, or if it has a sufficiently large receive buffer, it is safe to permanent tie the Host Ready input to ground using jumper JP1. In this way the ER400TRS is encouraged to immediately transmit received data. If, on the other hand, the host is not quick to process received data, it may control this module input via its serial

interface and jumper JP1 in accordance with its timing requirements.

Finally, the ER400TRS puts analogue information regarding received signal strength at your disposal.

### RSSI

#### (received signal strength indicator)

The transceiver has a built-in RSSI (Received Signal Strength Indicator) that provides an analogue output voltage that is inversely proportional to the RF energy present within the pass band of the receiver. It ranges from 0 V (maximum signal, -50dBm) to 1.2 V (minimum signal, -105 dBm) and has a slope of approximately 50 dB/Volt. This analogue output signal should only be connected to a high impedance load (>100 kΩ) and can be used to provide a measure of the signal strength and any interfering signals (noise) within band during the installation and operation of systems.

The hardware links between the host(s) and the radio modules are depicted in **Figure 2**. Because the ER400TRS operates in half-duplex bi-directional mode, both Host A and Host B can alternately transmit and receive.

## Easy Radio embedded software

The firmware running inside the ER400TRS module, and in particular the routines that look after the data transmission protocols, are proprietary, i.e., have been developed by LPRS themselves for their own products, hence do not reflect any kind of international standard. The software that does it all is remarkably simple, as well as easy to use from the outside. After all, when you are looking at 'just' a fast connection using simple microcontroller systems, no complex memory-hungry protocol stack is required like the one required by TCP/IP.

The embedded software of the ER400TRS is dubbed 'Easy Radio'. It handles three important functions:

1. parameter setting on the interface to the host (PC);
2. parameter setting on the RF parts;
3. executing the data transmission protocol.

The various parameters are loaded using ASCII command sequences listed in the Easy Radio software documentation. For example, the transmitter output power is stepped down to 5 mW by sending the command string ER\_CMD#P5 to the radio module. An overview is given in the text box 'Easy Radio Functionality'. On [www.lprs.co.uk/download/LPRS\\_kindly\\_supply\\_a\\_Windows-savvy\\_software\\_tool\\_that\\_allows\\_all\\_ER400TRS\\_parameters\\_to\\_be\\_given\\_the\\_desired\\_values,\\_and\\_convey\\_them\\_to\\_the\\_transceiver\\_module](http://www.lprs.co.uk/download/LPRS_kindly_supply_a_Windows-savvy_software_tool_that_allows_all_ER400TRS_parameters_to_be_given_the_desired_values,_and_convey_them_to_the_transceiver_module).

### Host interface parameters

The serial asynchronous interface may be used at one of five different (but very common) baud rates (2,400 to 38,400 bits/s). Characters are invariably transmitted and received in UART format. This is only applicable to the relevant host, with the ER400TRS running at a factory-determined speed of 19,200 bits/s. Hence the data speed on the radio link will always be 19.2 kbits/s.

### RF section parameter

Here users have the opportunity to select one of ten available radio channels (frequencies) between

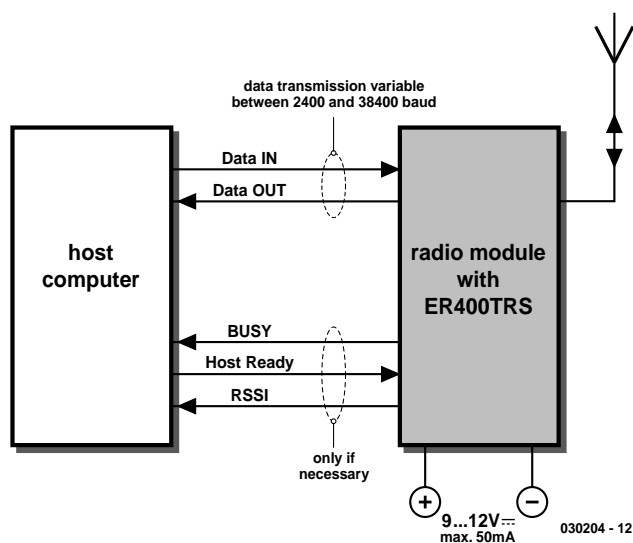


Figure 2. Basic configuration of a radio station for data transmission and reception.

433.25 MHz and 434.35 MHz, where the transmit and receive frequencies are always equal (i.e., no offset or shift).

The transmitter output power can be adjusted between 1 mW and 10 mW in ten steps. This may be useful to conserve battery energy when the module is used to cover a relatively small distance.

#### Data transmission protocol execution

The ER400TRS comprises a shared 128-byte transmit/receive buffer, via which the traffic is handled as follows:

First, the Host transmitter CPU checks if BUSY (= CTS on an RS232 link) is Low. Alternatively, it should idle a fixed time after the last trans-

mission, until the ER400TRS flags that it has finished its internal tasks, thus allowing the host CPU to release its data. Next, the data are written into the module's internal buffer, where they remain latched.

The data are transmitted when either the buffer is full after continuous reception of 128 bytes (if more are sent to the chip on one go, the excess ones are discarded), or if a pause occurs in the datastream after a byte is conveyed where the pause length is greater than two byte lengths. In this way, individual bytes may be 'broadcast'.

Before the data leave the transmitter a CRC (cyclic redundancy check) is run to generate a CRC byte which is added to the 'message' bytes, together with a preamble and

some more additional information (for example, the number of data bytes). Next, all bytes are Manchester-encoded and finally transmitted. During these activities, the BUSY line of the ER400TRS is logic High.

At the receiver side the data are Manchester-decoded, the extra information is separated and an error check is performed. During these activities, the BUSY line of the ET400TRS is logic High. In case data corruption is detected, all data is rejected and the receiving host does not get any data.

However, the transmitting host also does not get acknowledge information about the rejected 'telegram'. If it is required for the host to have confirmation that all data has been correctly received, then an acknowledge process has to be implemented using the ER400TRS firmware.

If all data has been received okay, the data are copied to the transmit/receive buffer in the receiving ER400TRS. Next, the chip waits for the receiving host to pull Host Ready Low (= RTS in RS232 terms). Following this the data are automatically and continuously fed to the host via the serial link. If this type of handshaking is not required, the Host Ready input may be permanently tied to ground using jumper JP2.

In case the Host Ready signal is not pulled low within 2.5 s from reception of the data, all received data are wiped from the buffer and the ER400TRS is ready for transmission or reception.

The data processing inside ER400TRS modules is invisible to the two hosts — all they need to do is supply and receive data via their RS232 ports.

## Antennas

No specialist knowledge is required on RF technology if you want to set up SRD-based data link like the one described in this article. There's one exception, though: the antenna.

The transceiver can be used with the various common types of antenna that match the 50 Ω RF Input/Output such as a whip, helical or PCB/Wire loop antennas — see **Figure 3**.

Whip antennas are resonant with a length corresponding to one quarter of the electrical wavelength ( $\lambda/4$ ). They are very easy to implement and can simply be a 'piece of wire' or PCB track, which at 433 MHz should be about 15.5 cms in length. This should be straight, in 'free space' (kept well away from all other circuitry) and should be connected directly to the Antenna pin of the transceiver. If the antenna is remote, it should be connected via a 50-Ω coaxial feeder cable (RG58U, RG174U). The quarter-wave antenna has the widest range of the three types dis-

## Easy Radio Functionality

The embedded microcontroller looks after the following Easy Radio functions:

- processing input and output data using a standard UART format (1 start bit, 8 databits, no parity, 1 stop bit)
- Manchester encoding/decoding of data for/from radio link
- Calculation and comparison of CRC checksum
- Radio protocol monitoring and control: transmission of preamble and sync bytes, removal of these bytes when receiving
- Programming of receive/transmit synthesizer for channel selection
- RF output power programming
- Operation of the external host interface at the desired baud rate
- Storage of up to 128 bytes for receive/transmit buffer
- Operation of the two handshaking signals
- Operational parameter back up in on-board EEPROM

cussed here, but only if it 'sees' a sufficiently large ground plane beneath it. Its disadvantage is mainly mechanical vulnerability and sensitivity to large metal objects or surfaces in its vicinity. It is also rather cumbersome in semi-portable equipment.

PCB Loop antennas are the most compact antennas but are less effective than the other types. They are also more difficult to design and must be carefully 'tuned' for best performance. At 433 MHz the loop should cover an area of 400-1000 mm<sup>2</sup> and be tuned to reso-

nance using a small (1.5-5 pF) capacitor. The PCB track acting as the loop may be as narrow as 1 mm. The feeder point should be at 15-25% of the overall length of the loop.

Helical antennas are also resonant and generally chosen for their more compact dimensions. They are more difficult to optimise than quarter wave antennas and are critical with regard to surrounding objects that can easily 'de-tune' them. They

operate most efficiently when there is a substantial ground plane for them to radiate against. For our project, a helical antenna can be made from 0.5-mm dia. (26 SWG) enamelled copper wire (ECW). Suitable dimensions are:

- 17 turns, 5 mm internal diameter, stretch to 34 mm length;
- 24 turns, 3.2 mm internal diameter, stretch to 19 mm length.

As to range and susceptibility to

## An Application using the Elektor 80C537 Board

Two built up radio modules are needed to realise the example application depicted in **Figure A**. At one side we find Host A, an 8051 controller system based on the 1997 Elektor Electronics 80C537 Microcontroller Board connected to an SRD radio module via the second serial interface of the 80C537 micro. A MAX232 level converter is not required here. On the radio module, the relevant pins are simply interconnected with wires (pin 11 to pin 14; pin 12 to pin 13). The development PC allows the software for the 80537 to be set while also acting as a display device for the received characters.

The other party is formed by Host B, a regular PC connected to a radio module via its serial port. At this side of the radio link, a MAX232 level converter is required. The PC initially runs an ordinary terminal emulation program (like HyperTerminal) just to display received characters and to send out (individual) characters.

Different applications were realised using this configuration, and the relevant software (C51 for the 80537 and Visual BASIC for the PC/Notebook) may be downloaded free of charge from the Elektor Electronics webserver at [www.elektor-electronics.co.uk/dl/dl.htm](http://www.elektor-electronics.co.uk/dl/dl.htm) (select this issue).

1. Individual key presses are transmitted by Host B, received by the 80C537 system and get displayed on the monitor of the development PC.
2. Host A sends individual characters or whole text strings — these appear on the monitor connected to Host B.
3. The 80C537 system is turned into a small weather station by extending it with sensors and/or actuators and an LC display. It employs the radio link to convey to Host B meteorological data like air pressure, temperature, relative humidity, etc. complete with time stamps. Host B runs a small Visual BASIC program displaying received data and writing an Excel compatible (log) file for inspection/evaluation at a later time. The system also allows texts to be sent from Host B to the LC display connected to the 80C537 system. **Figure B** shows a screendump of the program running on Host B. The hardware should also allow the control (by radio) of just about any actuator at the 'remote' site.

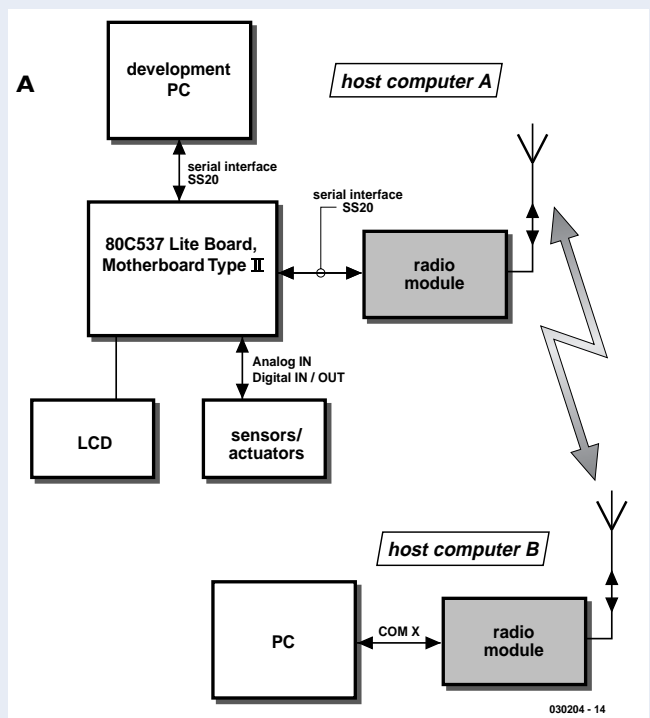


Figure A. A PC and a microcontroller system communicating over a radio link.

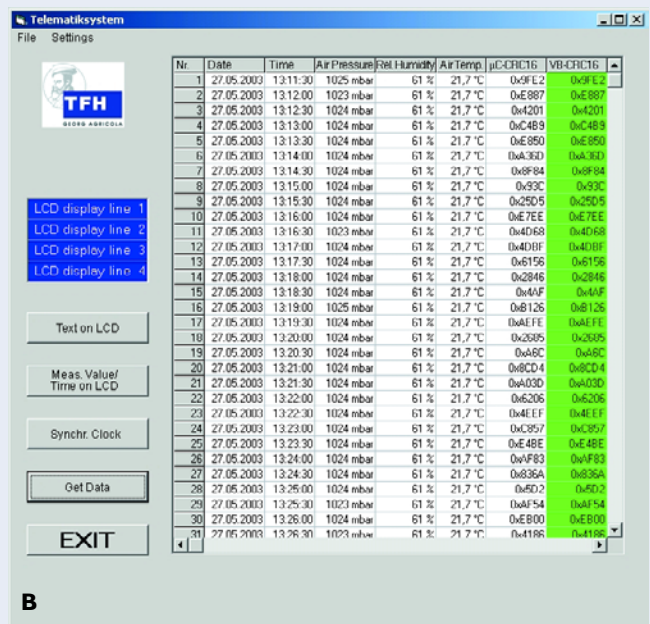


Figure B. Screendump of the weather station software running on Host B.

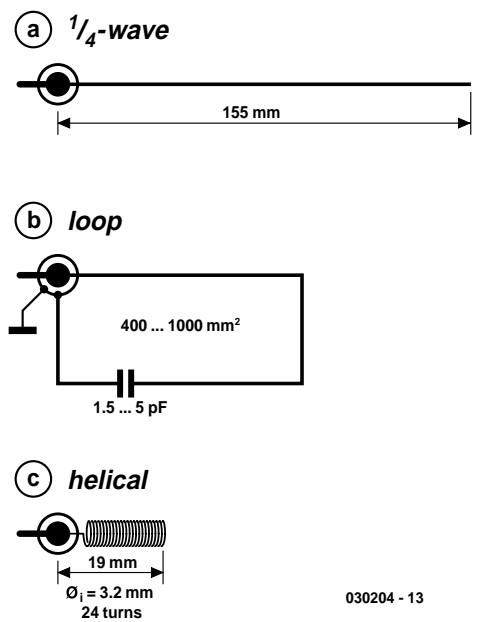
## Literature and Internet addresses:

- [1] LPRS: [www.lprs.co.uk](http://www.lprs.co.uk)
- [2] ER400TRS hardware manual:  
<http://www.lprs.co.uk/main/viewdatasheet.php?datasheetref=112>
- [3] Easy Radio software manual:  
[http://www.lprs.co.uk/pdf\\_directory/23861055851852.pdf](http://www.lprs.co.uk/pdf_directory/23861055851852.pdf)
- [4] 80C537 Microcontroller Board, Elektor Electronics June 1997.
- [5] Portal website for the SRD industry and radio regulation authorities:  
[www.lpra.org](http://www.lpra.org)

noise, the performance of the helical antenna lies roughly between that of the whip and the loop. It is, however, the least space consuming of the three. Moreover, it is easily tuned by stretching and compressing the coil until the greatest range is obtained.

## PCB and practical use

The printed circuit board for one radio module (Figure 4) is double-sided hence very compact. Populating the board should not present problems if you pay due attention to



030204 - 13

Figure 3. Pick your antenna.

## COMPONENTS LIST

### Resistors:

R1-R4 = 1kΩ8

### Capacitors:

C1, C3, C9 = 100nF  
C2, C4 = 10μF 16V radial  
C5-C8 = 1μF 16V radial

### Semiconductors:

D1 = 1N4148  
D2, D3, D5 = LED, green, 3mm, low current  
D4 = LED, red, 3mm, low current

IC1 = 7805CP  
IC2 = MAX232CP  
IC3 = ER400TRS from LPRS (see text)

### Miscellaneous:

JP1 = jumper with 2-way pinheader  
JP2 = jumper with 3-way pinheader  
K1, K3 = 2-way pinheader  
K2 = 9-way sub-D socket (female), angled pins, PCB mount  
K4 = BNC socket, 50Ω, PCB mount (Farnell # 365-0558)  
RS232 cable (non crossing)  
PCB, order code **030204-1** (see Readers Services page)

the correct fitting of the polarized parts.

To close off this article, we should reiterate that the use of SRDs is governed by strict regulations. On request, LPRS supply customers with a copy of the CE compliance document for their SRD modules, as well as pointers to legal information relevant to the use of SRDs in several countries.

(030204-1)

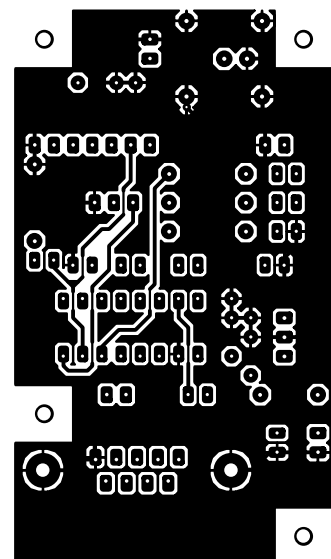
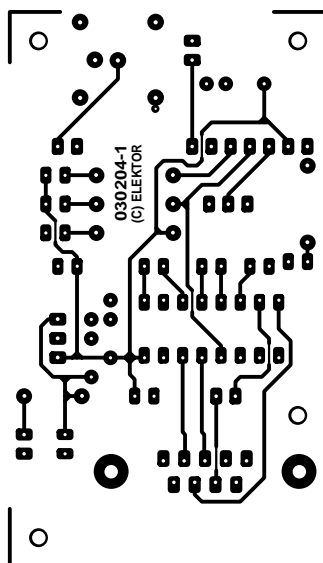
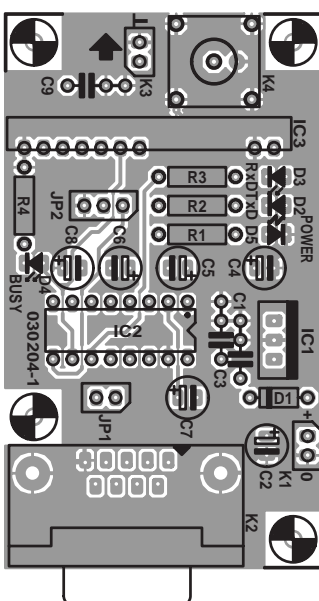


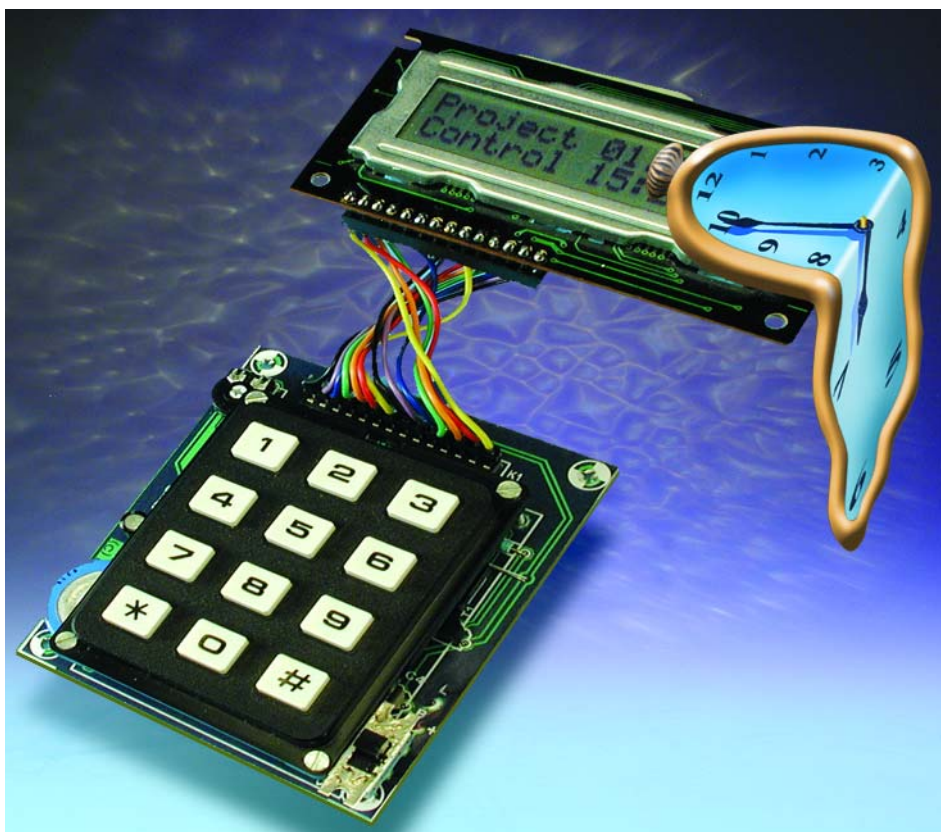
Figure 4. The PCB for the Wireless RS232 Link project is double-sided and easy to build up (board available ready-made).

# Project Timekeeper

Keep track of several projects at once using a PIC16F84

Design by W. Wätzig

This special timekeeper records individual time periods and adds them together to produce a total time. The circuit uses a real-time clock device, backed up by a 'Goldcap' capacitor.



time of (household) equipment, or the length of an experiment.

The circuit (**Figure 1**) is based around the familiar PIC16F84 8-bit microcontroller from Microchip and the Dallas Semiconductor DS1302 real-time clock (RTC). Information is displayed on a liquid crystal display with 2 rows of 16 characters. Control is via a telephone-type keypad with the digits '0' to '9' as well as star ('\*') and hash ('#'). The keypad is arranged as a matrix with three columns and four rows and is connected to port B. Buzzer B1 is also connected to this port via T1, and gives audible confirmation of each keypress. RA4 enables the keypad and buzzer. Port line RA2 enables and disables the display, while the display contrast can be adjusted using P1.

## Operation

Time measurement for project *N* is started by pressing '\*N'. *N* can be from 1 to 9. This causes the time to be read from the real-time clock and copied into EEPROM in the micro-processor. The display shows the value of *N* along with the date and time. The device can now be turned off: the RTC will continue to be powered from the reservoir capacitor.

Timing for the current project is stopped by pressing '##'. The time is again read from the RTC and the stored start time is subtracted from

When working on a project the time spent in each individual work session often needs to be recorded. Written notes may be made as follows:

- Period 1: October 7 from 8:13 to 8:55
- Period 2: October 7 from 8:56 to 9:32

and so on. The differences between the recorded times must be worked out, and then

added together to arrive at the total time spent. The device described here was developed to simplify this task. It can keep track of the total time spent on up to nine separate projects. Indeed, it can be used to measure time durations in any application as long as no great precision is required: for example, the duration of a competition, or the operating



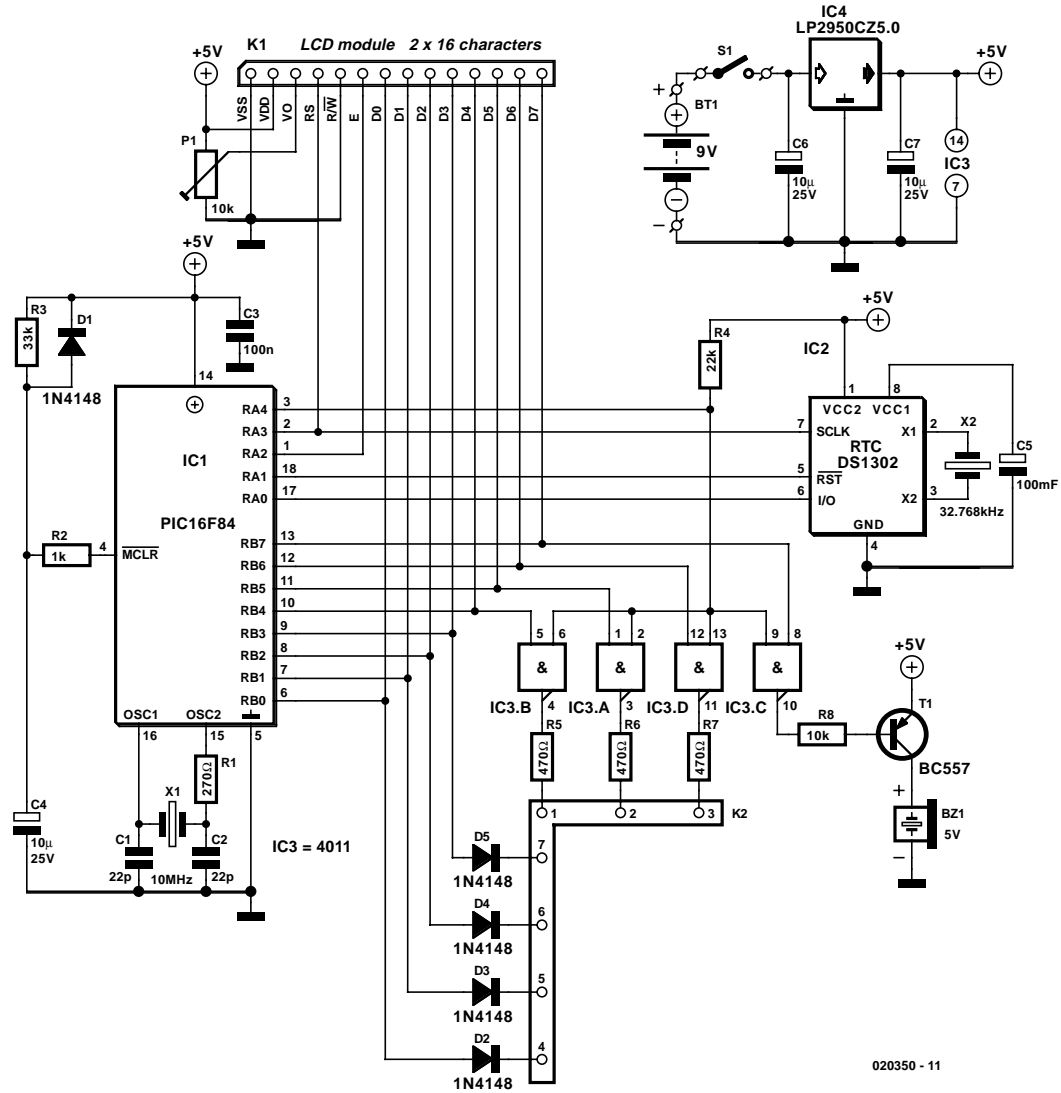


Figure 1. The circuit is based around the familiar Microchip PIC16F84 8-bit microcontroller and the Dallas Semiconductor DS1302 real-time clock (RTC).

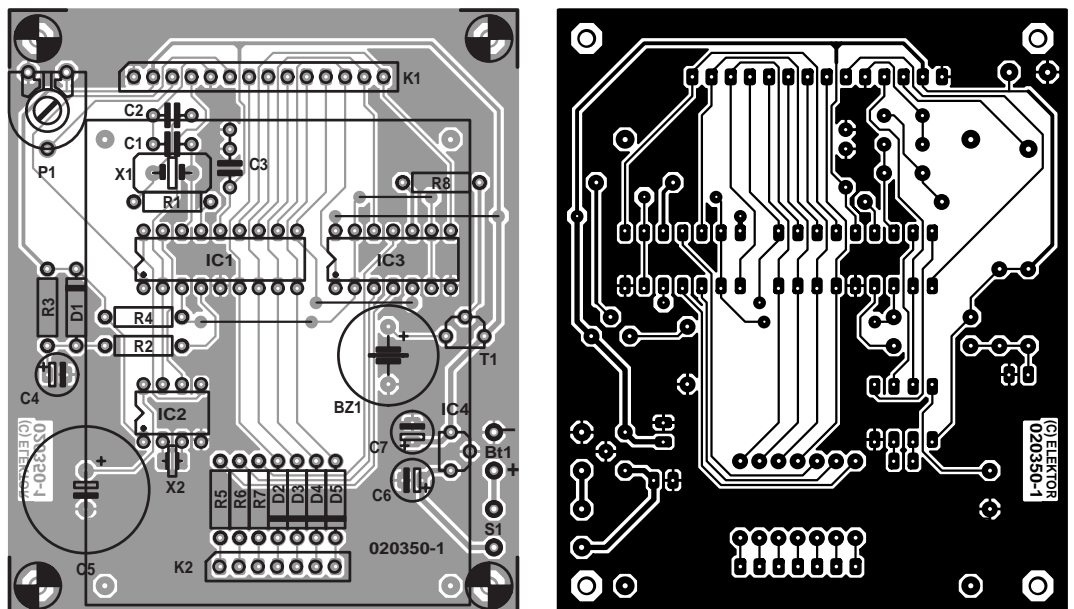


Figure 2. Printed circuit board layout and component mounting plan for the project timekeeper.

## COMPONENTS LIST

### Resistors:

R1 = 270Ω  
 R2 = 1kΩ  
 R3 = 33kΩ  
 R4 = 22kΩ  
 R5,R6,R7 = 470Ω  
 R8 = 10kΩ  
 P1 = 10kΩ preset H

### Capacitors:

C1,C2 = 22pF  
 C3 = 100nF  
 C4,C6,C7 = 10μF 25V radial  
 C5 = 100mF 5.5 V Goldcap

### Semiconductors:

D1-D5 = 1N4148  
 T1 = BC557  
 IC1 = PIC16F84-10P, programmed, order code **020350-41** (see Readers Services)  
 IC2 = DS1302  
 IC3 = 4011  
 IC4 = LP2950CZ5.0 (low-drop voltage regulator, 5V, TO92 case)

### Miscellaneous:

K1 = 14-way pinheader  
 K2 = 7-way socket strip  
 BZ1 = buzzer, 5 V<sub>DC</sub>  
 BT1 = 9V battery (6F22) with clip-on leads  
 X1 = 10MHz quartz crystal  
 X2 = 32.768kHz quartz crystal  
 S1 = on/off switch  
 2 x 16 character alphanumeric LCD Module  
 Telephone keypad (4x3 matrix), e.g., Conrad Electronics # 709840

PCB, available from The PCBShop

Disk (source & object code), order code **020350-11** or Free Download

## Table 1

### Control commands

<b>**N</b>	start project number N			
<b>##</b>	stop current project			
<b>#0*N</b>	clear time information recorded for project N			
<b>N</b>	show time information recorded for project N			
<b>##n</b>	set RTC register n	n=0	seconds	xx = 00-59
		n=1	minutes	xx = 00-59
		n=2	hours	xx = 00-23
		n=3	date	xx = 01-31
		n=4	month	xx = 01-12
		n=5	day of week	xx = 01-07
		n=6	year	xx = 00-99
<b>*#n</b>	read RTC register n			



it. The result is added on to the accumulated time for the relevant project stored in EEPROM. The result of the calculation is displayed in the format **days/hours:minutes**.

By pressing one of the number keys from 1 to 9 the accumulated time for the corresponding project is called up on the display for three seconds. It is shown in the format **dd hh mm**.

**Table 1** shows a list of the commands recognised by the project timekeeper. When first switched on, the time and date must be set. It is recommended to set the year first, followed by the month, day, hours, minutes

and seconds, in that order. Note also that the seconds register should be set to **zero** in order to start the RTC. After a reset (i.e., when power is first applied), the RTC sets the seconds register to **80** and enters a 'wait' mode.

The current consumption of the device is around 4 mA, and so a 9 V PP3-type battery should give a long life. Voltage regulator IC4 should be a low drop-out type in order to squeeze the last milliamp-hour out of the battery.

## Construction

A generously-proportioned printed circuit board (**Figure 2**) has been designed for the project timekeeper. There should be no particular difficulties encountered during construction, as long as care is taken to fit the polarised components the right way around. The three ICs can be fitted in sockets. The keypad matrix should fit neatly into seven-way connector K2, and the LCD in K1. This results in a compact unit which can be built into a small plastic enclosure.

(020350-1)

# FPGA Development Board

With a 300-Kgate Xilinx SpartanIIE device

By G. May DL3ABQ

In sharp contrast with plummeting prices of FPGA chips, suitable development systems for these interesting devices are costly and hard to find. However, that situation seems to have changed for the better with the arrival of a low-cost kit from Australia-based BurchEd.

The FPGA (Field Programmable Gate Array) is a programmable logic building block for use in complex digital circuits. Using either a hardware descriptive language or a graphics editor, users of FPGAs are able to design logic circuits that can be downloaded into an FPGA chip. The result (hopefully) is the FPGA acting as the desired logic configuration in all its complexity.

As may be expected, FPGAs contain a massive amount of universal elements consisting of gates and/or flip-flops, depending on the FPGA chip you have selected for the job. These elements may be interconnected to form configurations, hence the ability of an FPGA to perform countless digital functions designed by you, the user.

Most FPGAs are manufactured in SRAM (static random access memory) technology, and that is why they lose their configuration when the supply voltage disappears. To enable FPGAs to resume their normal task when the supply voltage returns, special configuration EPROMs are employed from which an FPGA can load a set of configuration data to enable it to carry out its programmed function. The **Xilinx PROM Programmer** described in the October 2003 issue of *Elektronics* is just the ticket for this sort of functionality.

Sure, there are also non-volatile FPGAs (for instance, 'Antifuse' types), but these devices

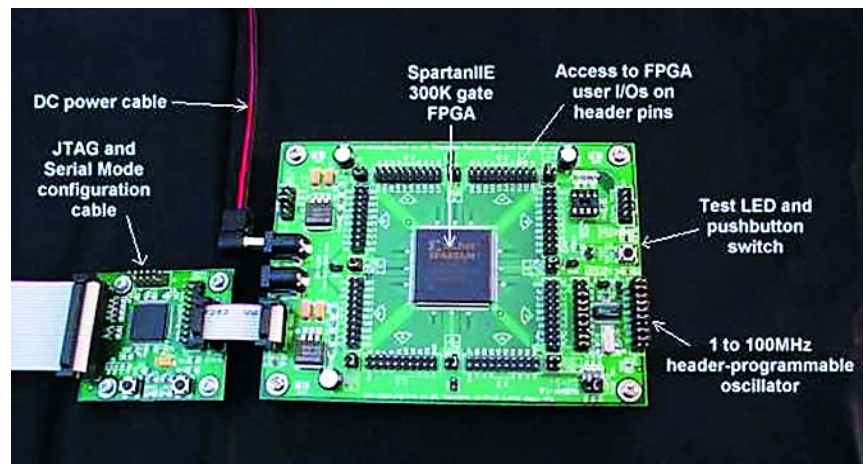


Figure 1. FPGA development board B5-X300 from BurchEd.

may only be programmed once. Leading manufacturers of FPGAs include Xilinx, Altera and Actel.

The CPLD (Complex programmable Logic Device) is another example of a programmable logic component. CPLDs contain several large logic structures not unlike those found in GALs (Generic Array Logic). It is therefore safe to say that a CPLD consists of an array of GALs. CPLDs generally employ non-volatile EEPROM cells for their programmable

elements. The main players in the CPLD arena are Lattice, Xilinx and Altera.

Depending on the size of the FPGA or CPLD, entire processors or similarly complex functions may be realised using their incredibly versatile elements. In the industry, these devices are not only used to build prototypes (in a very short time), but also at the heart of so-called reconfigurable systems, a prominent application range being signal processing.

The performance of such a component is usually expressed in 'gate equivalents', describing the number of theoretically available gates deep inside the FPGA. These figures should be used with care, however, because for reasons of product marketing they tend to be on the optimistic side.

Typical sizes of CPLDs are 1,000 to 50,000 gate equivalents, or anything up from 10,000 to a few million in the case of FPGAs.

Although smaller components have become available at relatively low cost, higher integration is still inevitably coupled to higher prices. To electronics designers, these new logic devices are an attractive alternative to using literally dozens of conventional logic ICs. Also, when the processing speed of a microcontroller is not sufficient for a certain application, programmable logic is a viable alternative.

## Development Board

The Australian manufacturer Burch Electronic Designs (BurchEd) [1] offers a cost effective development system for FPGAs (**Figure 1**). BurchEd's B5-X300 is based on a Xilinx FPGA from the SpartanIIIE series sporting a massive 300,000 gate equivalents. The system come with a download adapter for direct and easy connection to the printer port (Centronics) on your PC. This interface serves to convey configuration data from the PC to the FPGA chip.

The board proper contains a clock oscillator whose output frequency is set using jumpers. Also on the board is a socket for the configuration EPROM and pinheaders for easy access to the FPGA pins.

Besides the FPGA board, modules are offered that could easily form part of an application circuit. The following modules are currently offered:

- switch board
- LED board
- 7-segment displays
- memory extension
- interconnection board for RS232, mouse, keyboard and VGA
- CompactFlash interface
- IDE interface
- Flash configuration board

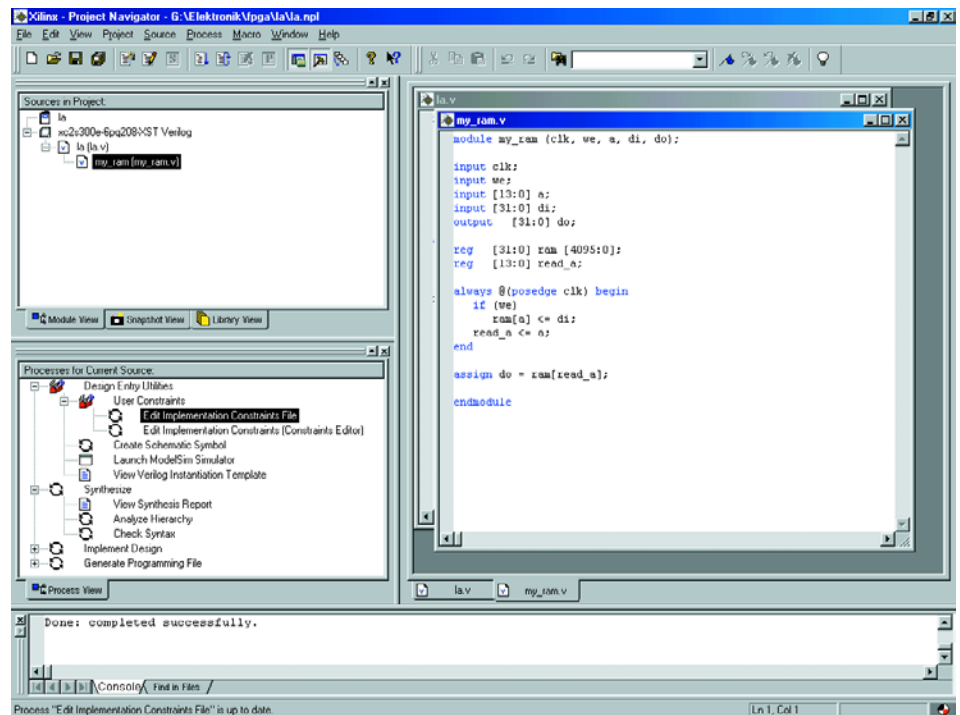


Figure 2. Xilinx FGA programming software in action.

The software development environment used with the BurchEd kit is Xilinx' **Webpack** freeware, which allows the component to be programmed in a number of different ways (see **Figure 2**). The software may be downloaded from the Xilinx website [2].

The BurchEd board is very simple to operate. In fact, the user guidelines found on the company's homepage should enable even inexperienced users of FPGAs to load his/her first program into the chip and make a LED flash on the board!

We are informed that BurchEd do not (yet) have a UK distributor so you have to contact them directly in Australia or via their French distributor, Hi Tech Tools [5]. The board costs AUS\$ 330.00, US\$ 215.00, £ 135.00 or € 195.00.

## Application examples

Unlike any of its competitors in the UK electronics magazine market, *Elektor Electronics* has a claim to several published, tried and tested projects employing configurable logic devices like GALs and CPLDs. Burch Electronic Designs on their homepage [1] offer a number of example programs for free down-

loading. Useful as they are to get started with FPGA technology, the programs are rather scanty. However, according to Tony Burch they will be extended in the near future so have a look by the time you read this.

A particularly interesting theme in relation to FPGAs covers the development of one's own processors. A large collection of material with specific relevance to this fascinating subject may be found at [3].

Finally, as you may have noticed already, a growing number of electronics designers is busy migrating old arcade-style games (say, Pacman) from their antediluvian platforms onto FPGAs. A wonderful website on this subject may be found at [4] where full details on the implementation may be downloaded.

(030216-1)

### Contact information

Burch Electronic Designs,  
PO Box 1548, Macquarie Centre, North Ryde,  
NSW 2113, Australia.  
Tel: +61 2 9614 3358,  
fax: +61 2 9614 3889.  
Internet: [www.BurchED.com](http://www.BurchED.com)

### Internet addresses

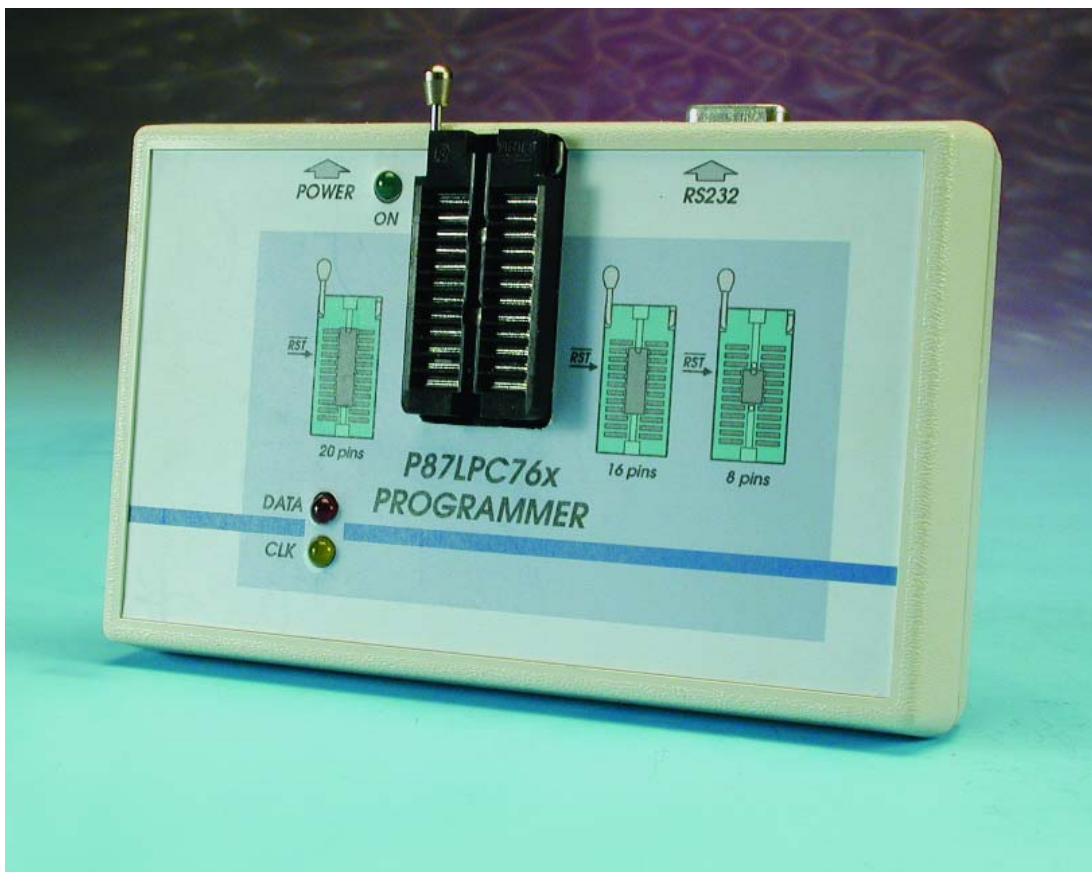
- [1] [www.burched.biz](http://www.burched.biz)
- [2] [www.xilinx.com/xlnx/xil\\_prodcat\\_landing-page.jsp?title=ISE+Webpack](http://www.xilinx.com/xlnx/xil_prodcat_landing-page.jsp?title=ISE+Webpack)
- [3] [www.fpgacpu.org/](http://www.fpgacpu.org/)
- [4] [www.fpgaarcade.com/](http://www.fpgaarcade.com/)
- [5] <http://www.hitechtools.com>

# P87LPC76x Programmer

A simple burner for a popular microcontroller family

Design by P. Luyckx

The programmer described in this article has been designed for the well-known P87LPC76x family of microcontrollers manufactured by Philips. These are 8-bit devices with an 8051-derived core and serial programming capability. The hardware we've in mind excels in simplicity and the programmer software may be obtained free of charge from our website.



The popularity of the 8-bit microcontrollers from the P87LPC76x series is mainly due to their versatility and ease of use. On the pros and cons of these devices, we are informed that programmers seem to value the limited complexity, the large number of outputs, the ample output current specification and (last but not least) the presence of an I<sup>2</sup>C bus. The chip can be programmed using a serial link, which means that in-system programming (ISP) is within easy reach. Having ISP on a microcontroller is a clear advantage to equipment manufacturers because it allows them to blow

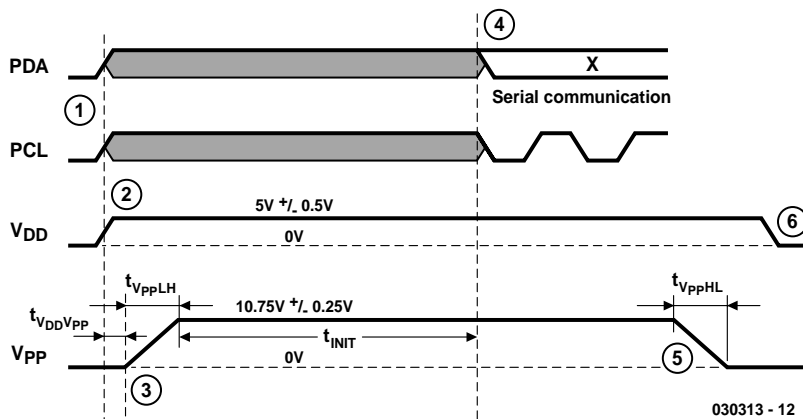


Figure 1. Logic sequence to initiate programming mode.

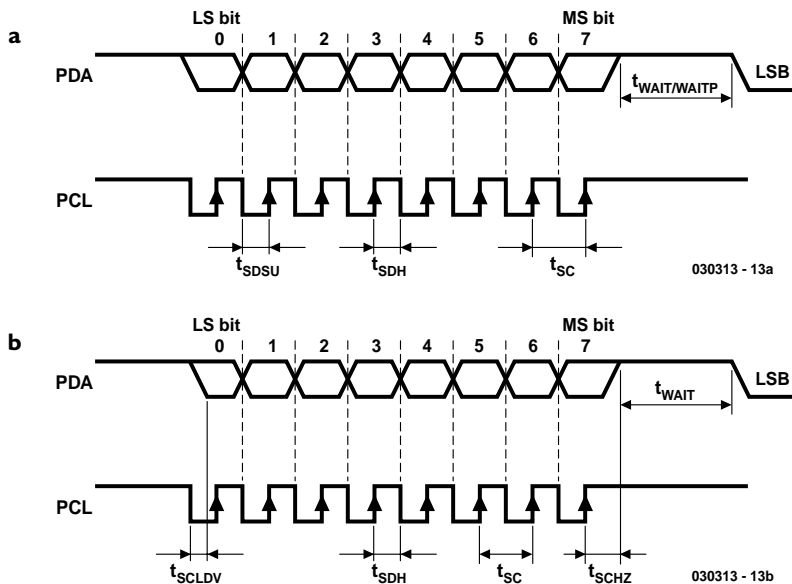


Figure 2. Pulse timing during chip programming (2a) and chip reading (2b).

the very latest version of the firmware into the controller, just before shipping the complete product. There is, however, one disadvantage that should not be left unmentioned: although we're dealing with EPROM types, there's no glass window on the current generation of P87LPC76x devices, hence they are really OTPs (one-time programmable ICs). Obviously, before you burn an OTP device you need to be sure the target circuit works, so the relevant software is best developed and debugged using a Flash board, for example, our incredibly popular **89C8252 Flash Microcontroller Board** (December 2001).

The circuit described here is driven by a PC via its serial port. These days, it may seem a little old-fashioned to employ the RS232 port, but in this case it was an important consideration to keep the hardware costs as low as possible. After all, the addition of a USB interface would set you back an additional £10 or so because of the need to use an USB controller chip.

The circuit diagram and the PCB layout make it overtly clear that we're dealing with an extremely simple and therefore easily reproducible bit of hardware. There are no exotic components. The software running on the host PC has been written in

Visual BASIC and can be obtained as a free download from our website at [www.elektor-electronics.co.uk](http://www.elektor-electronics.co.uk). Alternatively, those without access to the Internet (raise your hands please) may obtain the same on a floppy disk which may be ordered through our Readers Services.

### Programming

A certain protocol, illustrated in **Figure 1**, has to be followed to effectively switch the chip into programming mode:

- Initially, the PDA and PCL pins have to be switched to high impedance;
- Next, the +5-V supply voltage may be applied — note that the chip does not tolerate 'hot inserting';
- After 20  $\mu$ s, the programming voltage VPP may be applied to the relevant pin. This voltage must have a rise time of between 1  $\mu$ s and 100  $\mu$ s.
- The chip is now in programming mode. However, an additional 60  $\mu$ s delay has to be observed before the floating condition of PCL and PDA can be lifted and programming can commence.
- Leaving the programming mode is less critical, although we have to observe a fall time of 1–100  $\mu$ s for the removal of the programming voltage, after which the supply voltage may be switched off.

Programming a P87LPC76x chip involves sending commands to the serial interface on the chip via the PCL and PDA lines. Each command or program byte consist of eight bits. After eight clock cycles, all eight bits have been clocked into the processor, leaving the complete byte ready for fetching from the receive register. The serial interface on the P87LPC76x is identical to the 8051 compatible UART in mode 0.

Data is exchanged on the falling clock edge and clocked into the controller on the rising edge. The LSB always comes first, the MSB, last. This method applies to programming as well as reading the controller. When reading the chip, the data of the first bit is valid after 40 ns. An important point to note here is that when reading the last bit, the PDA pin has to be switched to high impedance 40 ns after the last rising edge. Consequently, sampling needs to take place before the last clock edge is issued.

A delay of at least 2  $\mu$ s has to be observed between two instructions. Similarly the system needs to wait at least 250  $\mu$ s at `strt_prgm`.

The diagrams in **Figure 2a** and **Figure 2b** show the timing diagram relevant to programming and reading respectively.

