leading the way

elektor
electronics

Multi Programmer on USB

Cool audio power: Class T

Have MS Word, will program

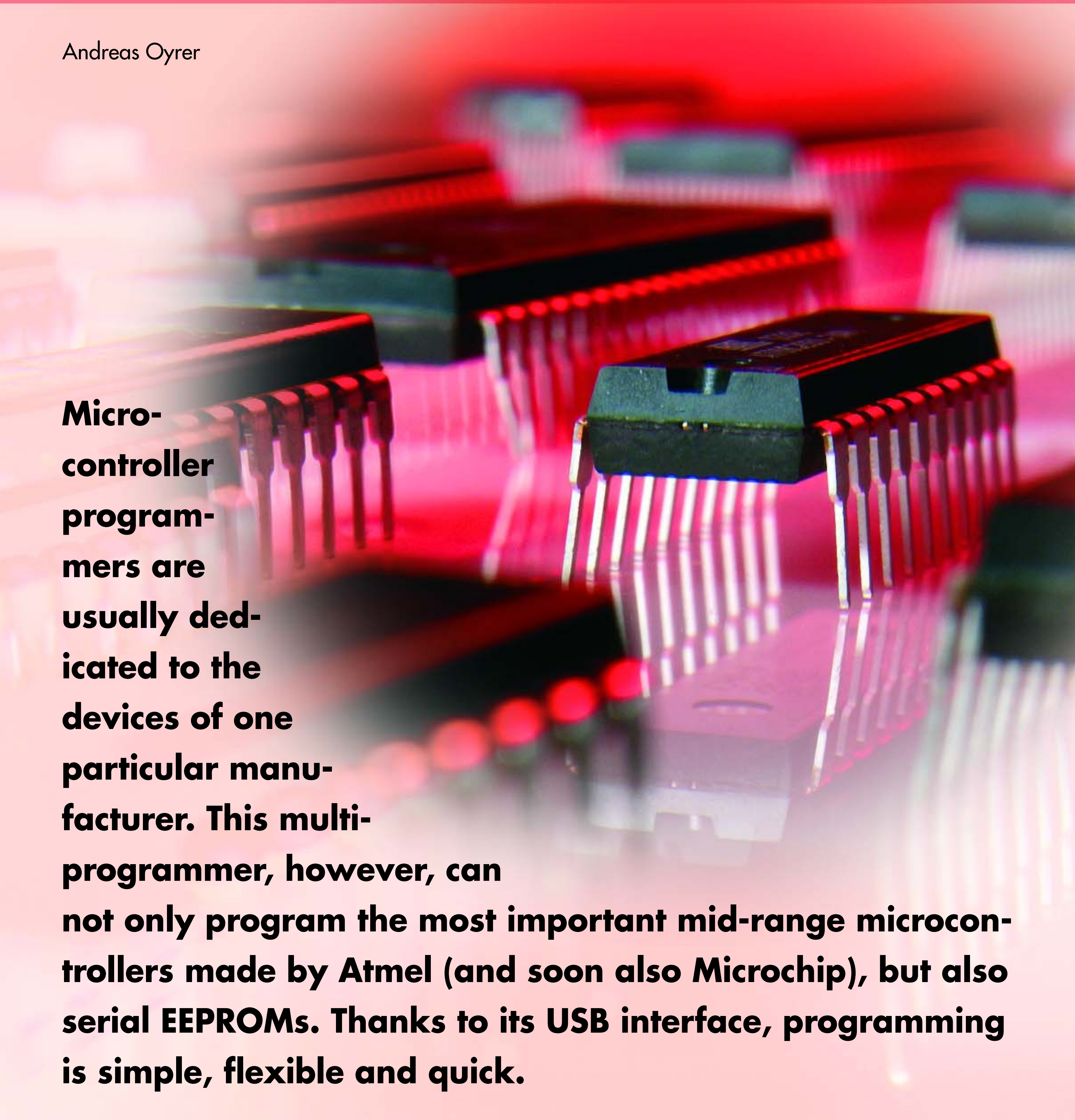
DAB: ten years after

www.elektor-electronics.co.uk
June 2004 • £3.70

Multi Programmer

one size fits all

Andreas Oyrer



Micro-controller programmers are usually dedicated to the devices of one particular manufacturer. This multi-programmer, however, can not only program the most important mid-range microcontrollers made by Atmel (and soon also Microchip), but also serial EEPROMs. Thanks to its USB interface, programming is simple, flexible and quick.