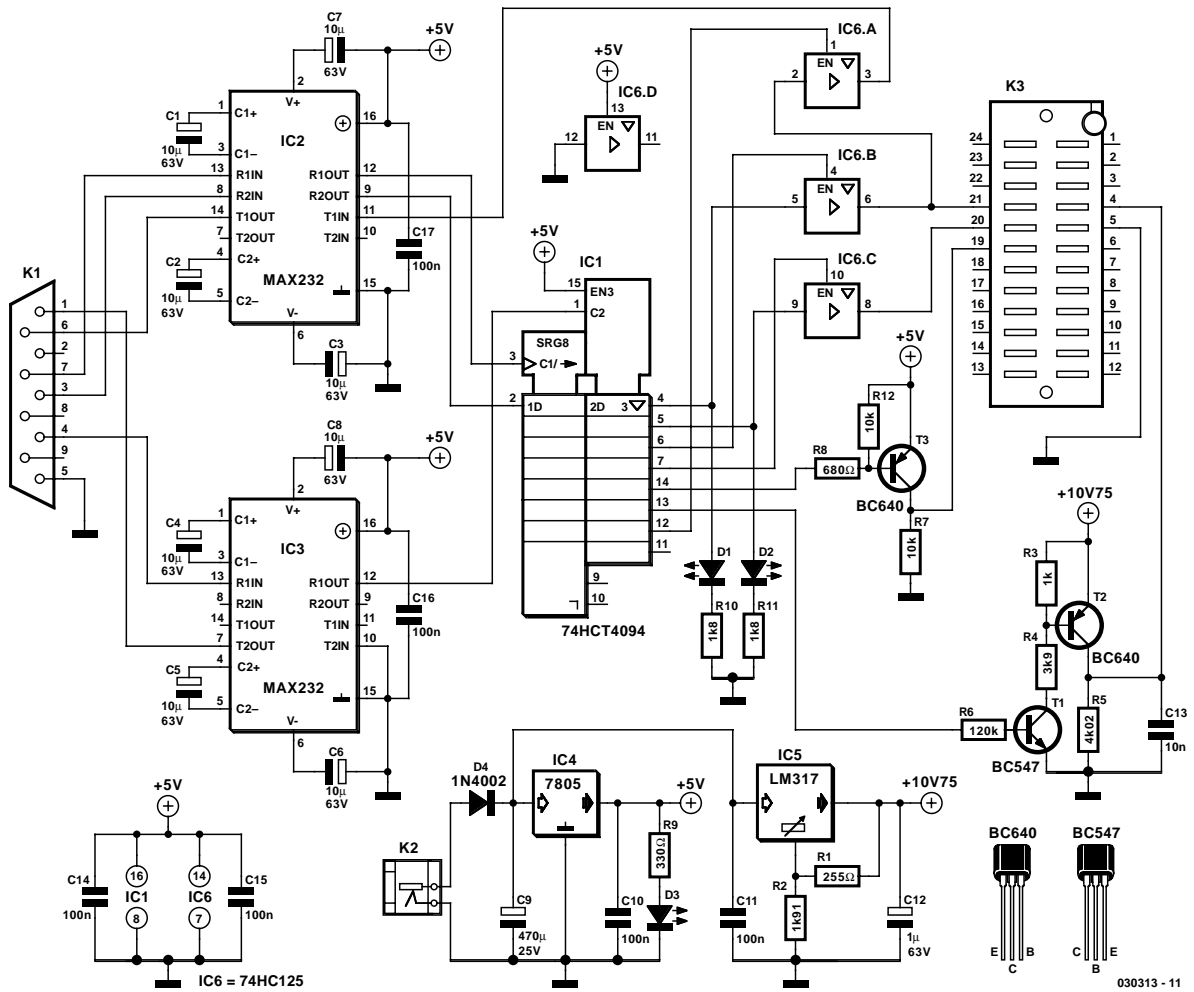


Figure 3. The programmer hardware can hardly be called complex.

**Hardware**

As you can see from the circuit diagram in **Figure 3**, the complexity of the hardware remains within reason. A handful of ICs, three transistors, a few passive components — and that’s it, really!

**Shift register**

The heart of the circuit is formed by IC1, a dual shift register. The first register allows eight bits to be clocked in serially. Using a strobe pulse, the contents are copied in parallel fashion into the second shift register, whose outputs are bonded out to pins. The device allows eight bits to be shifted ‘in’ at the clock rhythm without causing any state changes at the chip outputs. The state changes are not effective until the strobe input (pin 1) is pulled High. This feature comes in handy here because in our circuit seven control outputs have to be used simultaneously, while the serial interface ‘boasts’ only three!

Observing the order ‘LSB to MSB’ the var-

ious functions allocated to the outputs are described as follows:

- dataline for the controller to be programmed;
- clock line for the controller;
- control line to ‘high-Z’ the dataline (active low);
- control line to ‘high-Z’ the clock line (active low);
- controller supply line;
- controller programming voltage line;
- control line to ‘high-Z’ the read line (direction: to the PC) (active low);
- MSB of shift register is not used.

When copying (i.e., clocking) a number from the PC into the shift register, you have to observe that the LSB at the PC side corresponds to the MSB in the shift register. Also, 128 clock pulses are required to convey one byte to the controller. This is because the ‘amount’ of one bit

requires clocking a byte two times into the register. The only bit that differs within this byte is the clock bit — all others remain unchanged. Consequently the shift register has to receive two times eight or sixteen clock pulses for each bit read by the controller. A complete databyte then requires this operation to take place eight times.

**Supply voltage and buffer control**

As already mentioned, entering the programming mode calls for the data and clock lines to be held in high-impedance mode until the supply and programming voltages are applied. This condition is looked after by IC6, a quad tri-state buffer whose outputs are Low (0), High (1) or High-Z as a function of the voltage applied to its control input. The High-Z output state is brought about by pulling the control input logic High.

The supply and programming voltages required by the controller are each applied via a transistor switch, because CMOS ICs in general can not supply more than 4 mA. For the supply voltage a single driver using a p-n-p transistor proved sufficient. The switch for the programming voltage is a little more complex because it involves a level change from 5 V to 10.75 V. That's why we first create an open-collector output using T1, and then have T2 switch the programming voltage proper. Without T2, it would be impossible to completely turn off T2. Capacitor C13 ensures that the programming voltage edge has a rise time between 1  $\mu$ s and 100  $\mu$ s, while also eliminating undesirable RF noise.

LEDs D1 and D2 visualise the data and clock signal, respectively. In view of the limited current available, low-current types are mandatory in these positions.

#### From RS232 to TTL

The level conversion from RS232 to TTL levels is carried out by an old faithful, the MAX232 (IC2, IC3). This IC contains a pair of charge pumps that enable the +5-V TTL level to be converted to -10 V, in compliance with the RS232 standard. Because we're dealing with three control inputs, for convenience (and with easy availability in mind) two MAX232s are applied in our circuit.

#### Power supply

The circuit may be powered by an ordinary mains adaptor with an unregulated output voltage of 15 VDC. In many cases, a 12-VDC type will also suffice as it will easily supply 15 to 17 V at such light output load currents as expected here.

The internal 5-V supply voltage is obtained from three-pin voltage regulator IC4. The stabilised programming voltage is supplied by an adjustable regulator circuit built around the venerable LM317. Diode D4 acts as a polarity reversal protection and D3 as an on/off indicator.

## Software

The software is marked by a universal character, consisting of many different routines. This structure allows

enthusiastic programmers among you to modify or change the program to their heart's content. For example, it should be possible to implement a different interface without too much of an effort, just by changing the contents of the output routine and re-assembling the code.

A detailed discussion of each routine is unfortunately beyond the scope of this article, hence we limit ourselves to looking at the 'shell' of the software, i.e., the user interface and its actual use.

A screendump of the user window as it appears on the PC monitor is shown in **Figure 4**. First, we will briefly mention the button functions at the right-hand side of the window:

- 'Blank check' is available to see if the chip is unprogrammed, that is, it contains nothing but 'FFh' values (all bits at '1'). By programming, a bit reading '1' can be changed into a '0'.
- 'Read' to read the contents of the chip.
- 'Program' allows the chip to be programmed with the file you've just loaded.
- 'Verify': pressing this button launches a routine that runs a verification process on the chip contents after programming; if a discrepancy is found, an error report pops up.
- 'Test hex file' serves to check the hexadecimal file for checksum errors, as well as to check if the hex file size does not exceed the memory size of the chip you want to program.
- 'Auto' carries out all the above functions automatically (except 'Read').
- 'Read Chip ID' allows you to read out the exact type code of the P87LPC76x chip, which is useful in case the type print on the device has become hard to decipher. The result is displayed in a text window to the left of the button.

#### Configuration settings

The area in the upper left-hand corner allows you to tick the settings for the configuration bytes. To the right of it, you'll find the oscillator configuration, which is actually limited to just one setting! The same applies to

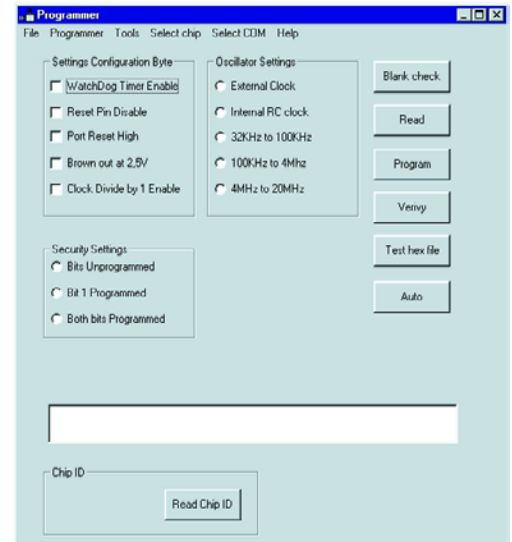


Figure 4. User interface.

the block 'Security settings' in the centre of the user screen. For the sake of completeness we should mention that the security routine looks after the value of the security byte and is called whenever this byte is required. In this way, the software ensures that the most recent value is always present in memory.

Between the block 'Security settings' and 'Chip ID' you'll find another text window showing hexadecimal strings read or due for programming.

#### Task bar

- The function 'File' with its options 'Open file', 'Close file', 'Save as' and 'Exit' will be mostly self-evident.
- 'Programmer': this function has been added to be able to work without the programmer connected to the PC. The options 'Online' and 'Offline' activate and de-activate the programmer, respectively.
- 'Tools': this fold-out window display has the same buttons as the ones directly accessible in the user window, with the exception of 'Read Chip ID'.
- 'Select Chip': here you select the chip from a list of available types. This function is carried out automatically when running the 'Read Chip ID' and 'Auto'.
- 'Select com.': using this fold-out window you can select a free serial ports on your PC.
- 'Help': the option 'Index' opens a help file containing brief explanations about the program functions. The option 'About' supplies some information about the program, the copyright holder and the author.

#### Extra function

There is an another useful function that's not found in the user window: all settings regard-



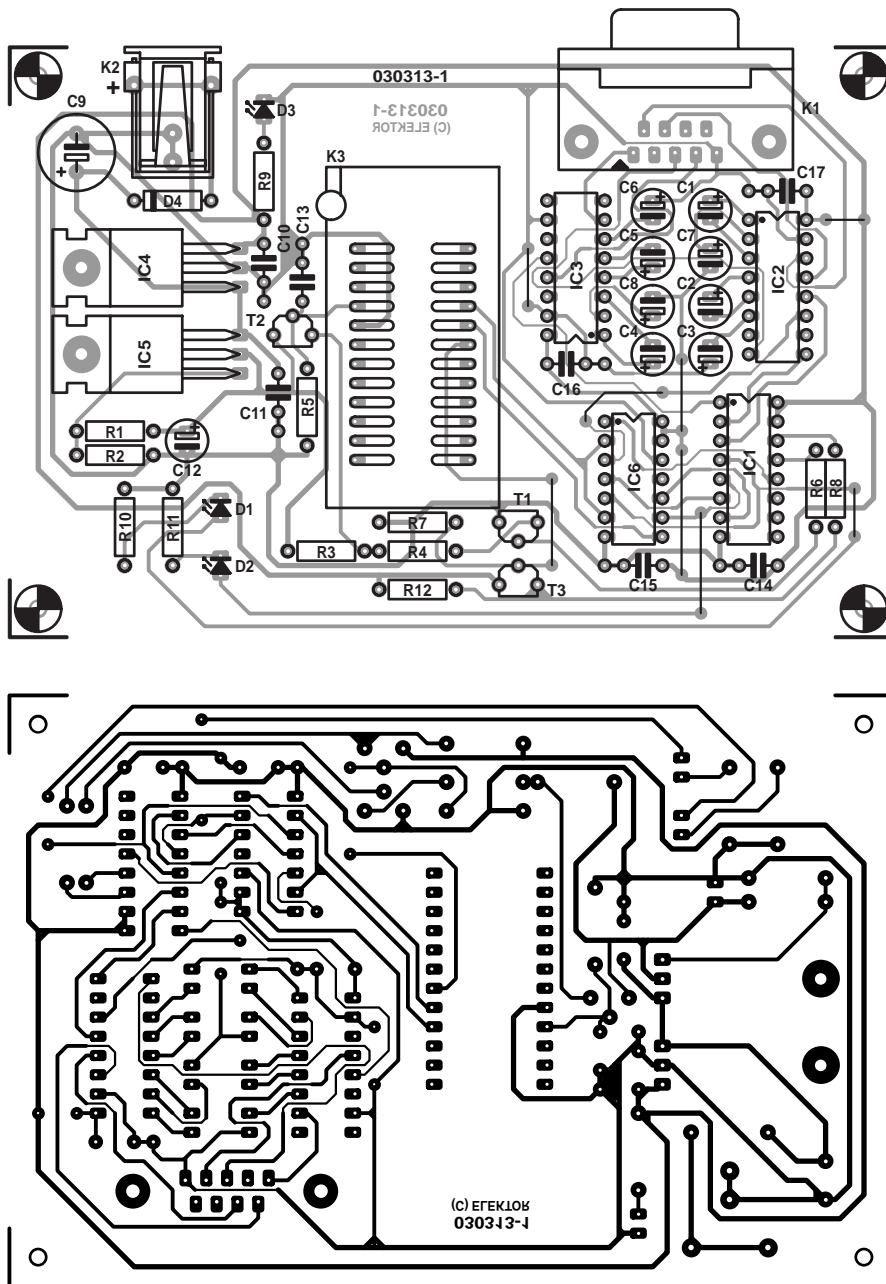


Figure 5. Copper track layout and component mounting plan of the PCB designed for the programmer.

ing the configuration bytes, the chip and the serial port may be stored in a configuration file. This works as follows: each time you quit the program, a message appears asking you whether or not to save the current settings. If 'yes' is selected, your settings will be saved on the C: drive in an ASCII file under the name 'settings.pl'. The next time the software is started, you are asked whether or not to use the settings from the previous session. If you select 'no' you start a new session with a 'clean slate'. In case 'yes' is selected the very first time you use the program, or if no configuration file has been saved, you are

treated to an error message.

## Construction

The copper track layout and component mounting plan of the printed circuit board designed for the programmer are shown in **Figure 5**. We believe it's reasonable to claim that this is a compact yet relatively uncluttered board. The ZIF socket for the controller to be programmed is found roughly in the centre of the board — on the final version of the board, this

## COMPONENTS LIST

### Resistors:

- R1 = 255Ω
- R2 = 1kΩ91
- R3 = 1kΩ
- R4 = 3kΩ9
- R5 = 1kΩ5
- R6 = 47kΩ
- R7,R12 = 10kΩ
- R8 = 680Ω
- R9 = 330Ω
- R10,R11 = 1kΩ8

### Capacitors:

- C1,C2,C3,C4,C5,C6,C7,C8 = 10μF 63V radial
- C9 = 470μF 25V radial
- C10,C11,C14,C15,C16,C17 = 100nF
- C12 = 1μF 63V radial
- C13 = 22nF

### Semiconductors:

- D1 = LED, low current, red
- D2 = LED, low current, yellow
- D3 = LED, green
- D4 = 1N4002
- T1 = BC547B
- T2,T3 = BC640
- IC1 = 74HCT4094
- IC2,IC3 = MAX232 CP
- IC4 = 7805 (TO220 case)
- IC5 = LM317T (TO220 case)
- IC6 = 74HC125

### Miscellaneous:

- K1 = 9-way sub-D socket (female), PCB mount
- K2 = mains adaptor socket, PCB mount
- K3 = 24-way wide ZIF socket 8 wire links
- PCB, order code **030313-1** (see Readers Services page)
- Disk, order code **030313-11** (see Readers Services page) or Free Download
- Mains adaptor, output 15VDC (or 12VDC, see text), 300 mA
- Serial cable, 1:1 (non-crossed)
- Enclosure: e.g., PacTek 145 x 90 x 30 mm

mounted flat onto the PCB surface.

The actual soldering work is not expected to cause any difficulty. Not a lot can go amiss if you stick to the component mounting plan and the components list references and part descriptions. Having said that, don't forget to fit each and every wire link on the board (there are eight of them) as the absence of any one of these can cause awkward problems. The photograph in **Figure 6** shows a correctly built up and fully functional programmer board.

Considering the small size of the board, finding a suitable enclosure should not be too difficult. We built our prototype into a PacTek case with outside dimensions 145x90x30 mm. As shown by the introductory photograph, a matching front panel was designed and produced to add the finishing touch to the prototype. The front panel is not available ready-made from Readers Services, but you can download its artwork from the *Elektor Electronics* website

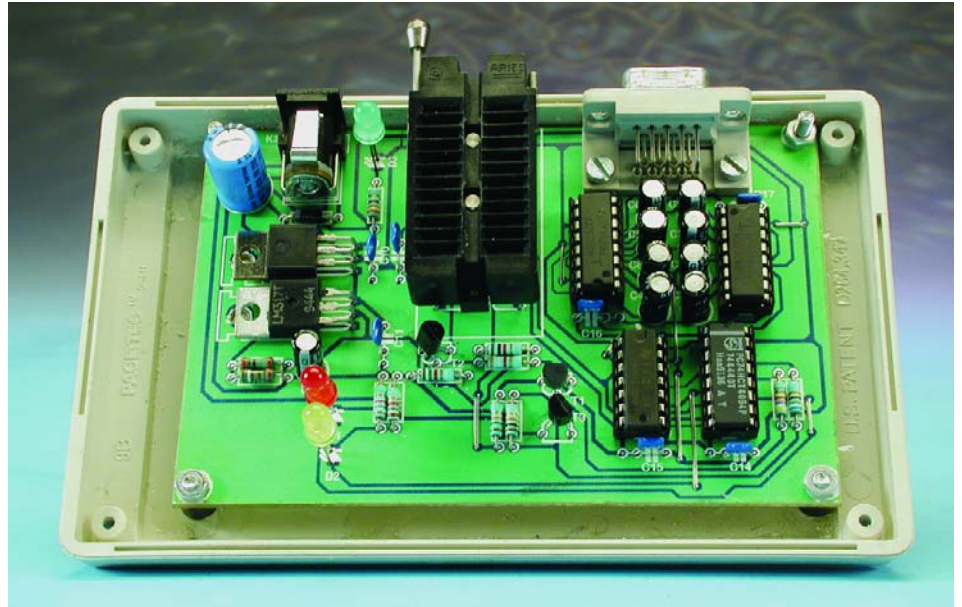


Figure 6. Assembled and fully working prototype of the 89LPC76x programmer.

under reference number **030313-F**.

Finally, an important remark: use a 1:1 (non-crossed) RS232 cable

between the PC and the programmer. A null (zero-) modem cable is **not** suitable.

(030157-1)

# Stepper Motors Uncovered (2)

Part 2 (final): a universal 4-channel unipolar stepper drive

Design by Timothy G. Constandinou

Having covered the fundamentals to stepping motors and drive systems, this second and final part provides a comprehensive design to a four-channel unipolar stepper drive with complete interface electronics for direct operation from a standard PC.



This second part of the article includes full details to build, test and use a low-cost 4-channel stepper motor drive which can be tailored to your applications. The project includes the RS-232 interface for direct connection to

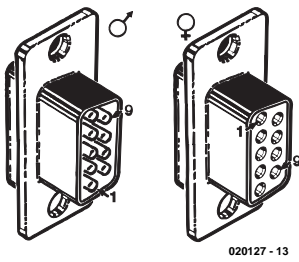
the PC, a custom high-level control language for executing commands sent to the controller and the drive electronics to power the motors. In addition, the PC communication soft-

ware will be explained in some detail allowing full customisation to your specific requirements. This software is compatible with all Microsoft Windows 32-bit platforms and was developed in Borland Delphi.

## The RS232 serial interface

The RS232 serial interface standard, defined over four decades ago, has remained a favourite for low bandwidth communications using the personal computer. Since virtually all PCs are shipped with at least one RS232 port, and many of today's microcontrollers have, or are easily extended with, RS232 interface circuitry, the RS232 port is an affordable as well as straight-forward option for home-built projects.

Normally recognised as a 9-pin sub-D plug labelled COM1 or COM2, the RS232 serial port has nine connections. Half duplex two-way communication can be achieved by using only three pins (2, 3 and 5). The complete pin-out of the port is shown in **Figure 1**. Unlike the standard TTL levels, RS232 data is bipolar, using +3 to +25 V to represent a logic '0' and -3 to -25 V to represent a logic '1'. This scheme makes relatively



020127 - 13

Pin	Signal
1	Data Carrier Detect (DCD)
2	Received Data (RxD)
3	Transmitted Data (TxD)
4	Data Terminal Ready (DTR)
5	Signal Ground (SG)
6	Data Set Ready (DSR)
7	Request to Send (RTS)
8	Clear to Send (CTS)
9	Ring Indicator (RI)

Figure 1 RS232 port pinning.

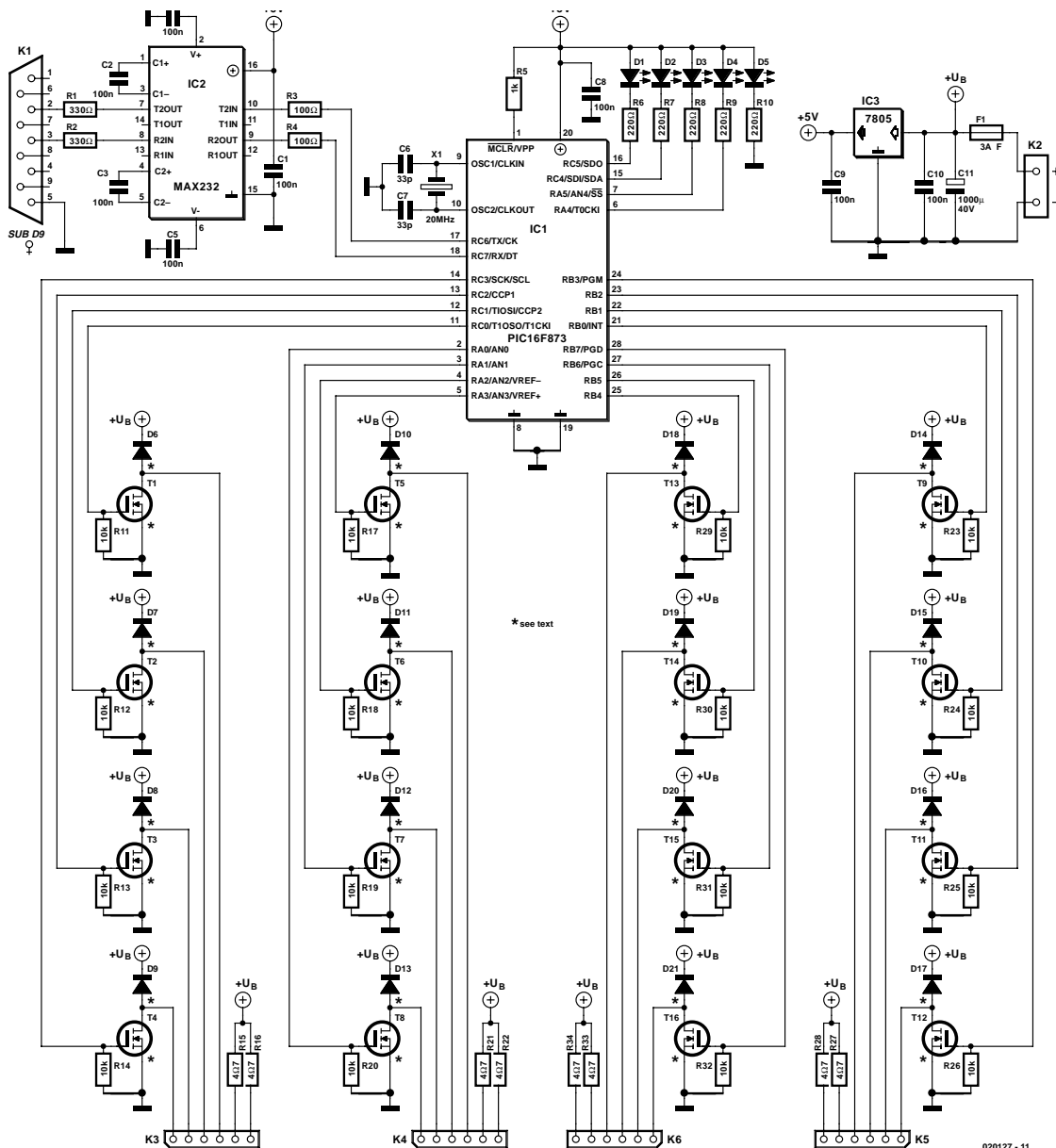
unique address. For access it is opened, the required data is transferred and then it is closed. The only additional requirement is that the port properties be set up ('configured') before usage; for example, data bitrate, parity bit and data timeout.

### Hardware

Figure 2 shows the circuit diagram of the stepper drive and interface. This is quite straightforward. Starting from the RS232 input (K1) the transmit (Tx) and receive (Rx) lines are connected to a level converter chip (IC2). As previously mentioned, this has the task of converting the RS232 bipolar voltage levels — for example, a swing of  $-9V/+9V$  to TTL swing (defined as  $+5V/0V$ ). Note that

long-distance communication possible — however, additional interface electronics are required to convert RS232 voltage levels to/from TTL.

At the computer end, communicating with just about any hardware port is much the same as handling a data file on disk. Each port has a



020127 - 11

Figure 2. Circuit diagram of the driver board.

this is internally done by using a switched capacitor technique to create a higher double-ended supply ( $\pm 9V$ ).

The TTL level-converted signals are then connected to the UART (Universal Asynchro-

nous Receive and Transmit) pins of the PIC microcontroller (IC1). The RS232 I/O pins have been connected through series resistors R1 and R2 and similarly, on the converted side,

R3 and R4, primarily for protection, in case something goes wrong!

The linear regulator (IC3) is required to provide the +5 V regulated supply to the PIC MCU and

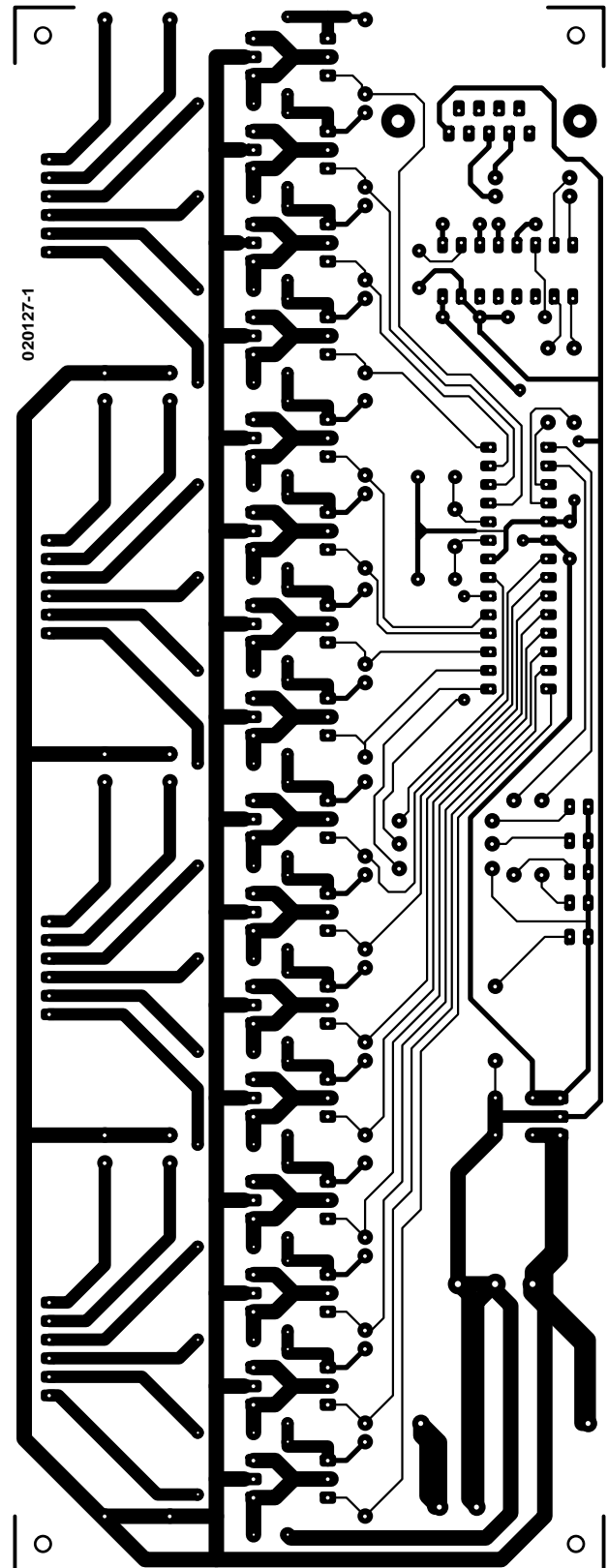
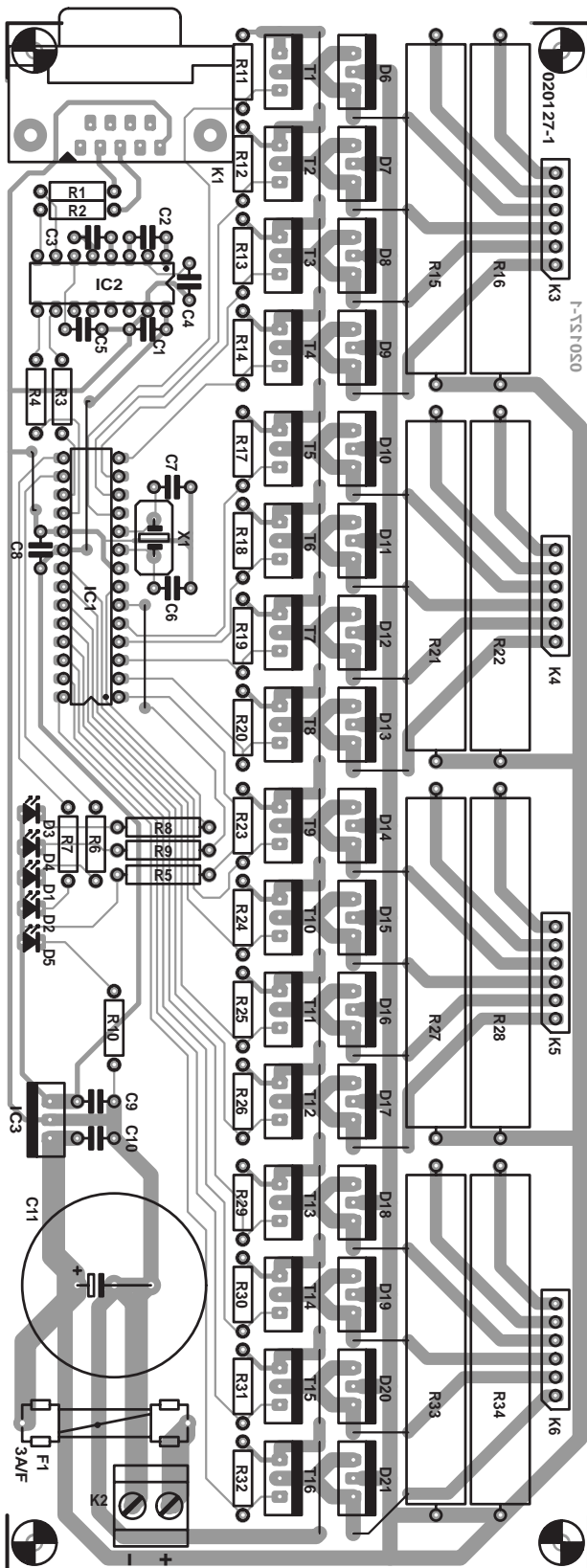


Figure 3. PCB design for the stepper motor driver board.

RS232 interface chips. IC1 employs capacitors C6, C7 and quartz crystal X1 in conjunction with an internal bistable to form a precision 20 MHz oscillator required by the UART. Pin 1 of the PIC is pulled high through R1, as resetting the microcontroller is not required. All remaining I/O ports (20 pins) are configured as outputs and connected to the stepper motor phase drives and LED indicators.

The stepper motor drive scheme used is a resistance-limited unipolar drive, suitable for 5, 6 and 8-wire low-power stepper motors. This provides a low-cost and simple means

to powering a unipolar winding. However, it does suffer from inefficiency due to power dissipated in the ballast resistors.

The phase drive circuit uses logic-level MOSFET devices driven directly from the microcontroller output to power the stepper motor windings. Various Logic Level FETs may be applied here, see the inset. Fast recovery diodes are required to provide a return path for the energy stored in the motor windings and to prevent damage to the MOSFET devices owing to back-EMF discharges. Again, a number of devices

### COMPONENTS LIST

#### Resistors:

- R1,R2 = 330Ω
- R3,R4 = 100Ω
- R5 = 1kΩ
- R6-R10 = 220Ω
- R11-R14,R17-R20,R23-R26,R29-R32 = 10kΩ
- R15,R16,R21,R22,R27,R28,R33,R34 = 18Ω 5 watt (see text)

#### Capacitors:

- C1-C5,C8,C9,C10 = 100nF
- C6,C7 = 33pF
- C11 = 1000μF 40V radial

#### Semiconductors:

- D1-D4 = LED, green, 3mm
- D5 = LED, red, 3mm
- D6-D21 = MBR2060CT (Farnell # 247-157) (see inset)

- IC1 = PIC16F873-20/SP (not available ready-programmed)
- IC2 = MAX232CPE
- IC3 = 7805CP
- T1-T16 = Logic-level MOSFET, for example, RFD14N05L (Farnell # 516-399) (see inset)

#### Miscellaneous:

- F1 = fuse, 3AF (fast) with PCB mount holder
- K1 = 9-way sub-D socket (female), PCB mount
- K2 = 2-way PCB terminal block, 5mm lead pitch
- K3-K6 = 6-way SIL pinheader
- X1 = 20MHz quartz crystal PCB, order code 020127-1 from The PCBShop
- Disk, contains all source code files, order code **020127-11** or Free Download

Table 1. Ballast resistor values (examples)

V <sub>supply</sub> (Volts)	I <sub>motor</sub> (Amps)	R <sub>motor</sub> (Ohms)	R <sub>ballast</sub> (Ohms)	P <sub>ballast</sub> (Watt)
15	1.00	5	10	5.0
20			15	7.5
25			20	10.0
30			25	12.5
15	0.500	15	15	1.9
20			25	3.1
25			35	4.4
30			45	5.6

are available to choose from, see the relevant inset. The ballast resistors are used to limit the current through the phase winding, but inevitably will dissipate power. The resistor values shown should be calculated for the specific stepper motor used. It is essential to have the manufacturer's data on the specific stepper motor including data on the winding impedance, as well as nominal current and voltage ratings. If you do not have this available it is not advisable to obtain just the resistance using a multimeter as no data will be available on the motors' real power ratings. **Table 1** gives an example of selecting ballast resistor value and rating for two different stepper motors for different supply voltages.

These values can be calculated as follows:

$$R_{ballast} = V_{supply} / (I_{motor} - R_{motor})$$

$$P_{ballast} = 0.5 (I_{motor}^2 \times R_{ballast})$$

Some points to note: because the motor is driven in full-step mode, the windings are only powered half the time, therefore the power rating for the relevant ballast resistor may be only half the energy dissipation normally expected. The voltage supply should be chosen to lie between 10 V and 30 V — the higher the supply, the more power delivered to the motor. This should be higher than the voltage rating of the motor — don't forget there is a voltage drop across the ballast resistor. Also, please note that the maximum current rating (per winding) that can be driven using this PCB should not exceed 1 A.

### Construction

All components in this circuit are assembled directly onto a PCB, whose copper track layout and component mounting plan are given in **Figure 3**. Sockets should be fitted for the two ICs with a DIL (dual-in-line) footprint, while IC3 should be soldered directly onto the PCB. It is advisable to firstly assemble the lower profile components such as links, resistors,

### Logic Level FETs and Fast Recovery Diodes

In this circuit, the choice of logic level FET (positions T1-T16) and fast recovery diodes (positions D6-D21) will be governed by availability and the power rating of the stepper motor(s) used.

FETs				
Type	I <sub>max</sub> (A)	U <sub>max</sub> (V)	R <sub>i</sub> (mΩ)	Note
RFD14N05L	14	50	100	Farnell # 515-399, Fairchild
BUK100-50GL	13.5	50	125	
BUK101-50GS	30	50	50	
IRLI2203N	61	30	7	
Diodes				
Type	I <sub>max</sub> (A)	U <sub>max</sub> (V)		
MBR1045CT	10	45		Farnell # 878-364
MBR1545CT	15	45		Farnell # 878-194
etc.				

## Listing I. Firmware source code.

```
// main.c – Main program code

#include <16f873.h>
#include <ports.h>
#include <protocol.h>
#include <delay (clock=20000000)
#include <rs232(baud=38400, xmit=tx, rcv=rc)

int astep=1, bstep=1, cstep=1, dstep=1;
long max=800, min=470;

// initialises the ports by defining whether the tri-state buffers should be input or output
void setup_ports(void) { set_tris_a(0x00);set_tris_b(0x00);set_tris_c(0xF0);set_uart_speed(38400); }

// resets one motor to initial state
void reset_motor(int motor) {
    if (motor==1) {output_low(a_1);output_low(a_2);output_low(a_3);output_low(a_4);output_high(led_a);}
    if (motor==2) {output_low(b_1);output_low(b_2);output_low(b_3);output_low(b_4);output_high(led_b);}
    if (motor==3) {output_low(c_1);output_low(c_2);output_low(c_3);output_low(c_4);output_high(led_c);}
    if (motor==4) {output_low(d_1);output_low(d_2);output_low(d_3);output_low(d_4);output_high(led_d);} }

// resets all ports to initial states
void reset_ports(void) { reset_motor(1);reset_motor(2);reset_motor(3);reset_motor(4);putc(ACKNOWLEDGE); }

// creates a delay which constitutes the step pulse duration
void delay_micro(long delay) { long n;for(n=1;n<=delay;n+=3)delay_us(6); }

// changes powered phases according to current step required
void power_motor(int axis, step) {
    if (axis==1) {
        if (step==1) {output_bit(a_1,1);output_bit(a_2,0);output_bit(a_3,0);output_bit(a_4,1);}
        if (step==2) {output_bit(a_1,0);output_bit(a_2,1);output_bit(a_3,0);output_bit(a_4,1);}
        if (step==3) {output_bit(a_1,0);output_bit(a_2,1);output_bit(a_3,1);output_bit(a_4,0);}
        if (step==4) {output_bit(a_1,1);output_bit(a_2,0);output_bit(a_3,1);output_bit(a_4,0);}
        output_low(led_a); }
    if (axis==2) {
        if (step==1) {output_bit(b_1,1);output_bit(b_2,0);output_bit(b_3,0);output_bit(b_4,1);}
        if (step==2) {output_bit(b_1,0);output_bit(b_2,1);output_bit(b_3,0);output_bit(b_4,1);}
        if (step==3) {output_bit(b_1,0);output_bit(b_2,1);output_bit(b_3,1);output_bit(b_4,0);}
        if (step==4) {output_bit(b_1,1);output_bit(b_2,0);output_bit(b_3,1);output_bit(b_4,0);}
        output_low(led_b); }
    if (axis==3) {
        if (step==1) {output_bit(c_1,1);output_bit(c_2,0);output_bit(c_3,0);output_bit(c_4,1);}
        if (step==2) {output_bit(c_1,0);output_bit(c_2,1);output_bit(c_3,0);output_bit(c_4,1);}
        if (step==3) {output_bit(c_1,0);output_bit(c_2,1);output_bit(c_3,1);output_bit(c_4,0);}
        if (step==4) {output_bit(c_1,1);output_bit(c_2,0);output_bit(c_3,1);output_bit(c_4,0);}
        output_low(led_c); }
    if (axis==4) {
        if (step==1) {output_bit(d_1,1);output_bit(d_2,0);output_bit(d_3,0);output_bit(d_4,1);}
        if (step==2) {output_bit(d_1,0);output_bit(d_2,1);output_bit(d_3,0);output_bit(d_4,1);}
        if (step==3) {output_bit(d_1,0);output_bit(d_2,1);output_bit(d_3,1);output_bit(d_4,0);}
        if (step==4) {output_bit(d_1,1);output_bit(d_2,0);output_bit(d_3,1);output_bit(d_4,0);}
        output_low(led_d); } }

// Moves a specified motor by a specified amount of steps in a specified direction.
int move(short direction, long steps, int axis, step) {
    long n, delay, accsteps;
    delay=max; accsteps=max-min;
    for(n=1;n<=steps;n++) {
        if(n<=accsteps)delay--;
        if(steps-n<=accsteps)delay++;
        if(direction==0)step--;else step++;
        if(step==0)step=4;
    }
}
```

```

        if(step==5)step=1;
        power_motor(axis, step); delay_micro(delay); reset_motor(axis); } return(step); }

// Reads in 2 bytes from the UART and returns a 16-bit integer (range 0-65535)
long readlong(void) { return(256*getc() + getc()); }

// Main Program
void main(void) {
    char incomm;
    long steps;
    setup_ports(); reset_ports();
    while(0==0) {
        output_low(led_a); output_low(led_b); output_low(led_c); output_low(led_d);
        incomm=getc();
        output_high(led_a); output_high(led_b); output_high(led_c); output_high(led_d);
        switch(incomm) {
            case RESET:      reset_ports();                          break;
            case SETUP_ACC:  min=readlong(); max=readlong();          break;
            case MOVE_A_FW:  steps=readlong(); astep=move(0, steps, 1, astep); break;
            case MOVE_A_RV:  steps=readlong(); astep=move(1, steps, 1, astep); break;
            case MOVE_B_FW:  steps=readlong(); bstep=move(0, steps, 2, bstep); break;
            case MOVE_B_RV:  steps=readlong(); bstep=move(1, steps, 2, bstep); break;
            case MOVE_C_FW:  steps=readlong(); cstep=move(0, steps, 3, cstep); break;
            case MOVE_C_RV:  steps=readlong(); cstep=move(1, steps, 3, cstep); break;
            case MOVE_D_FW:  steps=readlong(); dstep=move(0, steps, 4, dstep); break;
            case MOVE_D_RV:  steps=readlong(); dstep=move(1, steps, 4, dstep); break; } putc(ACKNOWLEDGE); } }

// ports.h – defines pin assignments

#define tx    PIN_C6
#define rc    PIN_C7
#define a_1   PIN_C3
#define a_2   PIN_C2
#define a_3   PIN_C1
#define a_4   PIN_C0
#define b_1   PIN_A0
#define b_2   PIN_A1
#define b_3   PIN_A2
#define b_4   PIN_A3
#define c_1   PIN_B3
#define c_2   PIN_B2
#define c_3   PIN_B1
#define c_4   PIN_B0
#define d_1   PIN_B7
#define d_2   PIN_B6
#define d_3   PIN_B5
#define d_4   PIN_B4
#define led_a PIN_A5
#define led_b PIN_A4
#define led_c PIN_C5
#define led_d PIN_C4

// protocol.h – defines communication protocol

#define RESET      1
#define ACKNOWLEDGE 2
#define SETUP_ACC 10
#define MOVE_A_FW  20
#define MOVE_A_RV  21
#define MOVE_B_FW  22
#define MOVE_B_RV  23
#define MOVE_C_FW  24
#define MOVE_C_RV  25
#define MOVE_D_FW  26
#define MOVE_D_RV  27

```



## Listing 2. Test program to run on the PC.

```

unit main;

interface

uses Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, QCom32, Buttons, ExtCtrls;

type
  TForm1 = class(TForm)
    QCPort: T_QCom32;
    Commport: TComboBox;
    xclgroup: TRadioGroup;
    setup_acc, move_a_rv, move_a_fw, move_b_rv,    move_b_fw, move_c_rv, move_c_fw, move_d_rv, move_d_fw, reset: TRa-
dioButton;
    parameter1, parameter2: TEdit;
    commportlabel, parameterlabel: TLabel;
    Executebutton: TBitBtn;
    autoreset: TCheckBox;
    procedure CommportChange(Sender: TObject);
    procedure ExecutebuttonClick(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure setup_accClick(Sender: TObject);
    procedure move_a_fwClick(Sender: TObject);
    procedure move_a_rvClick(Sender: TObject);
    procedure move_b_fwClick(Sender: TObject);
    procedure move_b_rvClick(Sender: TObject);
    procedure move_c_fwClick(Sender: TObject);
    procedure move_c_rvClick(Sender: TObject);
    procedure resetClick(Sender: TObject);
    procedure move_d_fwClick(Sender: TObject);
    procedure move_d_rvClick(Sender: TObject);
  private { Private declarations }
  public { Public declarations }
  end;

var Form1: TForm1;

Implementation {$R *.DFM}

  procedure TForm1.resetClick(Sender: TObject); begin parameter1.Enabled := FALSE; parameter2.Enabled := FALSE; end;

  procedure TForm1.setup_accClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := TRUE; end;

  procedure TForm1.move_a_fwClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_a_rvClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_b_fwClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_b_rvClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_c_fwClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_c_rvClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_d_fwClick(Sender: TObject); begin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.move_d_rvClick(Sender: TObject); egin parameter1.Enabled := TRUE; parameter2.Enabled := FALSE; end;

  procedure TForm1.CommportChange(Sender: TObject);
    begin QCPort.Port := Commport.ItemIndex + 1; end;

  procedure TForm1.FormShow(Sender: TObject); begin QCPort.Port := 1; CommPort.ItemIndex := 0; end;

```

```

procedure TForm1.ExecutebuttonClick(Sender: TObject);
var
  commandcode : char;
  command     : string;
begin
  Executebutton.Enabled := FALSE;

  if reset.Checked      then commandcode := char(1);
  if setup_acc.Checked then commandcode := char(10);

  if move_a_fw.Checked then commandcode := char(20);
  if move_a_rv.Checked then commandcode := char(21);
  if move_b_fw.Checked then commandcode := char(22);
  if move_b_rv.Checked then commandcode := char(23);
  if move_c_fw.Checked then commandcode := char(24);
  if move_c_rv.Checked then commandcode := char(25);
  if move_d_fw.Checked then commandcode := char(26);
  if move_d_rv.Checked then commandcode := char(27);

  QCPort.Open; setlength(command, 1);
command[1] := commandcode; QCPort.Write(command);

  if (parameter1.enabled) then
    begin setlength(command, 2);
      command[1] := char(strtoint(parameter1.text) div 256);
      command[2] := char(strtoint(parameter1.text) mod 256);
      QCPort.Write(command); end;

  if (parameter2.enabled) then begin
    setlength(command, 2);
    command[1] := char(strtoint(parameter2.text) div 256);
    command[2] := char(strtoint(parameter2.text) mod 256);
    QCPort.Write(command); end;

  while(QCPort.Read = '') do;

  if autoreset.Checked then begin
    setlength(command, 1);
    command[1] := char(1);
    QCPort.Write(command);
    while(QCPort.Read = '') do; end;

  QCPort.Close; Executebutton.Enabled := TRUE;
end;
end.

```

DIL sockets, ceramic capacitors, etc., mainly for convenience. Take special care to observe the correct polarity of all semiconductors and electrolytic capacitors before soldering. Also, the ballast resistors should be mounted slightly off the board surface as they will become hot during operation. It is advisable to use ceramic standoffs to space these resistors above the board.

If all four channels are not required, you may populate, for example, only two of the four channels of the stepper motor drivers.

When the soldering is finished, the PIC microcontroller and MAX232

ICs may be installed in their DIL sockets. You can program your own PIC for the project using the source code available under number **020127-11** on disk or from the Free Downloads section of our website at [www.elektor-electronics.co.uk](http://www.elektor-electronics.co.uk). For the more ambitious readers wanting to customize the PIC firmware or add functionality, a full overview including some guidelines is provided in the following section. It is advisable to test the project with the original firmware before attempting to modify it.

## The controller software

The PIC microcontroller's function is to receive commands from the PC via the RS232 port and execute them. It is responsible for generating the stepping sequence which will control the power delivered to the motor. This also produces the acceleration and deceleration cycles for optimal stepping response of a given motor. By having this low level interface the pulse timings can be guaranteed to be precise.

So why bother having a microcontroller at all? Why not control the stepper motor drive directly from the computer? Although such real-time control was possible in the past with DOS-based programs, unfortunately this is no longer the case. This is because of the

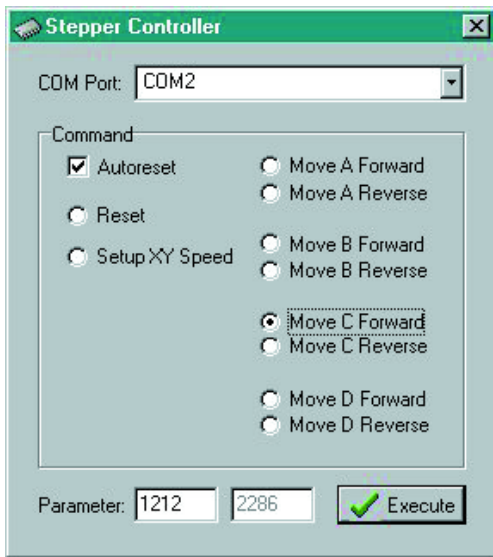


Figure 4. The stepper motor 'command' program in action on the PC.

multi-tasking and multi-threaded nature of recent 32-bit Windows operating systems, time-slicing the processor usage thus precluding stable and accurate timings.

The firmware for the project was programmed using an affordable third-party C compiler supplied by CCS, which is fully compatible with the Microchip MPLAB environment. For more details on this compiler, a full language reference is available online on the CCS website.

The code is divided into three files: main.c, protocol.h and ports.h. The main program is within main.c, with the pin assignments (to variable names) defined in ports.h and the custom communication protocol defined in protocol.h. This firmware source code is given in **Listing 1**.

The custom communication protocol used in this project is very simple. For every command a one-byte value is transmitted and if the command requires additional parameters these are sent in succession. For example, to tell the controller to move the specific motor in one direction for 1000 steps, three bytes are required, the first defining the command and the other two bytes specifying the number of steps (within the range: 0 to 65535). Depending on the initial command byte, the total length of transmission for that command is defined. After executing the command the microcontroller will reply with an acknowledge byte to notify the PC software that it is free to receive more commands if required.

The main program module firstly initialises and resets all the I/O ports including the UART with the bitrate set to 38,400 bits/s. The program then comprises an endless loop awaiting a single byte to be received on the

UART. On receiving the command byte, program control is given to the appropriate command section, which may receive further bytes on the UART.

The available commands are listed below:

**RESET** (byte 1): resets all I/O ports.

**SETUP\_ACC** (byte 10): Followed by an additional four bytes to set the minimum and maximum step delays for the stepper motor motion (both are 16-bit integers). On executing a MOVE command the step delays will initially be at maximum, reducing gradually in duration until the minimum delay has been reached. Further steps will have this minimum delay. Towards the end of the command cycle the step delays will increase until the maximum is again reached. This action implements the acceleration and deceleration in every MOVE command.

**MOVE\_A\_FW** (byte 20): Followed by an additional two bytes (one 16-bit integer) to specify how many steps motor A will move in the forward direction.

**MOVE\_A\_RV** (byte 21): Followed by an additional two bytes (one 16-bit integer) to specify how many steps motor A will move in the reverse direction.

**MOVE\_B\_FW** (byte 22)

**MOVE\_B\_RV** (byte 23)

**MOVE\_C\_FW** (byte 24)

**MOVE\_C\_RV** (byte 25)

**MOVE\_D\_FW** (byte 26)

**MOVE\_D\_RV** (byte 27)

These are as for **MOVE\_A\_FW** and **MOVE\_A\_RV** but for motors B, C and D respectively.

When programming your own PIC microcontroller, don't forget to turn off the DEBUG\_MODE feature. Ensure POWER\_ON\_RESET is enabled and disable the WATCHDOG\_TIMER and BROWN\_OUT\_DETECT features. Also ensure the clock speed is set to 20 MHz.

Recommended programmers and development kits for the microcontroller used here include the Elektor Electronics **PICProg 2003** (Septem-

ber 2003) and the Microchip PIC-START and ICD module (requiring an additional 28-pin header). Alternatively, Taylec Ltd. provide a very affordable equivalent to the ICD module (for under £50), fully compatible with the Microchip software, available for free download.

## The PC software

The PC software was programmed in Borland Delphi 4. A freeware (VCL) Visual Component Library was used in order to access the serial port called OCCOM32.

Included in **Listing 2** is a test program to illustrate how commands are sent through the RS232 port to the stepper motor controller. This is again available in the Free Downloads section on our website at [www.elektor-electronics.co.uk](http://www.elektor-electronics.co.uk).

It is important to ensure the OCCOM32 properties are set to exactly match the initialisation of the UART in the firmware, especially bitrate=38400. For each command to be sent to the controller, the port is opened, the required bytes are transmitted, then the program waits until it receives the acknowledge signal and finally the port is closed.

## Test and practical use

Before powering up, it is important to check all components are correctly placed and that the soldering is clean. Unplug all the stepper motors and power up. First, use an ammeter to check the current drawn from the power supply. Next, use a voltmeter to see if the supply rails are correct. If anything appears wrong at this stage, immediately power off and check the PCB and connections.

All five LEDs should light up when the circuit powers up properly. If this is the case, the microcontroller is up and running. However, if only one LED lights up, then there is power to the circuit but the microcontroller firmware is not being executed correctly. Assuming the microcontroller has been programmed successfully you should then check it receives supply voltage on the relevant pins. If all is in order then you should check the oscillator components (X1, C6 and C7) to ensure these are fitted correctly. If still no

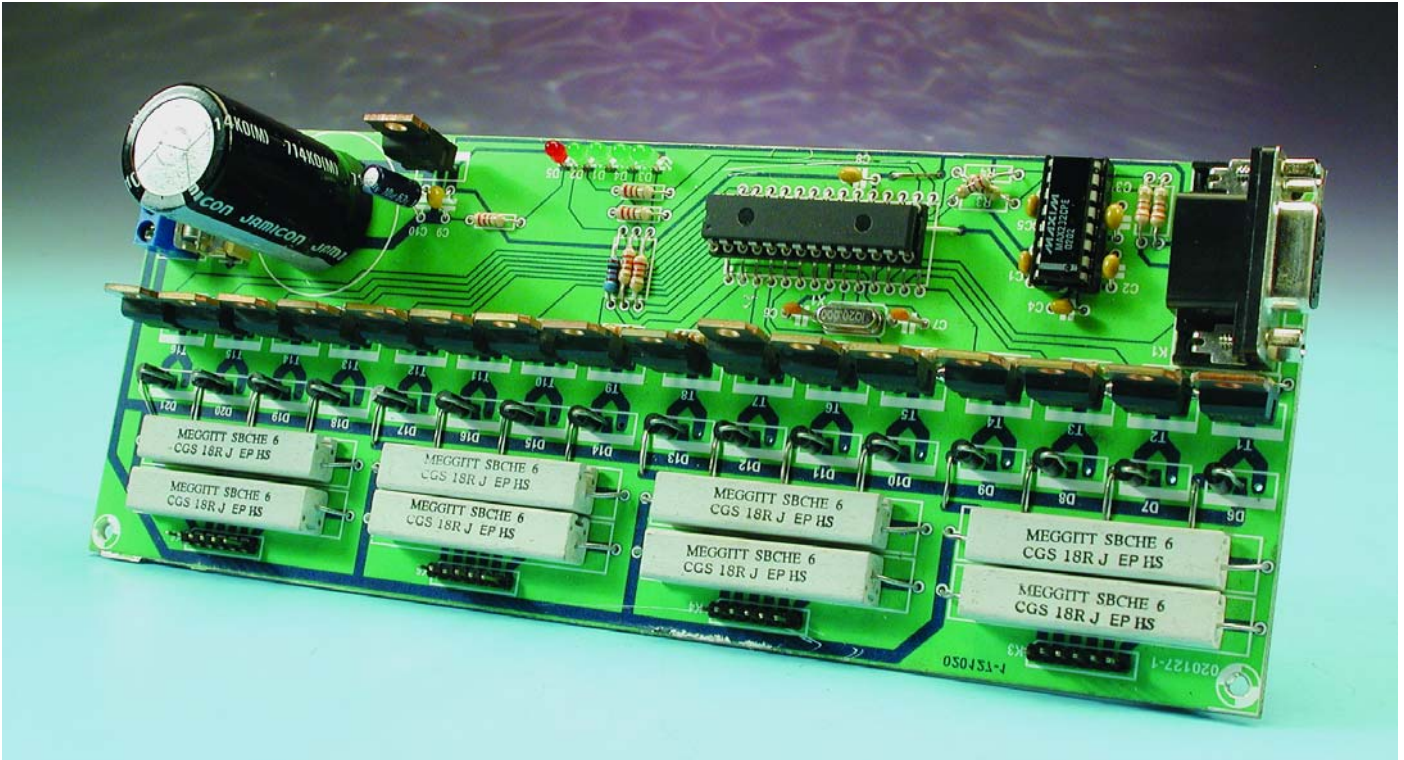


Figure 5. Our finished prototype of the stepper motor driver.

joy try reprogramming or replacing the microcontroller IC.

Once the circuit starts up correctly, use a 1:1 (non-crossed) D-9 female to D-9 male cable to connect the controller PCB to the computer RS232 port. Run the test software on the PC and select the correct COM port setting. Then try testing any of the commands. On sending a command, four LEDs should go out and one LED should light indicating which channel is in use. Once the command has been executed the four LEDs will light up. If this works as expected, turn off the controller PCB and connect a motor to one channel. It is important to ensure the phases and common taps are all correctly connected. Next, power up

again and retest. The motor should spin smoothly, accelerating and decelerating when starting and stopping. If the motor seems to skip or the movement is jerky, check that the phases are connected in the correct order and that the acceleration rate is not too fast for that stepper motor. Remember, lower delay rates mean faster rotation. If you set up the speed with equal delays, for example, 800-800, there will be no acceleration or deceleration. Most stepper motors should work with 500-1000 step delays.

Once all required channels have been tested and are found to be working, you can customize the command (PC) software or control (PIC) software to include your own commands and improvements. One powerful variation could be to multiplex the motors, enabling more than one axis to spin at any time. Applications of the driver board described here may be found in robotics, for accurate positioning of mechanical parts in telescopes, robots, cameras, etc., or for precision movement and placement as required in CNC machine tools.

(020127-2)

## Free Downloads

PIC and PC software (source code files). File number: **020127-11.zip**  
PCB layout in PDF format. File number: **020127-1.zip**

[www.elektor-electronics.co.uk/dl/dl.htm](http://www.elektor-electronics.co.uk/dl/dl.htm), select month of publication.

## Useful links

Microchip PIC 16F87X microcontroller datasheet:

[www.microchip.com/download/lit/pline/picmicro/families/16f87x/30292c.pdf](http://www.microchip.com/download/lit/pline/picmicro/families/16f87x/30292c.pdf)

Direct download link to the QCCOM32 VCL for RS232 I/O in Borland Delphi:

<http://geocities.com/scottpinkham/delphi/qccom32.zip>

Low-cost PIC development tools compatible with Microchip MPLAB environment:

[www.taylec.co.uk](http://www.taylec.co.uk)

A PIC C compiler compatible with Microchip MPLAB environment:

[www.ccsinfo.com](http://www.ccsinfo.com)

VCLs for hardware port access and control:

[www.programmersheaven.com](http://www.programmersheaven.com),  
[www.torry.net](http://www.torry.net), [www.codeguru.com](http://www.codeguru.com)

## Useful literature

'Serial Port Complete' by Jan Axelson, ISBN: 0965081923

'PICProg 2003',

Elektor Electronics September 2003.

# An Experimental DRM Receiver

## DDS meets DRM

By B. Kainka

Did you know that the BBC transmits digitally on shortwave and medium wave? The only trouble is DRM (Digital Radio Mondiale) radios are not widely available yet. This article describes how the *Elektor Electronics* DDS RF Signal Generator can be used together with a little extra hardware and some software to receive these high quality stereo broadcasts on a PC or Notebook.

The new digital broadcasting standard for short wave radio (and all bands below 30 MHz) is known as Digital Radio Mondiale (DRM). The system requires a high specification for the receiver but if you've already built the Direct Digital Synthesiser (DDS) frequency generator described in the October issue of *Elektor Electronics* the rest of the receiver design is simple. An alternative receiver design is also included in this article that does not need the DDS generator.

Take a look at the DRM transmission times and frequencies listed in English at [www.drm-dx.de](http://www.drm-dx.de). DRM transmissions can be picked up on a normal AM radio but all you can hear from the speaker is a high level of white noise as you tune into the signal. The DRM signal actually consists of a large number of QAM carriers that together produce a digital data stream representing the program content. Each carrier can have 64 states so the phase constant of the receive oscillator is very important to ensure reliable reception. See [1] for a more detailed background of the DRM standard. The smallest amount of phase noise in the mixer oscillator results in a degraded or unreadable input signal. This is where the DDS signal generator comes into its own; it has very low phase noise similar to a quartz oscillator and is also very accurately adjustable and stable.

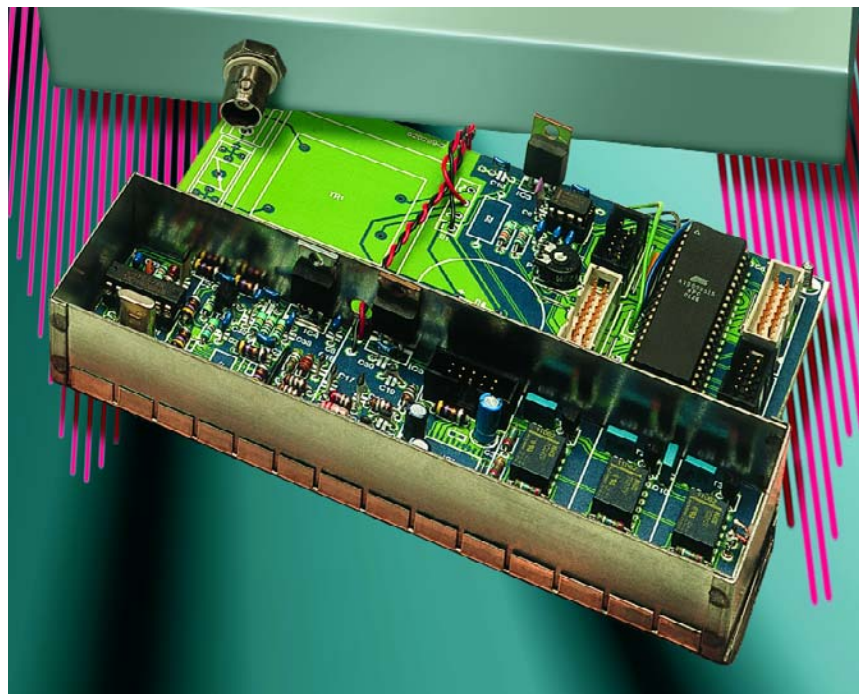


Figure 1. The Elektor Electronics DDS Generator.

Conventional oscillators using tuning capacitors or PLLs are not as suitable for this application. If you start off with the DDS generator the

rest of the receiver design becomes very simple. Even basic receivers will produce good results when a good quality oscillator is available.

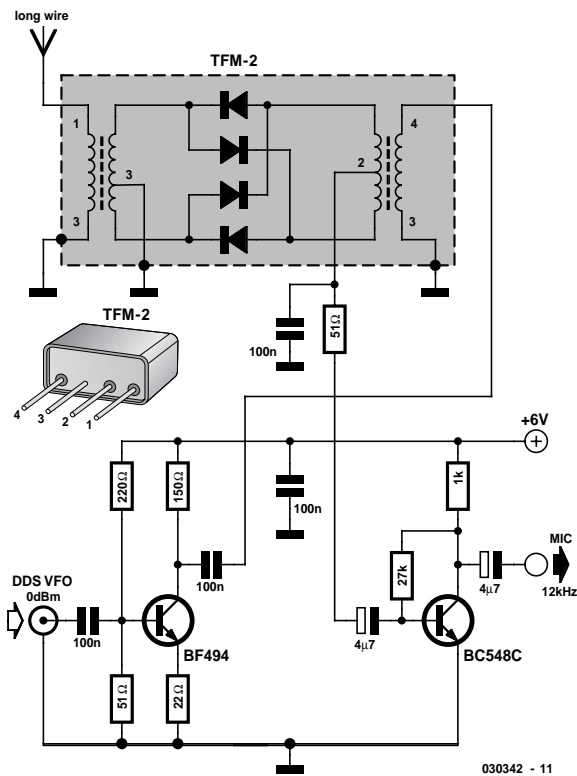


Figure 2. The 50 Ω Direct mixer.

The receiver design is very simple; the incoming RF signal is mixed down with a local oscillator (the DDS generator) to produce a 12 KHz signal, this is fed directly into the sound card input of a PC. Software now does all the tricky stuff like decoding the digital signal and converting it into an analogue signal before it is output to the PC's speakers.

### The direct mixer

The direct mixer is one of the most simple receiver concepts. The incoming HF radio signal is mixed down to produce a 12 KHz baseband signal in one single step. The local oscillator must therefore run at a frequency equal to the received signal plus or minus 12 KHz.

One disadvantage of the direct mixer is that it does not offer any rejection of the image frequency this means that the receiver is also sensitive to a signal 24 KHz away from the desired signal. If you are lucky there will be no other transmitters sending on this frequency. This 'feature' does give you two tuning points to receive the same signal for example if you wanted to tune into

the BBC World Service DRM transmissions sent out from the relay station in Sackville, Canada on 9795 KHz you could set the local oscillator frequency to either 9807 KHz or 9783 KHz. If another strong station is sending on 9771 KHz the signal will be mixed at the same frequency as the wanted signal (9783 KHz – 9771 KHz = 12 KHz) so in this case it is better to tune to the image frequency of 9807 KHz. The 'jamming' signal now produce a 36 KHz output signal that will be removed by the filter fitted to the input of the A/D converter in the sound card.

The DDS generator designed by G. Baars (Figure 1) and published in the October 2003 issue can produce both amplitude and frequency modulated output signal but these are not required for this application. The important features of this generator are the frequency range up to 30 MHz and the simple method of selecting the frequency using a keypad or rotary encoder control knob. The output signal level is adjustable up to 0 dBm (224 mV into 50 Ω) this ensures that the mixer can be driven at its optimum input signal level.

First tests were carried out using a diode ring mixer type TDM2 from Mini Circuits ([www.minicircuits.com](http://www.minicircuits.com)). A BF494 transistor is used as a wide band amplifier to raise the signal level from the DDS oscillator to around 7 dBm. A low noise transistor type BF548C is used as an AF amplifier at the output from the mixer to ensure that there is sufficient signal to drive the microphone input of the sound card. The mixer impedance of around 50 Ω ensures that it can handle large input signals.

The antenna can be just a long wire sited outdoors as high up as practical but if space is at a premium then an antenna length of between one and three metres should be sufficient.

Unlike most RF projects this design does not contain any tuned circuits so the component layout is relatively uncritical. Figure 3 shows the prototype circuit built on a small piece of perforated strip board.

### Decoder

Once you have attached an aerial to the receiver, switched on the DDS generator and connected the output of the simple receiver to the microphone input of your soundcard, the DRM Software Radio program from Fraunhofer-IIS can be started. This program is available online from the home page [www.drmtx.org](http://www.drmtx.org) and costs around 60 Euros (approx £ 45). An open source program has also been developed for DRM reception and we will take a closer look at this towards the end of the article.

The tuning example given earlier indicates the importance of selecting the correct frequency to avoid interfering stations, if you need to tune to the image frequency the

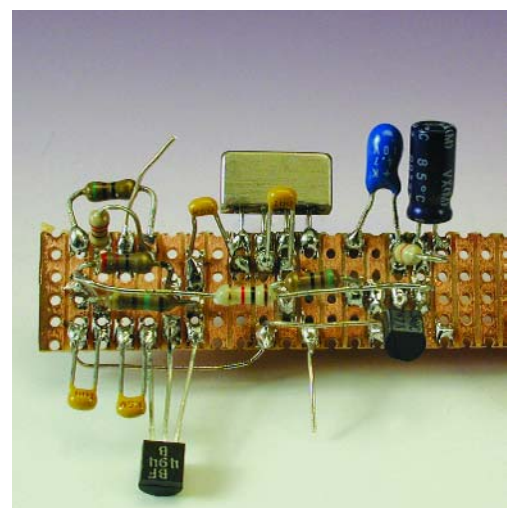


Figure 3. The prototype receiver.

recovered baseband signal will have its frequency spectrum mirrored or inverted so its important to be able to put the signal spectrum back the right way round. **Figure 4** shows the software option for correcting this inversion.

A window in the *DRM Software Radio* display shows the 10 kHz wide frequency spectrum of the received signal, this comes in handy if you are tuning manually to ensure that the flat-topped DRM signal is centred in the display. The display also has several 'virtual LEDs'. When a DRM signal is detected the first of these LEDs will light after a few seconds to indicate that the software has synchronised to the incoming signal next the data LED comes on and the station name together with other information such as the signal data rate and stereo/mono mode is displayed on the PC. If you are lucky and the signal strength is sufficient the third LED will illuminate after a few more seconds and then hey presto... high fidelity sound comes flooding from the speakers! All this from a short wave signal, it's certainly a long way from the fading, garbled, erratic sound that we are used to.

As with all short wave reception, the signal relies on reflection from the ionosphere to achieve its long range and all this digital technology does not protect the signal from the uncertainties of this propagation path so that distant weak stations will still be subject to audio dropouts.

The signal to noise ratio (S/NR) is displayed in real time, a figure of 25 dB is a very good value and even expensive short wave receivers would have trouble achieving this. The direct mixer produces good results as long as you are aware of the drawbacks of no image rejection and the absence of an automatic level control (ALC) of the received signal. On top of this it also requires a fairly long aerial to give a good input signal level.

The simple direct mixer has very wide band characteristics and does not require any input filter selection so that tuning across the entire short wave band is accomplished by simply changing the local oscillator frequency. In Europe the BBC World Service is currently transmitting DRM coded programs using transmitters at Orfordness (1296 kHz) and Rampisham (various SW frequencies). In addition Dutch and German World Service transmitters also broadcast English language programs for example Deutsche Welle on 15440 kHz in the mornings. When tuning into the broadcasts always try both frequencies to find optimum reception but it's often the case that strong stations are transmitting on both these frequencies and the signal gets swamped.

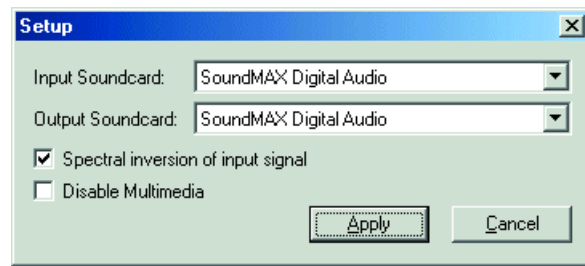


Figure 4. The program allows for spectral inversion of the received signal.

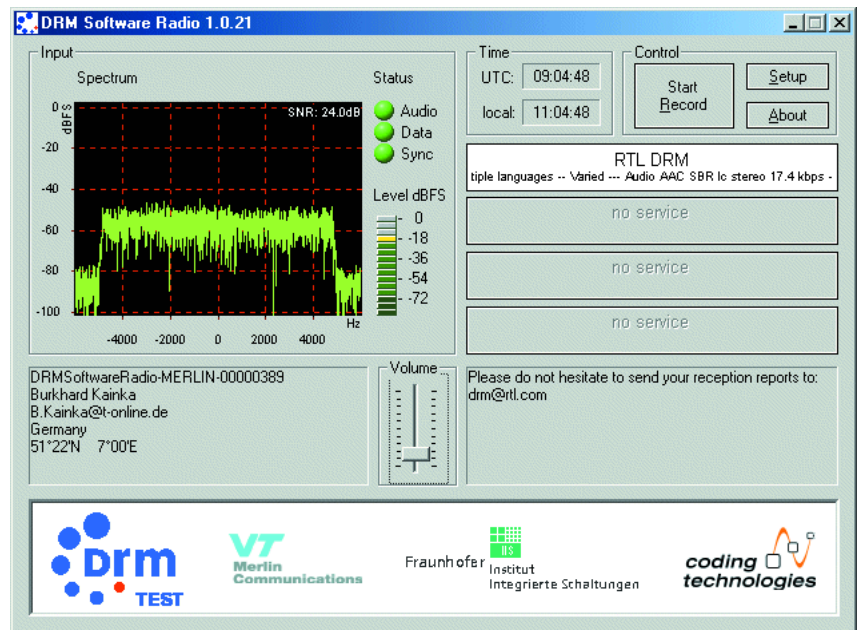


Figure 5. Radio Luxembourg reception.

### Homebrew diode ring mixer

The basic goal of this article was to build a DRM receiver and this has been achieved but the diode ring mixer used in the receiver design is quite an expensive component. Alternatives like the popular IE500 can also be used but you may also be interested in having a go at building your own mixer from discrete components.

A homebrew diode ring mixer can be built with four type BAR28 Schottky diodes and two wide band transformers. Each RF transformer is wound on an Amidon T37-2 (red) toroidal core with an  $A_L$  value of 40 nH/n<sup>2</sup>. Three lengths of 0.3 mm enamelled copper wire are wound together ten times through the core to produce the trifilar windings. Ten turns produces an inductance of

4  $\mu$ H giving the mixer an impedance of 50  $\Omega$  at 2 MHz. The wideband characteristic ensures that the entire short wave band can be received.

The ends of the windings can now be tinned and an ohmmeter used to identify each end of the windings. Two of the windings are now connected in series to form the two centre tapped windings connected to the diodes. The remaining winding on each core is used for the signal input (aerial) connection and the oscillator input. The four diodes can now be soldered in place to produce the finished mixer.

The homebrew mixer can best be tested with an oscilloscope. The upper trace shown in **Figure 8** shows the input on the secondary side of the oscillator transformer and the lower trace is the RF input. The oscillator signal shows limiting as the diodes are driven

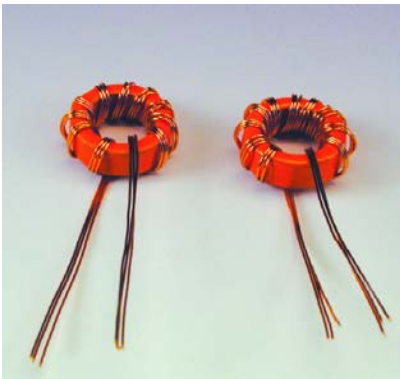


Figure 6. Winding the toroidal transformers.

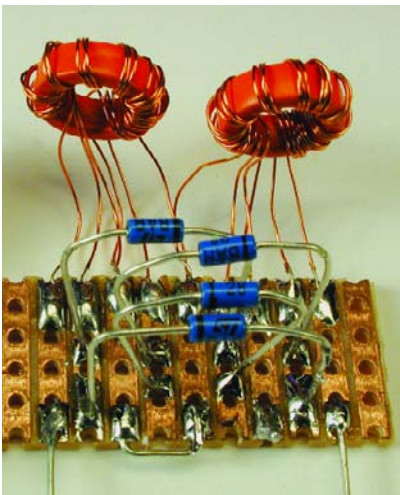


Figure 7. The trifilar wound ring mixer.

into conduction. At the RF input there is practically no evidence of the oscillator input signal, indicating that the mixer has good carrier suppression.

With an aerial attached and the homebrew mixer fitted in place of the commercial unit the receiver was tested again and worked just as well, but that's not to say that our mixer has a specification equivalent to the commercial unit but just that

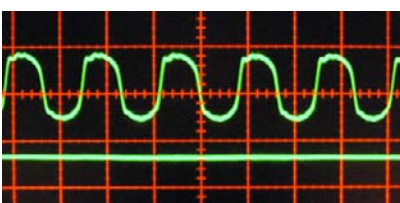


Figure 8. Oscillator drive signal and carrier suppression measurement.

it is more than adequate for receiving DRM broadcasts.

### Direct mixing with an NE612

A ring mixer is a relatively simple device and introduces a certain amount of attenuation in the signal path whereas an alternative type called an 'integrated balanced mixer' produces some mixing gain and also does not require a broadband transformer so altogether this device can offer a more economical solution.

A very popular low cost integrated mixer is the NE612, this device contains an internal oscillator and a fully symmetric mixer. An external oscillator with an output signal level between 200 mV<sub>pp</sub> and 300 mV<sub>pp</sub> can also be connected to pin 6. During tests with the *Elektor Electronics* DDS generator the optimum signal level was set to 250 mV<sub>pp</sub>.

The receiver circuit is shown in **Figure 9**. A long wire antenna is used without any input selector. A small fixed value inductor is used at the input and this circuit can receive DRM signals with a S/NR of 20 dB maximum. The performance of this design is not as good as the diode mixer when receiving strong signals because it can be more easily overdriven which produces intermodulation distortion and corruption of the DRM signal. The advantages of the integrated mixer are its signal gain and increased sensitivity. This type is therefore better than the diode mixer if you are using a short antenna and are only receiving low-level input signals.

### A manually tuned set

Would it be possible to build a DRM receiver without using the DDS signal generator? Well, in this application the DDS is used as a simple signal source so it would be possible to substitute an equivalent device like a manually tuned free running oscillator with a capacitor or diode for tuning but the oscillator needs to have very good stability and low phase noise. *DRM Software Radio* can tolerate a frequency offset of up to 500 Hz from 12 kHz. This degree of resolution is possible if bandspread

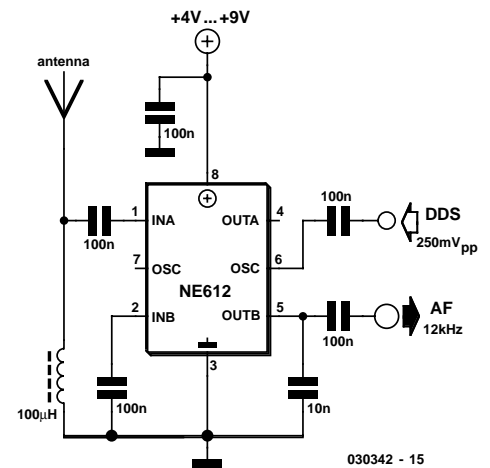


Figure 9. External tuning with a DDS generator.

tuning is used.

There are at least four frequencies in the 49-metre band used for transmitting DRM signals. **Figure 10** shows the circuit of a manually tuned receiver circuit that can be used on this band. The NE612 has an internal oscillator and requires an external adjustable tuned circuit. The oscillator has good frequency stability and a low phase noise and the coupled input filter gives effective rejection of out-of-band signals.

The oscillator input circuit connected to pin 7 of the NE612 is designed to give bandspread tuning over the 49-metre band. The coil consists of 20 turns on a 8 mm coil former and adjustment of the ferrite core gives a tuning range from 6 MHz to 7 MHz while a triple ganged air-spaced VHF tuning capacitor is used for fine-tuning. The coil in the antenna input circuit uses the same size former but this time the 20 turns is tapped at 5 turns and connected to pin 1 of the NE612. This coil is

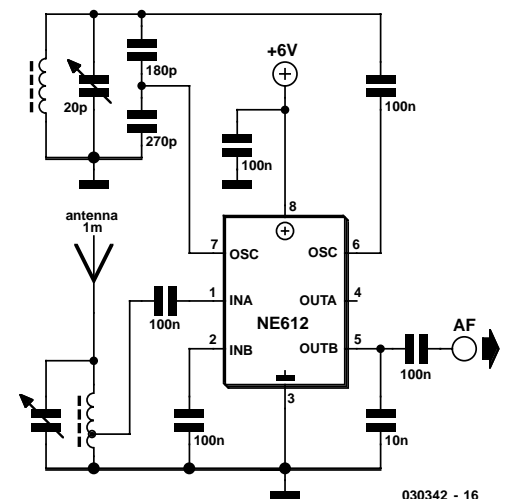


Figure 10. An integrated direct mixer.



not at all critical and you can experiment by connecting say a 120 pF fixed capacitor across the windings.

For testing the triple-ganged VHF tuning capacitor (salvaged from an old VHF tuner) was fitted into a screened enclosure for shielding and polystyrene capacitors were used throughout. The resulting circuit proved to be quite stable and remained on-station for many hours without adjustment.

The audio output signal level from pin 5 is suitable for direct connection to a line input and is also not too large for connecting to a microphone input. This receiver has been used to listen to DRM broadcasts on the 49 metre band and like any other direct-mixer it does not have image rejection so you need a certain amount of luck to be able to listen for any length of time without interference from other stations.

The receiver designs are a good first step in listening in to DRM signals but a different, more complicated receiver such as a superhet would solve the problem of image frequency and also give an improved SNR so that more distant stations could be picked-up.

## Free software

In addition to the commercially available DRM software already mentioned there is also a similar open-source program called DREAM which was developed by Volker Fischer and Alexander Kurpiers of the Institute for Communications Technology at Darmstadt University. The project DREAM can be found at:

[www.tu-darmstadt.de/fb/et/uet/fquet/mitarbeiter/vf/DRM/DRM.html](http://www.tu-darmstadt.de/fb/et/uet/fquet/mitarbeiter/vf/DRM/DRM.html).

All the source files can be downloaded from <http://sourceforge.net/projects/drm/>. The files are intended as a resource for software developers interested in the details of the decoder function. The authors have not made the compiled executable file available because some of software components are not intended for free distribution.

Those of you who are not interested in compiling the source files (or do not have a C++ engine installed) may be lucky and find a compiled version available for download from the Internet. Many colleges and universities use the original program and there are a number of sites that have the DREAM program to download. Try entering the keyword DREAM.EXE into a search engine and you will no doubt find several hits. These sites are not entirely reliable and cannot be recommended; one offered the source code along with separate DLL files whilst another had the complete program compiled in one file.

In principle the DREAM software does the

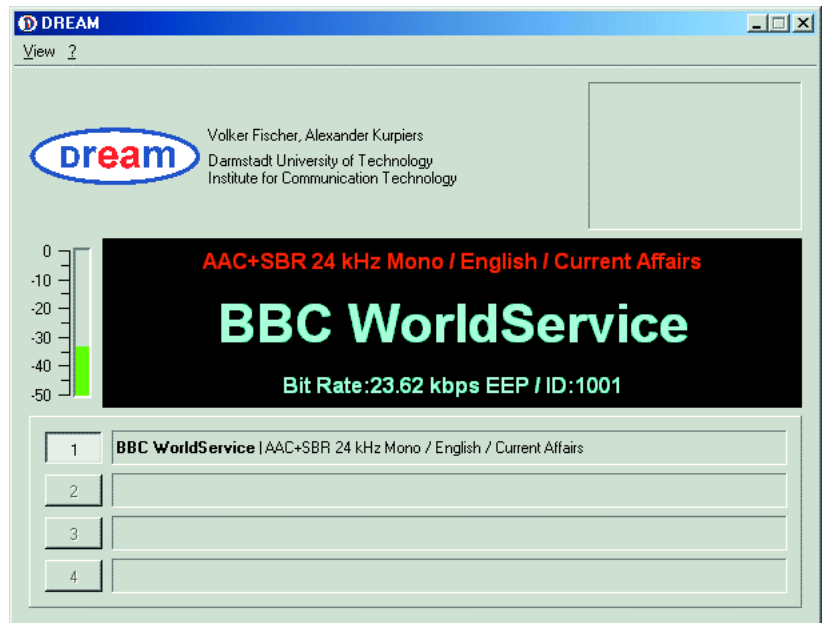


Figure 11. The DREAM program.

same job as the *DRM Software Radio* but has no multimedia possibilities. However it can handle different types of DRM transmissions including signals up to 20 kHz bandwidth and channel bundling. It also provides a good user dialogue with numerous options and display indicators. An advantage of the software is that the IF frequency need not be exactly 12 KHz, a wider range can be selected and this is ideal for homebrew receivers where the range may be limited or when an existing radio is converted for DRM reception.

The DREAM software (Figure 11) is more processor-hungry than the Fraunhofer-IIS software — in tests with a 1.3 GHz Pentium machine the *DRM Software Radio* ran without problems in the background while other applications such as Word, a graphics program and an Internet browser were used. DREAM on the

other hand is a bit of a loner and would only work happily when no other programs (including firewalls and virus alerts) were running. Calling up other applications would inevitably lead to a crash on our system set-up.

As a footnote the author has plans for a DRM receiver that may see the light of day as an *Elektor Electronics* project sometime in the future...

(030342-1)

## Literature:

- [1] H. Weber, **Digital Radio Mondiale**, Elektor Electronics December 2002.

## Weblinks:

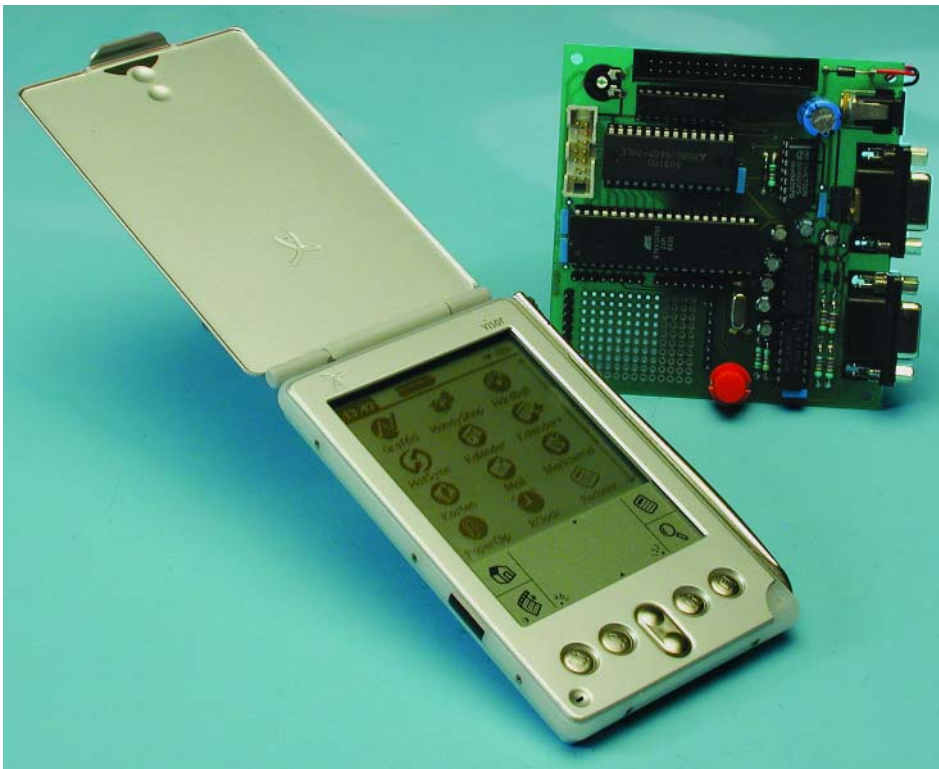
- [www.drm.org](http://www.drm.org) (general DRM info)
- [www.drmrx.org](http://www.drmrx.org) (DRM test project)
- [www.rnw.nl/realradio/html/drm.html](http://www.rnw.nl/realradio/html/drm.html) (Radio Netherlands DRM Site)
- [www.CodingTechnologies.com](http://www.CodingTechnologies.com) (DRM receivers)
- [www.iis.fraunhofer.de/dab/products/drmreceiver/index.html](http://www.iis.fraunhofer.de/dab/products/drmreceiver/index.html) (FhG DRM Software Radio)

# IrDA Interface

## Wireless communication from PCB to PDA

By B. Kainka

Wireless communication using infrared light has many advantages over the traditional mess of cables. This goes not only for PCs and laptops, but even for the humble microcontroller.



In particular, it is easier to set up communications with handheld PDA-style computers using infrared than using the RS232 interface. Some newer machines such as the Palm Zire do not even have an external RS232 interface and instead use a USB port for communications with a PC and an IrDA port to talk to other PDAs, mobile phones and other devices. It therefore seems a good idea to equip microcontroller systems such as the *Elektor Electronics* 89S8252 Flash Microcon-

troller Board (December 2001) with an IrDA interface.

### An IrDA UART for BASCOM-51

The IrDA interface looks rather complicated at first sight, especially considering the many layers of software. But at least at the lowest layer where bytes are physically commu-

nicated things are simple. The IrDA interface is a relatively normal serial interface using short pulses. Brief flashes of light are emitted for the start bit and for each zero data bit, i.e. in the bit positions where the RS232 TXD signal would be at a high level.

The pulse length is precisely specified. At data rates of up to 115,200 baud it must be at least 1.6  $\mu$ s and at most 3/16th of the bit time. On the Palm the shortest possible pulse length of 1.6  $\mu$ s is always used, which results in a power saving. Also, very short and intense pulses of infrared have a long range, as a specially-designed infrared receiver can easily distinguish them from ambient light, which generally contains no such short pulses.

**Figure 1** shows serial signals on an RS232 interface compared with those on an IrDA interface. The IrDA receiver just needs to extend the light pulses suitably in order to create a signal that can be read by a normal RS232 interface.

An IR transmitter simply needs to convert the serial signals into brief pulses. Dedicated ICs are available for this purpose. In the Palm the function is already integrated into the processor, and the IrDA signals are generated using a special setting of the internal UART. This special mode is of course not available in the venerable 8051, and so a suitable

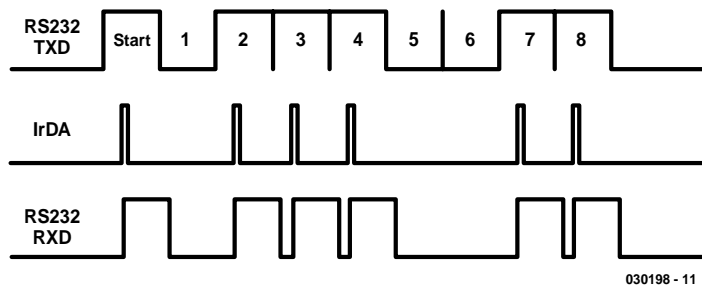


Figure 1. RS232 and IrDA signals.

IrDA UART must be implemented in software.

Simple IrDA routines can be implemented as assembler subroutines on the BASCOM-51. Subroutines Putir and Getir comprise the software UART. Data are passed between the assembler routines and the BASIC program using the accumulator by reading and writing the variable Acc. The transmit routine expects one byte in the accumulator. First it generates a pulse for the start bit on output P3.3. Here we use a pulse length of three instruction

cycles, which, with an 11.059 MHz crystal, is about 3 μs. We avoid using the shortest possible pulse length in order to relax the requirements on the drive circuitry. After the start bit follows a delay of one bit time. Then, one after another, the eight bits in the accumulator are tested and for each zero bit a pulse is generated. Comments in Listing 1 show the number of cycles taken by each instruction. The delay loop is set up so that the bit time is 104 μs which gives a data transfer rate of exactly 9600 baud.

The receive routine cannot poll the input signal directly as the short pulses could be missed. Instead, we use an interrupt. Input pin P3.2 is also the interrupt input Int0, and so we can write a suitable interrupt routine and enable interrupts on Int0. During initialisation we configure Int0 for edge-triggered interrupts so that each falling edge on the receiver input causes the interrupt service routine Puls to be called. The only action of the interrupt service routine is to set a flag bit. This bit remains set, indicating that an edge has been detected, until it is read and then reset. In this way we implement a kind of pulse stretching in software, and the very brief input pulses can be processed in a leisurely fashion.

The receive routine Getir is written in much the same way as would be done for a direct RS232 input. First the flag is cleared in case it has become set as a result of interference. The program then waits for a start bit. Once the flag is found to be set, the program waits for 150 % of a bit time, which takes us to the middle of the first data bit. Although a pulse (if any) will have occurred at the start of the bit time, testing for it in the middle of the bit time gives some tolerance to slight variations in data rates. If a pulse has not

**Listing 1. Port access over IrDA.**

```

' _____
' IrDA1.BAS
' _____

$regfile = "89s8252.dat"
Dim N As Byte
Dim Flag As Bit
Declare Sub Putirda
Declare Sub Getirda

Enable Interrupts
Enable Int0
Set Tcon.0 'INT0 falling edge.
On Int0 Puls Nosave 'Nosave: timing!
While 1 = 1
  Getirda
  P1 = Acc
  Waitms 1
  Acc = P1
  Putirda
  Waitms 1
Wend
End

Sub Putirda
$asm
  clr P3.3 ;1 start
  nop ;1
  nop ;1
  setb P3.3 ;1
  mov r2,#43 ;2
S1:
  djnz r2,s1 ;43*2
  mov r3,#8 ;2
  jnb ACC.0,S3 ;2,
  ;(10+86)*1.085μs=104μs
  clr P3.3 ;1 data puls
  nop ;1
  setb P3.3 ;1
  sjmp S4 ;2
S3:
  nop
  nop ;delay
  nop
  nop
S4:
  mov r2,#41 ;2
S5:
  djnz r2,S5 ;43*2
  rr a ;1
  nop ;1
  djnz r3,S2 ;2
  ;(14+82)*1.085μs=104μs
  clr Flag
  ret
$end Asm
End Sub

Sub Getirda
Flag = 0
$asm
Startbit:
  jnb Flag,Startbit
  clr Flag
  mov r2,#60 ;1.5 bit times
S6:
  djnz r2,S6 ;
  mov a,#0
  mov r3,#8
S7:
  rr a ;1
  jnb Flag,S8 ;2
  sjmp S9
S8:
  add a,#128 ;1
  nop
  nop
  nop ;delayh
  nop
  nop
S9:
  clr Flag ;1
  mov r2,#40 ;2*40
S10:
  djnz r2,S10 ;43*2
  djnz r3,S7 ;2,
  ;(9+7+80)*1.085μs=104μs
  ret
$end Asm
End Sub

Puls:
  Flag = 1
Return
    
```

## Listing 2. Transmitting and receiving port status using the Palm.

```
#irport.bas
open "com1:",9600,ir as #5
draw -1
while 1
  input n
  put #5,n
  a= fn wait(0.1)
  i=fn serial(5)
  if i>1 then dummy=get$(#5,0)
  n =get$(#5,0)
  t$=str$(n)+" "
  draw t$,75,60,2
  dummy =get$(#5,0)
wend
```

been received, the appropriate bit is set in the accumulator; if not, the bit is cleared. The following seven bits are processed in the same way at regular intervals, the bits in the accumulator being shifted one place each time.

The main program outputs the received bytes directly on port 1. After the data have been output the port state is read back and transmitted on the IrDA port. A delay of one millisecond is inserted between transmission and reception in order to allow the receiver part of the interface settling time to recover from the high-power pulses of its own transmissions.

This example provides a complete application allowing writing to and reading from an I/O port. Possible uses include switching external loads and reading switch positions. If only inputs are being used, the value 255 should be sent in order to switch the port to input mode and immediately receive back the read values.

Initial construction of a simple IrDA device should not present any problems. Using the 89S8252 development board as a base, all that is needed in the simplest case is an IR diode at the transmit output (P3.3) and a phototransistor and the receive input (P3.2). The circuit in **Figure 2** shows this minimal implementation, which can be enhanced to give greater range by using a better IrDA front end.

## Communication with the Palm

A programming language is needed to write quick and simple applications for small portable computers: BASIC is ideal. Fortunately, a BASIC interpreter for the Palm is available in the form of HotPaw BASIC, which can be experimented with for free. An evaluation version, unlocked for one month, can be

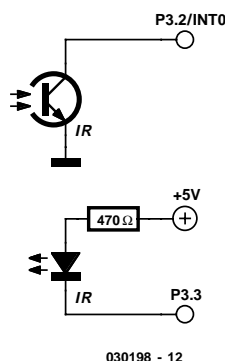


Figure 2. A simple IrDA interface for the 8051.

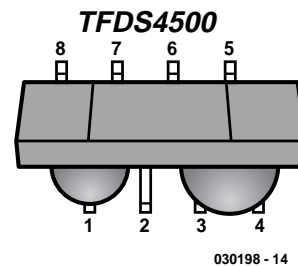


Figure 4. The surface-mount package of the TFDS4500.

downloaded from [www.hotpaw.com/rhn/hotpaw](http://www.hotpaw.com/rhn/hotpaw). After that, you must either pay about \$ 20 for the full version or continue with the evaluation version with one further small restriction: at most four programs can be kept in memory and run at one time.

The BASIC program examples shown here are available for download from the *Elektor Electronics* homepage in the form of plain text

files. In order to use them, the listings must be copied into the Palm desktop MemoPad via the clipboard using a text editor. They will be transferred onto the device at the next HotSync.

All newer PDAs which run PalmOS are equipped with an IrDA interface, and it is not difficult to drive the interface using HotPaw BASIC. **Listing 2** shows an example for writing to and reading from the

Pin	Name	Function
1	IR cathode	-
2	RxD	Receiver data output, open collector
3	V <sub>CC1</sub>	Power supply, 2.7 V to 5.5 V
4	GND	Ground
5	SC	Sensitivity control
6	NC	Not connected
7	TxD	Transmitter driver input, active high
8	IR anode	Anode connection for IR LED, connect to V <sub>CC</sub> via 7 Ω to 14 Ω resistor

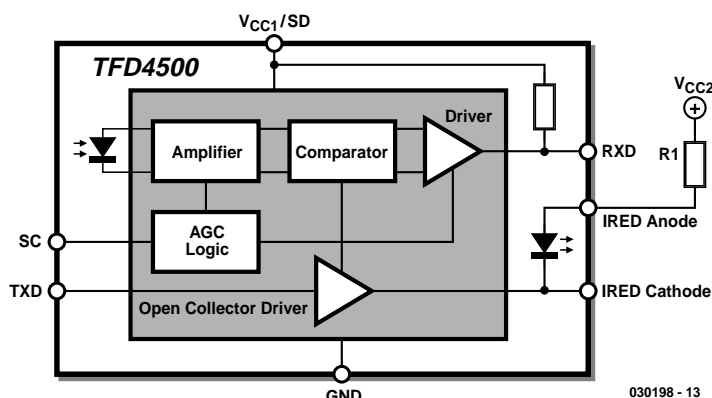


Figure 3. Block diagram of the TFDS4500.

## IrDA receiver for the PC

Many readers will not have a Palm PDA or an IrDA interface on their PC, but would nevertheless like to experiment with wireless communication. In this year's December issue we'll describe a simple circuit using the Vishay TFDU6102. Simpler, but with just the receive channel, is the circuit in **Figure A**. This circuit is built entirely from standard components (a phototransistor, two NPN transistors, and a capacitor) and has little more to do than simply stretch the pulses. The circuit is designed for operation at 9600 baud and can be connected directly to the serial interface of a PC, from which it also derives its power supply. For this to work, the DTR signal must be turned on. Miracles should not be expected of this simple circuit: it is sensitive to ambient light, and so the light path should be shaded. The range is also limited to a modest 10 cm.

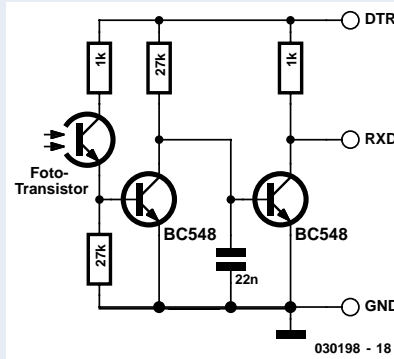


Figure A. A simple IrDA receiver for the PC.

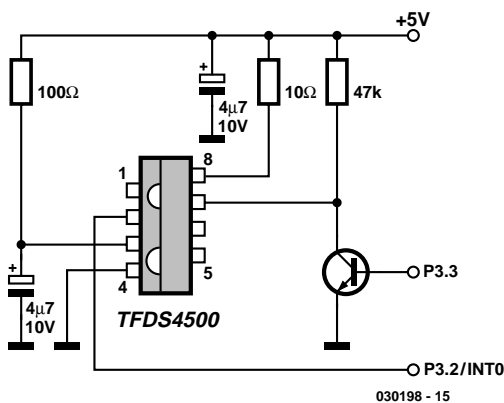


Figure 5. The improved IrDA adapter.

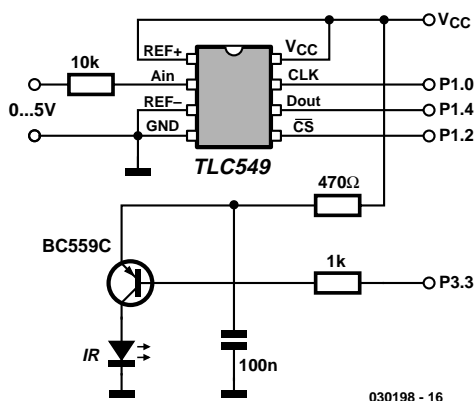


Figure 6. The analogue interface.

### Listing 3. Transmitting analogue readings.

```

'-----
' IrDatlc549.BAS
'-----

$regfile = "89s8252.dat"
Dim N As Byte
Dim Addat As Byte
Declare Sub Adtlc549
Declare Sub Putirda

While 1 = 1
  Adtlc549
  Acc = Addat
  Putirda
  Waitms 250
Wend
End

Sub Adtlc549
  Clk Alias P1.0
  Cs Alias P1.2
  Adin Alias P1.4
  Dim Count As Byte
  Addat = 0
  Cs = 0
  Clk = 0
  Cs = 1
  Delay
  Cs = 0
  For Count = 1 To 8
    Shift Addat, Left, 1
    If Adin > 0 Then Addat = Addat + 1
    Clk = 1
    Delay
    Clk = 0
  Next Count
End Sub

Sub Putirda
  $asm
  clr P3.3
  nop
  nop
  setb P3.3
  mov r2,#43
S1:
  djnz r2,s1
  mov r3,#8
S2:
  jb ACC.0,S3
  clr P3.3
  nop
  nop
  setb P3.3
  sjmp S4
S3:
  nop
  nop
  nop
  nop
  nop
S4:
  mov r2,#41
S5:
  djnz r2,S5
  rr a
  nop
  djnz r3,S2
  ret
$end asm
End Sub

```

ports on the Flash Microcontroller Board.

After each byte is output the program first reads back any echo byte that may have been received. To do this, the number of bytes in the interface buffer is first determined. If more than one byte has been received, the first byte in the buffer must be an echo of the byte that was transmitted, and the second is the reply from the microcontroller. After this a further dummy byte is read in order to clear any bytes from the buffer that may be the result of interference.

There are small hardware differences between the various Palm models. Although the Palm M100 will receive the echo of its own transmitted bytes, this is suppressed in the newer Palm Zire. The program shown here works with both models.

## An integrated IrDA transceiver

The TFDS4500 integrated IrDA transceiver includes a complete IR receiver with photodiode, preamplifier, gain control and output stage. A transmitter LED with driver stage is also included. The package includes lenses to focus the light beam and is made of a filter material that is only transparent to infrared. This relatively inexpensive IC simplifies building an IrDA interface enormously. The receiver outputs the IrDA pulses in negative logic, whereas the transmitter must be driven with positive pulses. The IC is designed for surface mounting, but can relatively easily be mounted onto a prototyping board (**Figure 4**).

**Table 1** shows the pinout of the IC. A capacitor should be connected between GND and VCC as close as possible to the device, since the sensitive receiver will otherwise be prone to interference. A resistor connected to pin 8 sets the current in the transmit LED. With a power supply voltage of 5 V a value of 10  $\Omega$  gives a peak current of around 250 mA. An additional 4.7  $\mu\text{F}$  electrolytic capacitor is required in order to avoid interference being generated on the power supply.