This multi-programmer is tailored to the requirements of the semi-professional user. It is not designed for the entire range of microcontrollers from one particular manufacturer, but rather for general use with standard 8-bit microcontrollers which have limited memory capacity. The hardware is capable of programming microcontrollers from more than one manufacturer (currently Atmel and Microchip) as well as serial EEPROMs. Since these use different programming algorithms and voltages, this is something of an unusual feature.

'Semi-professional' also means that the programmer is a development tool, and so must be suitable for use while debugging. The programmer must be fast, so that the job of getting software to work correctly does not become a chore.

The programmer must also be controllable, which is achieved by using a USB interface. HID (Human Interface Device) compatibility means that the device will work straightforwardly with versions of Windows from 98SE onwards. All that is needed to operate the multi-programmer is a spare USB port on the PC. The device takes its power from the USB port (it is 'bus-powered'), and so no mains supply is needed.

The microcontroller used does not have its own program memory, and so the firmware is downloaded directly from the PC over the USB cable when it is plugged in. Updating the firmware simply requires changing a file on the PC.

It is also possible to store the firmware in an EEPROM on the programmer board, accessed by the microcontroller on power-up. In this case the USB must be used to upload new firmware versions into the EEPROM.

A special feature of this device is that the microcontroller to be programmed does not have to be removed from the target circuit and put in the programming socket. The programmer has two ISP (in-system programming) interfaces available, one for Microchip microcontrollers, one for Atmel devices.

USB Microcontroller

At the heart of the hardware is the TUSB3210 (IC1) from Texas Instruments. This is an 8052-compatible microcontroller with a full speed (12 Mbit/s) USB interface, offering four I/O ports each with 8 port pins, a UART, a watchdog timer and an I^2C interface. The TUSB3210 does not have its own flash memory, and so the firmware must be reloaded every time power is applied. Software is loaded into the 8k-by-8 (8 kbyte) RAM memory by a built-in boot loader: this can be over the USB interface, or, alternatively, the software can be stored in serial EEPROM IC5 (a 24LC64). The EEPROM is connected to the I^2C interface pins SDA and SCL of the TUSB3210: its contents are read whenever the device is reset and copied to the TUSB3210's RAM. If the USB option is used a driver is required on the PC to send the software to the TUSB3210. Whether from the EEPROM or from the PC, once all the firmware has been copied into RAM the boot loader software disconnects from the USB. The program stored in the RAM is run and the device is then reinitialised over the USB.

Programming voltages

IC1 controls all the programming signals and voltages over its 32 I/O pins. In order to generate the programming voltages required by various microcontrollers the 5 V supply from the USB is converted to approximately 13 V using a step-up regulator. Normally the output voltage of the switching regulator would be a constant 12 V, but a diode in the feedback path of IC3 raises the output voltage by the forward voltage drop of the diode: this higher voltage allows PIC microcontrollers to be programmed.

The programming voltages are switched as required using p-channel and n-channel FETs. A voltage of 0 V, 5 V or 12 V can be made to appear on pin 1 or pin 31 of the programming socket. A voltage of 13 V is available for the MCLR signal on ISP connector K3, which is used for programming PICs. Diodes D5 and D7 reduce this voltage back down to 12 V: this lower voltage is used when programming Atmel microcontrollers.

TTL gates with open-collector outputs (type 74LS07) are used to drive the FETs. This allows a voltage of 0 V to appear between gate and source, ensuring that the transistors can be fully turned off. If used directly, the voltage on the I/O port pins could only rise to about 3.3 V, giving a gate-source voltage of 10 V: the transistors would then still conduct.

Some microcontrollers require a programming voltage on the reset or crystal inputs. In the case of the 90S1200, for example, a minimum voltage of $0.85 \cdot V_{CC}$ = 4.25 V (assuming a 5 V supply) is required on the reset input. Since the TUSB3210 runs from a 3.3 V supply it can only deliver a logic high level of 3.3 V; the remaining gates in IC4 are used to produce a high level of over 4 V.