The receiver output on pin 2 can be connected directly to the interrupt input of the microcontroller. In the transmit direction, however, an inverter is required, since the



Figure 7. Data plot.

TFDS4500 is expecting positive pulses. Of course, the polarity of the pulses could be changed in the software, but this has the severe disadvantage that in the default state after reset all the processor's ports are set high. The IrDA transmitter will then be continuously active while the processor is initialising. It is better to insert a transistor to invert the signal.

In comparison with the simple circuit of Figure 2 the TFDS4500 brings significant advantages. Reliable communication is possible at a range of up to around one metre.

## Analogue-to-IrDA interface

Using the software UART described above opens up many possibilities. Here we will suggest a simple instrumentation application using a TLC549 analogue-to-digital converter. The microcontroller continuously transmits readings over the infrared port. Since only transmission, and not reception, is required, the interface can consist of just a simple infrared LED. A PNP transistor carries the high pulse currents, and a range of up to two metres can be achieved. Thanks to the short pulses used in IrDA, the average current

### Listing 4. Displaying readings on the Palm.

```
#IrDAlog.bas
open "com1:",9600, ir as #5
while 1
  gosub AD
  Ualt =U
  draw -1
  for x=1 to 158
    gosub AD
    draw x-1,150-Ualt/2,x,150-U/2
    Ualt=U
    t$=str$(U)+" "
    draw t$,75,60,2
  next x
wend
close #5
end

sub AD
U=-1
while U=-1
  U =get$(#5,0)
wend
return
```

consumption is only around 1 mA.

The program uses a delay of 250 ms and thus sends four readings per second, which is suitable for long-term measurements. If required the speed can be increased up to about 500 readings per second.

The following Palm program in HotPaw BASIC shows a typical application for this device. Readings are continuously plotted on the screen and can be observed as on an oscilloscope.

Timing in this example is entirely the responsibility of the program in the microcontroller, and is in this case set at four readings per second. Depending on the application, it would be possible to take more frequent readings which could then be selectively discarded in the program running on the Palm: this would allow the timeframe to be configured on the PDA.

# Precision Measurement Central (5)

## Part 5: RS485 networking in practice

Design by J. Wickenhäuser

www.wickenhaeuser.com

In the previous instalments of this series featuring the MSC1201 board and microcontroller, we have already discussed many aspects of using it for various applications, ranging from a development platform (for which it is ideally suited) to a precision sensor in a network. In this instalment, we examine how the elements discussed earlier can be combined to form a complete system or network.

You can use the software described here as the starting point for developing your own applications. Despite the use of `print()` statements and floating-point computations, the code written for the application described in this instalment occupies only around 5 kB (by comparison, the free evaluation version of the suggested uC/51 compiler generates up to 8 kB of code).

Figure 1 depicts a network consisting of two MSC1210 boards and an RS232/RS485 converter. This network is intended to work with cable lengths of up to 1,000 metres, with the two MSC nodes at one end and the RS232/RS485 converter at the other end. The network is also powered via the converter.

### Construction aspects

Assuming a supply voltage of 15 V, each device connected to the network requires approximately 30 mA. In order for the nodes to function properly, the supply voltage must be at least 7.5 V. A certain amount of margin must be provided, since a

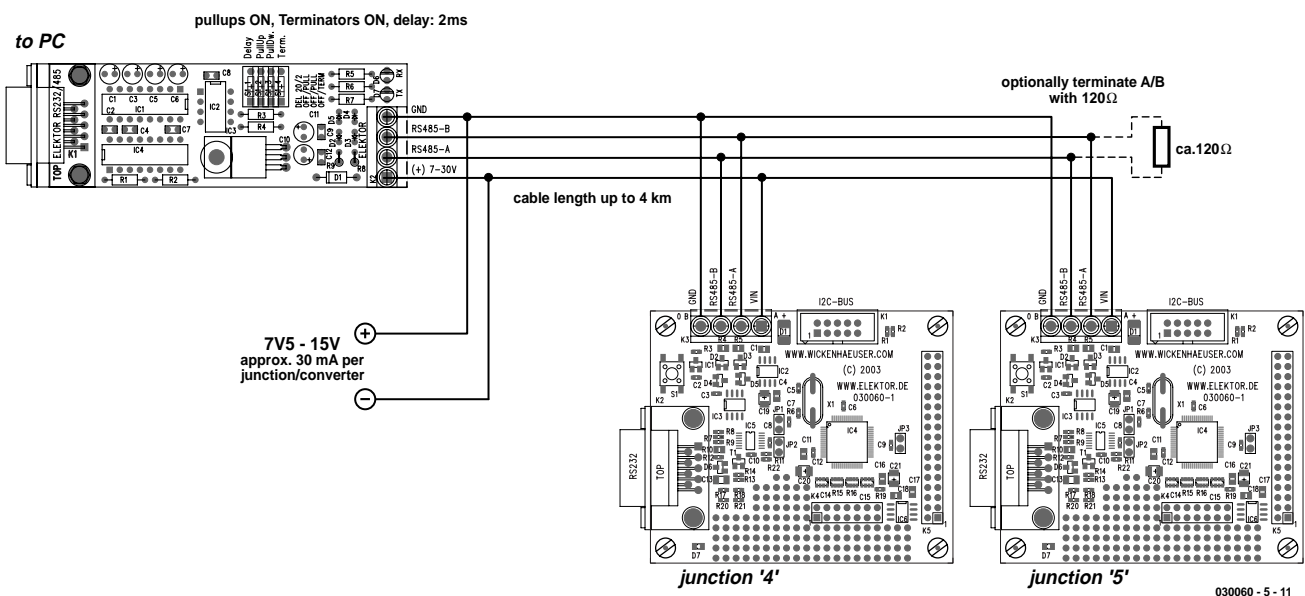


Figure 1. A network composed of two MSC1210 boards and an RS232/RS485 converter.

## Null (zero) modem cable

Almost all communications problems reported by readers in connection with the serial interface (RS232/V.24) can be traced back to the use of improper sub-D connectors and/or incorrect cables.

For the majority of microcontroller boards and projects with serial interfaces that have been described in *Elektor Electronics*, the RS232 connection is made using a 9-way sub-D socket (female connector) on the circuit board. In such cases the connecting cable is a simple serial interface cable (extension cable) with a socket at one end and a plug at the other end, with all pins connected one-to-one. This is also true of the connection between the PC and the RS485 adapter described here.

The MCS1210 forms an exception to this rule, since it is fitted with a sub-D plug (male connector). This means that a 'null modem' (or zero modem) cable must be used to connect the MSC1210 board to a PC. Such a cable has a 9-way socket (female connector) at each end. In addition, the connections between two sets of pins (1 & 4 and 2 & 3) are crossed over. Furthermore, pins 1 and 6 of each connector are directly connected to each other at the connector. The figure shows the wiring diagram of a null modem cable and the pin numbering of the 9-way sub-D socket (as seen from the front of the socket).

In order to be completely sure that you have the proper cable, you should check the connections using an ohmmeter or continuity tester. The jumpers between pins 1 and 6 must without fail be present in the connectors!

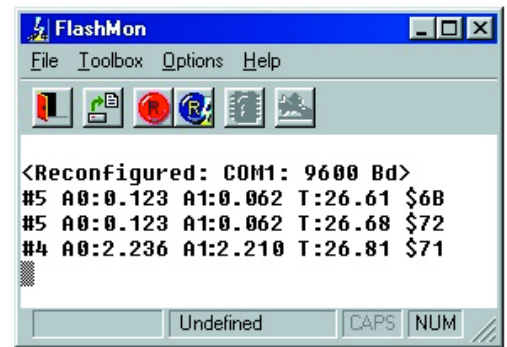
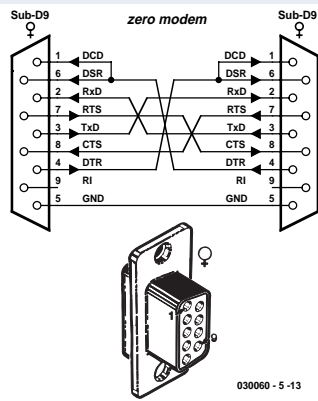


Figure 2. Measured values  
A0: value at A0 input relative to AGND, in volts  
A1: value at A1 input relative to AGND, in volts  
T: chip temperature in °C

exceed 7.5 V. This yields a lead resistance (copper resistance) of approximately 75 Ω (assuming 100 mA). Since the specific resistance of copper is approximately 0.02 Ω·m/mm<sup>2</sup>, a four-conductor cable with a cross-sectional area (c.s.a.) of 0.6 mm<sup>2</sup> is adequate. Fortunately, a cable with exactly these specifications is commercially available as a standard telecommunications cable (with a screen, no less!) at a relatively low cost.

In the case of very long cables, it may not be possible to power the entire RS485 bus from a single point. This was already discussed in the previous instalment. Thus before deciding to use a cable with 'super fat' conductors, you might be better advised to use a supplementary power supply.

With such long cables, it is normally necessary to terminate both ends of the RS485 bus. With the previously mentioned telecommunications cable, the suggested value of 120 Ω will almost always be correct.

## Network software

Once the network has been physically implemented, you still need some software. RS485 requires a reliable data transmission protocol using the parameters 9600 (bit rate), 8, N, 1 (by contrast, the PC and the converter communicate via RS232 at 57,000 bit/s).

A hash mark (#) serves as the start character. This is followed by the address of the destination node for the command. After this come the actual data (in the form of readable text), followed at the end by a checksum that allows the recipient to determine whether the received data are correct. In our case, the checksum is marked by a \$ character and consists of two bytes containing the lateral sum of all preceding characters after the start character. In normal practice, more complicated schemes are used, such as polynomial

node may sometimes draw additional current, such as when transmitting. With a capacity of 100 mA on the bus, you will in any case be

on the safe side.

With a total round-trip distance of 2000 metres, this means the voltage drop along the cable must not

## A professional application



The company GeoPrecision manufactures sensor networks, among other products. The illustrated sensor has been specifically developed for use in underground mining. Due to its high accuracy and stability, an MSC microcontroller is used for this purpose.

The complete circuitry is housed inside a 52-mm stainless-steel enclosure. The enclosure is waterproof and hermetically sealed. Nevertheless, new software can be downloaded whenever desired, and the calibration factors are externally accessible. This capability is provided by a small operating system in each sensor, which manages the free space in the Flash memory. This means that up to 30 kB of program memory are available for individually programming each sensor. Distance is no longer an issue, thanks to the use of RS485.

Naturally, for such an application it is highly important for the transmitted data to be absolutely correct. Consequently, checksums are used intensively. Nevertheless, each sensor essentially contains a 'shrunk' version of the MSC board, for which all of the software has been developed using the uC/51 compiler.



## List of articles

- Part 1: Precision Measurement Central (1): with an MSC1210 development platform (July/August 2003)  
Introduction to the overall concept of the MSC1210 experimenter's board
- Part 2: Precision Measurement Central (2): printed circuit board and software suite (September 2003)  
Printed circuit board, layout and components; software descriptions
- LC Display with I<sup>2</sup>C Bus (September 2003)  
I<sup>2</sup>C bus description and a user-friendly LC display for the MSC1210
- Part 3: Precision Measurement Central (3): Flash for all purposes (October 2003)  
The Flash memory and how to use it
- Part 4: Precision Measurement Central (4): RS485, or networking the MSC1210 micro (November 2003)  
Network architecture and a simple RS485/RS232 converter for the PC
- Part 5: Precision Measurement Central (5): RS485 networking in practice

checksums. There is a large amount of technical literature available on this subject in convenient 'pre-digested' form.

As already mentioned in the fourth instalment, a brief pause must be inserted between activating the transmitter section and sending the start character, in order to avoid transmission errors. The PC converter uses a time window for activating the transmitter section. For this reason, the pull-up and pull-down resistors should be connected, since otherwise the first character might be transmitted

incorrectly. The PC converter will thus be set to transmit when the first byte from the PC appears. All jumpers must be fitted in the PC converter, although the terminator is naturally only necessary if the converter is located at the end of the cable.

**Figure 2** shows a typical data packet. This was generated by the 'perfectly ordinary' terminal emulator program FlashMon, which comes with the uC/51 compiler.

In order to test the network using a standard terminal emulator, we can go even a step further and ignore the checksum when transmitting. It will then only be contained in the response. The demo program can only handle one command, which displays the measured value — but at least the source code is free.

When multiple boards are used as network nodes, the software accompanying this article should be downloaded to each of the MSC1210 boards, with each node being assigned its own ID (designated node id in the source code, with a default value of 5).

Now you can send a command to node 5 by typing #5<CR> (where <CR> refers to the Return key). The LED of this board will go on for approximately one second, while the LEDs of the other nodes will blink briefly. After this you will receive a measured value as shown in Figure 2. The node also outputs this

information in a somewhat more extensive form via its (local) RS232 interface (which is used to download the software).

A new version of the demo software produced using version V1.10.12 of the compiler is now available in directory  
...\\scr\\msc1210\\Elmet\\Elmet485\\ELM\_FLASH.

However, you do not necessarily have to download the entire compiler (12 MB) in one go, since the individual parts can also be downloaded from the *Elektor Electronics* website.

The MSC microcontrollers have many additional features, all of which are extremely interesting. With the elements that have been described in this series of articles, which now comes to a conclusion, you should not find it difficult to use them for your own purposes.

(030060-5)

## Links:

[www.wickenhaeuser.com](http://www.wickenhaeuser.com)

ANSI C compiler for the 8051 family, including the source code for all of the MSC1210 demo programs

[www.elektor-electronics.co.uk](http://www.elektor-electronics.co.uk)

Source of fully assembled and tested MSC1210 boards

[www.geoprecision.com](http://www.geoprecision.com)

Several practical applications

# Timebase Calibration for DDS RF Signal Generator

Design by G. Baars. PE1GIC

pe1gic@amsat.org

The DDS RF Signal generator described in the October 2003 issue of *Elektor Electronics* has an internal 10 MHz quartz crystal oscillator for the AD9851 DDS chip. Obviously, the output frequencies supplied by the generator are only correct if the quartz crystal is adjusted to exactly 10.000 MHz using trimmer C22. For many applications like repair work on 445 kHz or 10.7 MHz IF amplifiers, the frequency of the signal supplied by the instrument is not terribly important. However, 'netting' narrow band FM (NBFM) equipment like a 6-m (50 MHz) amateur radio transceiver does require a frequency-accurate test source. After all, if you adjust the transceiver with the aid of a signal generator, any error in the frequency supplied by the latter is copied by the local oscillator(s) of the 50-MHz receiver.

## Off-air — free and extremely stable

If you think that an £10k very high-end and hyper-accurate frequency source is required to calibrate your RF DDS Signal Generator, you will be pleasantly surprised to know that such sources are (1) free, (2) available in your home and (3) accurate to a level that should exceed most, if not all, amateur requirements. We're talking about MW broadcast transmitters (between 600 kHz and 1,500 kHz) in your region. Signals received from such a transmitter can be used without too much of a problem to calibrate the timebase oscillator in your signal generator. All you need is an add-on mixer that will produce 'zero beat' when the two frequencies are equal. This will be indicated by an oscilloscope.

## Practical circuit

The RF mixer shown in **Figure 1** is a dead conventional design around a dual-gate MOS-FET. The AM broadcast signal arrives at gate 1. A strong AM broadcast station can be selected between 600 kHz and about 1,500 kHz using an L-C tuned circuit consist-

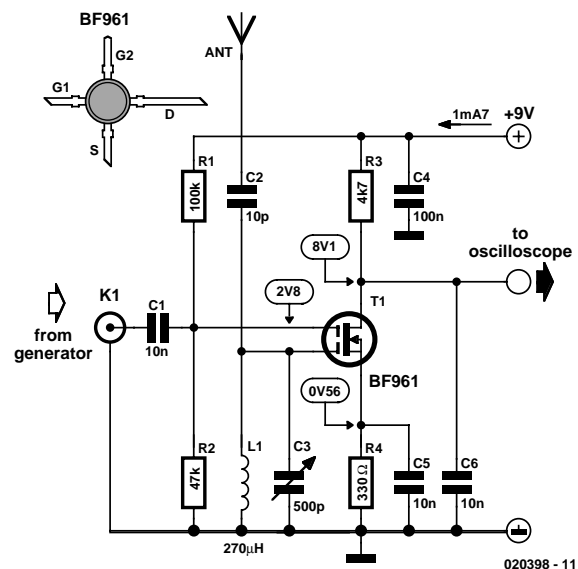
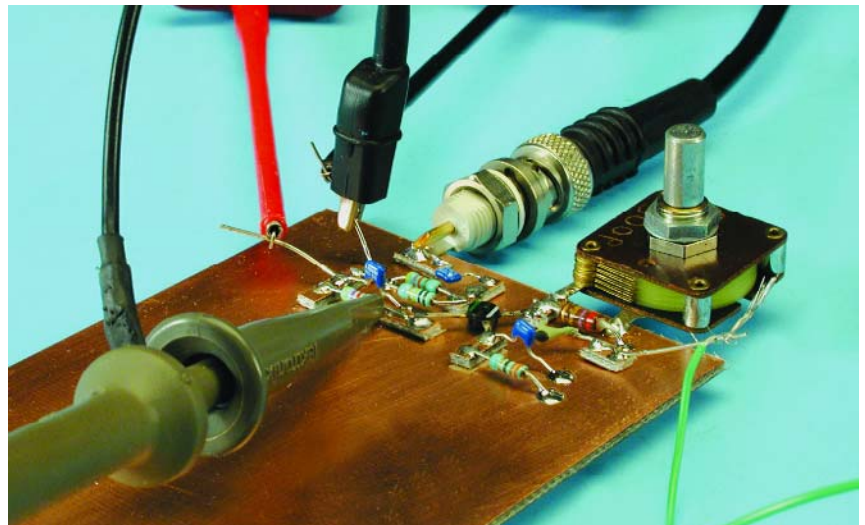


Figure 1. This beat detector employs an oscilloscope and an MW broadcast signal to calibrate the 10-MHz timebase oscillator in the DDS.

ing of a choke and a 500-pF mica tuning capacitor. The antenna consists of a few metres of wire. The RF DDS Signal generator is tuned to the official AM station frequency and its output signal injected into gate 2 via

a 10nF coupling capacitor. The mixer output signal can be taken from the drain of T1 and fed to an oscilloscope. The lower the frequency variation of the signal shown by the 'scope, the closer you are to

10.000 MHz with your calibration of the generator timebase.

### Practical use

Start by finding the strongest MW broadcast transmitter in your region. If necessary, use a frequency list to find a suitable station. Remember, the add-on mixer is not designed for selectivity or sensitivity so you need

to find a really strong signal. Set the oscilloscope to 2-ms timebase, AC-coupled and high sensitivity. Watch the trace on the 'scope, tune C1 for resonance and listen to a MW radio to make sure you're correctly tuned. Keep the antenna away from noise sources like fluorescent tubes and PCs. On the signal generator, set an output level of 0 dBm and enter the **exact** fre-

quency of the station as taken from the list. Watch the 'scope display as you carefully adjust C22 in the generator. At some point you will see the frequency difference ('beat' note) between the two sources in the form of slow amplitude variation(s). The optimum setting for C22 is obtained when the two signals approach 'zero beat' as close as possible. A calibration accuracy of less than 1 Hz should be within easy reach.

(020398-1)

# Digital Logic Compared

## Tackling compatibility issues

By David Daamen

In response to repeated requests, in this article we present an overview of a number of logic families and their properties. We also briefly examine certain important parameters. Finally, we explain what you need to pay attention to when working with digital logic.

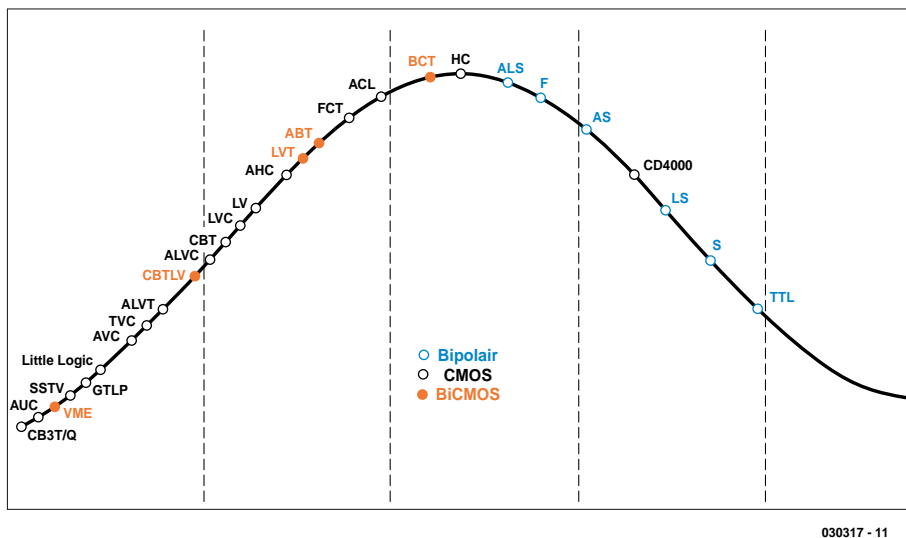


Figure 1. Market share of various logic families.

Although the ability of transistors to amplify signals excited the most interest from the very start, nowadays transistors are probably more often used as switches. It is precisely this (digital) application that forms the basis for this article, since a wide variety of logic functions can be implemented using combinations of switching transistors, thus enabling us to have transistors do our computations.

The importance of digital-logic building blocks can readily be seen from the history of the various 'logic families'. Many of them are already obsolete, and new ones are yet to come. This means that we certainly cannot deal with this entire subject within the scope of this article. To help give you a better idea of the present state of affairs, we direct

your attention to **Figure 1**.

In this capsule summary of the current market share of the various families (*source: Texas Instruments*), the families that are being phased out are shown at the right. As you can see, it is thus not a good idea to use TTL (for example) in new designs. At the far left, you can see a number of families that are just now being introduced. These families are smaller and even faster, or they may have lower power consumption. Incidentally, the latter two properties — high speed and low power consumption — are usually mutually contradictory, as we will see later on.

In this article, we concentrate on the families that are presently the most interesting for electronics hobbyists. For instance, we do not give any attention to the fast emitter-coupled logic (ECL) family, due to its excessively high price by hobbyist standards.

### What makes it tick

Among the older, still 'living' members of the collection of digital logic families, the previously mentioned TTL family is probably the best

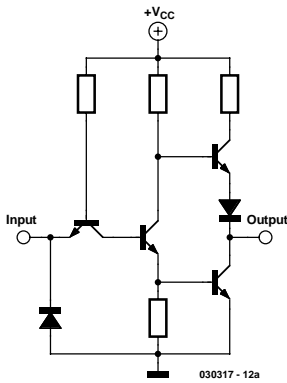


Figure 2a. A bipolar TTL inverter.

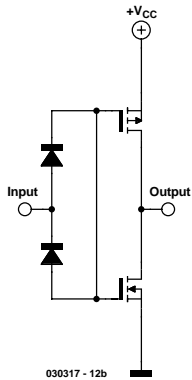


Figure 2b. A CMOS inverter.

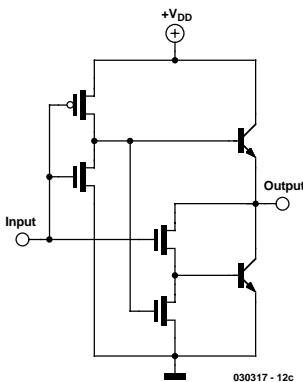


Figure 2c. A BiCMOS inverter.

known. A TTL IC is built using bipolar transistors. 'TTL' stands for 'transistor-transistor logic'. The structure of a simple gate using TTL technology, such as an inverter, is shown schematically in **Figure 2a**.

For comparison, **Figure 2b** shows the schematic of an inverter built with CMOS logic. CMOS ICs are constructed using n- and p-channel

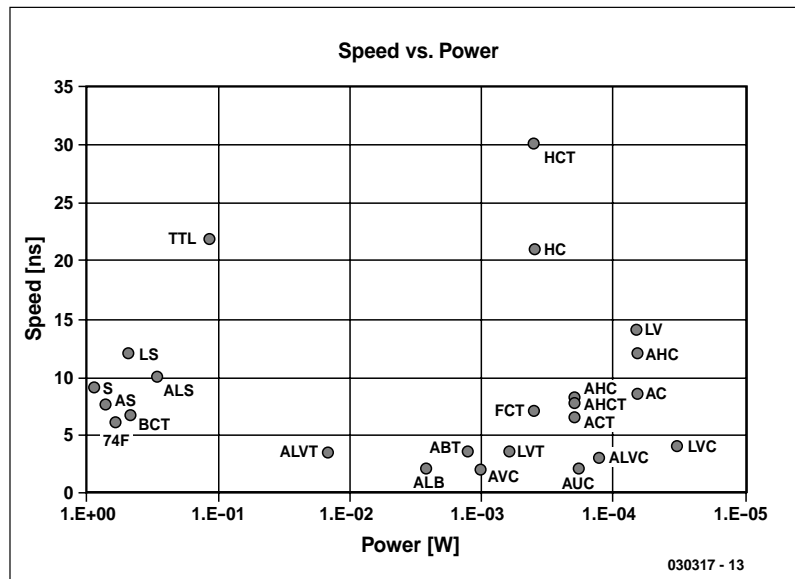


Figure 3. Speed versus power: a measure of performance.

MOSFETs, which are complementary types of field-effect transistors. This is why they are called 'CMOS logic', since 'CMOS' stands for 'complementary metal oxide semiconductor'. Finally, **Figure 2c** shows the internal schematic diagram of a BiCMOS inverter. This type of inverter is built using a combination of bipolar and CMOS transistors.

ICs made with CMOS technology use much less power than their bipolar predecessors, since in principle a CMOS IC consumes power only when it switches states. However, CMOS ICs have the disadvantage that their FET inputs are much more vulnerable to damage from static electricity and excessive input currents in general.

### Important parameters

For hobbyists, in many cases the availability and prices of the components to be used are the primary

considerations in designing a circuit. As applications become more professional, speed and power consumption quickly become important factors in selecting specific components or families of components. The 'speed-power product' is frequently used to compare the performance of different logic series. This product is shown graphically in **Figure 3** for a number of well-known families.

This chart is based on typical values of power consumption and speed for each of the families. In this case, the speed refers to the typical time it takes for a change at the input to affect the state of the output. This can be illustrated by the diagram in **Figure 4**, which shows that the time delay (propagation delay) for the associated AND gate is 3 ns. The output does not go high until three nanoseconds after the input goes high. The same applies to the high-to-low transition.

Another important parameter is the noise margin. This number expresses the difference in the voltage levels of the driver and the receiver for a particular logic value. We can thus calculate the noise margin for logic 0 and logic 1 as follows:

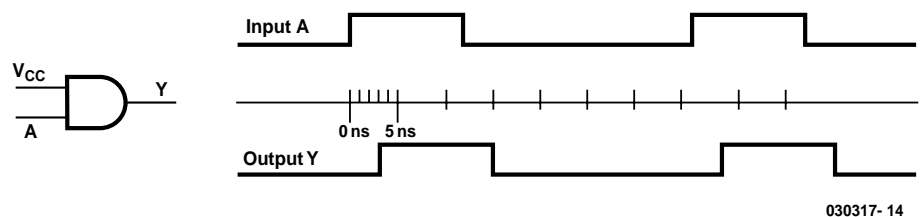
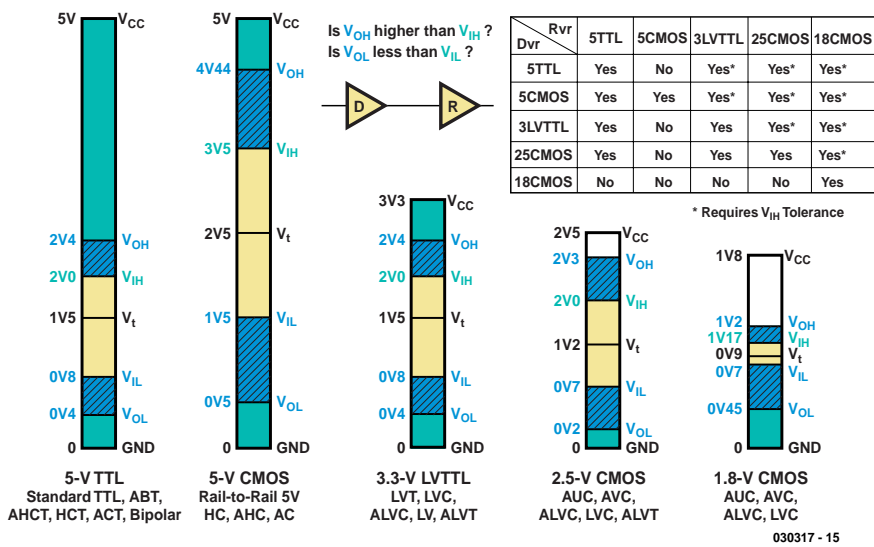


Figure 4. Propagation delay diagram for a CMOS AND gate.



$$\text{Noise Margin Output High} = V_{OH} [\text{driving device}] - V_{IH} [\text{receiving device}]$$

$$\text{Noise Margin Output Low} = V_{IL} [\text{receiving device}] - V_{OL} [\text{driving device}]$$

Dvr	Rvr	5TTL	5CMOS	3LVTTTL	25CMOS	18CMOS
5TTL	Yes	No	Yes*	Yes*	Yes*	Yes*
5CMOS	Yes	Yes	Yes*	Yes*	Yes*	Yes*
3LVTTTL	Yes	No	Yes	Yes*	Yes*	Yes*
25CMOS	Yes	No	Yes	Yes	Yes*	Yes*
18CMOS	No	No	No	No	No	Yes

\* Requires V<sub>IH</sub> Tolerance

Figure 5. Input and output levels. (Note: when calculating noise margins and compatibility, use the minimum values for 'Output High' and the maximum values for 'Output Low'. A compatibility marked with \* in the table is only true for ICs that can tolerate a high input voltage, which is referred to as 'V<sub>IH</sub> tolerance'. Refer to the data sheets of specific ICs.)

To calculate the noise margin for a logic 1, we thus take the voltage level of the driving gate for a high output (V<sub>OH</sub>) and subtract the voltage level for which the receiving gate detects a logic 1 (V<sub>IH</sub>). For a low level, the same procedure applies, but we use the values V<sub>IL</sub> and V<sub>OL</sub> instead.

The noise margin can also be used for other purposes than assessing the 'noise immunity' (for which higher values are better). It can be used to check whether certain families are compatible with regard to signal levels. A negative value for the noise margin is a sign of incompatibility!

The final important set of parameters that we want to examine here are the fan-in and fan-out. These numbers indicate how many gates a given gate can drive, and they are determined by the currents supplied and drawn by the gates.

$$\text{Fan-out Output High} = I_{OH} [\text{driving device}] / I_{IH} [\text{receiving device}]$$

$$\text{Fan-out Output Low} = I_{OL} [\text{driving device}] / I_{IL} [\text{receiving device}]$$

As you can see, these calculations must again be made for each of the two logic levels. Naturally, the smaller of the resulting numbers determines the actual fan-out. The fan-out calculations can be used not only for the gates of a particular family, but also among different families.

### Overview

Now that we have identified the most important parameters, it's time to present an overview. **Table 1** shows the supply voltages and the technologies used for a number of families, along with the logic level values under the 'Compatibility' heading. These values are further elaborated in **Figure 5**. Note that

**Table 1. Overview of a number of families.**

Family	Technology	Compatibility		Drive		Static Current	Speed
		Input V <sub>il</sub> /V <sub>Ih</sub>	Output V <sub>ol</sub> /V <sub>oh</sub>	I <sub>ol</sub> (mA)	I <sub>oh</sub> (mA)	I <sub>cc</sub> (mA)	T <sub>pd max</sub> (ns)
1.8 V							
AUC	CMOS	CMOS	CMOS	8	-8	0.01	2
2.5 V							
AVC	CMOS	CMOS	CMOS	8	-8	0.04	2
3.3 V							
ALVT	BiCMOS	CMOS	LVTTTL	24	-8	4.5	3.5
LVT	BiCMOS	LVTTTL	LVTTTL	64	-32	0.19	3.5
ALVC	CMOS	LVTTTL	LVTTTL	24	-24	0.04	3
LVC	CMOS	LVTTTL	LVTTTL	24	-24	0.01	4
ALB	BiCMOS	LVTTTL	LVTTTL	25	-25	0.8	2
AC	CMOS	CMOS	CMOS	12	-12	0.02	8.5
AHC	CMOS	CMOS	CMOS	4	-4	0.02	11.9
LV	CMOS	LVTTTL	LVTTTL	8	-8	0.02	14
5 V							
FCT	BiCMOS	TTL	TTL	64	-15	0.08	7
ABT	BiCMOS	TTL	TTL	64	-32	0.25	3.5
AHC	CMOS	CMOS	CMOS	8	-8	0.04	7.5
AHCT	CMOS	TTL	CMOS	8	-8	0.04	7.7
AC	CMOS	CMOS	CMOS	24	-24	0.04	6.5
ACT	CMOS	TTL	CMOS	24	-24	0.04	8
74F	Bipolar	TTL	TTL	64	-15	120	6
BCT	BiCMOS	TTL	TTL	64	-15	90	6.6
HC	CMOS	CMOS	CMOS	6	-6	0.08	21
HCT	CMOS	TTL	CMOS	6	-6	0.08	30
AS	Bipolar	TTL	TTL	64	-15	143	7.5
ALS	Bipolar	TTL	TTL	24	-15	58	10
LS	Bipolar	TTL	TTL	24	-15	95	12
S	Bipolar	TTL	TTL	64	-15	180	9
TTL	Bipolar	TTL	TTL	16	-0.4	22	22

the indicated values are only representative. When making the calculations, it is best to use values taken from the data sheet for a particular component. This also applies to the values listed for drive current and static current in Table 1. Like the delay time  $T_{pd(max)}$ , the listed values are average or typical values for a family.

### Using digital logic

When you actually use digital logic, there are a number of things you have to keep in mind. First, you have to watch out for possible incompatibilities among various versions made by different manufacturers. Each type of IC has a unique name, which can always be used to find a data sheet for it (on the Internet). Unfortunately, space limitations prevent us from describing all the possible options, but the naming convention for Texas Instruments ICs is shown in **Figure 6** as an example.

Naturally, you must also take maximum input currents into account in your design. Inputs should be protected against voltage spikes, such as may originate from electrostatic discharge (ESD). One way to do this is to connect a pair of reverse-biased diodes between the input and the supply voltage and the input and ground. These diodes will divert voltage spikes, thus protecting the input. Special transient protection diodes are commercially available, and there are also complete ICs based on the same principle, which can be used for applications such as protecting a complete bus with a single IC. By the way, most families can tolerate being touched by a person charged to 24 V; under normal circumstances, this provides adequate protection.

The next consideration is that connections between ICs, and between ICs and the 'outside world', must be kept as short as possible. As the operating speed increases, it quickly becomes necessary to regard long interconnecting lines as transmission lines. Phenomena such as reflections and overshoots can be controlled by using termination resistors. It is also better to avoid long parallel traces on circuit boards,

**Table 2. Summary list of manufacturers.**

	TI	Fairchild	Hitachi	IDT	ON	Pericom	Philips	Toshiba
Bipolar	ALS	ALS	—	—	—	—	ALS	—
	AS	AS	—	—	—	—	—	—
	74F	F	—	—	F	—	F	—
	LS	LS	—	—	LS	—	—	—
	S	S	—	—	—	—	—	—
	TTL	TTL	—	—	—	—	—	—
BiCMOS	ABT	ABT	ABT	—	—	—	ABT	ABT
	ALB	—	—	—	—	—	—	—
	ALVT	—	—	—	—	ALVT	ALVT	—
	BCT	BCT	—	—	BC	—	—	BC
	LVT	LVT	LVT	—	—	—	LVT	—
CMOS	AC/ACT	AC/ACT	AC/ACT	—	AC/ACT	—	—	AC/ACT
	AHC/AHCT	VHC	—	—	VHC	—	AHC	VHC
	ALVC	VCX	ALVC	ALVC	VCX	ALVC	ALVC	VCX
	AUC	—	—	AUC	—	—	AUC	—
	AVC	—	—	—	—	AVC	AVC	—
	CBT	FST	—	FST/QS	—	PI5C	—	—
	CBTLV	—	—	CBTLV	—	PI3B	—	—
	CD4K	CD4K	—	—	MC1400	—	—	—
	FCT	—	—	FCT	—	FCT	—	—
	HC/HCT	HC/HCT	HC/HCT	—	HC/HCT	—	HC/HCT	HC/HCT
	LV-A	LVQ/LVX	LV	—	LVQ/LVX	—	LV	LVQ/LVX
	LVC	LCX	LVC	LVC/LCX	LCX	LCX/LPT	LVC	LCX

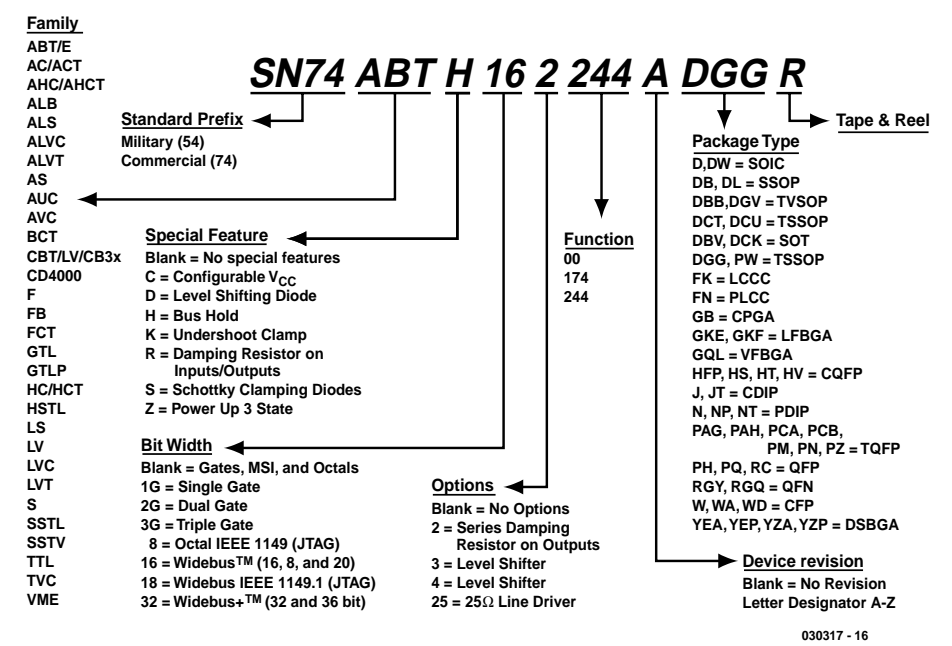


Figure 6. Texas Instruments device naming convention.

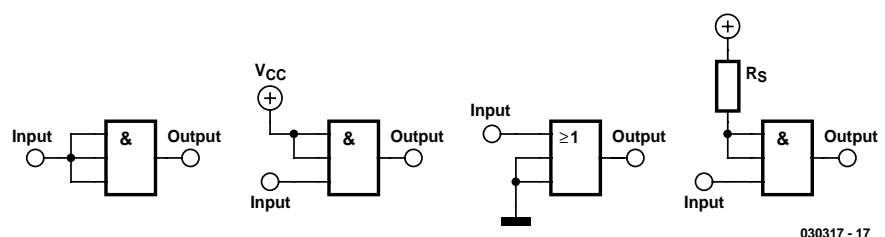


Figure 7. Do not leave unused inputs open.

### Table 3. Commonly used abbreviations of manufacturer names.

AD	Analog Devices	HM	Harris Microwave	SI	Siliconix
AM	Advanced Micro Devices	IDT	Integrated Devices Technology, IDT	SN	Texas Instruments, TI (Standard)
AT	Atmel	IRF	International Rectifier	SNJ	Texas Instruments, TI (MIL/QML Qualified)
bq	Benchmark	IP	Integrated Power	SPT	Signal Processing Technologies, SPT
CA	RCA (analog)	LM	National	SSD	Samsung Electronics
CD	RCA (digital)	M	Mitsubishi	SU	Signetics
CL	CComlinear Corp.	MACH	Vantis (MACH, PLD)	SY	Synergy Semiconductor
CS	Crystal Semiconductor	MAX	Maxim	TA	Toshiba
CS	Cherry Semiconductor	MB	Fujitsu	TC	Toshiba
CY	Cypress Semiconductor	MC	Motorola	TD	Pro-Electronics
DG	Siliconix	MN	Micro Networks	TL	Texas Instruments (analog, Lin- ear)
DS	Dallas Semiconductor	NDS	National Semiconductor	TMS	Texas Instruments
DM	National Semiconductor (digital)	NE	Signetics	X	Xicor
ED	IElectronic Designs Inc, EDI	PI	Pericom	XC	Xilinx
EL	Elanec	PM	PMI 'Analog Devices'	XR	Exar Corp.
EP	Altera (Classic series)	PWM	Siliconix	uA	Fairchild
EPC	Altera (EPROM)	QL	Quick Logic	UC	Unitorde integrated circuits
EPF	Altera (Flex series)	QSI	Quality Semiconductor	Z	Zilog
EPM	Altera (MAX series)	SA	Signetics	ZD	Zeltex
HA	Hitachi (analog)	SD	SGS Thomson		
HAT	Hitachi	SE	Signetics		
HD	Hitachi (digital)	SG	Silcon General		
HI	Harris				

due to the risk of crosstalk.

To prevent the supply voltage from 'collapsing' when signals are switched, the inductance of the supply lines must be as small as possible, which can be achieved by measures such as using large supply and ground planes. You should also use decoupling capacitors located as close as possible to the supply pins of the ICs, in order to counter this effect. Keep current paths as short as possible in the entire circuit (for EMC!).

Never leave unused inputs open. An undefined logic level can lead to spontaneous oscillations, which can cause problems for more than just the affected gate. An oscillating gate can destroy the entire IC, or perhaps even cause the whole circuit to malfunction. Four possible solutions are shown in **Figure 7**. The simplest solution is to simply connect the unused inputs of a multiple gate together. If you use this approach, be sure to check the fan-out of the driver IC! Another

solution, which does not create any fan-out problems, is to connect the unused input to one of the two logic levels, which usually means either the supply voltage or ground (shown in the middle of **Figure 7**). If you use TTL ICs, a series resistor ( $R_s$ ) can be inserted between the supply voltage and an unused input in order to limit the current drawn by the input.

Furthermore, with flip-flops, registers and latches, you have to pay attention to set-up and hold times. Actually, that falls outside the scope of this article, but in short, what it means is that the inputs must be stable for a certain length of time before and after an active clock edge. If you want to know more, we suggest you read the data sheets and application notes from the well-known manufacturers.

## Conclusion

The names of a number of well-known manufacturers, with the logic families that are currently in production and associated abbreviations, are listed in **Table 2**. A more extensive list of abbreviations, such as are found on the ICs, is given in **Table 3**.

(030317-1)

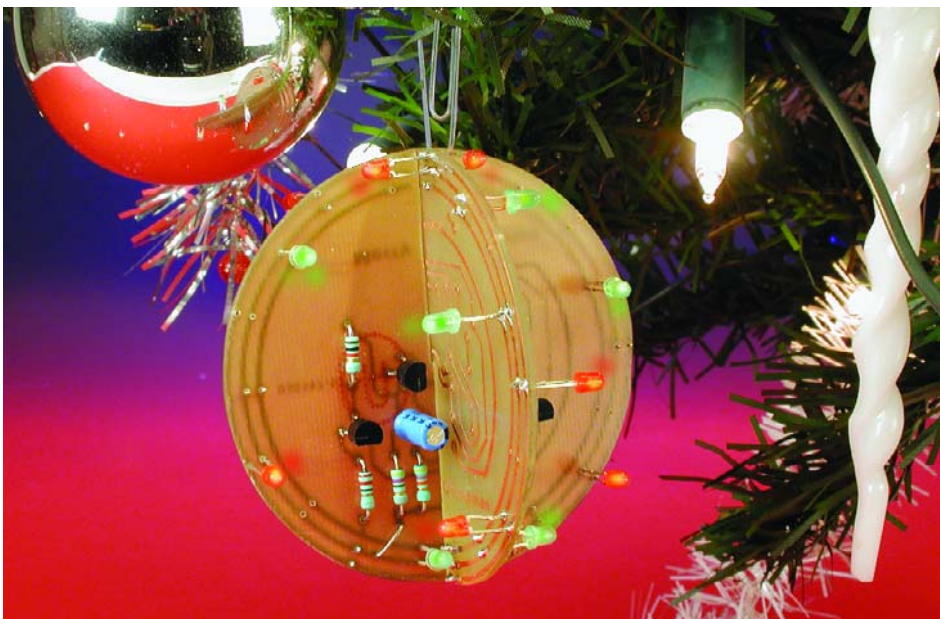


# LED Christmas Decoration

## A novel ornament

Design by M. Gaudin

As this issue comes through the letterbox, it's around the time that the Christmas decorations are taken out of the attic. And if you hurry up there is just enough time to add a homemade electronic ornament to your tree. This article shows you how to do just that



It has become something of a tradition that the December issue of *Elektor Electronics* includes a project for some kind of distinctive Christmas decoration. Since a true electronics hobbyist should not be content with a standard Christmas ornament, we have provided the opportunity to add a personal, homemade touch.

An electronic Christmas ornament generally falls in one of two camps: either it produces suitable sound effects or it produces light effects. A 'suitable sound effect' during these festivities quickly becomes a full-fledged rendition of a Christmas song, and this cannot be easily accomplished

without the use of dedicated ICs. For this reason the optical variant is usually chosen.

Even these can be made as complex as you like. A few years ago we took this route and designed a micro-processor controlled lights effect generator that had much to offer. But gauging from their reactions, it appears that most readers would prefer a simpler design for Christmas decorations. So this time round there is no complex circuit, no processor and no software, but just some simple electronics that can be con-

structed by anybody. The shape of the design makes it particularly appealing.

### An astable

A quick glance at the circuit diagram in **Figure 1** confirms that this really is a very simple circuit. Four transistors, a row of LEDs, two batteries for the supply, and that's about it. We would almost call it child's play.

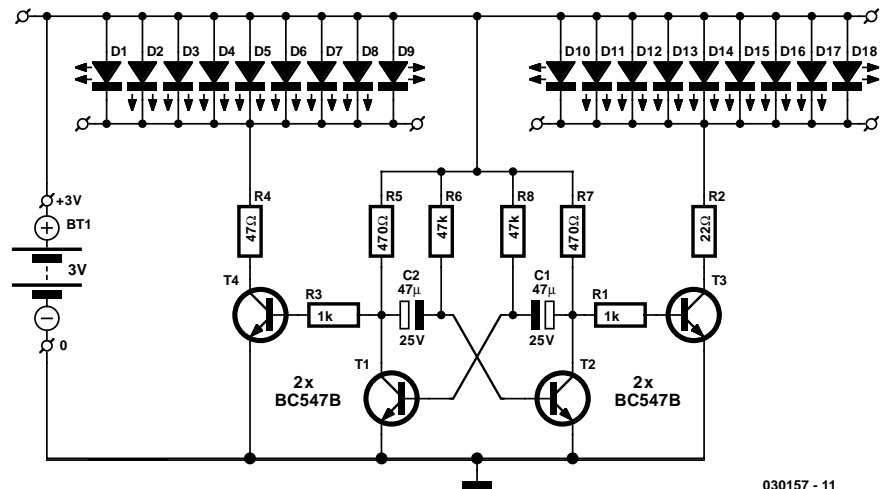
At the heart of the circuit are T1 and T2, configured as an astable multi-vibrator, which is usually called an 'astable' or 'AMV'. One of the two transistors in this circuit is always conducting, while the other is blocking. However, due to the regular charging and discharging of electrolytic capacitors C1 and C2 the conducting and blocking states of the transistors alternate as regular as clockwork. The circuit therefore doesn't have a stable state and so really lives up to its name. It is in fact a square wave oscillator, and circuits such as those round T1 and T2 are therefore often used as simple clock generators.

The only difference with those clock generators is that they usually operate at frequencies of several KHz or more, whereas in our astable the RC time-constants of R6/C2 and R8/C1 have been chosen to give a low enough frequency

that is visible to eye.

The collectors of T1 and T2 are connected to two driver transistors. They switch two rows of parallel-connected LEDs, which will light up alternately at the oscillator frequency. The frequency at which D1-D9 and D10-D18 alternately light up is about 2 Hz. Should you find this too fast or too slow, you can easily modify this by increasing or decreasing C1 and C2 respectively.

Resistors R4 and R2 are used to set the current through the LEDs. It



030157 - 11

Figure 1. The electronics in the bauble is limited to an astable multivibrator driving two groups of LED.

### COMPONENTS LIST

**Resistors:**

- R1,R3 = 1kΩ
- R2 = 22Ω
- R4 = 47Ω
- R5,R7 = 470Ω
- R6,R8 = 47kΩ

**Capacitors:**

- C1,C2 = 47μF 25V radial

**Semiconductors:**

- D1-D9 = LED, high-efficiency, red
- D10-D18 = LED, high-efficiency, green
- T1-T4 = BC547B

**Miscellaneous:**

- Two penlight (AA) batteries
- Battery holder for above PCB, order code **030157-I**, see Readers Services page

is noticeable that these resistors are unequal in value. The reason for this is that D1-D9 are red LEDs and D10-D18 are green LEDs.

These types of LEDs have two different characteristics. Firstly, the brightness of green LEDs is clearly less than that of the red ones. Secondly, the forward voltage drop across green LEDs is greater than that of red ones, so there is less of a voltage drop across R2 than across R4. R2 has therefore been given a smaller value so that a large enough current flows through the green LEDs.

### PCB

As we said earlier, what makes this circuit special is its PCB design. Since we wanted to make a bauble, the PCB obviously had to be adapted for that. **Figure 2** shows the result: a circular PCB in a 'bauble like' form, finished with a festive white lacquer on one side and for the occasion it has silver tracks. The control circuit is in the centre of the board, with the LEDs mounted along the outer edge. Along this edge are three circular tracks; the middle one is the positive supply, which goes to the anodes of all LEDs. The outer track is connected to the cathodes of the red LEDs and the inner track is for the

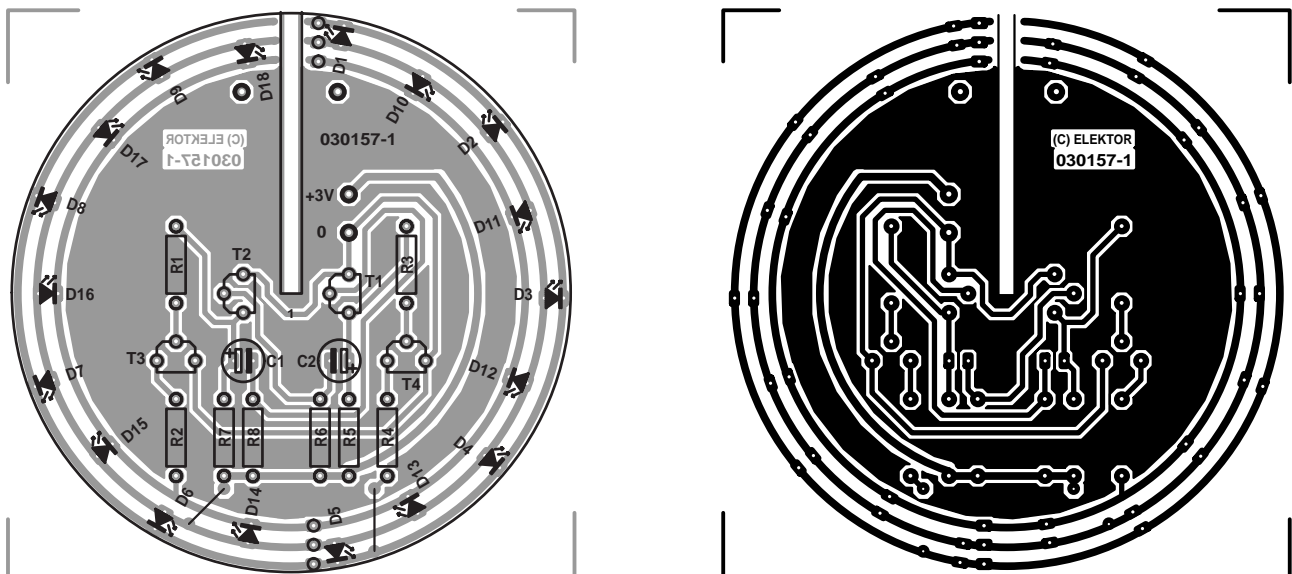


Figure 2. The PCB is shaped like a bauble, has an appropriate colour and can be extended with a second board.

cathodes of the green LEDs.

We found that the best effect was obtained with red and green LEDs mounted alternately on the board. You may also mount the LEDs on the track side, as well as on the component side. Do take care that you don't accidentally connect red and green LEDs in parallel, because the lower forward voltage drop of red LEDs will make just the red LEDs light up.

A noteworthy feature of the board is the provision of a slot, so that a second PCB can be inserted at right-angles, giving the bauble a three-dimensional shape. The second PCB should only contain the LEDs and the three circular tracks on both PCBs should be connected in parallel using two rows of three wire links. The PCBs already have holes for these. The main photo with this article shows clearly how this is done.

With two PCBs, the maximum number of LEDs obviously increases from  $2 \times 9$  to  $2 \times 18$ , which affects the values of resistors R2 and R4. When using the maximum number of LEDs ( $2 \times 18$ ), R2 should be reduced to  $15 \Omega$  and R4 to  $33 \Omega$ . However, when using two PCBs a good effect may also be obtained with

fewer LEDs. On our prototype we spread them out evenly and ended up with 12 LEDs per board. The values given in the circuit diagram for R2 and R4 are still suitable for use with this number of LEDs.

## Supply

As can be seen from the circuit diagram, the circuit requires a supply voltage of 3 V. Since the current consumption is fairly modest, the electronic bauble can easily be powered by two AA cells. Our prototype with  $2 \times 12$  LEDs had an average current consumption of about 35 mA, so two alkaline cells with a typical capacity of 1500 mAh should last for up to 50 hours. If that is insufficient, you could always use a stabilised 3 VDC mains adapter.

## And finally

Unfortunately, in practice there is a noticeable deviation in brightness

between LEDs. If you want to make sure that all LEDs have a similar brightness it may be best to buy a few extra and select the best ones. This is easily achieved using a small stripboard with an IC socket and a series resistor of either  $47 \Omega$  or  $22 \Omega$  to the 3 V supply. If you place several LEDs in the socket at the same time you can see clearly how well their brightness matches.

One thing we should mention about the PCB. If you look closely at Figure 2, you'll see that there are two wire links: one from the outer circular track to R4 and one from the middle circular track to R7. Don't forget these wire links; otherwise the circuit will certainly refuse to operate! One final remark: since the boards weigh very little it is perfectly safe to hang them up from the battery cables. However, for completeness the PCB also has two extra holes for a 'real' cord.

(030157-1)

# The Art of Soldering

## Making proper solder joints

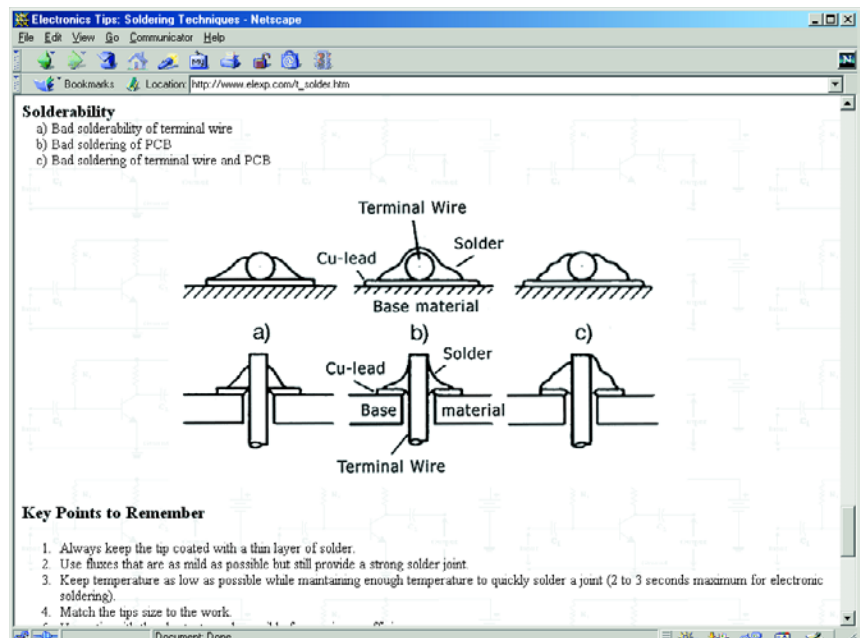
By Harry Baggen

Building up electronic circuits usually involves interconnecting all components using wires or circuit board tracks. Whatever the method, solder joints are used to ensure good mechanical stability and the best possible electrical conduction. Manual soldering is not difficult once you have some experience. If you are not too confident about your own skills in this area, a lot can be learned from articles and mini courses published on the Internet.

Many readers of this magazine will have done a lot of soldering in their lifetime, so they probably need not be told that good soldering skills are required if you want to repair electronics equipment or build your own using components, wires and/or a circuit board. And yet, it may be a good idea to have a good look at your solder joints, because their quality is of the utmost importance. Typical errors are (1) using far too much solder tin and (2) insufficient heating of the joint. Both imperfections can have negative consequences, maybe not straight away but certainly after some time, when you suddenly find that an electrical joint has developed a high junction resistance, or a component pin has come loose owing to mechanical stress or thermal effects.

Arguably, it makes sense for experienced engineers as well as for beginners to cast a critical eye over their soldering work. In order to help you, and supply some additional information, this month's instalment of Electronics On-Line concentrates on soldering courses on the Internet. In the process of searching for relevant information, we came across some quite useful information we'd like to share with you.

For a short, no-frills approach to the sub-



ject it is best to start at the **B2Mod-Kits** web pages [1], where ten pictures and some explanatory text describe the skill called soldering.

A more extensive story we would certainly recommend is the 'Online

Soldering School' published at Benchmarks, **The Electronics Technician's Web page** [2]. The process of soldering is examined in a number of steps, from the size of the soldering iron right up to desoldering com-

**B<sup>2</sup>MODKITS.COM**

WELCOME  
Contact B2ModKits

PRODUCTS  
VU-Meter KIT  
Bright LEDs & Clips  
PC Power Splitter

ORDERING  
Payment Methods  
Shipping  
View Cart / Checkout  
Powered by DigiKey

RESOURCES  
Soldering Techniques  
Tools  
Assembly Instructions:  
- VU-Meter KIT

GALLERY  
- Reviews, Case  
Gallery

PRESS  
- Reviews, articles,  
& testimonials


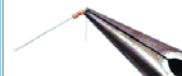


SITE MAP

### SOLDERING TECHNIQUES

First things first... The kits provide all the parts, it's your job to solder. Any damage or harm done to the kit or to yourself or your computer, as a result of poor or unsafe soldering techniques is your own responsibility. With that said, I've outlined some techniques to use when assembling and soldering on circuit boards.

#### BENDING THE LEADS

Your smallest components, the resistors and diodes, will need to have their wire leads bent, in order for you to feed them through the holes on the circuit board. Ideally, you want to have a clean looking circuit, and have your components as close to the surface of the circuit board as you can be.

	The best way to bend the leads is to take your pliers, and grip the wire a short distance from the side of the component.
	Then, take your finger, and push on the component to bend the wire to about a 90-degree angle, this will produce a nice clean bend.
	Doing the same thing to the other end will produce a nice clean looking bend, and can now be easily fed through the holes in the circuit board.
	When you put the component on the circuit board, you want to get it as close to the board as possible. If there is a lot of extra wire above the surface of the board, there is a better chance of it getting bent, damaged, or broken off.

the name **Der Kleine Lötkurs** parts 1-4 [4]. Even if you can not read German, the photographs alone are well worth visiting the site. The presentation is entirely practice-oriented and clearly laid out.

At [5] we found another website with a soldering course.

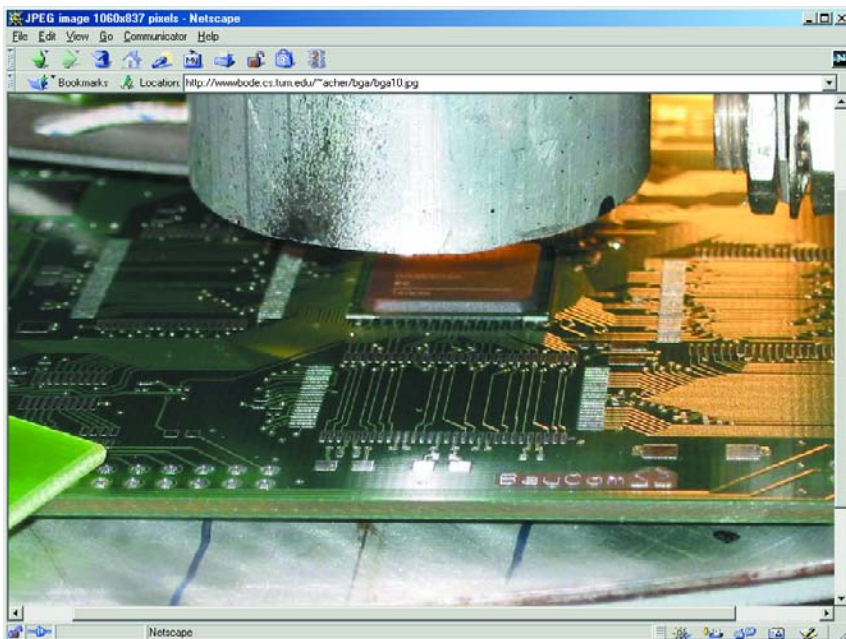
Those of you with a deeper interest in the subject or in need of additional information should have a look at **The Educational Encyclopedia** web pages [6]. Here you find a series of links to websites covering soldering and soldering techniques. The same website also offers an extensive collection of links to other subjects in the wide field of electronics.

Advanced techniques such as SMD soldering are also covered in courses. As you probably know, SMDs are miniature components without wires, requiring special soldering methods and materials to secure them on a printed circuit board. At **Soldering SMD** [7] the basic principles of SMD soldering are explained in a manner that's easy to follow.

Those of you with some experience in soldering SMDs may proceed to the next challenge: unsoldering SMDs and mounting new ones on existing PCBs. Rato Electronics provide a richly illustrated set of guidelines at the excellent **How To: SMD Rework** [8] web pages.

Finally, for Master Class level soldering, we should point you to the website published by **Georg Archer** [9] where you'll find out how to solder your own BGAs (ball grid arrays). The contacts on these ICs are no more than little spheres at the underside of the case that can not be accessed with the tip of a soldering iron. Using some tools and a lot of ingenuity Georg shows you how to solder BGAs 'by hand' despite everyone else telling you it's impossible!

(035074-1)



ponents and cleaning the tip of your iron.

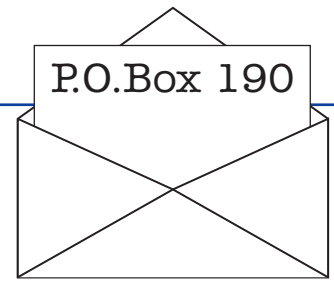
A more general and informative article about soldering may be found at **Electronics Tips: Better Soldering** [3]. Here you find a reprint of information originally published by Coopertools, the manufacturer of the well-known Weller soldering irons and stations. The story supplies details on the different lead/tin ratios used for soldering tin (with the resultant melting points) as well as on the operation of Weller soldering irons.

An excellent step-by-step course in soldering is available at the German-language Elexs website under

## Internet Addresses

- [1] B2ModKits.com: [www.b2modkits.com/Resources/Soldering/](http://www.b2modkits.com/Resources/Soldering/)
- [2] Benchmarks: [www.mts.net/~rmcgill/](http://www.mts.net/~rmcgill/)
- [3] Electronics Tips: Soldering Techniques: [www.elexp.com/t\\_solder.htm](http://www.elexp.com/t_solder.htm)
- [4] Elexs: Der kleine Lötkurs: [www.elexs.de/loet1.htm](http://www.elexs.de/loet1.htm)
- [5] A primer on hand soldering electrical connections: <http://et.nmsu.edu/~etti/fall97/electronics/solder.html>
- [6] The educational encyclopedia, soldering techniques, assembly process: <http://users.pandora.be/educyclopedia/electronics/soldering.htm>
- [7] Soldering SMD: [www.smdin.com/solderingsmds.html](http://www.smdin.com/solderingsmds.html)
- [8] Rato Elektronica NV - SMD Rework: [www.rato.be/nl/smd\\_rework.htm](http://www.rato.be/nl/smd_rework.htm)
- [9] DIY BGA Soldering: <http://www.bode.cs.tum.edu/~acher/bga/index.html>

We can only answer questions or remarks of general interest to our readers, concerning projects not older than two years and published in *Elektor Electronics*. In view of the amount of post and email received, it is not possible to answer all correspondence, and we are unable to respond to individual wishes and requests for modifications to, or additional information about, *Elektor Electronics* projects.



## Stepper Motors

Dear Sir — I am not much up on electronics, I can just read some circuits, yet when I tried to build a project in your magazine it really worked and I was rather pleased. Some mags are above my head and expect you to know what they are talking about and fill in the gaps yourself. Now there is another project I will have a go at, DC-Operated Stepper Motors (October 2003, Ed.). I have a small cod camera on my fish pond and rigged up some motors, belts and gears to pan and tilt, and looks the size of a beach bucket.

Now the parts, as you say, I can get from the ads in your mag, but the stepper motors are a bit hard to find, I might be looking in the wrong place, I spent nearly all day, Sunday, looking for stepper motors and found a lot, but the price they ask, it would be cheaper to buy new floppy drives and strip out the motor. It would have been nice if you had included, as you do for most, a www address or possible supplier like Bull Electronics who now has no motors, so could someone point me in the right direction? While on the net I found a few demos of circuit design and simulation, two I paid for before I found the demos, but found they were above my head. Not only that, the components they use were obsolete, as well as impossible to find in Maplin's catalogue. I like Electronics, Mechanics and computing principles, but then, they don't let you know the value or component number and you can only change very few parts. Now I would like a programme that I could put in your circuit and test it out, I know you test all circuits that you print, but I would like to play and change things to see what happens. A toy is nice, but more fun if it comes apart and back together and I think I would learn more if I knew what effect they have on a circuit. I probably could not afford what you use, but maybe you might know of something

that would do for me. I like your magazine because the circuits you show are very clear and even I with little knowledge of components can understand them, well, not all of them yet, but I will.

### Peter (by email)

Virtually all the stepper motors used in conjunction with our published projects have been obtained from scrapped floppy disks drives and old matrix printers, picked up from car boot sales, radio amateur rallies and computer flea markets.

We rarely state suppliers' addresses because mentioning one supplier is reason for a dozen others to complain they are not listed. In the case of stepper motors, we are simply not aware of suppliers of new motors as they will be very expensive and difficult to obtain as one-offs.

The November 2003 issue contains the first part of an in-depth article on Stepper Motors so you have something to look forward to.

## Simple Infrared Light Barrier

Dear Sir — I don't know if I will be able to explain it properly, but I will try to tell you what kind of tests I have done on the Simple Infrared Light Barrier (July/August 2002, Ed.). I built the transmission circuit, stage by stage, and I proved that it worked properly, sending the correct modulated signal to the reception circuit. I also built the reception circuit, and I could see that it did not work, because the output was always at 0 V level. I thought that it might be a construction problem, so I built it twice again, but the results were the same. I analysed the schematic circuit, as well as the specification of the IR receiver and the TLC555C, and I saw that the TLC555 couldn't work because with only R3 between pins 2 and 6 and ground, the voltage on THRESHOLD and TRIGGER prevents oscillation (also because pin 7 -DIS-

CHARGE- is needed for discharging the capacitor that the circuit should have).

I saw that the only way it could work was making a reset periodically to make the output oscillate, but the receiver must receive an oscillating signal to reset the circuit (I mean that the reception circuit can only oscillate when it receives an oscillating IR signal, so the tweeter produces sound when the beam is not interrupted (this is, always) because when the SFH5110 does not receive any signal, its output is at 5 V, so diode D1 does not discharge capacitor C2 and pin 4 of the 555 (RESET) stays at 5 V level. Because of the above I think the circuit doesn't work as described (of course it may be a failure of the Spanish publication) and I write to you to ask if I could see a picture of the correct circuit, so the alarm sounds when the beam is interrupted. Thank you very much for your help.

### Manuel Requena, Investigation engineer, CITEF

Wow, what a story! The answer is simple: the receiving IC TLC555 is not required to oscil-

late. You seem to have overlooked that a DC (or 'active') piezo buzzer should be used (see + and - symbols on Bz1).

## Projects cases

Hi there — this may be a stupid question, but where do you get your project-cases manufactured, and how do you go about getting them made? Every featured project has its' own case, specially designed to fit... I'm trying to find how to get such a case made, how much it costs, what the process is etc., and even Google (yes, it's true!) is coming up with no useful links :- (. Perhaps a by-story in a future issue? Or you could just tell me / print it on the letters page <grin>

### Simon (by email)

We do not get our project cases manufactured as you write. The cases used for our projects are off the shelf types manufactured by companies like OKW, Bopla, Retex, Pactec, Hammond, etc. selected from their catalogues. Where applicable the type numbers and case manufacturer name are given in the parts list published with the project.

## CORRECTIONS & UPDATES

### Valve Preamplifier (1)

September 2003, p. 25, 020383-1

In the Components List for the Power Supply board, the lead pitch of capacitor C13 should be amended to read: 15 mm.

### Real RS232 for Laptop PCs

December 2002, p. 66, 014099-1

Please note that C4 is shown with the wrong polarity in the circuit diagram. On the printed circuit board overlay, the same goes for C1, C2, C3 and the supply connections.

### DDS RF Signal Generator

October 2003, p. 14, 020299-1

In some cases, the display reading does not change when the rotary encoder is turned quickly. The problem is solved by connecting a 10-nF capacitor between pin 13 of the microcontroller and ground.

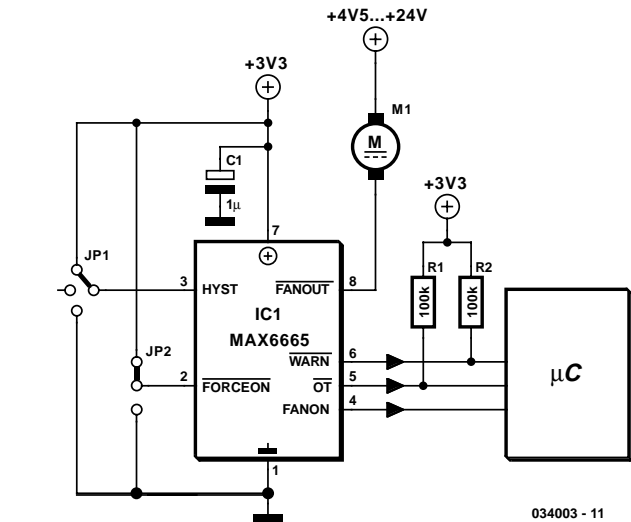
# Fan Controller using just Two Components

G. Kleine

The Maxim MAX 6665 ([www.maxim-ic.com](http://www.maxim-ic.com)) provides a complete temperature-dependent fan controller. It can switch fans operating at voltages of up to 24 V and currents of up to 250 mA. The IC is available from the manufacturer in versions with pre-set threshold temperatures between +40 °C (MAX6665 ASA 40) and +70 °C (MAX6665 ASA 70). The device's hysteresis can be set by the user via the HYST input, which can be connected to +3.3 V, connected to ground, or left open. The following table shows the hysteresis values available:

HYST	Hysteresis
open	1 °C
ground	4 °C
+3.3 V	8 °C

The other pins of the SO8 package are the  $\overline{\text{FORCEON}}$  input and the status outputs  $\overline{\text{WARN}}$ ,  $\overline{\text{OT}}$  and FANON. The test input  $\overline{\text{FORCEON}}$  allows the fan to be run even below the threshold temperature. The open-drain output  $\overline{\text{WARN}}$  goes low when the temperature rises more than 15 °C above the threshold temperature, while the open-drain output  $\overline{\text{OT}}$  indicates when the



temperature is more than 30 °C above the threshold. The push-pull output FANON can be used to indicate to a connected microcontroller that the fan is turned on.

## Replacement for Standard LCDs

P. Goossens

Many circuits published in *Elektor Electronics* use LC displays. These displays are usually only intended to show text. Such displays are much less expensive than LCDs that can display complex graphic images.

The legibility of most LC displays is without question good with normal lighting and proper contrast adjustment. Unfortunately, it degrades with reduced ambient light. Some LCDs have a 'backlight', which is a light source located behind the actual screen. When a backlight is used, the entire screen lights up, except where pixels are to be made visible. These locations are dark.

The presentation of the display can be improved by having the letters light up while the rest of the screen remains dark. This could be achieved using software, but due to the way the displays are made, it's not possible to darken the entire screen. There is a type of display that does not have this problem, which is called a 'VFD'. Such displays show bright pixels against a dark background, instead of making the pixels dark. However, this type of display has the disadvantage that it requires a high voltage to illuminate the pixels.

Fortunately, the Japanese firm Noritake has recently launched a series of VFD modules (the CU series) that is compatible with standard LC displays. These displays have converters to generate the high voltage needed to illuminate the pixels, and the programming interface and connector are exactly the same as those of standard LCD modules. This means that in any project that uses a standard LCD, the display can be replaced by a CU-series VFD from this manufacturer.

Of course, you must take into account that these modules draw more current than standard LCD modules. This is hardly surprising, since they generate their own light. For a standard display with  $16 \times 2$  characters, the maximum current consumption is approximately 150 mA, while the current consumption of a  $4 \times 40$  type can be as much as 550 mA!

If you are interested in these new displays, you might want to take a look at the manufacturer's website at

[www.noritake-itron.com](http://www.noritake-itron.com).

These displays may not be readily available, but you can certainly enquire at your local electronics shop. In Europe, Noritake has offices in the UK and Germany.



# Secure On/Off Pushbutton



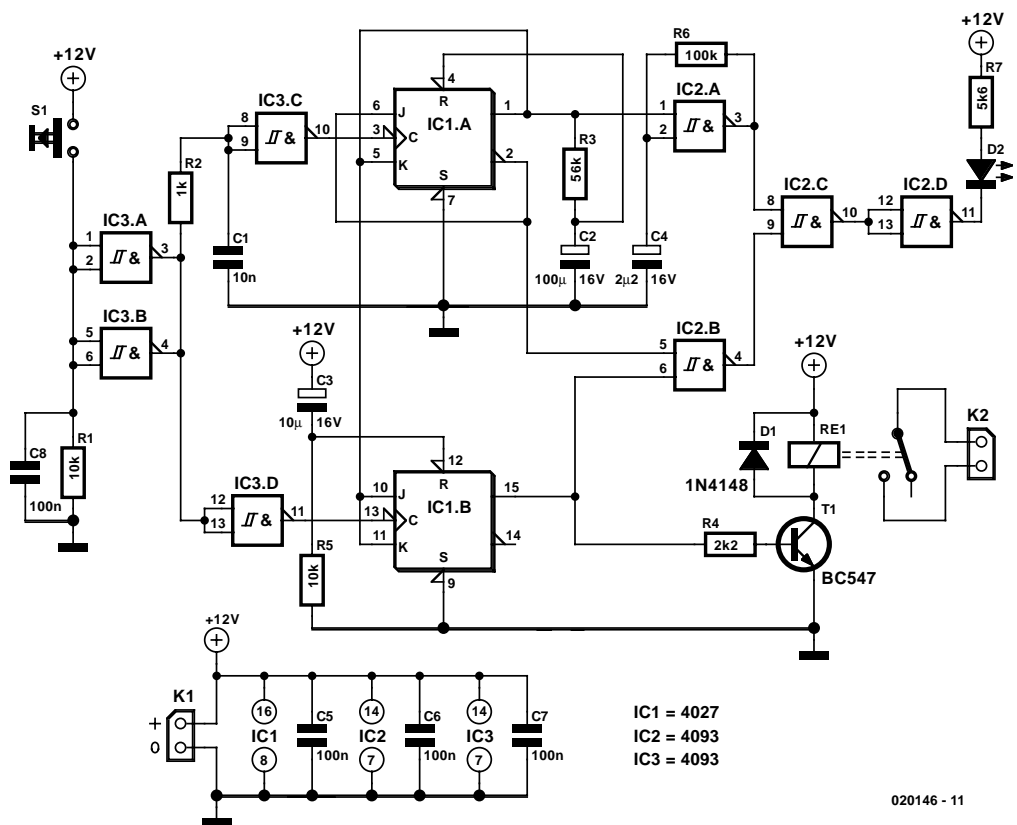
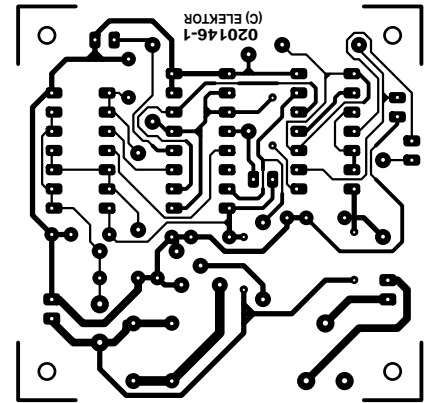
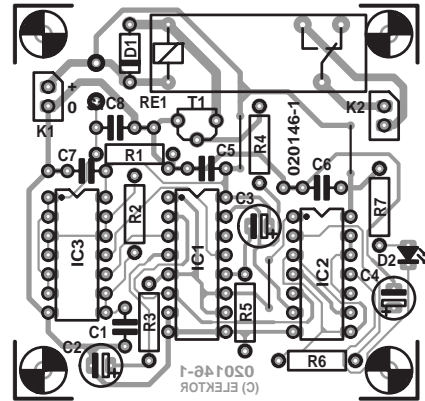
L. Libertin

The author used to employ a switching central that allows mains appliances among his PC equipment to be individually switched on and off by pressing pushbuttons. One of these pushbuttons switches the PC on and off. Good economics, but not perfect, because awkward situations can arise when a wrong pushbutton is pressed, causing the PC to be switched off instead of, say, powering up the printer.

A solution was found in the use of relay which is switched on and off only if the relevant pushbutton is pressed twice within a certain period. The first press causes a LED inside the pushbutton to flash for about 7 seconds. If you press again within this period, the relay is energised and the LED lights permanently. The switch-off procedure is identical. In this way, the switch is given a level of security.

After the first press on S1, IC1.A is switched on and the resulting state is indicated by D2 starting to flash in the rhythm defined by oscillator IC2.A. The flash rate is determined by components R6 and C4. At the same time, the second flip-flop IC1.B is enabled via its J/K inputs. Capacitor C2 is slowly charged via R3. Once the switching threshold is reached, flip-flop IC1.A is reset again.

Within this period, the pushbutton has to be pressed again. IC1.B, now operating as a data flip-flop (toggling on the positive pulse edge), toggles



- IC1 = 4027
- IC2 = 4093
- IC3 = 4093

**COMPONENTS LIST****Resistors:**

R1, R5 = 10k $\Omega$   
R2 = 1k $\Omega$   
R3 = 56k $\Omega$   
R4 = 2k $\Omega$   
R6 = 100k $\Omega$   
R7 = 5k $\Omega$

**Capacitors:**

C1 = 10nF 5mm lead pitch  
C2 = 1000 $\mu$ F 16V radial  
C2 = 100 $\mu$ F 16V radial  
C2 = 2 $\mu$ F 16V radial  
C5-C8 = 100nF

**Semiconductors:**

D1 = 1N4148  
D2 = LED, red, low current

T1 = BC547  
IC1 = 4027  
IC2, IC3 = 4093

**Miscellaneous:**

K1, K2 = solder pin  
Rel = relay, 12V (e.g. Siemens V23057-12V)  
S1 = pushbutton, 1 make contact

and causes the relay to be energised. Also, IC2.B then causes D2 to light constantly (or remain off when the relay is switched off). Capacitor C3 ensures that the relay contact can not close of its own should the mains voltage disappear.

The circuit may be built on a PCB of which the layout is shown here. As usual the artwork file may be obtained from the Free Downloads section of the Publishers' website.

# 2708 Replacement

K. Walraven

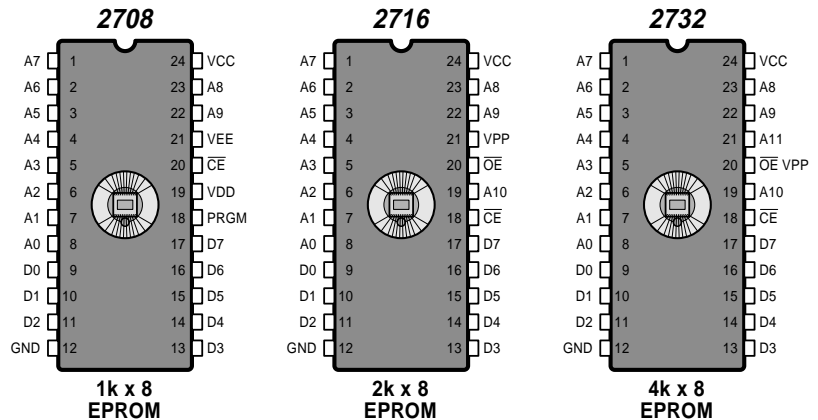
At our editorial offices we are surprised (and delighted) how often we are being asked about relatively ancient circuits. In not a few cases, these circuits contain components that are currently no longer available, such as, for example, the 1 K size EPROM 2708, which required three(!) power supply voltages. The 2716 and 2732 are parts that are still reasonably easily obtained and programmed, and they also have 24 pins. These parts require only a single 5 V power supply.

Because the two additional power supplies of +12 V and -5 V are no longer required, these two pins have become available on the 2732 and 2716. But the 2716, being 2 K in size, needs an additional address line, of course, and the 2732 (4 K) needs two more. These extra address lines are simply connected to ground permanently, so that both the 2716 and 2732 behave like a 1 K EPROM, just like the 2708.

Pin 19 (VDD on the 2708) became address line A10 on the 2716 and 2732. This line is connected straight to ground. Pin 21 (VBB on the 2708) became the programming pin on the 2716. In order to be able to read the EPROM, this pin must be connected to the +5 V power supply. On the 2732, this pin became address line A11, and therefore has to be connected to ground.

Unfortunately there have been additional changes to pins 20 ( $\overline{CE}$ , chip enable) and 18 (programming voltage). Pin 18 on the 2708 is the programming voltage but is the  $\overline{CE}$  (chip-enable) on the 2716 and 2732. Pin 20 used to be  $\overline{CE}$  on the 2708 but is  $\overline{OE}$  (output enable) on the 2716 and 2732. The 2708 did not have an  $\overline{OE}$  pin.

The simplest solution is as follows: continue to use pin 20 to enable the data output from the chip. This is possible because



034022 - 11

$\overline{CE}$  for the 2708 is now used as  $\overline{OE}$  for the 2716 and 2732, provided pin 18 ( $\overline{CE}$ ) is permanently connected to ground. This way, the chip is continuously selected, but the data only appears on the outputs when  $\overline{OE}$  becomes active. The only disadvantage is that there is no reduction in current consumption. We can imagine that you may have become a little confused after this long-winded story. Hence a set of foolproof instructions:

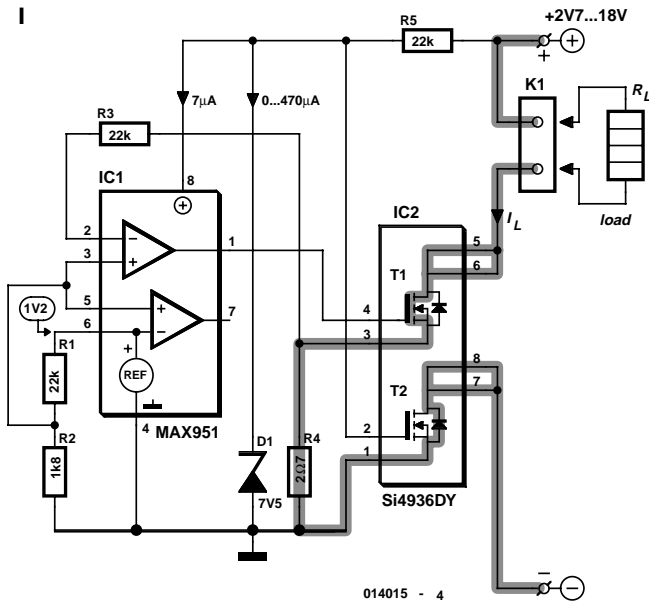
To replace a 2708 with a 2716:

1. Break the track to pin 18 and connect pin 18 to ground.
2. Break the track to pin 19 and connect pin 19 to ground.
3. Keep the connection to pin 20.
4. Break the track to pin 21 and connect pin 21 to +5 V.

To replace a 2708 with a 2732:

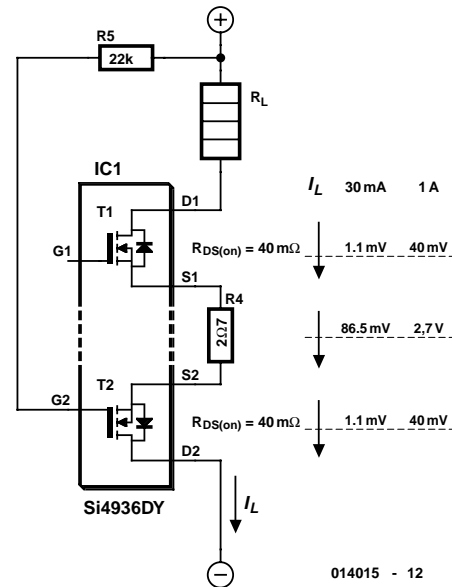
1. Break the track to pin 18 and connect pin 18 to ground.
2. Break the track to pin 19 and connect pin 19 to ground.
3. Keep the connection to pin 20.
4. Break the track to pin 21 and connect pin 21 to ground.

# Constant Current Sink



014015 - 4

2



014015 - 12

## K. Thiesler

The precision current sink shown in **Figure 1**, which uses a Si4936 DY dual MOSFET (IC1) as a controlled load resistance, is very energy-efficient and thus particularly suitable for use in battery-powered equipment. The voltage drop across current-sense resistor R4 should be approximately 85 mV. IC2 and R4 form the actual current sink, which has a very low dropout voltage. The supply voltage can be reduced to as low as the operating voltage of the load. If the supply voltage is greater than this, transistor T1 in IC2 regulates the supply current. Each of the two MOSFETs T1 and T2 can switch 4.6 A. IC2 is only available in an SO-8 SMD package. **Figure 2** shows the voltages across T1, T2 and R4 for various currents.

IC1, which is a MAX951, includes a 1.200-V reference potential source that is brought out to pin 6. R1 and R2 determine the voltage present at the non-inverting input of opamp IC1 (approximately 90 mV). The sense voltage across R4 is applied to the inverting input, where it is compared with the voltage on pin 3. T2 is only included in the circuit to provide reverse-polarity protection. The maximum allowable voltage between the source and gate also determines the maximum allowable supply voltage for this circuit.

The values of resistors R4 and R2 must be adjusted to match the desired constant current value. The circuit is primarily intended to be used with small currents. If the load current is greater than 200 mA, the value of shunt resistor R4 should be reduced.

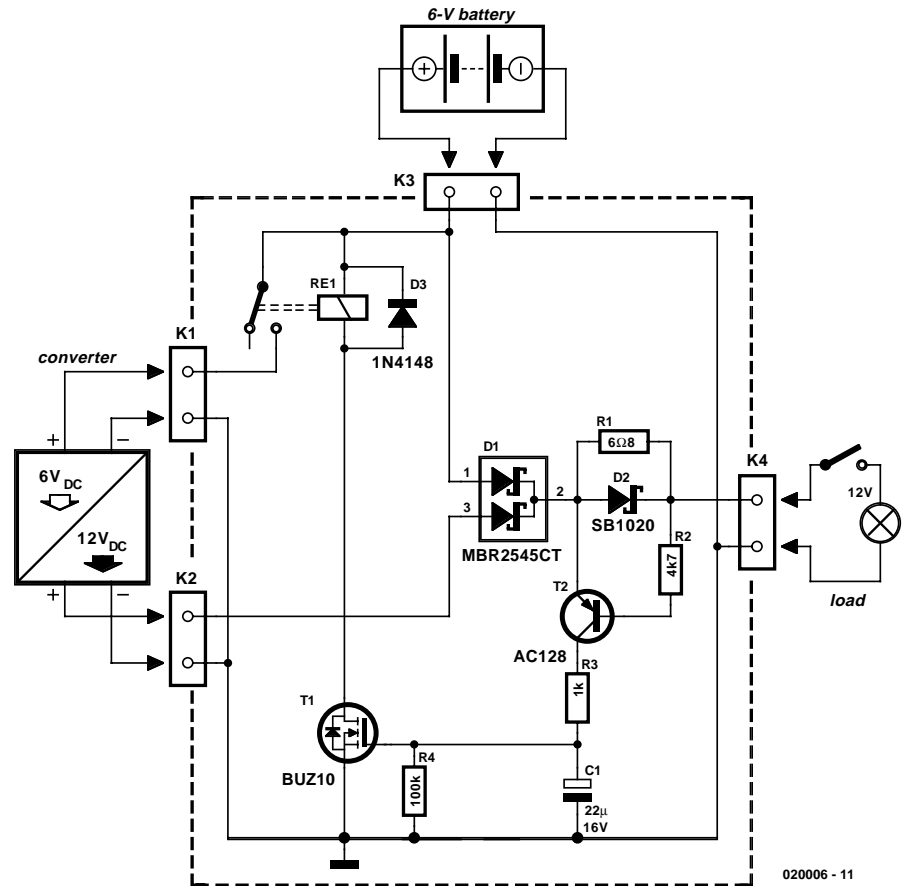
# Automatic Switch for Voltage Converters

C. Wolff

New applications for DC voltage converters, such as the 'workhorse' LT1070, arise every day. These converters can be adapted to nearly every imaginable ratio of input and output voltages. However, all of these circuits and devices have the same shortcoming, which is that they lack an on/off switch. Especially when they are used as a source of 6-V / 12-V power for a car radio, this is highly impractical. The circuit described here adds automatic load detection to the converter.

For use in a car, the additional circuitry must be small and fit into a compact enclosure together with the converter. Since the battery voltage and ambient temperature vary over wide ranges, a simple form of load detection must be used. Besides this, the voltage drop across the load sensing circuitry must naturally be as small as possible. This can be achieved by using 'ultra-modern' SiGe technology.

The 6 V from the battery and the 12 V from the converter are combined in the MB R2545 dual diode. Consequently, a voltage of at least 6 V is always applied to the radio (for memory retention). If the radio is switched on, it draws a current from the 6-V battery, which may be around 100 mA. This current produces a voltage across R1. If this voltage is 75 mV or greater, the AC128 germanium transistor starts conducting and charges electrolytic capacitor C1, which is connected to the gate of the BUZ10. The MOSFET energises RE1 and thus connects the supply voltage to the converter. As a result, 12-V power is connected to the radio. The resulting increased current causes the voltage drop across R1 to increase, which is undesirable, so a 10-A Schottky diode is connected in parallel. The total voltage drop is thus approximately 0.6 V. The RC network connected to the BUZ10 ensures that the transistor always remains switched on for at least several seconds, to prevent the circuit from 'chattering' with varying current consumption. If the load is switched off, the AC128 cuts off, the electrolytic capacitor discharges and the relay again disconnects the voltage converter. The residual current consumption is so small that the circuit can also be connected ahead of the ignition switch. The Schottky diodes need only be rated for the necessary volt-



ages and currents, and above all, they should have the lowest possible saturation voltage. The exact type is not critical. Two separate diodes can also be used. A small heat sink for the MBR diode won't hurt, but this is normally not essential. Practically any type of PNP germanium transistor that is still available or on hand can be used (AC125, AC126 and AC128 work perfectly). It may be necessary to modify the value of R1. In combination with the germanium transistor, R1 determines which level of current will be ignored (for memory retention) and which level of current will cause the converter to be switched on. With the component values shown in Figure 1, this level is between 10 mA and 25 mA. It is recommended to measure the quiescent current (at 6 V) and switch-on current of the load and then simulate the switching process using dummy load resistors. When selecting the 6-V relay, ensure that its contacts have an adequate current rating. The actual value can be significantly greater than the nominal output current. With a load of 5 A at 12 V and a converter efficiency of 70 percent, the current through the relay contacts rises to 14.3 A.

# Infrared Proximity Detector/Alarm

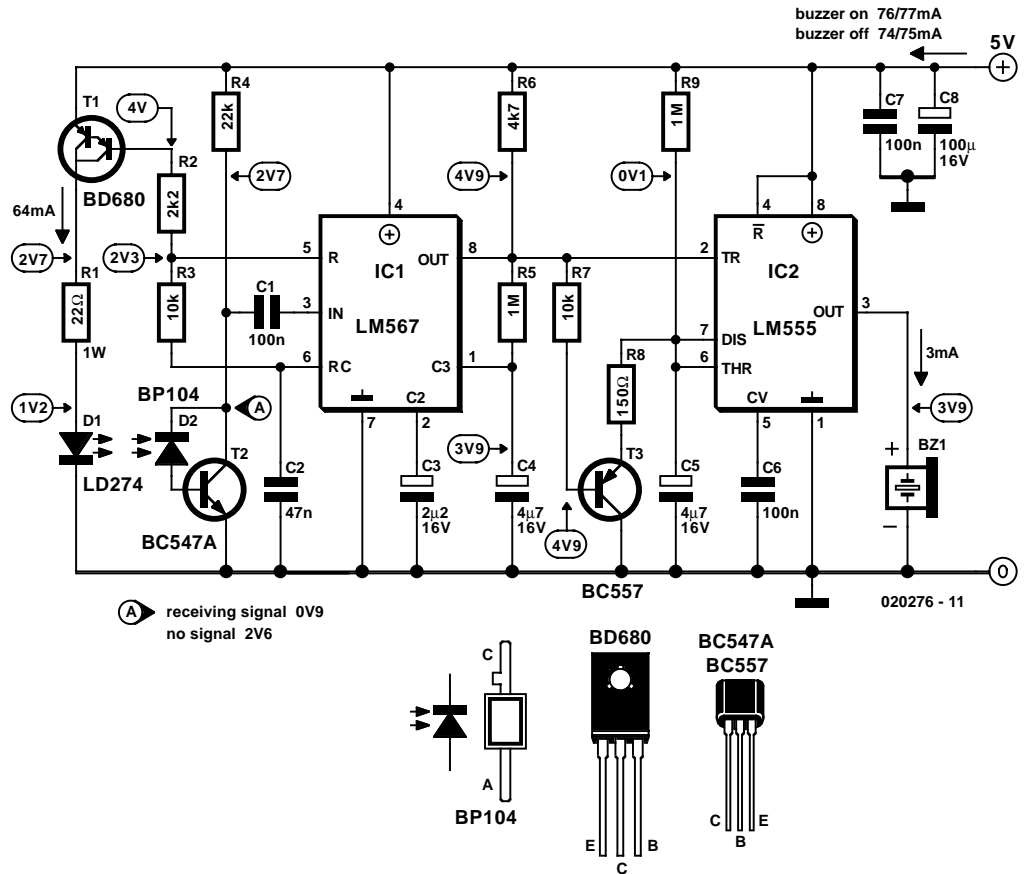
T. K. Hareendran

This circuit can be built from readily available low-cost components, some of which may even be hiding in your junkbox!

The indicated value of 22 Ω for resistor R1 causes an average current of about 65 mA through infrared emitter D1. Because the IRED is pulsed at a duty factor of about 50% through the action of T1 and IC1, a peak current of 128 mA flows during every half cycle. This may seem a lot but in fact is well within the safe specification of the LD274. The LM567 PLL IC is configured to supply a switching frequency of about 20 kHz.

When the infrared beam emitted by D1 is reflected by a nearby object, IC1, through receiver diode D2 and transistor T2, receives the recovered 20 kHz signal at its input, pin 3. Because the '567 PLL is then locked, the IC output (pin 8), drops low, triggering the 555 chip in monostable mode (IC2) and so causing acoustic actuator Bz1 to sound. The monostable remains on as long as the reflected signal is being received. Because of the presence of T3, capacitor C5 is allowed to charge only when no signal is being received. In that condition, the 555 is turned off automatically after a time determined by R9-C5. Using the component values shown, this will be about 5 seconds.

Obviously D1 and D2 should be mounted such that the latter can only pick up reflected infrared light. The choice of the two infrared components used in this circuit will be uncritical but



they must be 'band' compatible, i.e., generate (D1) and respond to (D2) the same wavelength. The operating point of the receiver input circuit is rather dependent on ambient daylight levels and the value of R4 may need to be adjusted a little to ensure a voltage of between 1.5 V and 4 V on the collector of T1 when no signal is being received.

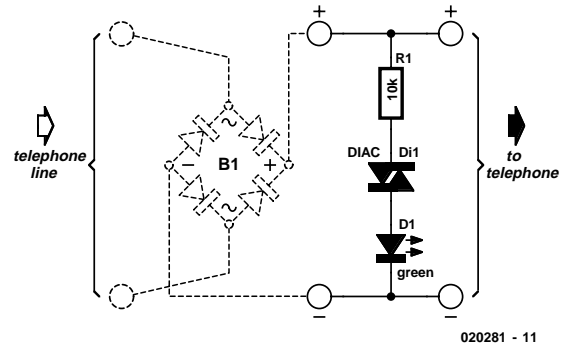
Some dc buzzers cause a lot of back-emf so it may be necessary to insert a diode in series with the output of IC1. If necessary, this diode should preferably be a Schottky type because of the inherent low voltage drop of about 0.4 V as opposed to 0.65 V for a typical small-signal silicon diode.

# Telephone Free Indicator

R. J. Gorkhali

Depending on local regulations and the telephone company you happen to be connected to, the voltage on a free telephone line can be anything between 42 and 60 volts. As it happens, that's sufficient to make a diac conduct and act like a kind of zener diode maintaining a voltage of 38 V or so. The current required for this action causes the green high-efficiency LED in the circuit to light. Line voltages higher than about 50 V may require R1 to be changed from 10 kΩ to a slightly higher value. When the receiver is lifted, the line voltage drops to less than 15 V (typically 12 V) causing the diac to block and the LED to go out.

The circuit diagram indicates + and - with the phone lines. However, in a number of countries the line polarity is reversed when a call is established. To make sure the circuit can still function under these circumstances, a bridge rectifier may be added as indicated by the dashed outlines. The bridge will



make the circuit independent of any polarity changes on the phone line and may consist of four discrete diodes, say, 1N4002's or similar.

Finally, note that this circuit is not BAPT approved for connection to the public switched telephone network (PSTN) in the UK.

# Telephone Line Indicator

R. van Orsoy de Flines

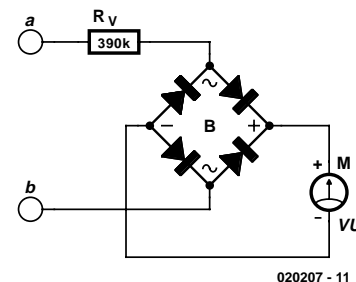
With the aid of an (old) moving coil instrument it is very little effort to make a simple voltmeter that, at a glance, indicates the status of a telephone line. Because the input impedance of this circuit is very high, there is no problem in having it permanently connected to the line, since it only draws a tiny amount of current.

The schematic shows that the circuit consists of no more than a series resistor, a bridge rectifier and a moving coil meter. The value of the resistor depends on the sensitivity of the moving coil meter. In his prototypes, the author used old VU meters that require  $250\ \mu\text{A}$  for full-scale deflection. A resistor value of  $390\ \text{k}\Omega$  appeared to be optimal for these meters. For a  $100\text{-}\mu\text{A}$ -instrument, this resistor value will have to be increased to about  $680\ \text{k}\Omega$ . The starting point, when selecting a resistor value is that when the telephone is not in use, the meter should deflect about  $2/3$ rd of full scale.

The amount of meter deflection indicates the three different states of the telephone line:

1. The deflection is very small: the line is in use (voltage 5 to 12 V).
2. The deflection is  $2/3$ rd of full scale: the line is not in use (voltage typically 48 V).
3. Full-scale deflection: ring signal (60 to 90 V AC).

Because the idle voltage and certainly the ring voltage are high enough to be dangerous, it is recommended that the circuit is constructed in such a way that it presents no hazard when touched.





# AVR Dongle

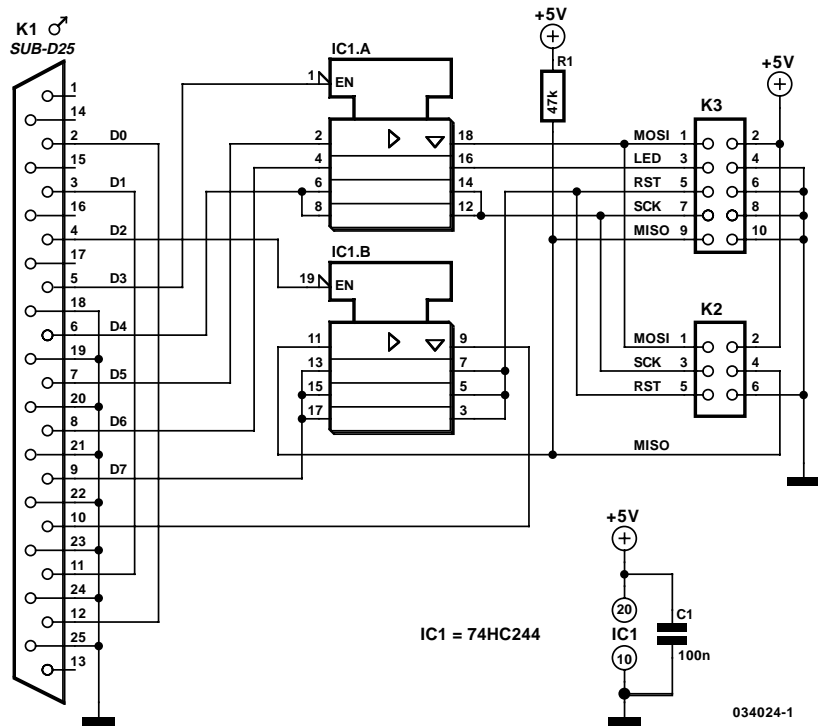
P. Goossens

This circuit is intended to program AVR controllers such as the AT90S1200 via the parallel port. The circuit is extremely simple. IC1 provides buffering for the signals that travel from the parallel port to the microcontroller and vice versa. This is essentially everything that can be said about the circuit.

The two boxheaders (K2 and K3) have the 'standard' ISP (in system programming) pinout for the AVR controllers. The manufacturer recommends these two pinouts in an attempt to create a kind of standard for the in-circuit programming of AVR-controllers. These connections can be found on many development boards for these controllers.