Programming socket

Most devices can be programmed directly in socket IC5. Crystal X2 provides a clock source for Atmel 89Cxx and 89Sxx type microcontrollers. Because of the limited number of I/O pins offered by the TUSB3210 only a

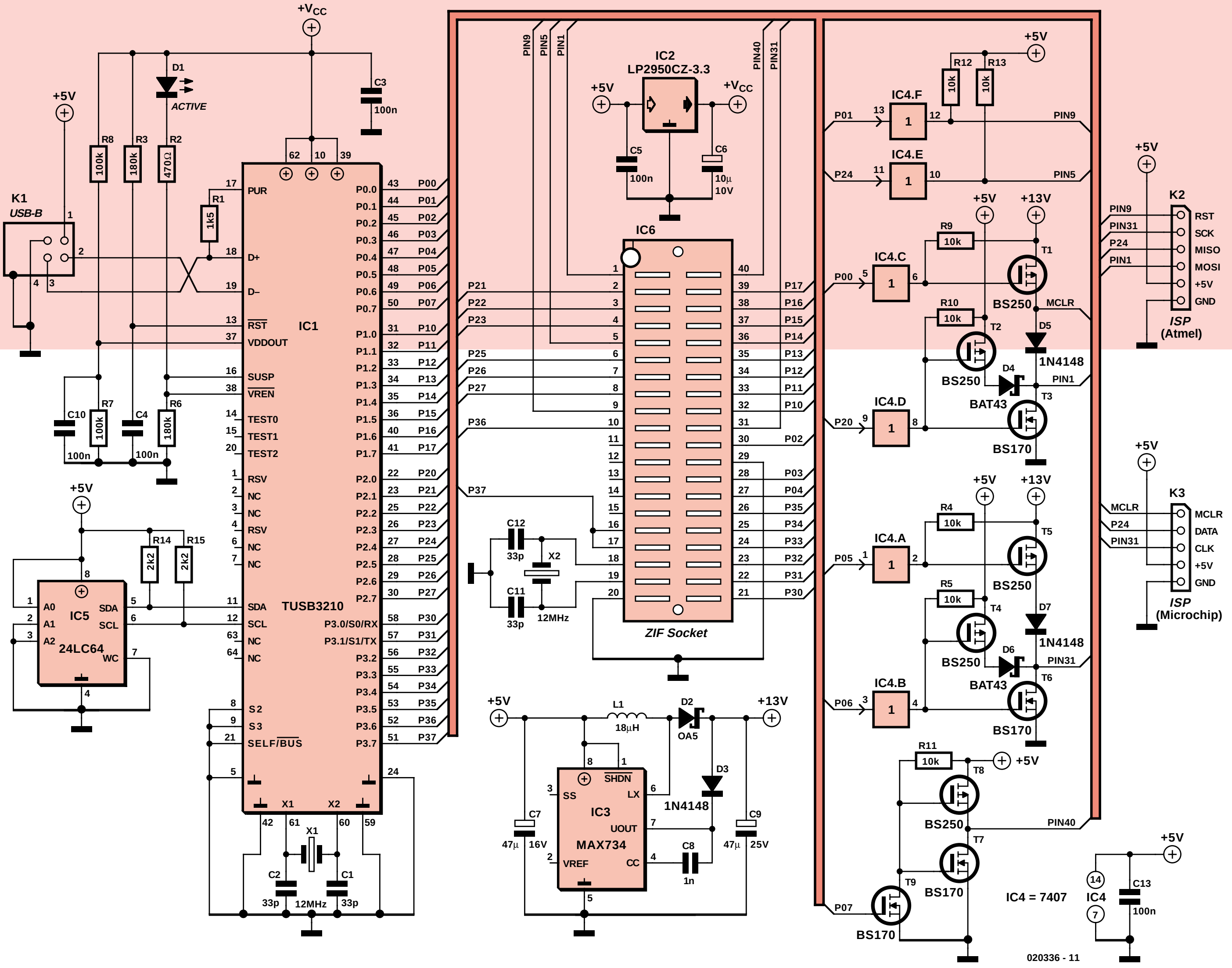


Figure 1. The TUSB3210 includes I²C and USB interfaces, and so the only additional hardware needed is a step-up converter and a few MOSFETs.

certain range of microcontrollers can be programmed. Other (larger) microcontrollers can, however, be programmed using the ISP connectors.

On closer inspection, you will notice that there is no ground connection on pin 10, as would be required, for example, by an 89Cx051. Since the supply current is not particularly high during programming, it is sufficient to provide the ground via port pin P3.6 of IC1. The voltage does indeed rise a little above ground, but remains within reasonable limits.