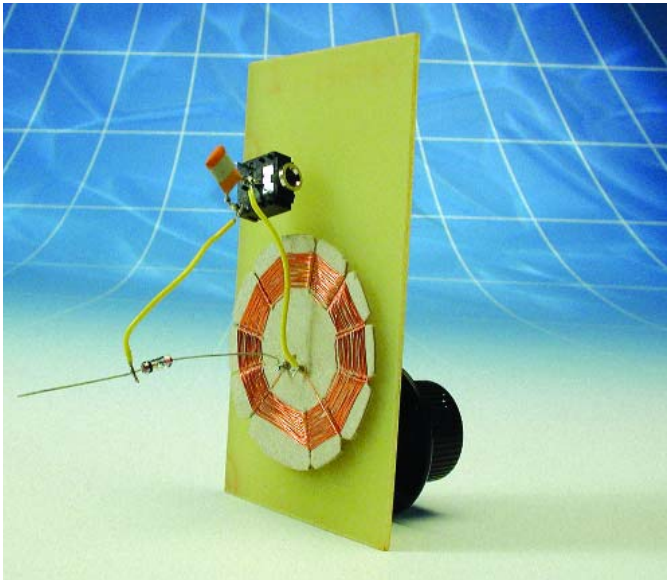
The software carries out the actual programming task. It is therefore necessary to have a program (ATMEL AVR ISP), which is available as a free download from <http://www.atmel.com>.

The construction of the circuit will have to be made on standard prototype board, since we didn't design a PCB for this circuit. This should not present any difficulties considering the small number of parts involved. We recommend that inexperienced



builders first make a copy of the circuit and cross off each connection on the schematic once it has been made on the board. This makes it easy to check afterwards whether all connections have been made or not.

# Card Radio



## G. Stabe

Among some of our modern contemporaries, 'musical' postcards evoke strong reactions of astonishment about hyper-modern microcontroller technology. However, such flat melody memories would only have elicited a weary smile from our forefathers.

As early as 1928, there are reports that radio cards with the dimensions of a regular postcard and a thickness of only a few millimetres were being made. These cards concealed a basketwork coil with a sliding tap for tuning the frequency of the received signal, a fixed capacitor and a miniscule detector device consisting of a small crystal with a 'whisker' contact. A similarly simple circuit can also be implemented using current resources. For this, you will need an interesting local medium-wave transmitter and a high-impedance headphone (1–2 k $\Omega$ ), as well as a good aerial (such as a metal downpipe or an earthed radiator). The aerial is connected to an LC resonant circuit tuned to the frequency of the local transmitter, and a diode provides the demodulation. The necessary capacitance following the diode is provided by the cable to the headphone or amplifier.

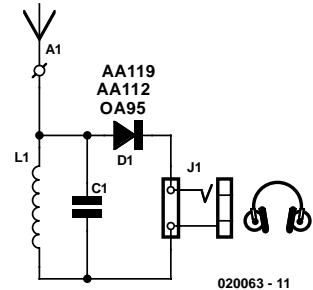
The coil can be made using a circular piece of stiff cardboard with a diameter of a couple of centimetres. Cut an odd number of slots into the cardboard disc. Then wind enamelled cop-

per wire (diameter 0.15–0.2 mm) back and forth through the slots. Forty turns will give an inductance of around 80  $\mu\text{H}$ . The coil looks like the bottom of a reed basket, which explains its cryptic name in RF jargon.

To tune the coil to the frequency of the local transmitter and determine the required frequency of the resonant circuit, connect a dual-gang or multiple-gang variable capacitor (500–1000 pF) to the coil, with the stator sections (the fixed portion of the capacitor plates) connected in parallel. The rotor sections, which are connected to the shaft of the rotary capacitor, must without fail be connected to ground in order to prevent a 'hand effect' while tuning. Incidentally, the resonant-circuit formula cannot be used to determine the tuning capacitance, since it ignores the effect of the aerial. After the capacitor has been adjusted, estimate the value of the capacitance (or even better, measure it), dig out a suitable fixed capacitor from your parts box and solder it to the coil at the centre of the cardboard disc, along with a general-purpose germanium diode (AA119, AA112, OA95, etc.). Secure the capacitor and diode with glue. For terminals, you can use 4-mm tubular rivets for miniature plugs, as shown in the photo.

A suitable 'enclosure' can be made from 'customer discount' cards in credit-card format (you probably already have more than you really need). Use one card as the 'circuit board' for the receiver, and cut an opening in a second card to receive the circuitry. Ideally, this card should be thick enough to fit the full height of the receiver. The cover is formed by a third card. After a final check, glue or rivet the cards together, and your card radio is finished. It's not high-end, but it has astonishingly good performance for such a simple circuit.

One final glimpse into the past: already in the 1930s, such fixed-tuned detector receivers were available in the form of 'Berlin plugs', 'Hamburg plugs', and so on, for receiving local transmitter signals in various locations.



# IrDA Interface

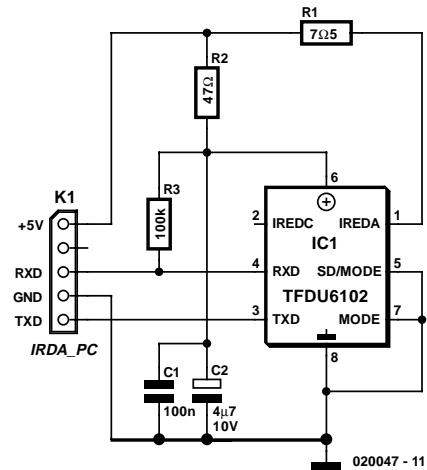
## A. Bitzer

Many modern motherboards are equipped with an infrared data interface compliant with the IrDA standard, but this interface not very often used. However, it is not difficult to build a data transmission module and connect it to the corresponding header. As can readily be seen from the schematic diagram, this doesn't exactly involve a large array of ICs. This is because transceiver ICs are available for the IrDA standard, so only a few passive components have to be added to obtain an operational circuit. The author has successfully built this circuit many times using the TFDU5102 from Vishay Semiconductors (formerly Telefunken). If this IrDA transceiver is no longer available (it has been officially discontinued), the largely pin- and function-compatible TFDU6102 can be used without any problems. This IC is faster and meets the latest IrDA specification.

The TFDU6102 low-power receiver IC supports IrDA at data rates up to 4 Mbit/s (FIR), HP-SIR, Sharp ASK, and carrier-based remote control modes up to 2 MHz. The IC contains a photodiode, an infrared emitter and CMOS control logic. The IC also has internal protection against electromagnetic immisions and emissions, so no external screening is necessary. The IC works with a supply voltage of 2.7–5.5 V, so it is suitable for use in desktop PCs, notebooks, palmtops, and PDAs. It is also used in digital still and video cameras, printers, fax machines, copiers, projectors, and many other types of equipment.

The author has designed a printed circuit board for the IrDA module that is only 20 × 20 mm square. Of course, this means that all of the components are SMD types. The TFDU6102 in the 'babyface' package is available in upright and flat versions. Here the upright version (suffix 'TR3') is used. Thanks to its small size, the assembled circuit board can easily be placed behind a drive bay cover or the like. It is connected to the motherboard by a five-way flat cable. The pin assignments for header X1 must match the mating connector on the motherboard.

After you have fitted the module, you may have to edit the BIOS settings to activate the UART for IrDA operation. These



## COMPONENTS LIST

### Resistors:

R1 = 7Ω5 (shape I210)  
R2 = 47 Ω (shape I206)  
R3 = 100 k (shape I206)

### Capacitors:

C1 = 100nF (shape I206)

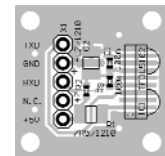
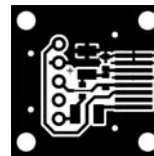
C2 = 4μF7 (shape I210)

### Semiconductors:

IC1 = TFDU6102TR3  
(Vishay) (Farnell)

### Miscellaneous:

X1 = 5-way SIL pinheader



settings enable the (Windows) operating system to boot the new device and automatically install it. You may have to briefly insert the Windows CD to modify the settings. There is an abundance of free programs on the Internet that use the IrDA interface.

# Solar Relay

W. Zeiller

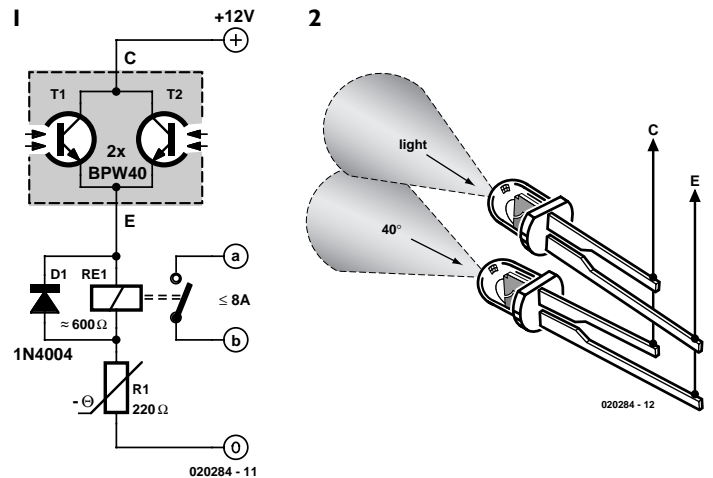
With extended periods of bright sunshine and warm weather, even relatively large storage batteries in solar-power systems can become rather warm. Consequently, a circuit is usually connected in parallel with the storage battery to either connect a high-power shunt (in order to dissipate the excess solar power in the form of heat) or switch on a ventilation fan via a power FET, whenever the voltage rises above approximately 14.4 V. However, the latter option tends to oscillate, since switching on a powerful 12-V fan motor causes the voltage to drop below 14.4 V, causing the fan to be switched off. In the absence of an external load, the battery voltage recovers quickly, the terminal voltage rises above 14.4 V again and the switching process starts once again, despite the built-in hysteresis.

A solution to this problem is provided by the circuit shown here, which switches on the fan in response to the sweltering heat produced by the solar irradiation instead of an excessively high voltage at the battery terminals. Based on experience, the risk of battery overheating is only present in the summer between 2 and 6 pm. The intensity of the sunlight falling within the viewing angle of a suitably configured 'sun probe' is especially high precisely during this interval. This is the operating principle of the solar relay.

The trick to this apparently rather simple circuit consists of using a suitable combination of components. Instead of a power FET, it employs a special 12-V relay that can handle a large load in spite of its small size. This relay must have a coil resistance of at least 600  $\Omega$ , rather than the usual value of 100–200  $\Omega$ . This requirement can be met by several Schrack Components relays (available from, among others, Conrad Electronics). Here we have used the least expensive model, a type RYII 8-A printed circuit board relay.

The light probe is connected in series with the relay. It consists of two BPW40 phototransistors wired in parallel. The type number refers to the 40-degree acceptance angle for incident light. In bright sunlight, the combined current generated by the two phototransistors is sufficient to cause the relay to engage, in this case without twitching. Every relay has a large hysteresis, so the fan connected via the a/b contacts will run for many minutes, or even until the probe no longer receives sufficient light.

The NTC thermistor connected in series performs two functions. First, it compensates for changes in the resistance of the



copper wire in the coil, which increases by approximately 4 percent for every 10 °C increase in temperature, and second, it causes the relay to drop out earlier than it otherwise would (the relay only drops out at a coil voltage of 4 V). Depending on the intended use, the 220- $\Omega$  resistance of the thermistor can be modified by connecting a 100- $\Omega$  resistor in series or a 470- $\Omega$  resistor in parallel. If the phototransistors are fastened with the axes of their incident-angle cones in parallel, the 40-degree incident angle corresponds to 2 pm with suitable solar orientation. If they are bent at a slight angle to each other, their incident angles overlap to cover a wider angle, such as 70 degrees. With the tested prototype circuit, the axes were oriented nearly parallel, and this fully met our demands.

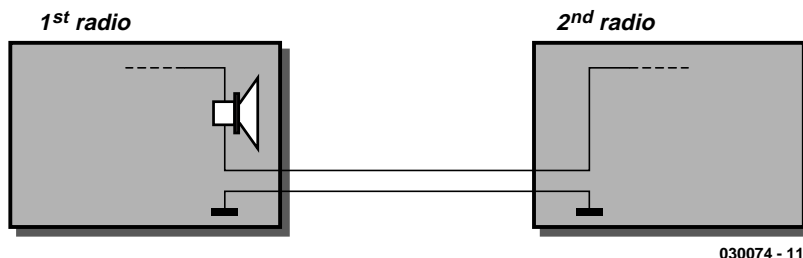
The automatic switch-off occurs quite abruptly, just like the switch-on, with no contact jitter. This behaviour is also promoted by the NTC thermistor, since its temperature coefficient is opposite to that of the 'PTC' relay coil and approximately five times as large. This yields exactly the desired effect for energising and de-energising the relay: a large relay current for engagement and a small relay current for disengagement. Building the circuit is actually straightforward, but you must pay attention to one thing. The phototransistors resemble colourless LEDs, so there is a tendency to think that their 'pinning' is the same as that of LEDs, with the long lead being positive and the short lead negative. However, with the BPW40 the situation is exactly the opposite; the short lead is the collector lead. Naturally, the back-emf diode for the relay must also be connected with the right polarity. The residual current on cloudy days and at night is negligibly small.

# Poor Man's MW/LW Wideband Noise Reduction

G. Baars

Many radio signals in the medium and long wave bands (MW/LW) but also shortwave (SW) are infested by noise of wide variety and of such levels that weak stations are virtually obliterated. The worst noise is the wideband variety which stretches across several hundred kilohertz across the band. However the very fact that the noise is wideband in nature allows it to be suppressed using a little known and inexpensive method of using a second receiver tuned just beside the frequency you want to listen to.

The principle is illustrated in the drawing. The second receiver effectively isolates the noise by adding it in anti-phase to the wanted signal. As shown the loudspeaker outputs of the two radios are connected in phase while the loudspeaker in the second radio is disconnected or removed. With the second receiver tuned a little higher or lower than the main one, the



030074 - 11

loudspeaker in the main receiver is fed with the difference between the two signals. Using the volume adjustment on the second radio, a setting can be found that should result in considerable reduction of wideband noise. If an external antenna is used, it may be connected in parallel to the receiver inputs. Ferrite rod antennas need to be turned in the same direction. Best results are obtained when two identical radios are used.

# RS232 Voltage Regulator

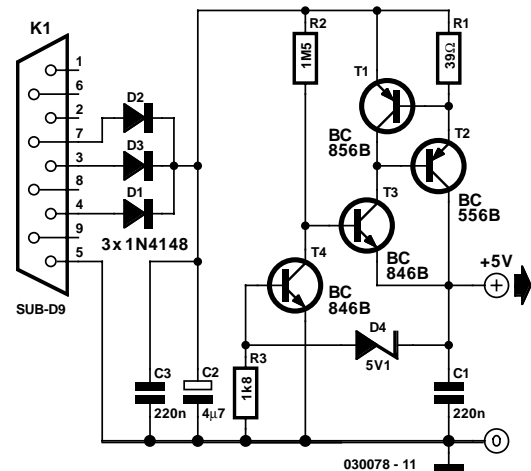
M. Müller

There are many small applications where it would be preferable to power a device directly from an RS232 (V.24) interface, avoiding a mains power supply. Most ICs require 5 V, and the interface can provide a current of around 8 mA, almost all of which would be consumed by a readily-available voltage regulator, leaving nothing for the actual circuit.

Using just four transistors we can construct a voltage regulator with current limiting which will allow us to draw more than the permitted 8 mA from an RS232 interface without damaging it. The example circuit in is configured for an output voltage of 5 V from an input voltage of at least 8 V, and a short-circuit current of 19 mA. The current drawn by the regulator itself is only 0.2 mA.

The circuit appears very simple, but it is more cunning than it looks. Few people appreciate what a handy device the transistor is. To meet the requirements for the circuit, the gains of the transistors need to be controlled carefully. Here only B-class devices are used, which have a gain of between about 220 and 280.

Diodes D1 to D3 extract the positive voltage from the serial interface. Current limiting is achieved via resistor R1 and transistor T1. As soon as the voltage across the resistor reaches 0.7 V (at 18 mA with  $R1 = 39 \Omega$ ) the transistor turns on and thus turns off the output voltage by turning off T2. The output voltage of 5 V is set by Zener diode D4. Note that the output voltage is only approximate: beware when using components which have narrow supply voltage tolerances. When the Zener diode voltage and the voltage across transistor T4 are added together, the total is 5.8 V. However, because of T3, the diode



is operating at a low current and the actual threshold for T4 is 4.9 V.

The main regulation loop is built around R2 and T2. The high value of R2 (1.5 MΩ) is important, since this limits the maximum current through T2. At the output we would like to be able to draw a maximum current of 19 mA. The base of T2 must therefore be supplied with exactly 1/220 (the gain of the transistor) of 19 mA, and likewise the current into the base of T3 should be just 1/220 of 80 μA. With an input voltage of 9 V the voltage drop across R2 will be 3.3V, and so a current of 2.2 μA will flow. Transistor T3 multiplies this current by 220 to 0.5 mA, which is also the minimum quiescent current of the circuit.

# Vehicle Interior Lighting with Switch-Off Delay

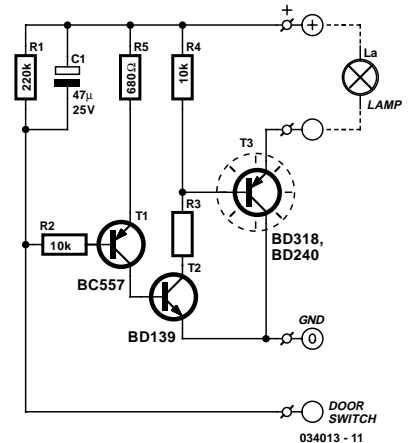
L. Libertin

Nowadays, a switch-off delay for the vehicle interior lighting is a naturally a standard feature. However, with certain models having only spartan fittings, or older-model vehicles, you're left sitting in the dark as soon as you climb in and close the door. That calls for an aftermarket accessory!

The author built this circuit using 'normal' components (with leads), but in the SMD manner, which means fitting the components on the copper side. The only holes drilled in the circuit board were the four fixing holes, and the entire assembly was firmly attached to the surface of the heatsink for power transistor T3 (the author used a finned heat sink rated at 7.2 °C/W). The heatsink is at ground potential. A value of 1 Ω was used for R3 with satisfactory operation of the darlington. The light goes on when the door is opened. After the door is

closed, it continues to illuminate the interior of the car at full brightness for around 30 seconds, after which it slowly dims. Approximately 1 minute after the door is closed, the quiescent current drops to zero.

(034013-1)



# Boolean Expression Parser

R. Sridhar

Given a Boolean expression or its minterms, the program described here prints the simplified ('reduced' or 'parsed') expression.

Anyone who's ever designed a fairly complex digital circuit will know that what looks like a tangled network of logic gates and inverters may often be simplified by writing down everything that happens in the circuit in the form of Boolean expressions and then discovering that these can often be condensed in a way that allows you to actually cut down on the number of ICs to be used! The program is also useful if you need to design logic with nothing more front of you than a large truth table describing the functionality.

The program was written to run under DOS which means that it is not particularly user friendly but then again it's free!

Boolean equations may be entered in two ways. One is to simply enter the equation, the other, to define the minterms. The first method is fairly straightforward. The use of brackets is not allowed. An inverted signal is entered by adding an apostrophe (') behind it, for example,  $A\bar{B}$  becomes  $AB'$ , and  $\bar{A}\bar{B}$  is entered as  $A'B'$ . The minus sign (-) is used to terminate the entry.

The use of minterms is practical if the Boolean equation is not available, but you do have a truth table. Using this method you mark those positions in the truth table that should use a output to become '1'. The value -1 is used to terminate the entry. Next, the program will ask you how many variables should be

used.

An example may help to explain the method. The table below has to be entered into the program by typing

```
1[Enter]3[Enter]-1[Enter]
```

and then replying '3' to the number of variables prompt (we need A, B and C).

A	B	C	OUT
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

After a valid entry the program will print the simplified Boolean expression on the screen.

The program executable code as well as the source code written in C++ may be obtained from the Free Downloads section at the publishers' website, [www.elektor-electronics.co.uk](http://www.elektor-electronics.co.uk). The file number is **020391-11**, see month of publication.



# Step-Up Booster Powers Eight White LEDs

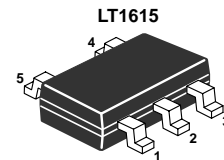
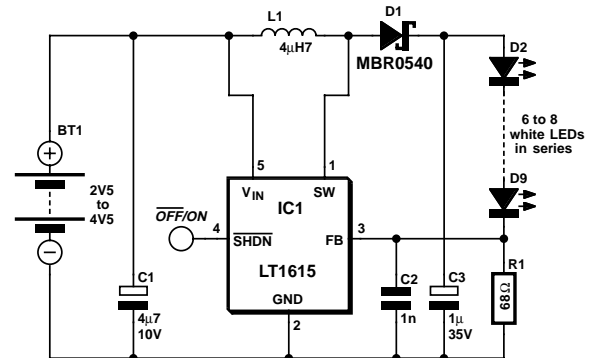
D. Prabakaran

Tiny white LEDs are capable of delivering ample white light without the fragility problems and costs associated with fluorescent backlights. They do pose a problem however in that their forward voltage can be as high as 4 V, precluding them being powered directly from a single Li-Ion cell.

Applications requiring more white LEDs or higher efficiency can use an LT1615 boost converter to drive a series connected array of LEDs. The high efficiency circuit (about 80%) shown here can provide a constant-current drive for up to eight LEDs. Driving eight white LEDs in series requires at least 29 V at the output and this is possible thanks to the internal 36-V, 350-mA switch in the LT1615. The constant-current design of the circuit guarantees a steady current through all LEDs, regardless of the forward voltage differences between them. Although this circuit was designed to operate from a single Li-Ion battery (2.5V to 4.5V), the LT1615 is also capable of operating from inputs as low as 1 V with relevant output power reductions.

The Motorola MBR0520 surface mount Schottky diode (0.5 A 20 V) is a good choice for D1 if the output voltage does not exceed 20 V. In this application however, it is better to use a diode that can withstand higher voltages like the MBR0540 (0.5 A, 40 V). Schottky diodes, with their low forward voltage drop and fast switching speed, are the best match. Many different manufacturers make equivalent parts, but make sure that the component is rated to handle at least 0.35 A. Inductor L1, a 4.7- $\mu$ H choke, is available from Murata, Sumida, Coilcraft, etc.

In order to maintain the constant off-time (0.4 ms) control scheme of the LT1615, the on-chip power switch is turned off only after the 350-mA (or 100-mA for the LT1615-1) current limit is reached. There is a 100-ns delay between the time when the current limit is reached and when the switch actually turns off. During this delay, the inductor current exceeds the current limit by a small amount. This current overshoot can be beneficial as it helps increase the amount of available output current for smaller inductor values. This will be the peak current passed by the inductor (and the diode) during normal operation. Although it is internally current-limited to 350 mA,



020349 - 11

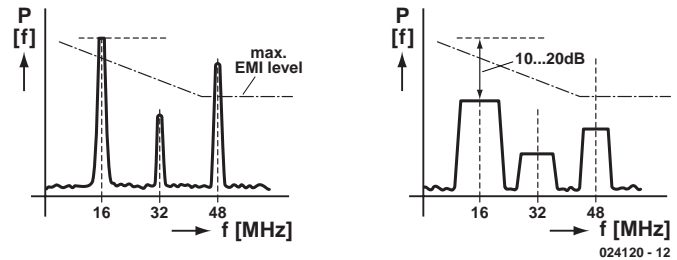
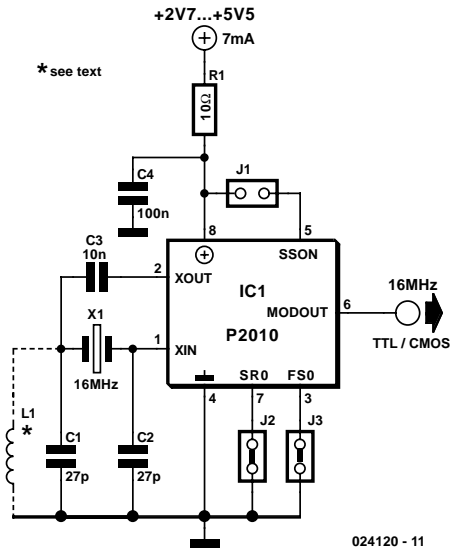
the power switch of the LT1615 can handle larger currents without problems, but the overall efficiency will suffer. Best results will be obtained when  $I_{PEAK}$  is kept well below 700 mA for the LT1615.

The LT1615 uses a constant off-time control scheme to provide high efficiencies over a wide range of output current. The LT1615 also contains circuitry to provide protection during start-up and under short-circuit conditions. When the FB pin voltage is at less than approximately 600 mV, the switch off-time is increased to 1.5 ms and the current limit is reduced to around 250 mA (i.e., 70% of its normal value). This reduces the average inductor current and helps minimize the power dissipation in the LT1615 power switch and in the external inductor L1 and diode D1.

The output current is determined by  $V_{ref}/R1$ , in this case,  $1.23V/68 = 18 \text{ mA}$ .

Further information on the LT1615 may be found in the device datasheets which may be downloaded from [www.linear-tech.com/pdf/16151fa.pdf](http://www.linear-tech.com/pdf/16151fa.pdf)

# EMC Improver



FS0	SR0	Frequency range	Spreading range
1	0	10 to 20 MHz	± 1.50 %
1	1	10 to 20 MHz	± 2.50 %
0	0	20 to 35 MHz	± 1.25 %
0	1	20 to 35 MHz	± 2.00 %

## G. Kleine

All electronic equipment these days must conform to the requirements of the electromagnetic compliance (EMC) regulations. A particular problem is the radiation of interference, for example the clock signal, from digital circuits. The problem is conventionally attacked using shielding and complex smoothing circuits.

A simpler and cheaper approach is offered by the company PulseCore in the form of a crystal oscillator IC. It distributes the energy in the interference signal over a band of frequencies, rather than concentrating it at one frequency. The energy at any one particular frequency is therefore reduced. In practice reductions in interference of 10 dB to 20 dB can be

achieved. The technique used is known as 'Spread Spectrum'. The P2010 device includes a crystal oscillator, suitable for frequencies between 10 MHz and 35 MHz, and the spread spectrum jitter circuit. ICs designed to work at higher frequencies are also available. Crystal X1 is designed for fundamental frequency oscillation; for overtone crystals coil L1 ensures that the crystal oscillates at the correct (third or fifth) harmonic. The frequency range is selected using pin FS0, while SR0 allows one of two different spreading ranges to be selected (see table). In the table a '0' indicates that Br 2 (respectively Br 3) bridged while a '1' indicates that Br 2 (respectively Br 3) is open. The device draws a current of about 7 mA and operates with 3.3 V and 5 V logic. Br 1 allows the clock jittering to be disabled for test purposes.

# Universal Clock Generator

R. Zenzinger

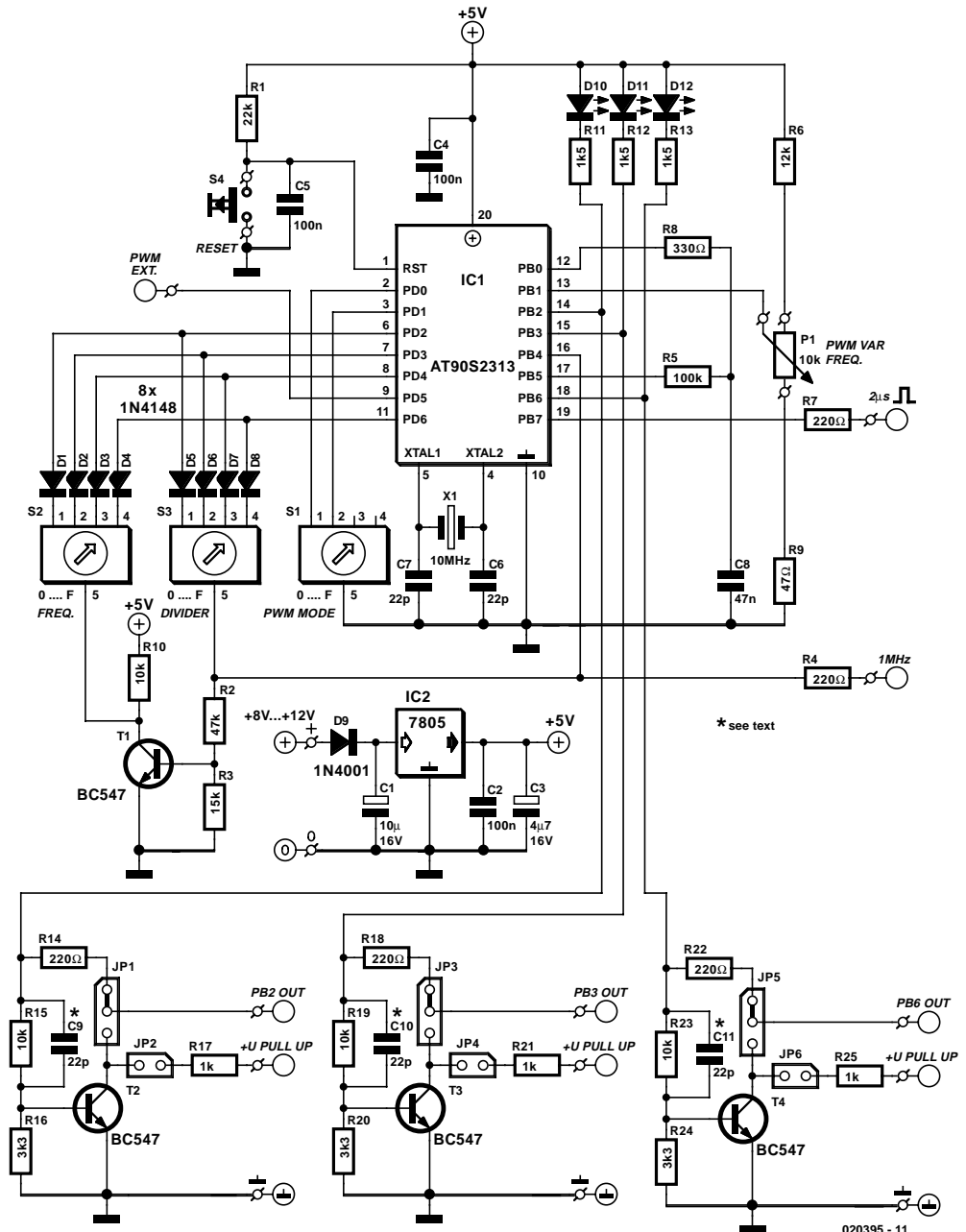
This universal clock generator is implemented using an Atmel AT90S2313 microcontroller, so it does not require very many external components. It is a versatile clock generator for use in pulse generator and timer circuits in a lab setting. It provides approximately 100 reference frequencies with 'crystal accuracy', and it can be used to implement dimmer circuits, generate arbitrary pulse waveforms for simulations, as a freely settable frequency generator and much more. Operation and adjustment are simple and easy to understand.

As can be seen, all of the microcontroller ports are fully used. Port lines PC2–PD4, PD6 and PB4 have dual functions. This is not a problem for a microcontroller as long as the software is suitably designed. In this case, two BCD switches are read using the same inputs. The switches are selected using PB4 and inverter T1, and they are isolated from each other by diodes D1–D8.

The 1-MHz clock signal is also output via PB4. During normal operation, the 1-MHz output is inactive; only when it is in the 1-MHz mode does the microcontroller enter a permanent loop, in which the other functions are anyhow not used.

Each of the outputs PB2, PB3 and PB6 is connected to a transistor stage, which can be configured using jumpers according to the intended application. The microcontroller outputs can be fed out either directly or inverted via the transistor stages. Another jumper can be used to connect a pull-up resistor, which can be connected externally to any desired voltage below the maximum specified value for the transistor. With the given component values, the output circuits are short-circuit proof for transient shorts.

For some applications, it may be necessary to alter the component values. If necessary, suitable external circuitry can also be added. Capacitors C9–C11 are only necessary if it is impor-



tant to have extremely steep pulse edges. These capacitors accelerate the switch-on response of the transistors and reduce the delay time when the transistors switch off.

The frequency or pulse duty cycle is set using PB0, PB1 and PB5. Via PB1, the microcontroller measures the time required for capacitor C8 to charge to a threshold level of 2.2 V. Within this range, the charging curve of the capacitor is still relatively linear. The maximum value is set by R6, and the minimum value by R9; these can be modified if desired. For good long-term stability, C8 should be a polystyrene type if possible. The microcontroller inputs are configured to use internal pull-up resistors.

The following is a summary of the utilisation of the I/O lines:

**PB0, PB1 & PB5**

Analogue signal processing using a simple RC network. The comparator in the microcontroller switches when the voltage on PB0 exceeds the value on PB1. The capacitor is charged and discharged under program control via PB5.

**PB2 & PB6**

Clock frequency on complementary outputs, switchable during operation. Approximately 100 fixed frequencies and six variable ranges can be selected. The outputs can be used directly or buffered and inverted by open-collector drivers.

**PB4**

1-MHz clock output when the fixed-frequency divider switch is set to '0' and Reset is pressed. This mode can only be exited by pressing Reset, which causes the output to go high and instigate the changeover via BCD switch S3. In normal operation, the output is low.

**PB7**

Pulse edge marker for each switching transition on PB2 and PB6. Pulse width 2  $\mu$ s. This output is active for both fixed and variable frequencies.

**PB3**

Pulse width modulator (PWM) output; operates in parallel with the frequency outputs (8-bit resolution). One of four PWM clock frequencies can be selected using PDO & PD1 (19.6 kHz, 2.45 kHz, 306.4 Hz or an external clock on PD5). The PWM can be adjusted over a range of approximately 0–99.5 percent of the supply voltage using an analogue voltage (P1). In the 1-

**Controls**

**FREQUENCY (S2, BCD):** selects the basic frequency

Fixed: 0.1, 1, 10, 100, 1000, 10 000, 50 000 Hz (settings 1–7)  
 Variable: 50 000, 10 000, 100, 100, 10, 1 Hz (settings 9–14)  
 The clock generator is stopped (disabled) for settings 0, 8, and 15.

**DIVIDER (S3, BCD):** divides the basic frequency in 15 steps

(1:1 to 1:15). The selected division factor only becomes effective after Reset or when the FREQUENCY setting is changed. If step 0 is selected and Reset is pressed, the 1-MHz mode is activated.

**PWM MODE (S1, BCD):** selects one of four clock

frequencies. The PWM operates in parallel to the frequency outputs. The pulse width can be adjusted using a potentiometer or an analogue voltage.

1-MHz mode, the PWM continues to operate with its most recent setting. Same output options as PB2 and PB6.

**PD0 & PD1**

Configuration inputs for selecting the PWM clock frequency using DIP switches or a rotary BCD switch. The setting is captured on Reset (see Table 1).

**PD2, PD3, PD4 & PD6**

Configuration inputs for fixed-frequency and divider settings using two BCD switches. The frequency setting is captured immediately following any changes to the input values; the divider setting is only captured on Reset, at which time S3 is evaluated, following which PB4 switches back to S2.

Fixed frequencies								
Divider (S3)	Frequency (S2) (frequency in Hz)							
	0	1	2	3	4	5	6	7
0	1-MHz mode (activated by Reset)							
1	stopped	0.1	1	10	100	1,000	10,000	100,000
2	stopped	0.05	0.5	5	50	500	5,000	25,000
3	stopped	0.0333	0.3333	3.333	33.33	333.3	3,333	16,666
4	stopped	0.025	0.25	2.5	25	250	2,500	12,500
5	stopped	0.02	0.2	2	20	200	2,000	10,000
6	stopped	0.0166	0.1666	1.666	16.66	166.6	1,666	8,333
7	stopped	0.0143	0.143	1.43	14.3	143	1,430	7,143
8	stopped	0.0125	0.125	1.25	12.5	125	1,250	6,250
9	stopped	0.0111	0.1111	1.111	11.11	111.1	1,111	5,555
A	stopped	0.01	0.1	1	10	100	1,000	5,000
B	stopped	0.0091	0.091	0.91	9.1	91	910	4,545
C	stopped	0.00833	0.0833	0.833	8.33	83.3	833	4,166
D	stopped	0.0077	0.077	0.77	7.7	77	770	3,846
E	stopped	0.00714	0.0714	0.714	7.14	71.4	714	3,571
F	stopped	0.00666	0.0666	0.666	6.66	66.6	666	3,333

(Values rounded to 1% as necessary)

Variable frequencies	
FREQUENCY (S2)	Frequency range (Hz)
8	stopped
9	196 – 50,000
A	39 – 10,000
B	3.9 – 1,000
C	0.39 – 100
D	0.039 – 10
E	0.0039 – 1
F	stopped (free)
Frequency range resolution: 8 bits (255 steps)	

Pulse width modulator (PWM)			
PWM MODE (S1)	PB1	PB0	PWM clock
0	1	1	external (PD5)
1	1	0	19.6 kHz
2	0	1	2.45 kHz
3	0	0	306.4 Hz

## PD5

External PWM clock frequency. If Reset is pressed when S1 = 0, the PWM is switched to the external clock frequency.

The circuit can be powered by a small (mains adapter) power supply providing an output voltage of 8–12 V. On the circuit

board, a fixed voltage regulator converts the supply voltage to a stabilised 5-V level. D1 protects the circuit against reverse-polarity connection of the power supply. Decoupling capacitors should be soldered to the circuit board as close as possible to the supply pins of the microcontroller. The reset switch is debounced using an RC network. The crystal must have a frequency of 10 MHz, since all of the calculations in the software are based on this value. It should be located as close as possible to the microcontroller, along with the associated capacitors.

The pushbutton switch and BCD switches are also fitted to the PCB. It is a practical idea to use IC sockets to fit the BCD switches. This allows the switches to be fitted external to the PCB if necessary and connected using cables and plugs.

The provisions of the EMC Directive should generally be observed in the construction of the generator. After all, here you are working with a fast microcontroller with a 10-MHz clock frequency. The decoupling capacitors directly connected to the microcontroller crystal should be located as close as possible to the microcontroller, and the pertinent capacitors should be connected directly to ground. It is recommended to have the largest possible ground plane or a gridded ground reference plane, use short or screened wiring and use free-wheeling diodes with inductive loads.

(020395-1)

The microcontroller software, including the source code, is available on diskette from Readers Services under order number 020395-11 or free of charge via the Elektor Electronics website.

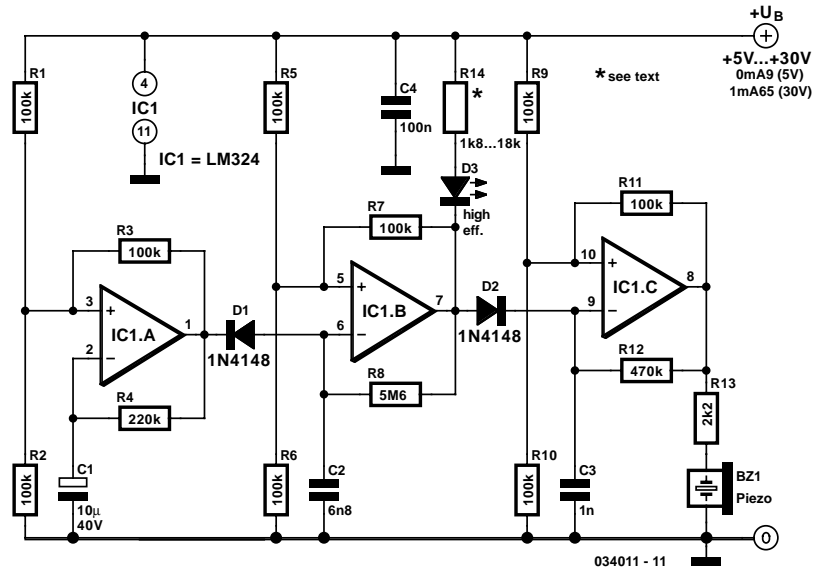
# Electronic Telephone Ringer

L. Libertin

This circuit produces a ringing sound similar to that made by more recent telephones. It consists of three almost identical oscillators connected in a chain, each generating a squarewave signal. The frequency of each oscillator depends on the RC combination: R4 and C1 around IC1.A, R8 and C2 around IC1.B and R12 and C3 around IC1.C. The pairs of 100 kΩ resistors divide the asymmetric power supply voltage (between 5 V and 30 V) so that, in conjunction with the 100 kΩ feedback resistors (R3, R7 and R11) either one third or two thirds of the supply voltage will be present at the non-inverting inputs to the opamps. The voltage across the capacitor therefore oscillates in a triangle wave between these two values.

The first oscillator is free-running at a frequency of approximately 1/3 Hz. Only when its output is high, and D1 stops conducting, can the second oscillator run. The frequency of the second oscillator is about 13 Hz, and optional LED D3 flashes when it is running. When the output of the second oscillator is low, the third is allowed to run. The frequency of the third oscillator is around 1 kHz, and this is the tone that is produced. The second oscillator is not absolutely necessary:

its function is just to add a little modulation to the 1 kHz tone. A piezo sounder is connected to the output of the third oscillator to convert the electrical signal into an acoustic one. The current consumption of the circuit is just under 1 mA with a 5 V power supply, rising to about 1.65 mA with a supply voltage of 15 V.



# IR-S/PDIF Transmitter

T. Giesberts

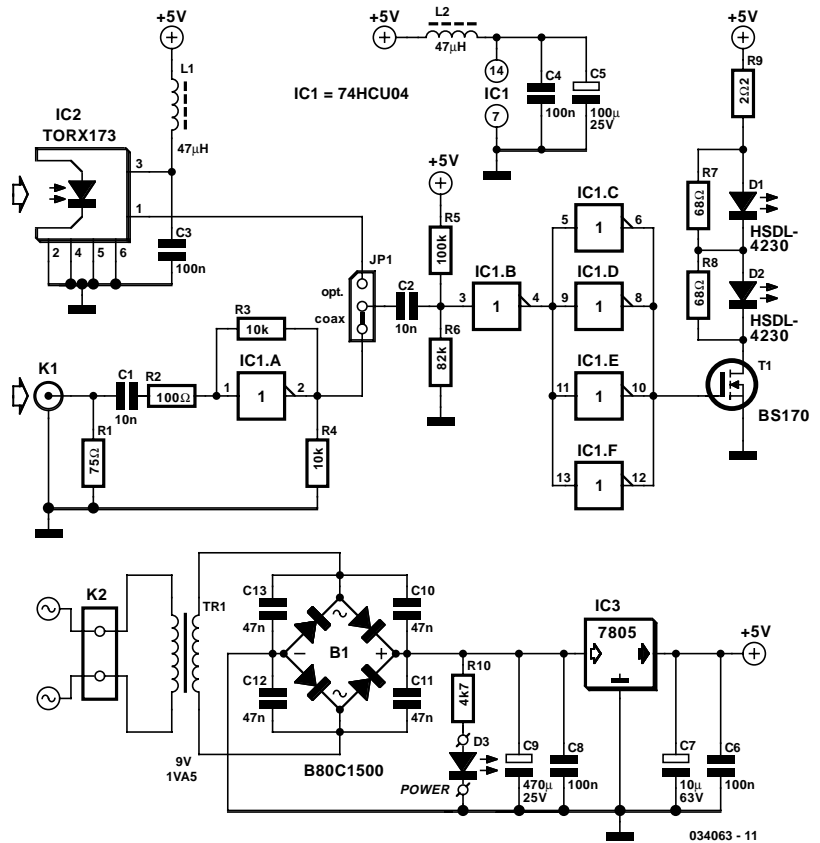
The best-known ways to transmit a digital audio signal (S/PDIF) are to use a standard 75- $\Omega$  coaxial cable or Toslink optical modules with matching optical cable. Naturally, it can happen that for whatever reason, you cannot (or don't wish to) run a cable between the equipment items in question. With a wireless solution, you have the choice of a wideband RF transmitter or an optical variant. Here we describe a simple optical transmitter. The matching IR-S/PDIF receiver is described elsewhere in this issue.

Although designing such an IR transmitter/receiver system does not have to be particularly difficult, in practice there are still several obstacles to be overcome. For one thing, the LEDs must have sufficient optical switching speed to properly pass the high frequencies of the S/PDIF signal, and they must also produce sufficient light intensity to deliver a noise-free signal at the receiver over a reasonable distance. At a sampling frequency of 48 kHz, it's necessary to be able to transfer pulses only 163 ns wide! The LEDs selected here (Agilent HSDL-4230) have optical rise and fall times of 40 ns, which proved to be fast enough in practice. With a beam angle of only 17°, they can also provide high light intensity. The downside is that the combination of transmitter and receiver is highly directional, but the small beam angle also has its advantages. It means that fewer LEDs are necessary, and there is less risk of continuously looking into an intense infrared source.

The circuit is essentially built according to a standard design. The S/PDIF signal received on K1 is amplified by IC1a to a level that is adequate for further use. JP1 allows you to use a Toslink module as the signal source if desired. JP1 is followed by a voltage divider, which biases IC1b at just below half of the supply voltage. This causes the output level of the buffer stage driving switching transistor T1 to be low in the absence of a signal, which in turn causes IR LEDs D1 and D2 to remain off.

The buffer stage is formed by the remaining gates of IC1. This has primarily been done with an eye to elevated capacitive loading, in the unlikely event that you decide to use more LEDs. A small DMOS transistor (BS170) is used for T1; it is highly suitable for fast switching applications. Its maximum switching time is only 10 ns (typically 4 ns).

Getting D1 and D2 to conduct is not a problem. However, stopping D1 and D2 from conducting requires a small addition to what is otherwise a rather standard IR transmitter stage, due to the presence of parasitic capacitances. This consists of R7 and R8, which are connected in parallel with the LEDs to



quickly discharge the parasitic capacitors. The drawback of this addition is naturally that it somewhat increases the current consumption, but with the prototype this proved to be only around 10 percent.

With no signal, the circuit consumes only 25 mA. With a signal, the output stage is responsible for nearly all of the current consumption, which rises to approximately 170 mA. In order to prevent possible interference at such high currents and avoid degrading the signal handling of the input stage, everything must be well decoupled. For instance, the combination of L2, C4 and C5 is used to decouple IC1. The circuit around T1 must be kept as compact as possible and placed as close as possible to the voltage regulator, in order to prevent the generation of external interference or input interference. If necessary, place a noise-suppression choke (with a decoupling capacitor to ground) in series with R9. Note that this choke must be able to handle 0.3 A, and if you use additional stages, this rating must be increased proportionally.

The circuit should preferably be fitted into a well-screened enclosure, and it is recommended to provide a mains filter for the 230-V input of the power supply. For the sake of completeness, we have included a standard power supply in the schematic diagram, but any other stabilised 5-V supply could be used as well. LED D3 serves as the obligatory mains power indicator.

# Power Buzzer

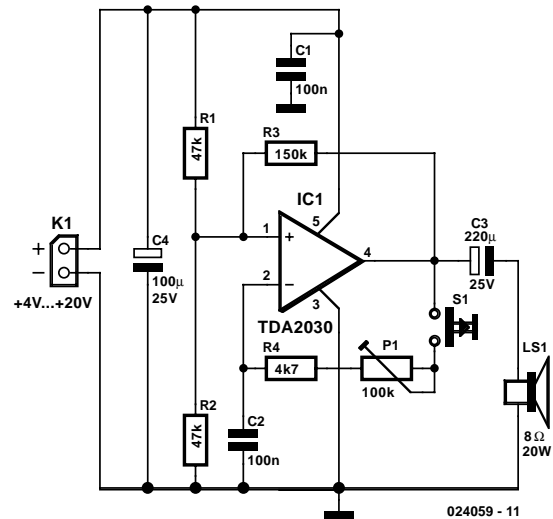
## G. Baars

How often on average do you have to call members of your family each day to tell them that dinner is ready, it's time to leave, and the like? The person you want is usually in a different room, such as the hobby room or bedroom. A powerful buzzer in the room, combined with a pushbutton at the bottom of the stairs or in the kitchen, could be very handy in such situations.

The heart of this circuit is formed by IC1, a TDA2030. This IC has built-in thermal protection, so it's not likely to quickly give up the ghost. R1 and R2 apply a voltage equal to half the supply voltage to the plus input of the opamp. R3 provides positive feedback. Finally, the combination of C2, R4 and trimmer P12 determines the oscillation frequency of the circuit. The frequency of the tone can also be adjusted using P1. There is no volume control, since you always want to get attention when you press pushbutton S1.

Fit the entire circuit where you want to have the pushbutton. The loudspeaker can then be placed in a strategic location, such as in the bedroom or wherever is appropriate. Use speaker cable to connect the loudspeaker. Normal bell wire can cause a significant power loss if the loudspeaker is relatively far away.

The loudspeaker must be able to handle a continuous power of at least 6 W (with a 20-V supply voltage). The power quickly drops as the supply voltage decreases ( $P = U_{\text{rms}}^2 / R_L$ ).



The power supply for this circuit is not particularly critical. However, it must be able to provide sufficient current. A good nominal value is around 400 mA at 20 V. At 4 V, it will be approximately 25 mA. Most likely, you can find a suitable power supply somewhere in your hobby room. Otherwise, you can certainly find a low-cost power supply design in your *Elektor Electronics* archive that will fill the bill!



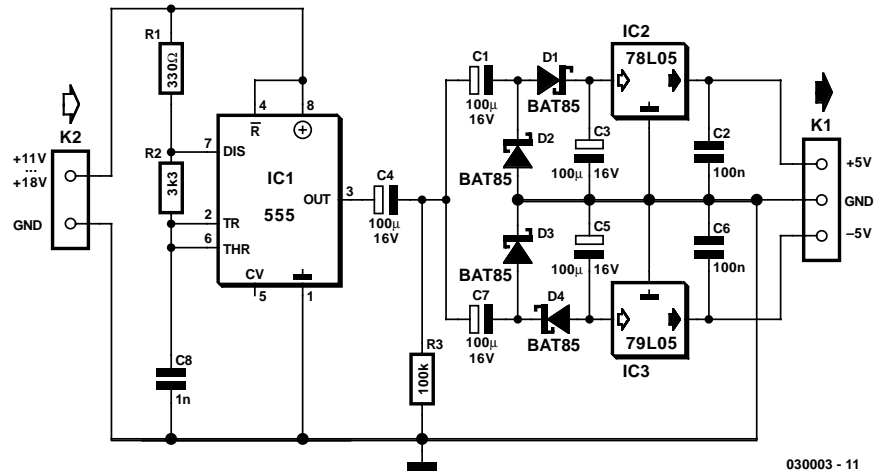
# 555 DC/DC Converter

L. de Hoo

It is all too often necessary to augment the power supply of an existing electronic circuit because exactly the voltage that you need is missing. The circuit presented here may provide a solution in a number of cases, since it can be used to convert a single-ended supply voltage into a balanced set of supply voltages. That's not so remarkable by itself, but the special feature of this circuit is that this is accomplished without using difficult to obtain, exotic ICs. All of the components used in the circuit are ones that every electronics hobbyist is likely to have in a drawer somewhere.

The heart of the circuit is formed by an 'old reliable' 555 timer, which is wired here as a free-running oscillator with a frequency of approximately 160 kHz. The oscillator is followed by two voltage-doubling rectifiers, consisting of C1, D1, D2, C3 and C7, D3, D4, C5. They are followed in turn by two voltage regulators to stabilise the positive and negative voltages generated in this manner.

The duty cycle of the 555 is set to approximately 50 percent using R1 and R2. The square-wave signal at the output of the timer IC has a DC offset, which is eliminated by C4 and R3. The amplitude of the output signal from the 555 is approximately equal to the supply voltage less 1.5 V, so with a 12-V input voltage, there will be a square-wave signal on pin 3 with an amplitude of approximately 10.5 V<sub>pp</sub>. With respect to ground (across R3), this is this +5 V / -5 V. Although this yields a symmetric voltage, its positive and negative amplitudes are



030003 - 11

somewhat too small and it is not stabilised. In order to split the square-wave signal into sufficiently large positive and negative amplitudes, C1/D2 are added for the positive voltage, causing the positive half to be doubled in amplitude. For the negative half, the same effect is achieved using C7/D3. Following this, the two signals are smoothed by D1/C3 and D4/C5, respectively. Both voltages are now high enough to be input to normal 5-V voltage regulators, yielding symmetric +5-V and -5-V supply voltages at the output.

The input voltage does not have to be regulated, although it must lie between +11 V and +18 V. The maximum output current is  $\pm 50$  mA with an input voltage of 12 V. This circuit is an excellent choice for generating auxiliary voltages, such as supply voltages for low-power opamps. Naturally, the fact that the converter can be powered from the in-vehicle voltage of a car is a rather attractive feature.

(030003-1)

# IR-S/PDIF Receiver

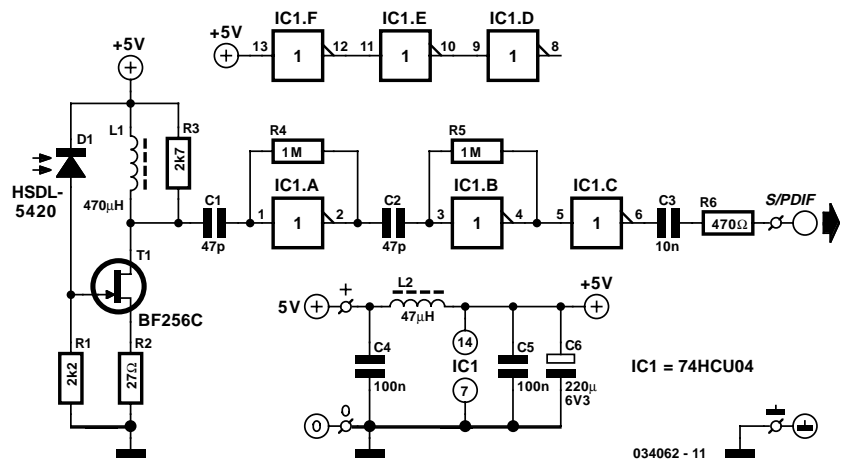
T. Giesberts

This simple circuit proves to achieve surprisingly good results when used with the IR-S/PDIF transmitter described elsewhere in this issue. The IR receiver consists of nothing more than a photodiode, a FET and three inverter gates used as amplifiers. The FET is used as an input amplifier and filter, due to its low parasitic capacitance. This allows R1 to have a relatively high resistance, which increases the sensitivity of the receiver. The bandwidth is primarily determined by photodiode D1, and with a value of 2k2 for R1, it is always greater than 20 MHz.

The operating current of the FET is intentionally set rather high (around 10 mA) using R2, which also serves to ensure adequate bandwidth. The voltage across R2 is approximately 0.28–0.29 V. The combination of L1 and R3 forms a high-pass filter that allows signals above 1 MHz to pass. L1 is a standard noise-suppression choke. From this filter, the signal is fed to two inverters configured as amplifiers. The third and final inverter (IC1c) generates a logic-level signal. This 74HCU04 provides so much gain that there is a large risk of oscillation, particularly when the final stage is loaded with a 75-Ω coaxial cable. In case of problems (which will depend heavily on the construction), it may be beneficial to add a separate, decoupled buffer stage for the output, which will also allow the proper output impedance (75 Ω) to be maintained in order to prevent any reflections.

When building the circuit, make sure that the currents from IC1 do not flow through the ground path for T1. If necessary, use two separate ground planes and local decoupling. Furthermore, the circuit must be regarded as a high-frequency design, so it's a good idea to provide the best possible screening between the input and the output.

With the component values shown in the schematic, the range is around 1.2 metres without anything extra, which is not especially large. However, the range can easily be extended by using a small positive lens (as is commonly done with standard IRDA modules). In our experiments, we used an inexpensive



magnifying glass, and once we got the photodiode positioned at the focus after a bit of adjustment, we were able to achieve a range of 9 metres using the same transmitter (with a sampling frequency of 44.1 kHz). This does require the transmitter and receiver to be physically well aligned to each other. As you can see, a bit of experimenting certainly pays off here!

It may also be possible to try other types of photodiode. The HDSL-5420 indicated in the schematic has a dome lens, but there is a similar model with a flat-top case (HDSL-5400). It has an acceptance angle of 110°, and with the same level of illumination, it generates nearly four times as much current.

The current consumption of the circuit is 43 mA with no signal and approximately 26 mA with a signal ( $f_s = 44.1$  kHz). That is rather high for battery operation, but it can be handled quite readily using a pair of rechargeable NiMH cells. Incidentally, the circuit will also work at 4.5 V and even 3 V. If a logic-level output is needed, C3 at the output can be replaced by a jumper.

Finally, there is one other thing worth mentioning. With the HDSL-5400 that we had to play with, the cathode marking (a dark-blue line on the side below one lead) was on the wrong side (!). So if you want to be sure that the diode is fitted properly, it's a good idea to measure the DC voltage across R1, which should be practically zero.

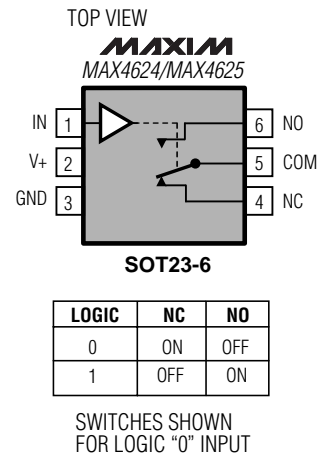
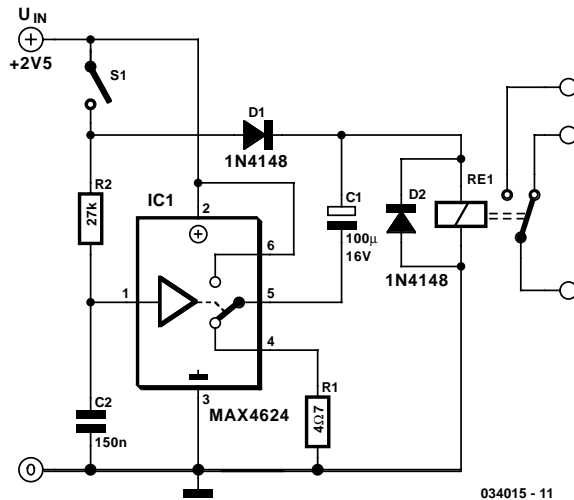
# Reducing Relay Power Consumption

Source:  
Maxim Design Showcase

Relays are often used as electrically controlled switches. Unlike transistors, their switch contacts are electrically isolated from the control input. On the other hand, the power dissipation in a relay coil may be unattractive for battery-operated applications. Adding an analogue switch lowers the dissipation, allowing the relay to operate at a lower voltage. The circuit diagram shows the principle. Power consumed by the relay coil equals  $V^2/R_{COIL}$ . The circuit lowers this dissipation (after actuation) by applying less than the normal operating voltage of 5 V. Note that the voltage required to turn a relay on (pickup voltage) is usually greater than that to keep it on (dropout voltage). In this respect the relay shown has specifications of 3.5 and 1.5 V respectively, yet the circuit allows it to operate from an intermediate supply voltage of 2.5 V. **Table 1** compares the relay's power dissipation with fixed operating voltages across it, and with the circuit shown here in place. The power savings are significant.

When SW1 is closed, current flows through the relay coil, and C1 and C2 begin to charge. The relay remains inactive because the supply voltage is less than its pickup voltage. The RC time constants are such that C1 charges almost completely before the voltage across C2 reaches the logic threshold of the analogue switch inside the MAX4624 IC. When C2 reaches that threshold, the on-chip switch connects C1 in series with the 2.5 V supply and the relay coil. This action causes the relay to be turned on because its coil voltage is then raised to 5 V, i.e., twice the supply voltage. As C1 discharges through the coil, the coil voltage drops back to 2.5 V minus the drop across D1. However, the relay remains on because the resultant voltage is still above the dropout level (1.5 V).

Component values for this circuit depend on the relay characteristics and the supply voltage. The value of R1, which protects the analogue switch from the initial current surge through C1, should be sufficiently small to allow C1 to charge rapidly, but large enough to prevent the surge current from exceeding the specified peak current for the analogue switch.



**Table 1. Power dissipated by relay**

Voltage (V)	Current (mA)	Total Power Dissipation (mW)
5 (normal operating voltage)	90	450
3.5 (pickup voltage)	63	221
2.5 (application circuit)	45	250

The switch's peak current (U1) is 400 mA, and the peak surge current is

$$I_{PEAK} = (V_{IN} - V_{D1}) / R1 + R_{ON}$$

where  $R_{ON}$  is the on-resistance of the analogue switch (typically 1.2  $\Omega$ ). The value of C1 will depend on the relay characteristics and on the difference between  $V_{IN}$  and the pickup voltage. Relays that need more turn-on time requires larger values for C1.

The values for R2 and C2 are selected to allow C1 to charge almost completely before C2's voltage reaches the logic threshold of the analogue switch. In this case, the time constant  $R2C2$  is about seven times  $C1(R1 + R_{ON})$ . Larger time constants increase the delay between switch closure and relay activation.

The switches in the MAX4624 are described as 'guaranteed break before make'. The opposite function, 'make-before break' is available from the MAX4625. The full datasheets of these interesting ICs may be found at

<http://pdfserv.maxim-ic.com/arpdf/MAX4624-MAX4625.pdf>

# Motorcycle Battery Monitor

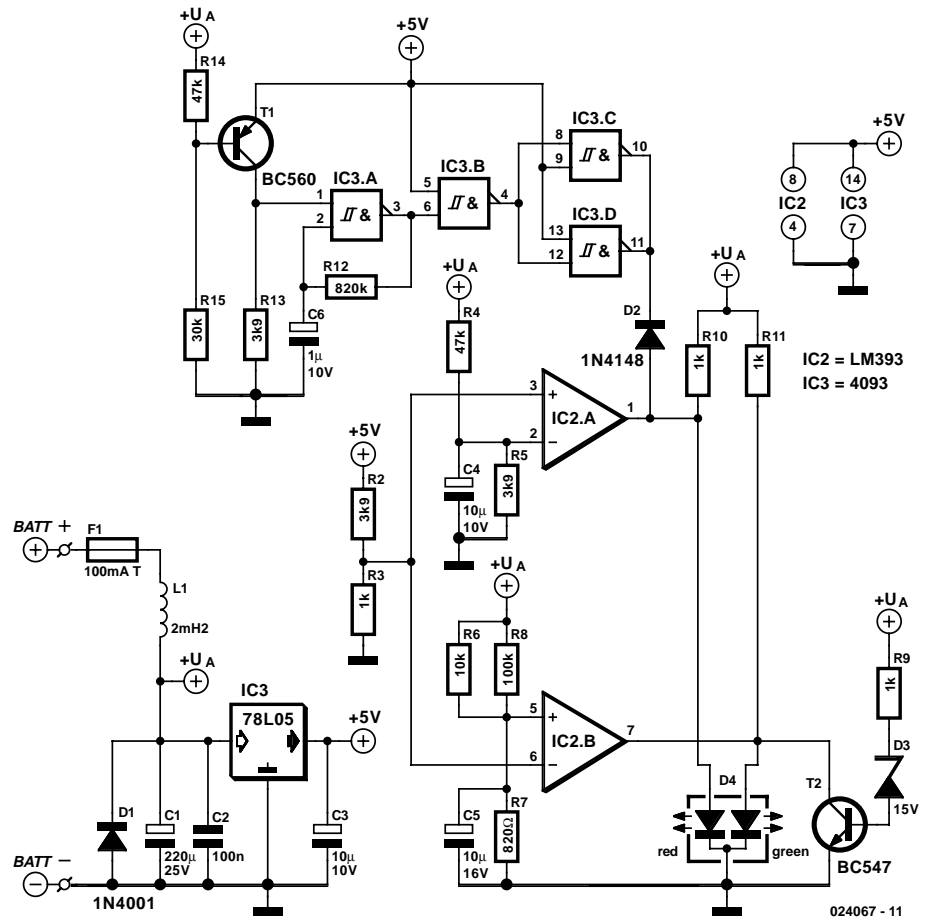
A. Eschhold

A circuit for monitoring the status of the battery and generator is undoubtedly a good idea for motorcyclists, as for other motorists. However, not every biker is willing to drill the necessary holes in the cockpit for the usual LED lamps, or to screw on an analogue accessory instrument. The circuit shown here manages to do its job with a single 5-mm LED, which can indicate a total of six different conditions of the onboard electrical system. This is done using a dual LED that can be operated in pulsed or continuous mode (even in daylight). Built on a small piece of prototyping board and fitted in a mini-enclosure, the complete circuit can be tucked inside the headlamp housing or hidden underneath the tank.

The heart of the circuit is IC2, a dual comparator. The comparator circuit is built without using any feedback resistors, with the indication being stabilised by capacitors C4 and C5 instead of hysteresis. Small 10- $\mu$ F tantalum capacitors work well here; 220- $\mu$ F 'standard' electrolytic capacitors are only necessary with poorly regulated generators. Voltage regulator IC1 provides the reference voltage for IC2 via voltage divider R2/R3.

The onboard voltage is compared with the reference voltage via voltage dividers R4/R5 and R6/R7, which are connected to the inverting and non-inverting comparator sections, respectively. Using separate dividers allows the threshold levels to be easily modified by adjusting the values of the lower resistors.

IC2a drives the anode of the red diode of LED D4 via pull-up resistor R10. The anode of the green diode is driven by IC2b and R11. T2 pulls R11 to ground, thereby diverting the operating current of the green diode of the LED, if the voltage of the electrical system exceeds a threshold level of 15 V (provided by Zener diode D3). The paralleled gate outputs on pins 10 and 11 of IC3 perform a similar task. However, these gates have internal current limiting, so they can only divert a portion of the current from the red diode of the LED. The amount of current diverted depends on the battery voltage. The two gates are driven by an oscillator built around IC3a, which is enabled via voltage divider R14/R15 and transistor T1 when the battery voltage is sufficiently high. Depending on the state of IC3a, the red diode of the LED blinks or pulses. The circuit is connected to the electrical system via fuse F1 and a low-pass filter formed by L1 and C1. If you cannot obtain a low-resistance choke, a 1- $\Omega$  resistor can be used instead. In this

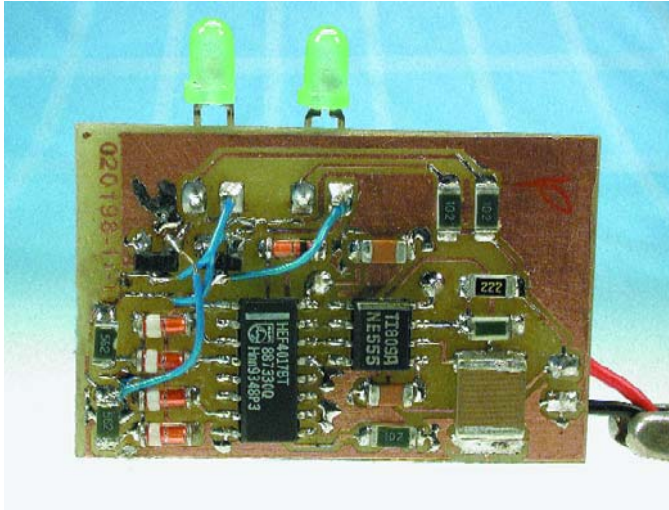


case, the values of C3, C4 and C5 should be increased somewhat, in order to help stabilise the indication. D1 protects the circuit against negative voltage spikes, as well as offering protection against reverse-polarity connection. Due to its low current consumption (less than 30 mA), the circuit could be connected directly to the battery, but it is better to power it from the switched positive voltage.

(024067-1)

Indication	Ignition on, lights off	Motor rpm > 2000
Blinking red	Battery deeply discharged ( $U_{BATT} < 7.1 V$ )	Electrical system defective
Pulsing red	Battery empty ( $7.1 V < U_{BATT} < 11.3 V$ )	Generator not charging
Continuous red	Battery half full ( $11.3 V < U_{BATT} < 12.1 V$ )	Generator barely charging
Continuous yellow	Battery full ( $12.1 V < U_{BATT} < 13.0 V$ )	Generator overloaded
Continuous green	$13.0 V < U_{BATT} < 16.0 V$	Normal state
Dark	$U_{BATT} \geq 16.0 V$	Regulator defective

# Alternating Blinker for Model Vehicles

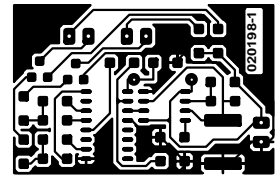
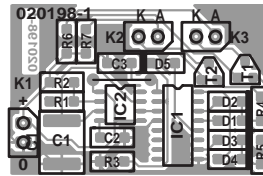


U. Neubert

This type of blinker is primarily used in model vehicles, where it can be used to achieve a realistic imitation of the dual flasher bar on an ambulance or fire engine. The LEDs, which are connected to the two open collectors, blink twice on the right and then twice on the left. The visual effect is simply fantastic. The electronics consists of a simple 555-based clock generator and a 4017 decimal counter. The 555 is wired as a free-running oscillator.

The timer output signal on pin 3 is fed to the Count input of the 4017, whose outputs go high sequentially. Each time an output goes high, the previous output goes low, as it should be with a decimal counter.

2



## COMPONENTS LIST

(all parts SMD)

### Resistors:

- R1 = 2kΩ
- R2 = 180kΩ
- R3,R6,R7 = 1kΩ
- R4,R5 = 5kΩ

### Capacitors:

- C1 = 1μF 16V
- C2 = 10nF
- C3 = 100nF

### Semiconductors:

- D1-D5 = BAS32
- D6,D7 = LED, low current\*
- T1,T2 = BC847
- IC1 = 4017T
- IC2 = NE555

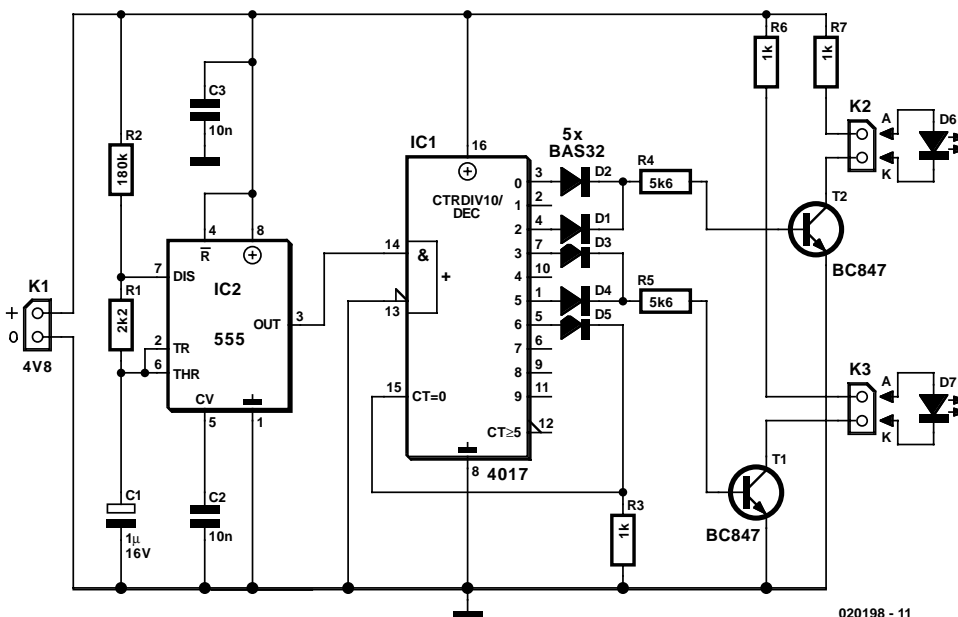
### Miscellaneous:

- K1 = supply connections

\* not SMD, see text

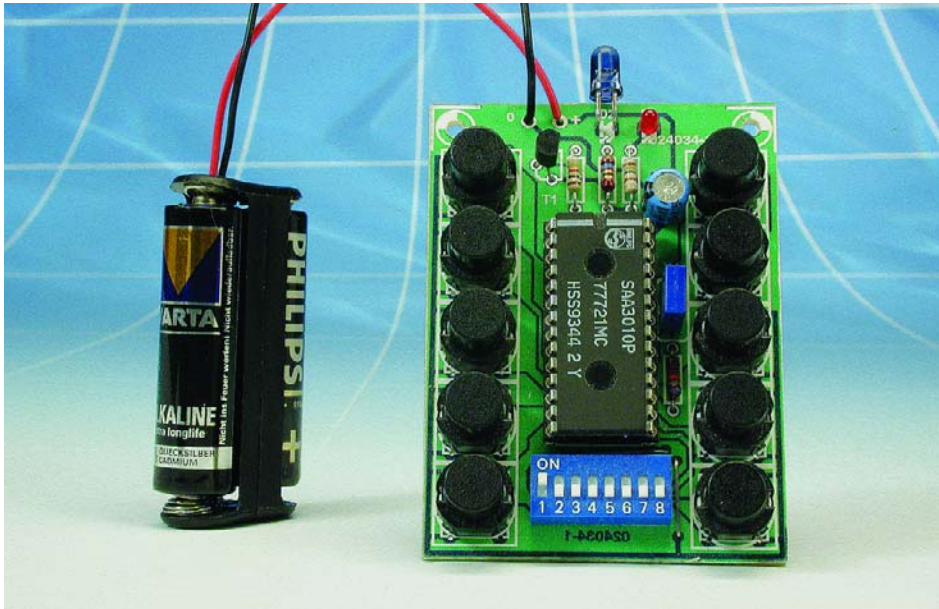
The 'program' runs in six steps. In the first and third steps, T2 conducts, while in the fourth and sixth steps T1 conducts. Each time one of the transistors conducts, the connected LED flashes. On the second and fifth clock pulse, both transistors are switched off and the LEDs are dark. The leading edge of the seventh clock pulse resets the counter via Q6, and the cycle starts again from the beginning. Other blinking patterns can also be implemented using different logical-OR combinations of the IC outputs.

The open-collector outputs allow other loads, such as relays or incandescent lamps, to be connected in place of the LEDs. If this is done, R6 and R7 should be replaced by 0-Ω resistors or wire bridges. However, the load current should not exceed 300 mA. Circuit construction using the printed circuit board shown in **Figure 2** is not difficult, despite the use of SMD components, so it should work properly right from the start.



020198 - 11

# Small RC5 Transmitter



ing stocks — and that could remain true for some time to come. It is thus certainly worthwhile to design a simple RC5 remote control unit, especially given the availability of the PT2211 from Princeton in Taiwan and the HT6230 from Holtek, which are nearly identical alternatives.

Our objective is to have a versatile transmitter that can communicate with several different devices. As this can only be achieved by using a user-modifiable address code, an 8-section DIP switch is used to allow the system address to be selected within the range of 0–7. According to the RC5 protocol, this makes it possible to address the devices listed in **table 1** — and for our purposes, the

**F. Wohlraabe**

We have published many articles in past issues of *Elektor Electronics* about IR data transmission using the Philips RC5 code. A prerequisite for these using these circuits has always been an RC5 remote control made by Philips or Loewe, or else a 'home-brew' RC5 transmitter based on the SAA3010 encoder IC. Philips has now officially discontinued its RC5 encoder, effective the end of last year. Nevertheless, it does not appear to be difficult to obtain small quantities of this IC from exist-

'DIY system address' (7) is naturally especially interesting.

A jumper (wire bridge) allows other address ranges to be alternatively selected, as shown in **table 2**. This allows the RC5 transmitter to work with all possible RC5 receivers.

Using the ten pushbutton switches, ten different commands can be sent to the device selected by JP2 and S11. Buttons S1 and S2 have a special role here: depending on the setting of

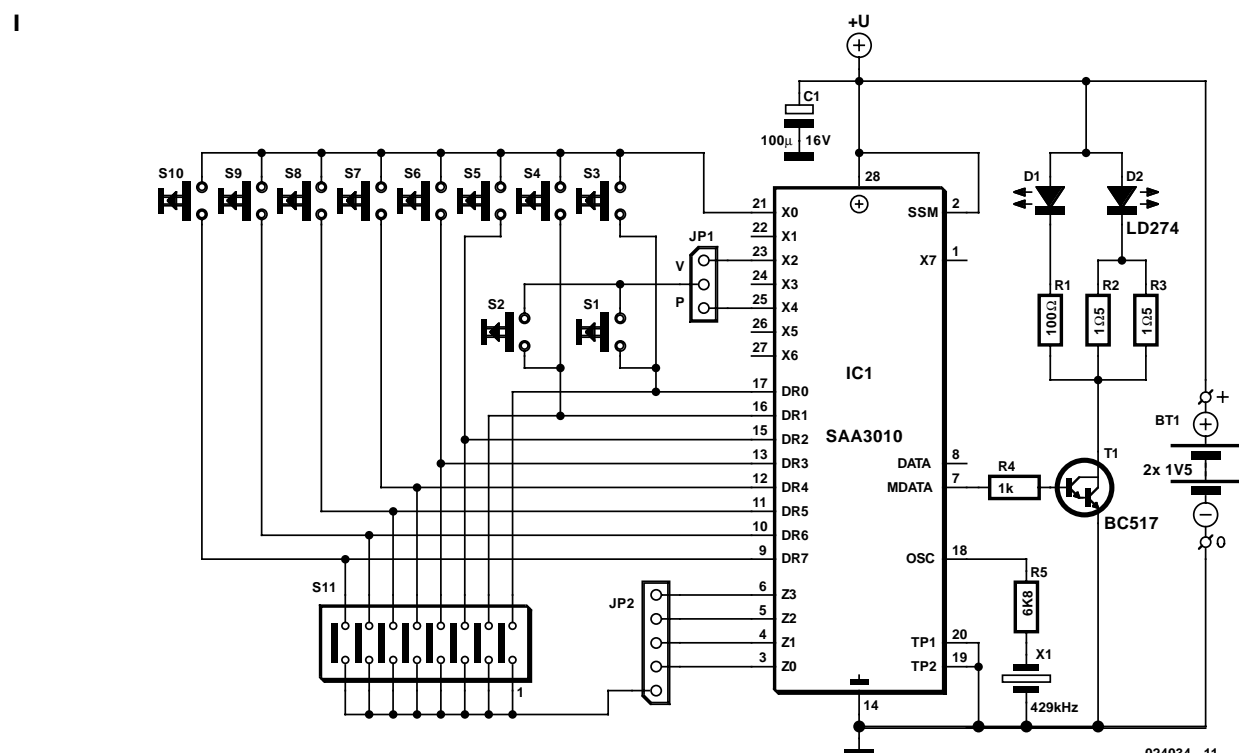


Table 1

SII	Address	Device
1	0	TV1
2	1	TV2
3	2	Videotext
4	3	Extension TV1/TV2
5	4	Laser Vision Player
6	5	VCR1
7	6	VCR2
8	7	Reserved

JP1 (which may be a jumper, a wire bridge or a small slide switch), S1 and S2 control the volume (audio) or the preset values, as shown in **table 3**. This makes our simple RC5 transmitter compatible with the Model Remote Control circuit design published in the May 2001 issue of *Elektor Electronics*.

Table 2

Address range	Jumper JP2
0-7	1-2
8-15	1-3
16-23	1-4
24-31	1-5

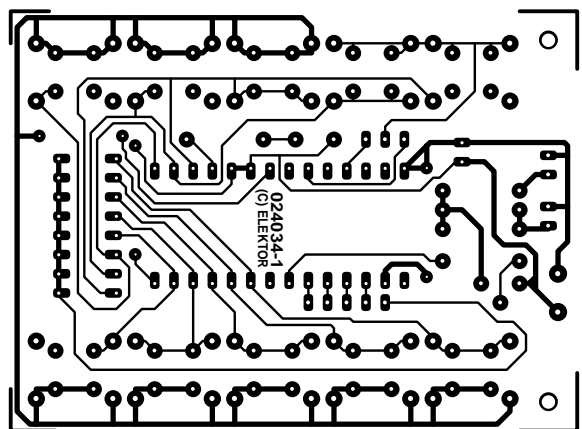
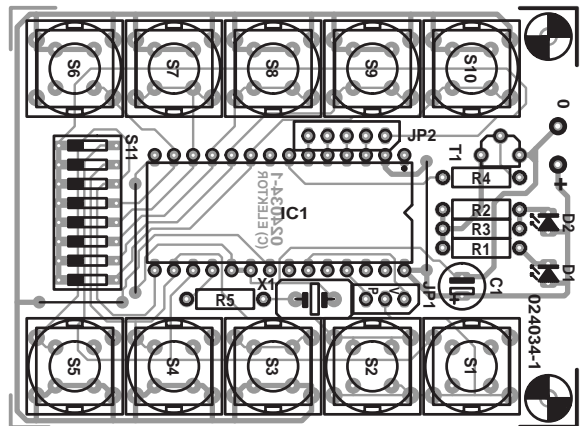
In other respects, the circuit shown in **Figure 1** corresponds to the specifications in the data sheet. The SSM pin is connected to  $V_{pp}$ , since address selection is hard-wired using the DIP switch. Philips prescribes a resonator with a 429 kHz clock frequency for compliance with the timing specified for the RC5 protocol. Using a 455 kHz resonator (which is easier to obtain) is not allowed, since it would cause the bit intervals to be shortened by around 40  $\mu$ s. However, this could be taken into account in a DIY receiver design.

A BC517 Darlington transistor is used as a driver for the IR

Table 3

Button	Command	JP1	Command
S3	0	-	0
S4	1	-	1
S5	2	-	2
S6	3	-	3
S7	4	-	4
S8	5	-	5
S9	6	-	6
S10	7	-	7
S1	16	1-2	+ Volume
S2	17	1-2	- Volume
S1	32	2-3	+ Preset
S2	33	2-3	- Preset

2



### COMPONENTS LIST

**Resistors:**

- R1 = 100 $\Omega$
- R2,R3 = 1 $\Omega$ 5
- R4 = 1k $\Omega$
- R5 = 6k $\Omega$ 8

**Capacitor:**

- C1 = 100 $\mu$ F 16V radial (low profile)

**Semiconductors:**

- D1 = LED, red, 3 mm
- D2 = LD274
- T1 = BC517

IC1 = SAA3010 (see text)

**Miscellaneous:**

- BT1 = two 1.5V batteries with holder
- S1-S10 = pushbutton, PCB mount (e.g. ITT/Shadow D6)
- X1 = 429kHz 2-pin ceramic resonator
- JP1 = 3-way pinheader with jumper, or wire link
- JP2 = wire link
- S11 = 8-way SIP switch

transmitter diode (D2), which emits in the 950-nm region, because of its high current gain — which is so great that it can also drive a normal LED ‘on the side’ as an operational indicator. The current level is set to give the remote control transmitter a range of around eight metres.

The circuit works with a supply voltage of 3 V. Two AAA or AA cells can be used as a power source. In the quiescent state, the load on the batteries is less than 10  $\mu$ A.

We have designed a circuit board for this simple IR transmitter (**Figure 2**). The layout can be downloaded from the *Elektor Electronics* website (Free Downloads, item **024034-11**).

# FM Remote Control Receiver

T. Giesberts

It is often required to switch electrical appliances from a distance (the garage light, when you're still in your car for example) without there being a direct line of sight between the transmitter and receiver. This rules out the use of infrared, so we have to use radio signals instead. An ideal

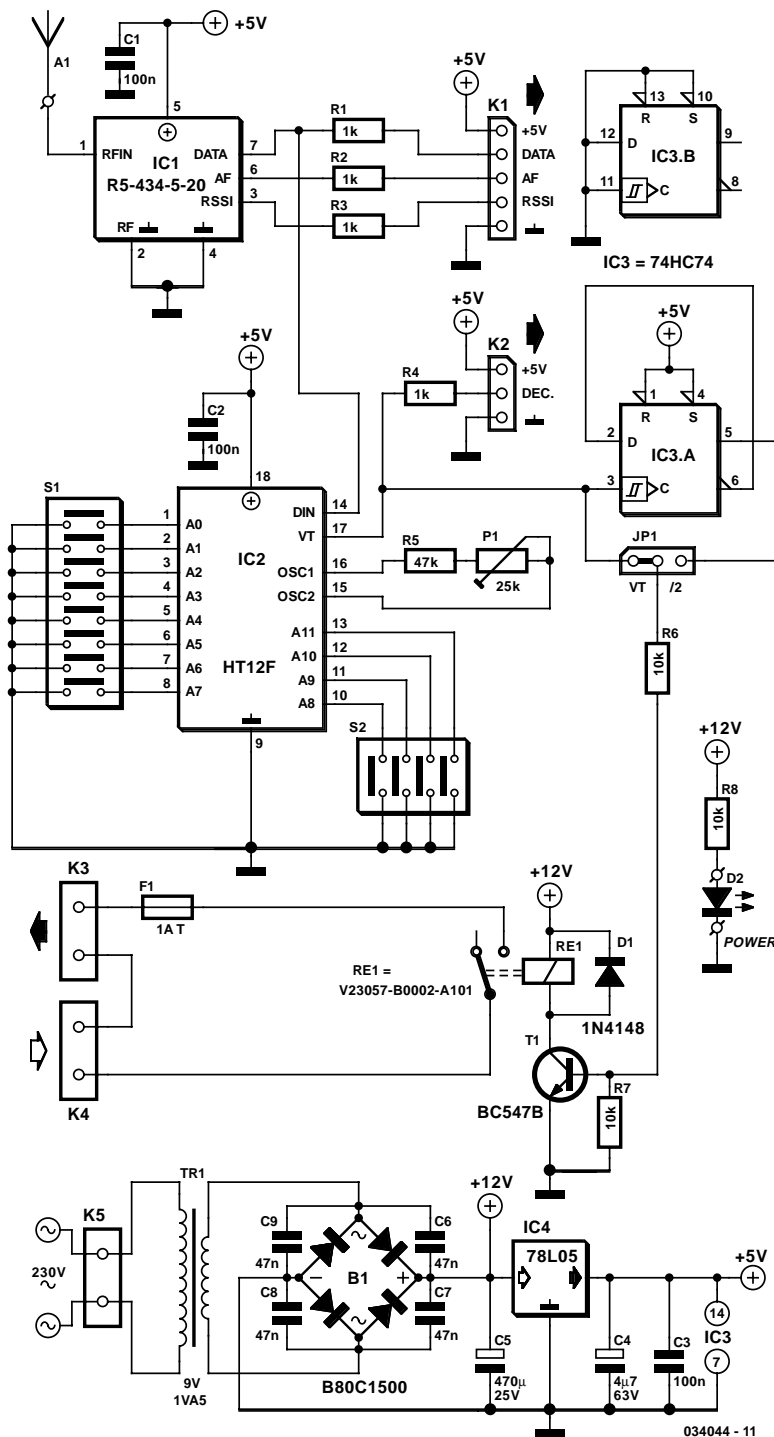
solution for this application is provided by transmitter and receiver modules, which operate at a frequency of 433 MHz and are available ready-made. This circuit complements the small FM transmitter that uses such a module. Here we describe the associated receiver, which picks up the transmitted signals using an R5-434-5-20 receiver module made by R.F. Solutions (this 20 kbps version is now also stocked by Farnell). This integrated receiver has been tuned to a frequency of 433.92 MHz, exactly the same as for the transmitter.

To prevent interference and unauthorised use the transmitter sends out a coded signal. This is processed in our receiver by decoder IC HT12F, made by Holtek. P1 and R5 are used to tune the oscillator frequency of the decoder to that of the encoder in the transmitter. In this way any possible variations due to tolerances or a different battery voltage can be compensated for by P1. The data bits are set up using solder bridges on the PCB (S1 and S2).

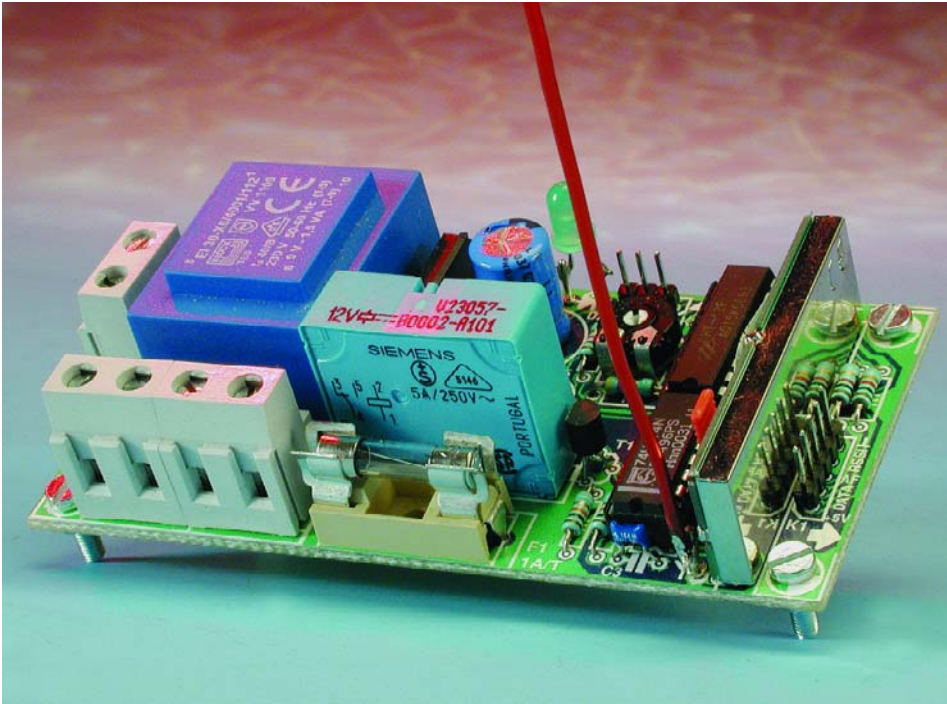
With jumper JP1 the output can be diverted through a divide-by-two circuit (half of IC3, a dual D-type flip-flop 74HC74), so that relay RE1 can remain energised without the presence of a signal. The make contact of the relay connects a contact of terminal block K4 to K3 via a fuse. The other contacts are connected directly. The relay and terminal blocks are placed far enough from mains voltages on the PCB to comply with Class II safety specifications. It is also possible to connect terminal block K4 with K5 in order to safely switch mains voltages. In this case you should make sure that the 'live' mains voltage is connected via the relay and fuse.

The circuit has a mains power supply on-board, making it ready for use as soon as the construction is complete. The relay and mains power indicator are driven directly from the smoothed transformer output, keeping the load on the small 5 V regulator (78L05) to a minimum. It should now only have to supply a few mAs.

The use of a ready-made receiver module simplifies the construction of this circuit and also makes it more reliable. Apart from the RF input (RF IN) the module has a data and analogue output (AF). There is also a status output that gives an indication of the RF signal strength (RSSI). These last two signals are not used in this application. There is another article in this issue (Amplifier with Squelch), which uses these two analogue signals with an extra circuit and a modified version of the transmitter. For this reason all outputs of the receiver module have been made available on a 5-way pin header (K1), along with the supply volt-







down. The small transmitter section can be cut easily from the board. The remaining receiver PCB is well-organised and easily populated (don't forget the wire-link underneath IC3!). It is possible that jumper JP1 may be a tight fit between IC3 and receiver module IC1. All connectors have been placed along the edges of the PCB. If you make a connection to K1 you should take care that you don't make contact with the metal shielding of the module.

(034044-1)

age. R1 to R3 protect the outputs from potential short circuits. The supply on this header is only meant to provide a few mAs!

Just as for the transmitter, the aerial has to be mounted as close as possible to the 'RF IN' pin. The construction of the aerial is described in the article for the transmitter. The output of the decoder (as well as the supply voltage) is brought out on a separate pin header (K2), making the logical signal available to circuits that need it.

The transmitter and receiver PCBs have been combined on one board to keep the cost

## COMPONENTS LIST

### Resistors:

- R1-R4 = 1k $\Omega$
- R5 = 47k $\Omega$
- R6,R7,R8 = 10k $\Omega$
- P1 = 25k $\Omega$  preset

### Capacitors:

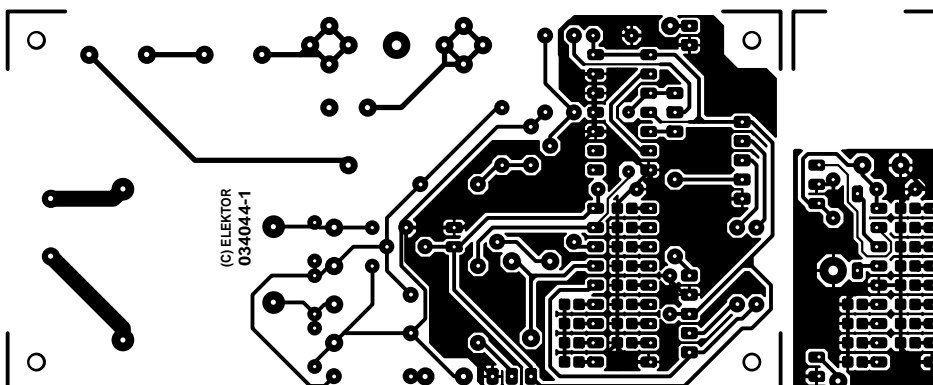
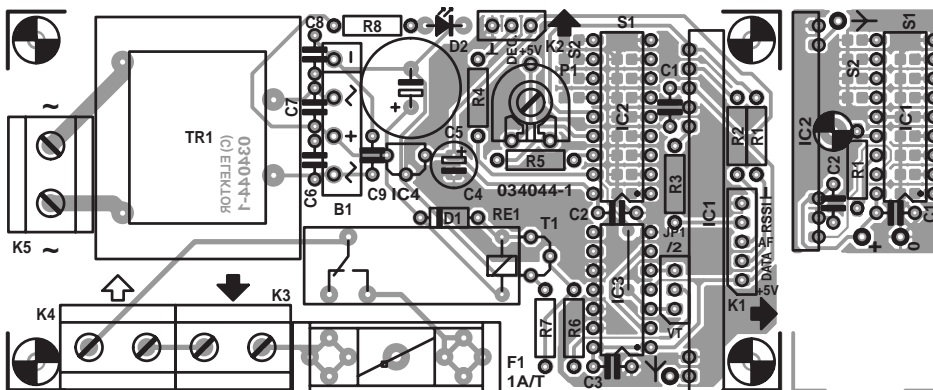
- C1,C2,C3 = 100nF ceramic
- C4 = 4 $\mu$ F 63V radial
- C5 = 470 $\mu$ F 25V radial
- C6-C9 = 47nF ceramic

### Semiconductors:

- D1 = 1N4148
- D2 = LED, high-efficiency
- T1 = BC547B
- IC1 = R5-434-5-20 from R.F. Solutions (Farnell # 352-4383)
- IC2 = HT12F from Holtek (Maplin)
- IC3 = 74HC74
- IC4 = 78L05

### Miscellaneous:

- JP1 = 3-way pinheader with jumper
- K1 = 5-way pinheader
- K2 = 3-way pinheader
- K3,K4,K5 = 2-way PCB terminal block, lead pitch 7.5 mm
- S1,S2 = solder links
- B1 = B80C1500 (rectangular case) (80V piv, 1.5A)
- TR1 = mains transformer, 9V/1.5VA, e.g., Block type EI30-X6/4001/112 VV1109
- F1 = fuse, IAT (time lag), with PCB mount holder
- RE1 = V23057-B0002-A101 vertical card relay 12 V/330  $\Omega$ / 8 A PCB, order code **034044-1** (transmitter and receiver)



# Cheap and Cheerful Transistor Tester

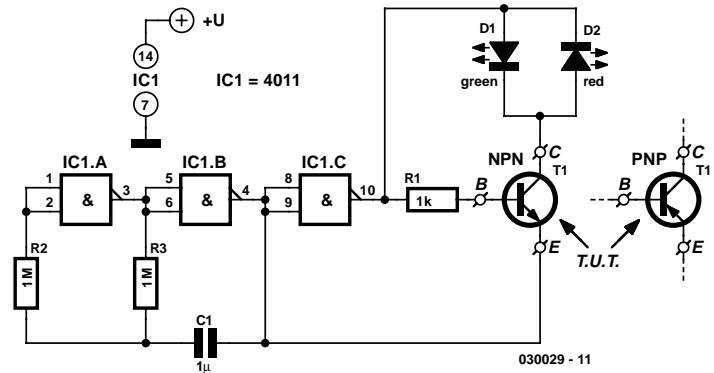
R. J. Gorkhali

By using a simple visual indicating system, this small transistor tester allows you to run a quick 'go/non-go' check on NPN as well as PNP transistors. If the device under test is a working NPN then the green LED (D1) will flash, while the red counterpart will flash for a functional PNP device. However if the transistor is shorted, both LEDs will flash, and an open-circuit device will cause the LEDs to remain off.

The circuit is based on just one CD4011B quad NAND gate IC, four passive parts and two LEDs. The fourth gate in the IC is not used and its inputs should be grounded. Alternatively, you may want to connect its inputs and output in parallel with IC1.C to increase its drive power to the transistor test circuit.

IC1.A and IC1.B together with R2, R3 and C1 form an oscillator circuit that generates a low-frequency square wave at pin 4. This signal is applied to the emitter of the transistor under test as well as to inverter IC1.C. The inverted signal from IC1.C and the oscillator output then drive the test circuit (LEDs, device under test, R1) in such a way that the voltage across that part of the circuit is effectively reversed all the time.

For example, with an NPN transistor under test, when pin 10 is High and pin 4, Low, current flows through LED D1 and the forward biased transistor. However, no current will flow when



pins 10 and 4 change states, since the transistor is then reverse-biased. The green LED, D1, will therefore flash at the rate determined by the oscillator. As you would expect to happen, a PNP transistor will be forward biased when pin 10 is Low and 4, High, enabling current to flow through the red LED in that case.

A supply rail of around 3 V (two series connected 1.5-V batteries) should be adequate. To prevent damage to the transistor under test, supply voltages higher than 4.5 V should not be used. Because the LED currents are effectively limited to a few mA by the output of IC1.C (also slightly dependent on the supply voltage), it is recommended to use high-efficiency devices for D1 and D2.

# Cat and Dog Repeller

I. Fietz

Nowadays, just about every house has an outside lamp with a motion sensor. Such a device eliminates the need to feel your way to the front door, and it apparently also scares away intruders. The only problem is that free-running dogs and cats in the neighbourhood have little regard for such lamps and continue to deposit their excrement in the garden, once they have found a habitual location there for this purpose.

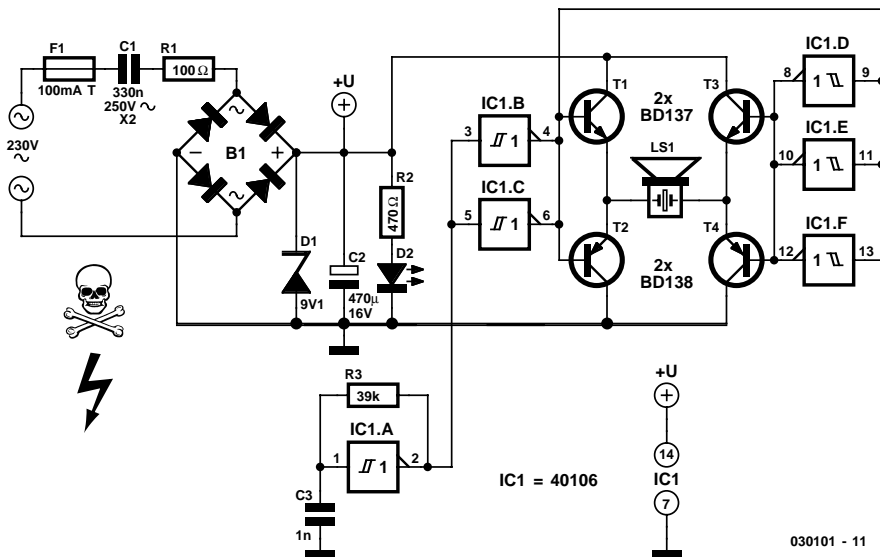
This gave rise to the idea of connecting a sort of siren in parallel with the outside lamp to clearly advise dogs and cats that they are not welcome. Naturally, it would be nice to avoid startling the entire neighbourhood with this alarm signal.

Here we can take advantage of the fact that dogs and cats have a significantly better sense of hearing than people. Not only are their ears more sensitive, they can also perceive significantly higher frequencies. With people, the upper limit is around 18 kHz, but dogs and cats can hear frequencies in excess of 20 kHz. We can take advantage of this by building a siren that emits a frequency just above 20 kHz. This will scare off dogs and cats, but people will simply not hear it.

All we need for this is an oscillator with an amplifier stage and a tweeter that can reproduce such high frequencies, such as a piezoelectric tweeter. The schematic diagram shows how easily this can be implemented.

The power supply for the entire circuit is formed by the components up to and including C2. The 230-V leads are connected in parallel with the motion-sensor lamp. C1 and R1 provide capacitive coupling to reduce the 230 V to an acceptable voltage. A DC voltage of approximately 9.1 V is generated from this voltage using a bridge rectifier and D1, filtered and buffered by C2.

The oscillator is built around R3, C3 and IC1a. The frequency of this oscillator is rather dependent on the specific characteristics of IC1, so the values shown here should be regarded



as guidelines. If the oscillator frequency is too high, it can be reduced by increasing the value of R3 and/or C3. If the frequency is too low (which means that the siren tone it is audible), the value of R3 and/or C3 should be increased.

The square-wave signal from the oscillator is applied to the input of an H bridge composed of several Schmitt triggers in combination with the final output stages (T1–T4). This approach causes the peak-to-peak value of the square wave signal to be twice the supply voltage. As a result, a respectable 18 V is obtained across the piezoelectric tweeter, which is sufficient to produce a quite loud whistle tone.

When building the circuit, you should bear in mind that it is directly powered from 230 V and not electrically isolated from the mains network. It is thus necessary to avoid contact with all of the components when the circuit is in use. In practice, this means that the circuit must be fitted into a well-insulated, waterproof box. If you want to test the circuit, it is a good idea to first discharge C1 using a resistor, since it can hold a dangerous charge. You must also ensure that components F1, C1, R1 and B1 all have a mutual insulation separation of at least 6 mm!

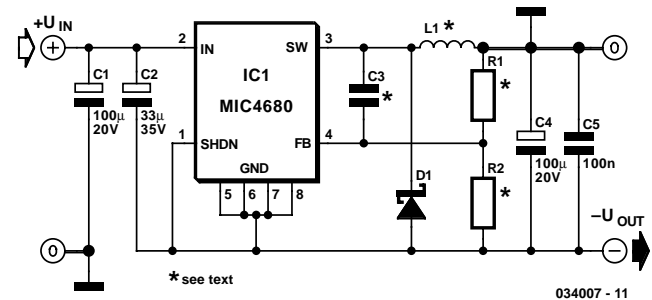
# Voltage Inverter Using Switch-Mode Regulator

G. Kleine

This circuit uses a step-up switch-mode regulator, which is usually used to produce a positive supply, to generate a regulated negative output voltage. The device used here is the MIC4680 from Micrel ([www.micrel.com](http://www.micrel.com)), but the idea would of course work with similar regulators from other manufacturers. Because of coil L1, which performs the voltage conversion by the intermediate storage of energy in the form of a magnetic field, the output is effectively isolated from the input. We can therefore connect the right-hand side of L1 to ground rather than to the positive output without causing a large current to flow. Then we connect the ground pin of the regulator IC and all the components connected to it as the negative voltage output, isolated from ground. The components on the output side of the regulator are connected as usual: flywheel diode D1, coil L1 and the voltage divider formed by R1 and R2. These last two components set the output voltage, according to a formula given in the data sheet. Example component values for the MIC4680 used here are given in the table.