PC software

The software running on the PC is written in Delphi 7. The menu language can be set (under Setup, as shown in **Figure 2**) to English, French or German. This setting, along with all other settings, is stored in the registry and automatically recalled when the program is next run.

Also under Setup are options to control whether signature bytes are read, and whether the device memory contents are verified after programming.

The device type is set under the Device menu item (see **Figure 3**). There are two sub-menus available here: Socket (i.e., IC5) and ISP Connector (i.e., using connectors K2 and K3). Currently only the Socket option is available. The next choice is between Atmel MCU and serial EEPROM. Under Atmel MCU the available ranges of microcontrollers are 89Cx051, 89C5x, 89Sx and the two AVR microcontrollers, 90S1200 and 90S2313. Of course, functions such as lock bit and fuse bit programming are supported.

On 89C5x microcontrollers only lock bits 1 and 2 can be programmed, since there are not enough port pins available on the TUSB3210 to allow programming of lock bit 3. None of the lock bits can be programmed on 89Sx-type microcontrollers. If a type 90S1200/90S2313 microcontroller is selected, then when configuring the second fuse bit two variants (the RCEN fuse bit and the FSTRT fuse bit) are displayed. If the microcontroller type is detected, the text changes to show the name of the fuse bit supported by the device in question.

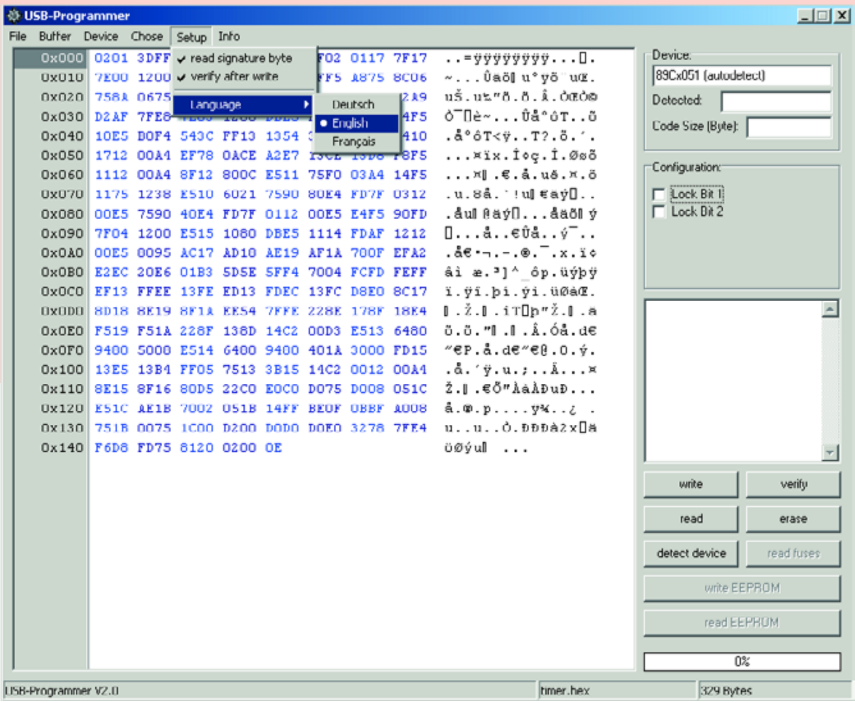
If a serial EEPROM device is selected, there is in some cases more than one device type ending in the same digits (the digits correspond to the size of the memory). Devices in the 24AAxx and 24CxxC series with the same capacity differ, however, in their page size: this is the number of bytes that form one 'row' in the memory that can be programmed in one cycle (approximately 2 ms). The bigger the page size, the quicker the overall programming operation.

All programming functions such as program, verify, erase, read, program EEPROM, read EEPROM, read fuse or lock bits and detect device are directly available either using buttons or under the Action menu (see **Figure 4**). If the microcontroller has been selected as autodetect, then it can be tested using Detect Device. This causes the signature bytes to be read from the device. The bytes are analysed and information including the memory capacity, programming voltage and exact part number are displayed in the upper right-hand corner of the window under Device.

When an action is selected, the signature bytes are first automatically read out of the microcontroller before the action is carried out. The signature byte test can be disabled by deactivating the Read signature bytes option in the Setup menu. This might be necessary if a fault in the microcontroller makes it impossible to read out the signature bytes.