The input voltage should lie within the permitted range for the regulator used, and must in any case be at least as great in magnitude as the desired output voltage (here +5 V or +12 V), so that the step-down regulation technique can work.

It is important to take care when building this circuit to mount



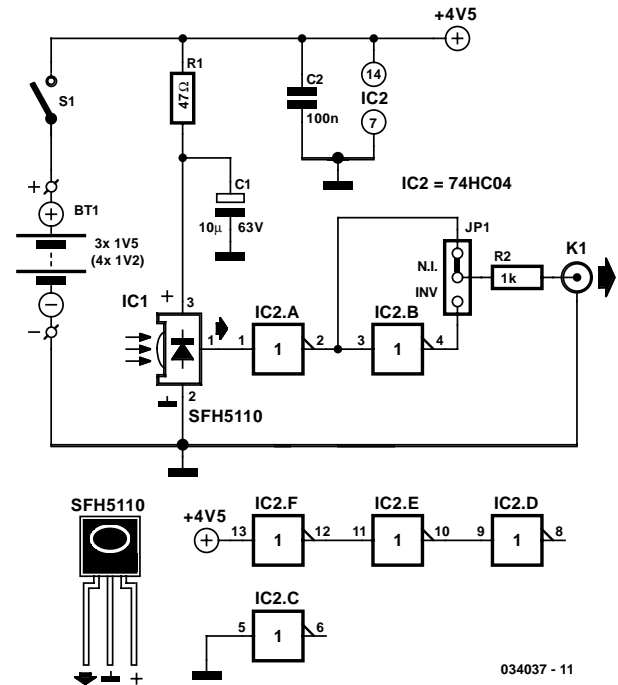
$U_{OUT}$	$I_{OUT}$	C3	D1	L1	R1	R2
-5 V	0.5 A max.	10 nF	SS14 (1 A, 40 V)	15 µH	3 kΩ	1 kΩ
-12 V	0.15 A max.	22 nF	ES1B (1 A, 100 V)	33 µH	8.87 kΩ	1 kΩ

the regulator using an insulator, since generally the GND pin of the device is connected to the heatsink tab. Also, the ON/OFF control input cannot be driven using a normal logic signal, since the regulator's ground reference is the output voltage rather than ground itself. If the ON/OFF function is required, a level shifter or optocoupler must be used.

# IR Remote Control Receiver

With many audio systems consisting of separate units, you'll often find that due to economic reasons only the amplifier has a remote control receiver module. The control signals are then sent to the other units using patch cables. The tuner and CD player, for example, won't have a built-in receiver module. When the tuner from such a system is bought separately it can therefore not be used directly with a remote control, which is a big disadvantage in practice. The only way in which this can be accomplished is to connect an IR receiver to the input used by the patch cable. And that is exactly what this circuit is for. In practice it is not always clear which signal should be used and what its polarity should be. However, it will most likely be a demodulated signal. For these reasons we've combined a standard IR receiver module and two inverters. The first inverter also functions as a buffer, since the output of the receiver module has a high impedance. The output of the receiver module is active low, so the first inverter outputs a non-inverting signal. The second inverter inverts this signal again. Jumper JP1 is used to select which of the signals is presented at the output. R2 protects the output from short circuits or possible overloading of the electronics in the equipment it's driving (for example when the input circuit uses 3 V logic). R1/C1 suppress any possible supply spikes.

Batteries are suitable for the power supply, because the circuit only takes about 1 mA. With a set of four rechargeable batteries with a capacity of 1800 mAh the circuit can function continuously for 2.5 months. Four NiMH cells and a charger are therefore perfect for the power supply. If you can be sure that the circuit will always be switched off when not



in use, you could also use three ordinary alkaline batteries (AA cells). Because of their slightly larger capacity they will probably last for about half a year. When making your choice you should of course keep in mind that rechargeables are better for the environment.

# LED Flasher

T. Giesberts

The idea behind this economical flashing LED indicator is very simple: it uses a very low frequency oscillator to drive two LEDs. As can be seen in the circuit diagram, the LEDs are connected in anti-parallel. At every change in level there will only be a current flow for a short time, due to the large capacitor (C3) that is connected in series with D1 and D2. When C3 is charging D1 lights up and when it is discharging D2 lights up. The current through the LEDs is limited by R4. When the circuit is switched on the peak current is determined by R4 and for a red LED will be just over 7 mA:

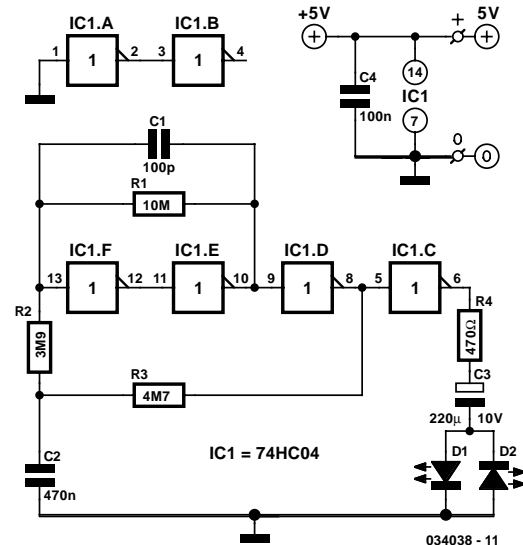
$$(5 - 1.6) / 470 = 7.2 \text{ mA.}$$

This value has been chosen because it is the maximum safe current for certain types of low-current LEDs. When the circuit is in operation the (dis)charging of C1 determines the maximum current. The electrolytic has a minimum voltage remaining which is equal to the LED voltage, causing the pulses to be smaller than the very first pulse:

$$(5 - 2 \times 1.6) / 470 = 3.8 \text{ mA peak.}$$

If one LED is sufficient then you can replace either D1 or D2 with a Schottky diode; the pulse through the remaining LED then becomes slightly larger. The use of two LEDs has the advantage that the discharge current of C3 is also used by the indicator, which makes the circuit more efficient.

The oscillator can of course be replaced by a different type, for example the classic Schmitt-trigger/inverter type using a 74HC14 and RC network. The version used here has the advantage of offering a better stability. C1 makes IC1f switch faster. The resistors have been made as large as possible, so



the adjustment of the flash frequency is best done by varying the value of C2. The period with the component values given is about 6 seconds, so a LED will flash every 3 seconds.

The current consumption is determined mainly by IC1f, since this is almost working as a linear amplifier. The circuit doesn't draw more than about 1.3 mA in total. The LEDs used should preferably be red 'low-current' or 'high-efficiency' types. The current pulses will then be slightly larger and the efficiency of red LEDs is still better than that of green or yellow ones.

# Battery Saver Switch



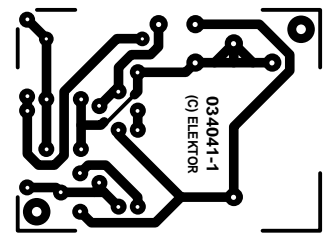
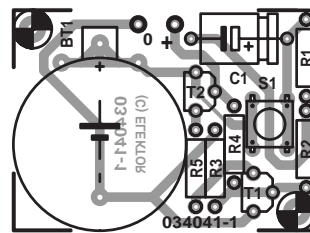
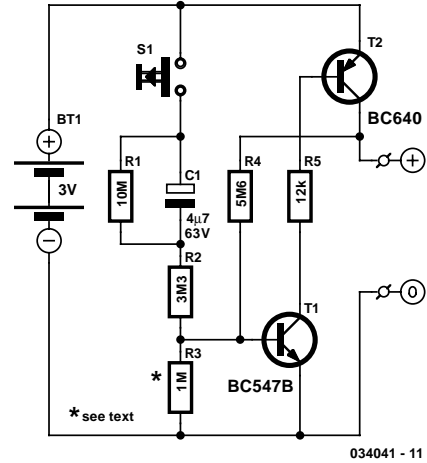
T. Giesberts

There are circuits that consume only a little power or require power occasionally for short periods of time. An example of this last type is the 'FM Remote Control Transmitter'. This type of circuit can easily be powered by a small lithium button cell. The CR2032 immediately comes to mind, since it is a popular battery used in many applications and is widely available.

The circuit shown here was designed to protect these button cells from short circuits or extended current drain. Its primary function is to restrict the time that power can be drawn from the cell after a button has been pressed. In order to keep the construction as easy as possible we have designed two small PCBs that can accommodate the three most popular types of battery holders for the CR2032, including those with solder pins. There are at least two types of battery holder available for single cells: a small and wider version. Their diameters are 22.75 and 27.76 mm respectively. The PCB has been made a bit larger for use with the wider type. When such a battery holder is ordered it is not always clear if it will be a small or wider type, hence the two PCB layouts.

The circuit consists of no more than a simple comparator using two transistors. Because of its simplicity it is likely that the value of R3 has to be adjusted to obtain the required switching characteristics. In one of our prototypes the best value for R3 was 1 MΩ and in another it was 2.2 MΩ. It should be chosen such that the circuit can switch on properly with a supply voltage of about 2 V.

Determining the correct value for R3 is very easy. Using a variable power supply, the voltage should be decreased slowly



## COMPONENTS LIST

### Resistors:

- R1 = 10MΩ
- R2 = 3MΩ
- R3 = 1MΩ\*
- R4 = 5MΩ
- R5 = 12kΩ

### Capacitors:

- C1 = 4μF 63V axial

### Semiconductors:

- T1 = BC547B
- T2 = BC640

### Miscellaneous:

- S1 = 6-mm 'tactile switch', e.g., MCDTS6-5R (Farnell # 312-1033)
- BT1 = 3 V CR2032 with PCB mount holder (dia. 22.75 or 27.76 mm)\*

\* see text

(starting at about 3 V), and C1 should be discharged every time before pressing S1 again. At 4 V or just above the circuit should still switch off (keep pressing S1 and slowly increase the supply voltage). The switching characteristics also depend upon the current gain of both transistors and the voltage drop across the base/emitter junctions. When there is a short circuit the current is limited to a few mA's because in that case R4 won't provide extra bias to T1.

The circuit is so simple that it should be easy to adapt it for your own needs. There is another standard battery holder (also with a diameter of 27.76 mm) that takes two CR2032 cells on top of each other, giving a nominal supply voltage of 6 V. In that case you will have to determine the component values yourself. R1 discharges C1 when the switch is open. This time has been made fairly large on purpose, about 50 seconds! The

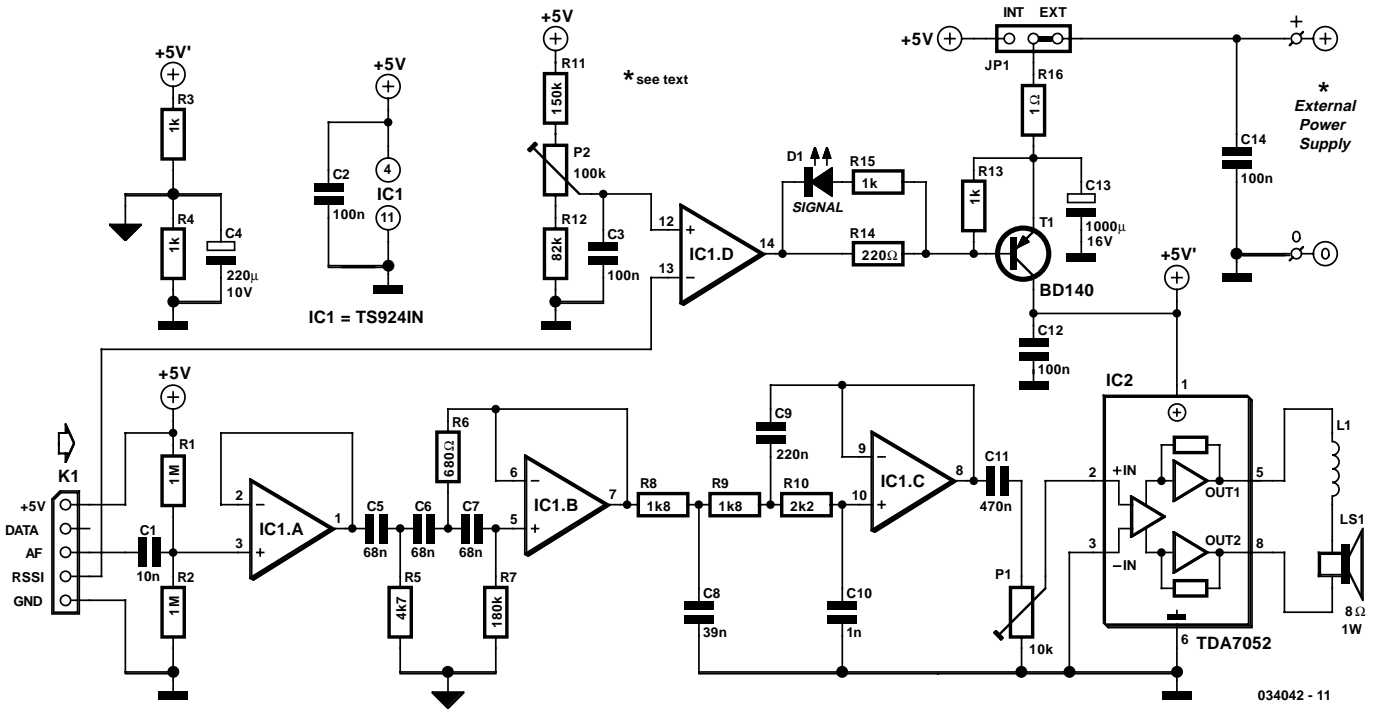
advantage of this circuit is that as the battery voltage becomes lower, T1 stops conducting earlier and hence saves the battery even more. With C1 fully discharged and a fresh battery the on-time is between about 15 to 20 seconds. We have assumed that the circuit will be used in applications where S1 will only be switched on momentarily. When S1 is accidentally

switched on for longer then C1 will prevent the battery from discharging quickly. The current drain will then be below  $0.3 \mu\text{A}$ .

A construction tip: if the circuit is housed in a small case it may be easier to mount the switch on the solder side of the PCB.



# Amplifier with Squelch



034042 - 11

## T. Giesberts

This addition to the receiver part of the 'FM Remote Control' is mainly intended for use as a simple intercom or P.A. system. With a few changes this circuit could possibly also be used with different receiver modules. In the previously mentioned project the decoder and dual D-type flip-flop (IC2 and IC3) are not required for use with this circuit.

The main feature of this circuit is that the supply voltage to the power amplifier is switched on or off automatically. Implementing a mute state this way is very simple and also saves on current consumption (especially important in battery powered applications). Opamp IC1d, which is configured as a comparator, compares the signal strength of the RX5 receiver with a level set by P2. The range of P2 is a little bit larger than the voltage provided by the RSSI output of the receiver module, and are 1.23-2.74 V and 1.27-2.63 V respectively. IC1d switches T1, which then provides a supply voltage to the power amplifier. Potential divider R13/R14 makes T1 switch cleanly and the voltage across R14 is used via D1 to indicate when a signal is received.

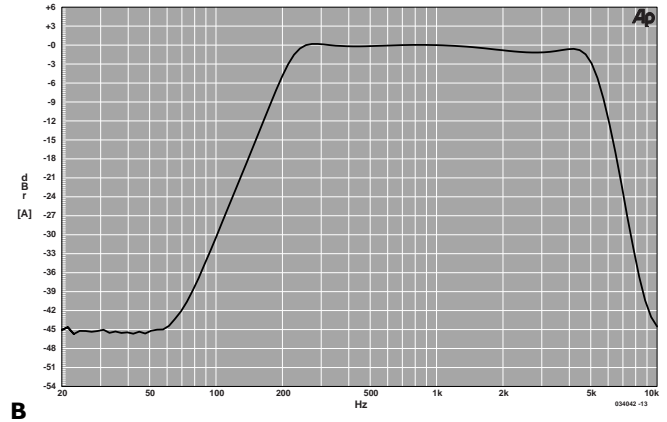
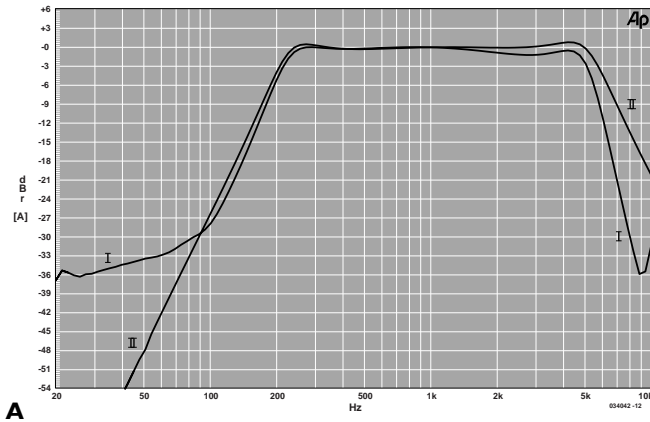
For the power amplifier a small bridge amplifier is used, which comes in an 8-pin DIL package: the TDA7052. This is a neat little chip that requires no external components apart from a volume control (P1). It did appear that this integrated amplifier had a tendency to oscillate when long leads were used to connect the loudspeaker. Air-cored coil L1 (6 turns of 1 mm CuL wire, with an inner diameter of 10 mm) offers reasonable pro-

tection against this, but it is better still to use such a coil in each of the loudspeaker leads.

A variant of the amplifier is the TDA7052A. This has an internal DC controlled volume control, which is controlled with a voltage on pin 4 (not connected here). Although we haven't tested this, this pin can probably be left unconnected, which means that the A-type can be used without having to make any changes.

Connector K1 in the circuit diagram corresponds to K1 in the 'FM remote control'. Apart from the RSSI signal strength output, use is also made of the demodulated LF signal (AF output). It becomes clear that the transmitter and receiver modules of this remote control were not really designed for audio when the distortion of the signal is measured when it has gone through the transmitter and receiver. This turned out to be about 3%. The speech quality is good, due to the heavy use of filters in this circuit. IC1b is used to make a 3<sup>rd</sup> order highpass filter with a turnover frequency just above 200 Hz and round IC1c is a similar lowpass filter with a turnover frequency of about 5.5 kHz. Both filters are Chebyshev types with a ripple of only 1 dB. **Figure A** shows two graphs: I is measured with the transmitter/receiver/amplifier combination, II is just for the amplifier. The filters also reduce any noise introduced by the transmitter or receiver.

As a comparison we have also taken some measurements using a previous generation of the transmitter and receiver modules (the transmitter was a TX2-433-40-5V and the receiver an RX2-433-14-5V, both made by Radiometrix). With



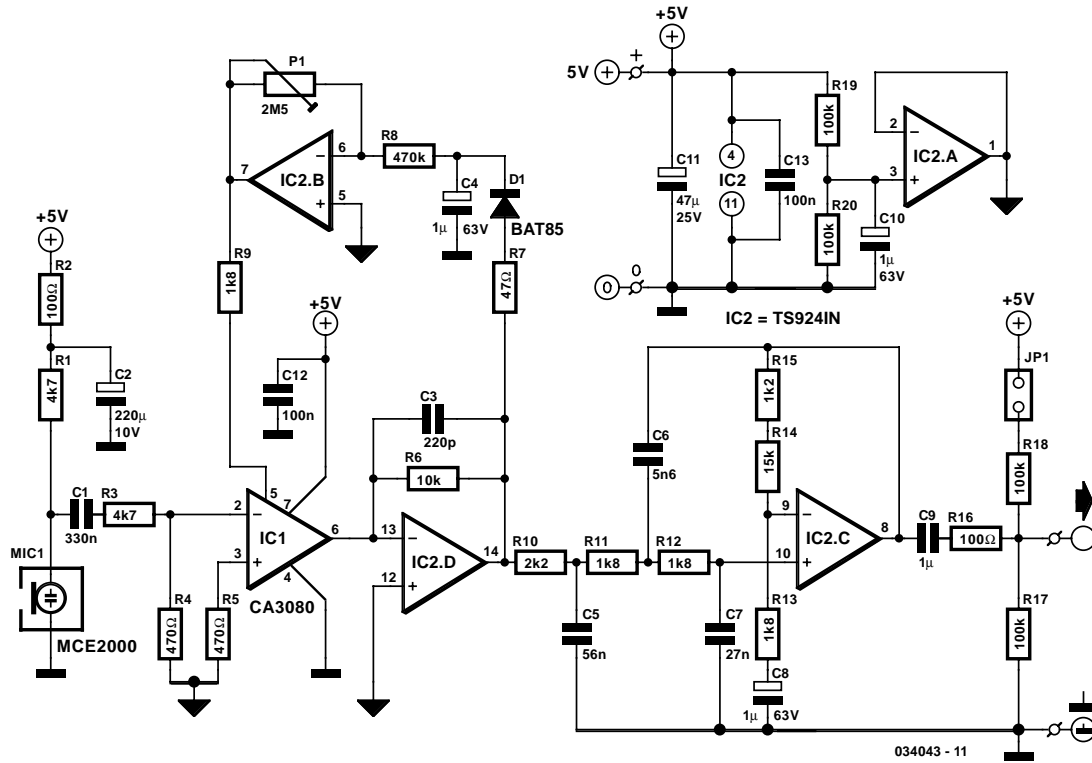
optimum modulation (1.2 V at the transmitter module, 4.5 V supply voltage, compressor set for fixed gain) we measured a distortion of only 0.4%; this is quite a big difference! And with a similar measurement as in figure A-I the noise floor at low frequencies seemed to have moved from  $-36$  dB to  $-45$  dB. The disadvantage is that these modules only have a carrier-detect output and require some extra circuitry to change this into a 'signal strength' output.

The current consumption of this circuit when it fully drives an  $8 \Omega$  speaker is about 0.22 A. The amplifier can then deliver 400 mW into  $8 \Omega$ . The 'FM Remote Control Receiver' is not

capable of delivering such a large current, which is why jumper JP1 gives a choice of using either an internal or external power supply. The supply voltage for the filters and comparator can still be derived from the receiver via K1, but if use is made of an external 5 V source then it makes sense to use that to supply the whole circuit.

Without an audio signal the current consumption is about 33 mA. When the signal strength is below the threshold the circuit goes into a mute state and the current consumption drops to just 4.5 mA.

# Compressor for Electret Microphone



T. Giesberts

The 'FM Remote Control Receiver' has a connector where an analogue output is made available. To make a simple intercom or P.A. system the associated transmitter needs a microphone pre-amplifier that outputs a signal at the correct level. And that is exactly the function of this circuit. Actually, this design is adapted from a circuit published last year ('AM Modulator for Intercom'). A few things have been changed so that it can work with the 5 V supply from the transmitter module.

The OTA (IC1) used here is the single version (CA3080), which has slightly different characteristics from the dual CA3280. The quad opamp is the same rail-to-rail TS924IN, made by ST. The turnover frequency of the filter (3<sup>rd</sup> order 1 dB Chebyshev) has been increased slightly to improve the intelligibility of speech and is now about 5.5 kHz. The filter now amplifies the signal by a factor of 10. In practice it is possible that due to various tolerances and the fact that the opamp is not perfect, the filter characteristic shows some deviation from that required. In our prototype it was necessary to change R15 into 2k7 to straighten the response curve.

The DC current variation at the output of the OTA and the resulting offset variation at the output of current/voltage converter IC2d is such that the gain of IC2d has to be substantially smaller than in the 'old' design. Otherwise the output could easily rise to the supply voltage at low signal levels. The value of R6 has therefore been made smaller by a factor of 10.

This has reduced the gain of the circuit by 20 dB, which is compensated for in the filter.

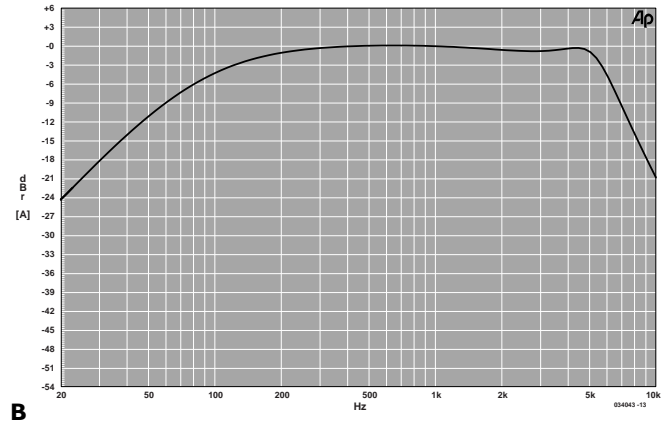
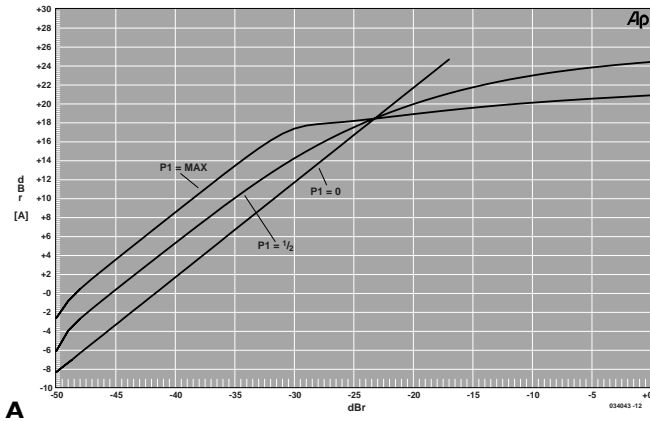
The amplitude of the signal from IC2d is fed back as a control current to the OTA by peak rectifier D1/C3 and inverting amplifier IC2b. R7 limits the loading on IC2d. P1 can be used to adjust the amplifier between a fixed gain and maximum compression.

**Figure A** shows clearly what effect the circuit has. 0 dB corresponds to 100 mV. The maximum gain, with P1 set to maximum compression, is about 48 dB (250 Ω) for small signals. The minimum gain is about 20 dB (10 Ω). The OTA is then slightly overdriven and the distortion becomes several percent! With a fixed gain selected (P1 shorted) the gain is about 42 dB (125 ×).

The middle curve was measured with P1 in its central position. The curve drawn for a fixed gain (the straight line) doesn't finish at the edge of the graph because the end of the line corresponds to the maximum possible output level, which is 25 dB ( $\approx 1.76$  V or  $5 / 2\sqrt{2}$ ).

**Figure B** shows the frequency response. The low turnover frequency is mainly determined by C8 (and to a lesser extent by C1) and is about 120 Hz.

The current consumption is about 7 mA. When the circuit is battery powered we recommend the use of three AA cells, because the circuit still works perfectly at 4.5 V. If you want to use a higher supply voltage (maximum 12 V for the de TS924IN and 30 V for the CA3080, but you should also think of the volt-



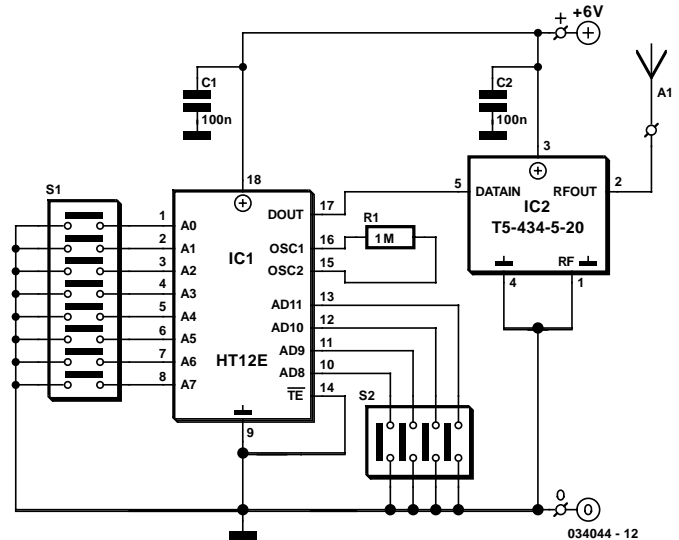
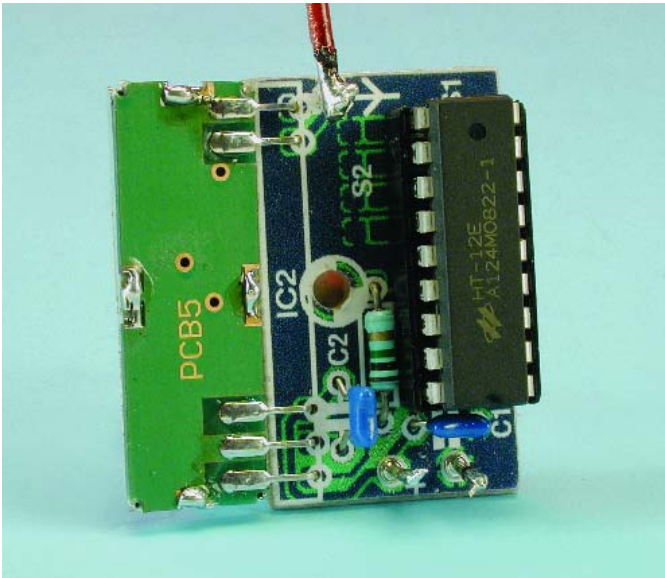
age across the electret microphone!) you have to keep in mind that the maximum current through R9 (which is  $I_{ABC}$ ) is only 2 mA. When we consider a maximum chosen current of 1 mA and the maximum output voltage of IC2b (half the supply voltage, which is 2.5 V), then the value of R9 should be  $(2.5 - 0.7) \text{ V} / 1 \text{ mA} = 1.8 \text{ k}\Omega$ . The value of 0.7 V corresponds to the potential between pin 5 and earth.

For a larger safety margin R9 is calculated with the full supply voltage and a current of 2 mA:  $(5 - 0.7) \text{ V} / 2 \text{ mA} = 2\text{k}2$  (rounded upwards). Of course the regulation will then be different (a little less gain).

This circuit and the transmitter module can therefore be fed from the same 5 V supply. Because the transmitter requires a DC offset at its input, a resistor is connected to +5 V via a jumper, which biases the output to half the supply voltage. With the jumper open R17 functions as a load resistor when the output is not connected, because C9 still has to charge up even without a load.

If you're designing a PCB for this compressor then it makes sense to include the transmitter module as well. The current consumption then increases by about 10 mA.

# FM Remote Control Transmitter



T. Giesberts

This extremely simple transmitter, consisting of only an encoder IC and a 433 MHz licence-exempt TX module, was designed to remotely switch simple appliances on and off. The associated receiver has a relay that can be switched permanently or momentarily and is suitable for various applications (including mains).

For the transmission of a unique signal an encoder is indispensable. For this we've used the HT12E made by Holtek, which we've used previously. Since this is an old favourite there is no need to go into detail and we'll just mention that the oscillator is set to about 3 kHz by R1.

The radio frequency part consists of a standard FM transmitter module from R.F. Solutions, part number T5-434-5-20, which makes the circuit easy to construct and improves its reliability.

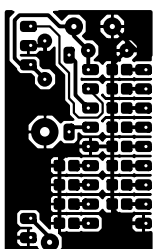
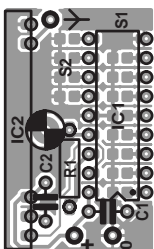
The transmitter module works at a frequency of 433.92 MHz and has a range of about 400 m according to the manufacturer.

The transmitter module has five pins. Apart from 'data in' (pin 5) and the supply (pin 3), there is a common ground for data and supply (pin 4). Last, but not least, is the RF output (pin 2) with its associated ground (pin 1). The best results are achieved when a 15.5 cm length of stiff wire is used for the aerial. If you want to keep the circuit compact and build it into a small handheld case, you should use a helix aerial. This is wound like a coil, which will be about 34 mm long and consists of 17 turns with an inner diameter of 5 mm. We've used 0.9 mm enamelled copper wire (ECW) for this, which keeps its shape reasonably well (you could use the smooth end of a 5 mm drill bit for winding the wire). The aerial has to be mounted as close as possible to pin 2 of the module. The transmitter module is also available from several other sources, although the part number may be slightly different. Examples are the QFMT5-434-5 from Warwick Wireless Limited and the QFMT5-433B5 from OKWelectronics ([www.OKWelectronics.com](http://www.OKWelectronics.com)).

A small PCB has been designed for the transmitter, with room for the transmitter module (IC2) to lie flat along the length of the board. For practical reasons the board is combined with that for the receiver PCB (see 'FM Remote Control Receiver') and needs to be cut from it.

Solder bridges on S1 and S2 are used to set the address and data bits. The current consumption with a supply voltage of 4.5 V (three AA cells for example) is about 9 mA. With a supply voltage of 6 V the current consumption rises to 12.5 mA. These figures apply to the 5 V version of the transmitter module. This is specified for use with a supply between 4.5 and 5.5 V, but has an absolute maximum rating of 10 V, so 6 V won't do it any harm. Furthermore, the 5 V version will still function with a supply of only 2 V, although the range is then much less.

Since the circuit consumes very little current the power can



## COMPONENTS LIST

### Resistors:

R1 = 1MΩ

### Capacitors:

C1, C2 = 100nF ceramic

### Semiconductors:

IC1 = HT12E (Holtek)  
 IC2 = T5-434-5-50 R.F. Solutions  
 (Farnell # 352-4371)

### Miscellaneous:

S1, S2 = solder bridges  
 PCB: see 'FM Remote Control Receiver'

also be provided by button cells. Elsewhere in this issue is a circuit that combines a battery holder for one or two lithium button cells with a miniature automatic switch on a small PCB

(see 'Battery Saver Switch'). That circuit is ideal for use as a power source for the transmitter.

(034044-2)

# LCD module in 4-bit Mode

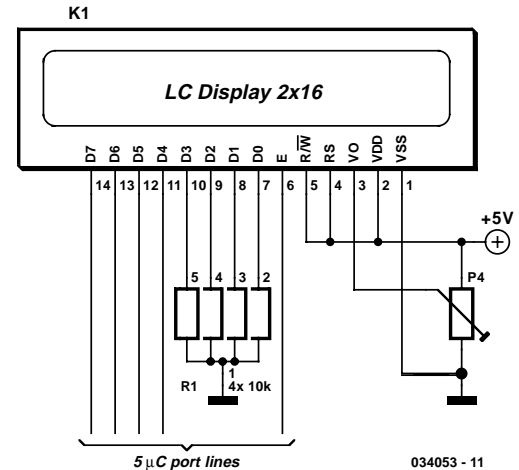
L. Lemmens

In many projects use is made of alphanumeric LCDs that are driven internally by Hitachi's industry-standard HD44780 controller. These displays can be driven either in 4-bit or 8-bit mode. In the first case only the high nibble (D4 to D7) of the display's data bus is used. The four unused connections still deserve some closer attention. The data lines can be used as either inputs or outputs for the display. It is well known that an unloaded output is fine, but that a floating high-impedance input can cause problems. So what should you do with the four unused data lines when the display is used in 4-bit mode?

This question arose when a circuit was submitted to us where D0-D3 were tied directly to GND (the same applies if it was to +5 V) to stop the problem of floating inputs. The LCD module was driven directly by a microcontroller, which was on a development board for testing various programs and I/O functions. There was a switch present for turning off the enable of the display when it wasn't being used, but this could be forgotten during some experiments. When the  $R/\overline{W}$  line of the display is permanently tied to GND (data only goes from the microcontroller to the display) then the remaining lines can safely be connected to the supply (+ve or GND). In this application however, the  $R/\overline{W}$  line was also controlled by the microcontroller.

When the display is initialised correctly then nothing much should go wrong. The data sheet for the HD44780 is not very clear as to what happens with the low nibble during initialisation.

After the power-on reset the display will always be in 8-bit



mode. A simple experiment (see the accompanying circuit) reveals that it is safer to use pull-down resistors to GND for the four low data lines. The data lines of the display are configured as outputs in this circuit ( $R/\overline{W}$  is high) and the 'enable' is toggled (which can still happen, even though it is not the intention to communicate with the display). Note that in practice the RS line will also be driven by an I/O pin, and in our circuit the  $R/\overline{W}$  line as well. All data lines become high and it's not certain if (and if so, for how long) the display can survive with four shorted data lines.

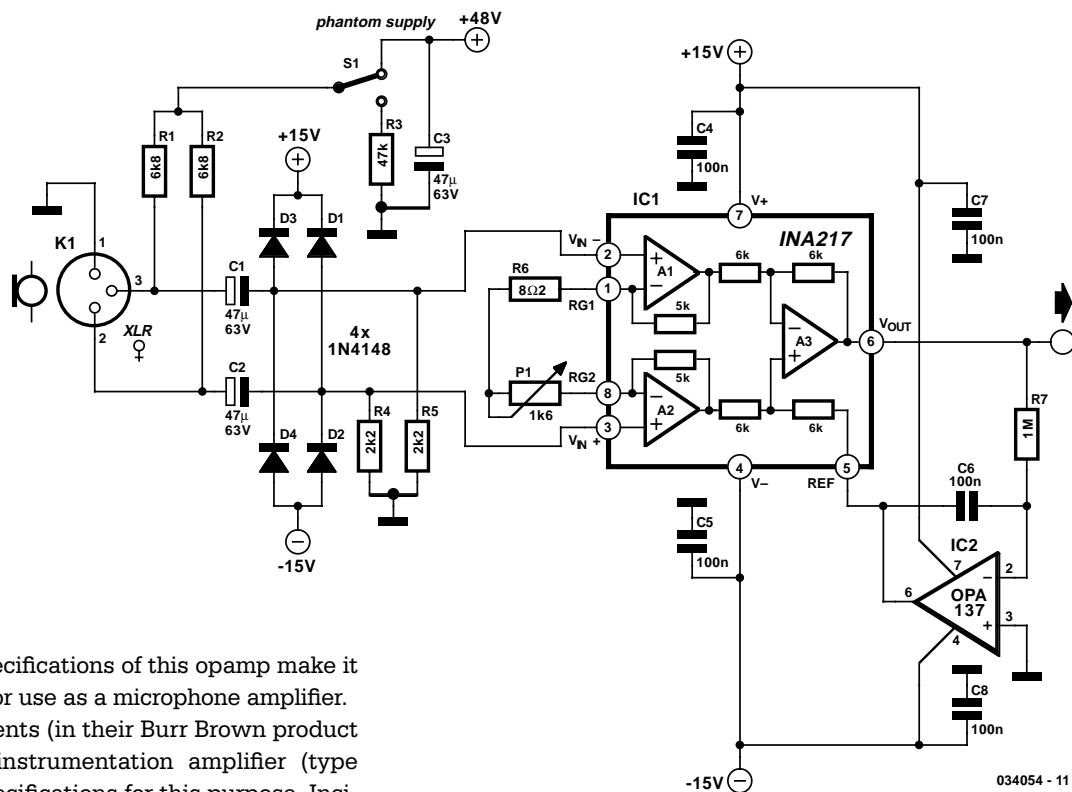
The moral of the story is: in 4-bit mode you should always tie D0-D3 via resistors to ground or positive.

# Balanced Microphone Amplifier

T. Giesberts

A number of years ago (November 1997), we published a design for a stereo microphone preamplifier with balanced inputs and a phantom power supply. The heart of this circuit was a special Analog Devices IC, the SSM2017. Unfortunately, this IC has been discontinued. In its place, the company recommends using the pin-compatible AMP02 from its current product line. However, and again unfortunately, the specifications of this opamp make it considerably less suitable for use as a microphone amplifier. By contrast, Texas Instruments (in their Burr Brown product line) offer an integrated instrumentation amplifier (type 1NA217) that has better specifications for this purpose. Incidentally, this IC is also recommended as a replacement for the SSM2017. It features internal current feedback, which ensures low distortion (THD + noise is 0.004 % at a gain of 100), low input-stage noise (1.3 nV/ $\sqrt{\text{Hz}}$ ) and wide bandwidth (800 kHz at a gain of 100). The supply voltage range is  $\pm 4.5 \text{ V}$  to  $\pm 18 \text{ V}$ . The maximum current consumption of the 1NA217 is  $\pm 12 \text{ mA}$ . The gain is determined by only one resistance, which is the resistance between pins 1 and 8 of the IC.

The circuit shown here is a standard application circuit for this instrumentation amplifier. R1 and R2 provide a separate phantom supply for the microphone connected to the amplifier (this is primarily used with professional equipment). This supply can be enabled or disabled using S1. C1 and C2 prevent the phantom voltage from appearing at the inputs of the amplifier. If a phantom supply is not used, R1 and R2 can be omitted, and it is then better to use MKT types for C1 and C2. Diodes D1–D4 are included to protect the inputs of the 1NA217 against high input voltages (such as may occur when the phantom supply is switched on). R4 and R5 hold the bias voltage of the input stage at ground potential. The gain is made variable by including potentiometer P1 in series with R6. A special reverse log-



taper audio potentiometer is recommended for P1 to allow the volume adjustment to follow a linear dB scale.

The input bias currents (12  $\mu\text{A}$  maximum!) produce an offset voltage across the input resistors (R4 and R5). Depending on the gain, this can lead to a rather large offset voltage at the output (several volts). If you want to avoid using a decoupling capacitor at the output, an active offset compensation circuit provides a solution. In this circuit, a FET-input opamp with a low input offset (an OPA137) is used for this purpose. It acts as an integrator that provides reverse feedback to pin 5, so the DC output level is always held to 0 V. This opamp is not in the audio signal path, so it does not affect signal quality. Naturally, other types of low-offset opamps could also be used for this purpose.

The current consumption of the circuit is primarily determined by the quiescent current of IC1, since the OPA137 consumes only 0.22 mA.

(034054-1)

(from a Texas Instruments application note)



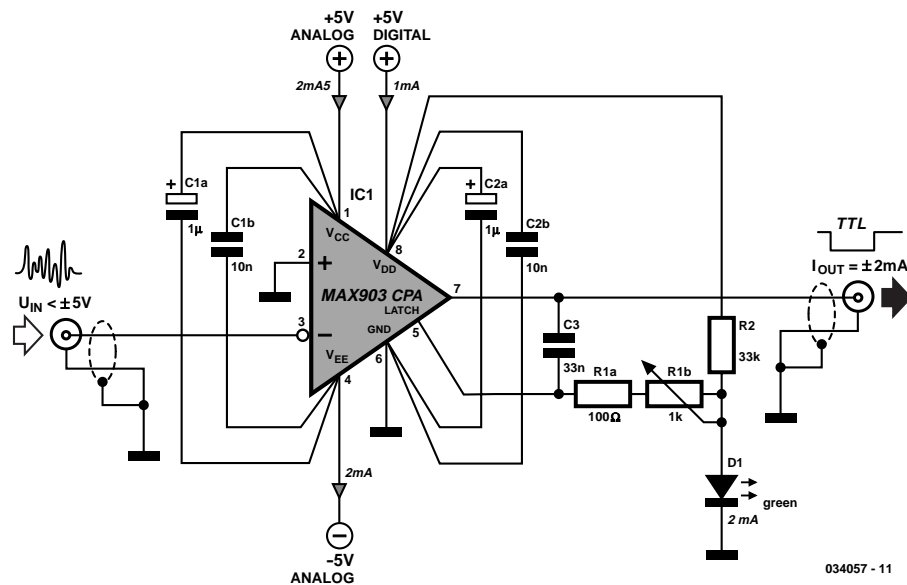
# Spike Detector for Oscilloscopes

K. J. Thiesler

Dynamic flip-flops ignore pulses at their inputs that are shorter than 40 ns or do not have TTL levels. This means that TTL flip-flops are poorly suited to capturing noise pulses having unknown durations and magnitudes. Anyone who has ever tried to observe very short laser pulses (15–25 ns) is familiar with this problem. By contrast, this circuit can detect impulses with widths less than 8 ns and amplitudes between +100 mV and +5 V. The heart of this circuit is formed by a MAX903, a very fast comparator with internal memory. The IC has separate supply pins for its analogue and digital portions. The analogue portion is powered by a symmetrical  $\pm 5\text{-V}$  supply. This allows the detector to also handle input voltages that are negative relative to ground. The internal memory and output stage operate from a single-ended +5-V supply, so the output signal has proper TTL levels.

The MAX903 (IC1) has a special internal memory circuit (latch). The latch either connects the output of the internal comparator directly to the signal output or stores the most recent TTL level and blocks the output of the internal comparator, causing the most recent TTL level appears at the output. This allows short input pulses to be stretched to any desired length. Despite its extremely short switching times, the MAX903 consumes only a modest 18 mW.

In the quiescent state, the voltage on the Latch input (pin 5) is at 1.75 V. This reference voltage is provided by LED D1, which draws its current via R2. In this state the latch is transparent, and a **positive** edge at the input appears will appear as a negative transition on the output after a propagation delay of 8 ns ( $t_{PD}$ ). This only happens if the peak voltage on the input is more positive than ground potential. C1 passes this change in the output voltage level to the Latch input (pin 5). As soon as the voltage on the Latch input drops below 1.4 V, the internal latch switches to the Hold state. In this state, the output is no longer connected to the comparator, and the output remains low for the duration of the latch hold time, regardless of what happens with the input signal. The latch hold time is determined by the time constant of the C3/R1 net-



work; it has an adjustment range of 100–500 ns. Pulses of this length can be readily observed using practically any oscilloscope.

This latch function in this circuit is only triggered if the input signal has a rising edge that crosses the zero-voltage level. The internal latch remains transparent for signals in the range of  $-5\text{ V}$  to  $0\text{ V}$ , so such pulses will not be stretched. If only positive input voltages are anticipated, the negative supply voltage is not necessary and the circuit can be powered from a single +5-V supply.

A fast circuit such as this requires a carefully designed circuit board layout. All connections to the IC must be kept very short. Decoupling capacitors C1 and C2 should preferably be placed immediately adjacent to the supply pins. Pin 3 of the IC can be bent upward and soldered directly to a length of coax or twisted-pair cable (air is still the best insulator). If a coax cable is used, the unbraided screen must not be formed into a long pigtail. It's better to peel back a short length of the screen, wrap a length of bare wire around it and solder it directly to the ground plane. The supply traces for the analogue and digital portions must be well separated from each other, and each supply must be well decoupled, even if only a single supply voltage (+5 V) is used. The preferred solution is to use two independent voltage regulators.

# Model Train Identification Circuit

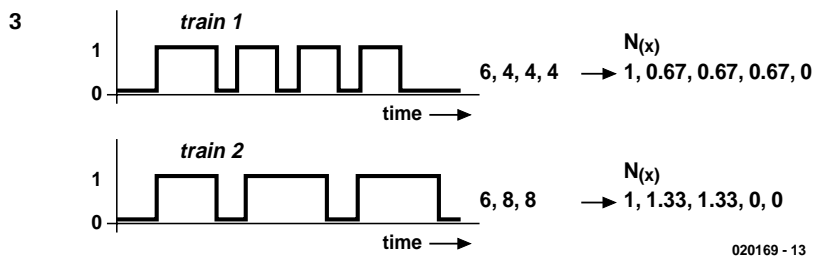
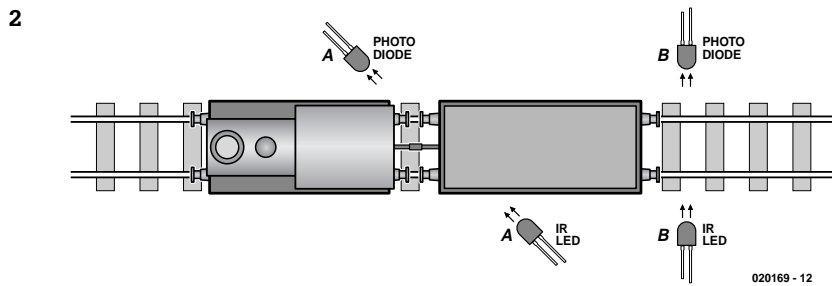
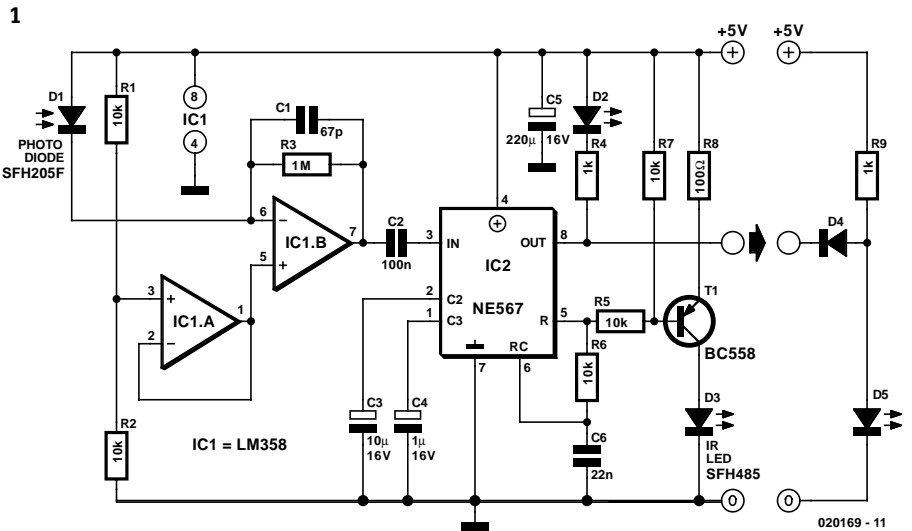
M. Hajer

This circuit is intended to be used to detect trains on a model railway track. It is partially based on designs described in past issues of *Elektor Electronics*. As can be seen from **Figure 1**, it consists of a simple infrared transceiver built around an LM358 and an NE567. With the component values shown in the schematic, it has a working range of around 40 to 50 cm. When several detectors are used, crosstalk among the detectors can be avoided by setting the frequency of the NE567 to a different value for each detector. The LED and series resistor shown at the far right can be added to indicate that a track section is occupied.

The most interesting aspect of this circuit is how it is used. For this, refer to **Figure 2**. If the transmitter and receiver are placed in the 'A' positions, the circuit acts as a 'normal' train detector. The diagonal placement makes it very reliable as a train detector. If instead the transmitter and receiver are placed in the 'B' positions, the circuit can be used for train identification. When a train passes through the gate formed by the diodes in the B positions, the light beam is interrupted at certain intervals, which generates a sort of 'bar code' for the train configuration in question. Using a PC or microcontroller, it is then possible to distinguish various trains from each other using this bar code. This in turn makes it possible to have different trains travel differently on the track, or to handle them differently at the mimic station.

**Figure 3** shows two examples of such bar codes. Here train 1 consists of a locomotive with three short wagons, while train 2 has two long wagons. Measuring the lengths of the individual pulses yields the series '6-4-4-4' for train 1 and the series '6-8-8' for train 2. Train speed information can be eliminated by normalising the values to the first number of the series. This yields a series that indicates how long the wagons are relative to the length of the locomotive, regardless of the speed of the train. If zeroes are added as necessary to achieve a certain train length and the detected values are compared with all previously stored train configurations, surprisingly reliable train identification is possible.

The comparison can be performed in a variety of manners. One way that works well is to determine the quadratic error between the measured value and all stored 'reference trains'.



However, this method is sensitive to differences in train length, so a reference train with one less wagon will not be recognised. The error estimate can be strongly reduced by taking train length into account (by reducing the reference trains to the same length as the measured train).

The numerical value determined using quadratic error estimation is also a measure of the reliability of the estimate. An error threshold can be used to distinguish between trains that have been recognised and trains that are uncertain. In the author's system, the light gate operates at a frequency of approximately 20 Hz, which yields good train recognition.

A final tip: if you place the transmitter and receiver diodes at the level of the floor of the wagons, you can prevent the light from passing through carriage windows to the receiver. This prevents measurement errors, and it allows flat and open wagons to be properly sensed.