The Read action reads the entire memory contents out of the device. The number of bytes to be read is determined from the information in the signature bytes or by the digits at the end of the part number in the case of a serial EEPROM. If, in the case of a microcontroller, the signature byte has not been read, then the maximum possible memory size for a device in the selected series is used. For example, if the 89Cx051 series is selected, then 4 kbyte will be used, since this is the memory capacity of the biggest device in this series, the 89C4051.

Under the Buffer menu you can choose whether the data in the buffer can be altered using the hex editor (Buffer editable). You can also select whether data in the buffer is synchronised with the data in the previously-opened file before any write or verify action is started (Update buffer from file).



```
0x000  0201 3DFF                     F02 0117 7F17   ..=ÿÿÿÿÿÿÿ...□.
0x010  7E00 1200                     FF5 A875 8C06   ~...Ûaö▯ u°ÿõ¨uŒ.
0x020  758A 0675                         2A9         uš.u‰"ŏ.ŏ.Â.ÒŒÒ®
0x030  D2AF 7FE8                         4F5         Ò¯□è~...Ûå°ôT..õ
0x040  10E5 D0F4 543C FF13 1354 3         410        .å°ôT<ÿ..T?.ö.´.
0x050  1712 00A4 EF78 0ACE A2E7 13CE 13D8 F8F5       ...¤ïx.Î¢ç.Î.Øøõ
0x060  1112 00A4 8F12 800C E511 75F0 03A4 14F5       ...¤▯.€.å.uð.¤.õ
0x070  1175 1238 E510 6021 7590 80E4 FD7F 0312       .u.8å.`!u▯€äý□..
0x080  00E5 7590 40E4 FD7F 0112 00E5 E4F5 90FD       .åu▯@äý□...åäõ▯ý
0x090  7F04 1200 E515 1080 DBE5 1114 FDAF 1212       □...å..€Ûå..ý¯..
0x0A0  00E5 0095 AC17 AD10 AE19 AF1A 700F EFA2       .å€•¬.-.®.¯.x.ï¢
0x0B0  E2EC 20E6 01B3 5D5E 5FF4 7004 FCFD FEFF       âì æ.³]^_ôp.üýþÿ
0x0C0  EF13 FFEE 13FE ED13 FDEC 13FC D8E0 8C17       ï.ÿî.þí.ýì.üØàŒ.
0x0D0  8D18 8E19 8F1A EE54 7FFE 228E 178F 18E4       ▯.Ž.▯.îT□þ"Ž.▯.ä
0x0E0  F519 F51A 228F 138D 14C2 00D3 E513 6480       õ.õ."▯.▯.Â.Óå.d€
0x0F0  9400 5000 E514 6400 9400 401A 3000 FD15       ”€P.å.d€”€@.0.ý.
0x100  13E5 13B4 FF05 7513 3B15 14C2 0012 00A4       .å.´ÿ.u.;..Â...¤
0x110  8E15 8F16 80D5 22C0 E0C0 D075 D008 051C       Ž.▯.€Õ"ÀàÀÐuÐ...
0x120  E51C AE1B 7002 051B 14FF BE0F 0BBF A008       å.®.p....ÿ¾..¿ .
0x130  751B 0075 1C00 D200 D0D0 D0E0 3278 7FE4       u..u..Ò.ÐÐÐà2x□ä
0x140  F6D8 FD75 8120 0200 0E                        öØýu▯ ...
```

Device: 89Cx051 (autodetect)

Detected:

Code Size (Byte):

Configuration:

□ Lock Bit 1

□ Lock Bit 2

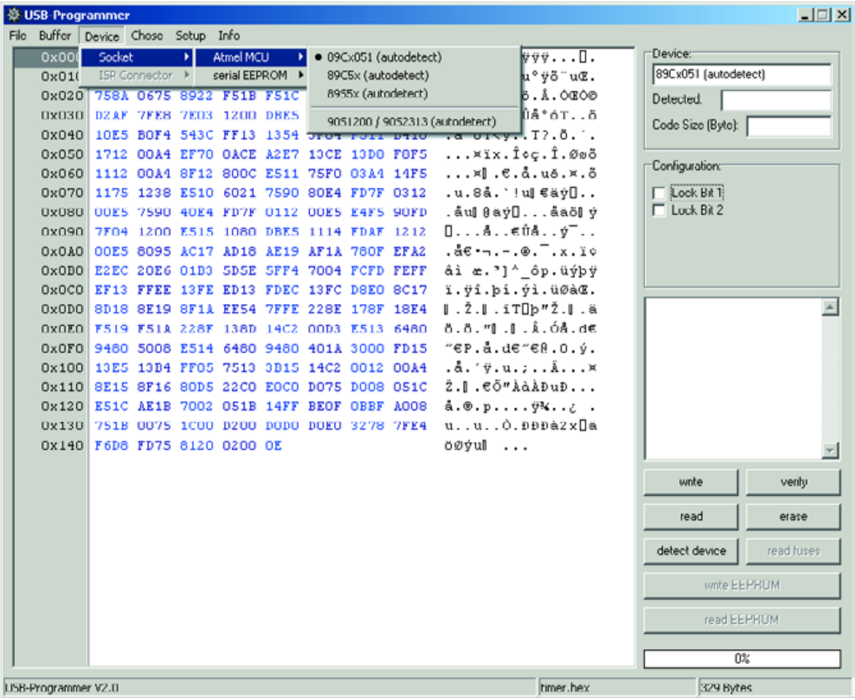
0%

USB-Programmer V2.0 | timer.hex | 329 Bytes

Figure 2. The setup menu

Protocol

A protocol is of course required to ensure that data communications between the PC and the programmer are kept in step. The first byte from the PC to the programmer contains information on the selected microcontroller or memory: the value 1 specifies the 89Cx051 series, the value 2 the 89C5x and 89Sx series. The second byte gives the selected action: 1 to read the signature bytes, 2 to erase and so on. Succeeding bytes contain further information, for example the programming voltage for an 89C5x microcontroller or the page size for a serial EEPROM. When programming, an extra byte is



```
0x00                                                 ÿÿÿ...□.
0x01                                                 u°ÿõ¨uŒ.
0x020  758A 0675 8922 F51B F51C                      õ.Å.ÒŒÒ®
0x030  D2AF 7FE8 7E03 1200 DBE5                      Ûå°ôT..õ
0x040  10E5 B0F4 543C FF13 1354                      .T?.ö.´.
0x050  1712 00A4 EF70 0ACE A2E7 13CE 13D0 F0F5       ...¤ïx.Î¢ç.Î.Øøõ
0x060  1112 00A4 8F12 800C E511 75F0 03A4 14F5       ...¤▯.€.å.uð.¤.õ
0x070  1175 1238 E510 6021 7590 80E4 FD7F 0312       .u.8å.`!u▯€äý□..
0x080  00E5 7590 40E4 FD7F 0112 00E5 E4F5 90FD       .åu▯@äý□...åäõ▯ý
0x090  7F04 1200 E515 1080 DBE5 1114 FDAF 1212       □...å..€Ûå..ý¯..
0x0A0  00E5 8095 AC17 AD18 AE19 AF1A 780F EFA2       .å€•¬.-.®.¯.x.ï¢
0x0B0  E2EC 20E6 01D3 5D5E 5FF4 7004 FCFD FEFF       âì æ.³]^_ôp.üýþÿ
0x0C0  EF13 FFEE 13FE ED13 FDEC 13FC D8E0 8C17       ï.ÿî.þí.ýì.üØàŒ.
0x0D0  8D18 8E19 8F1A EE54 7FFE 228E 178F 18E4       ▯.Ž.▯.îT□þ"Ž.▯.ä
0x0E0  F519 F51A 228F 138D 14C2 00D3 E513 6480       õ.õ."▯.▯.Â.Óå.d€
0x0F0  9480 5008 E514 6480 9480 401A 3000 FD15       ”€P.å.d€”€@.0.ý.
0x100  13E5 13D4 FF05 7513 3D15 14C2 0012 00A4       .å.´ÿ.u.;..Â...¤
0x110  8E15 8F16 80D5 22C0 E0C0 D075 D008 051C       Ž.▯.€Õ"ÀàÀÐuÐ...
0x120  E51C AE1B 7002 051B 14FF BE0F 0BBF A008       å.®.p....ÿ¾..¿ .
0x130  751B 0075 1C00 D200 D0D0 D0E0 3278 7FE4       u..u..Ò.ÐÐÐà2x□a
0x140  F6D8 FD75 8120 0200 0E                        öØýu▯ ...
```

Device: 89Cx051 (autodetect)

Detected:

Code Size (Byte):

Configuration:

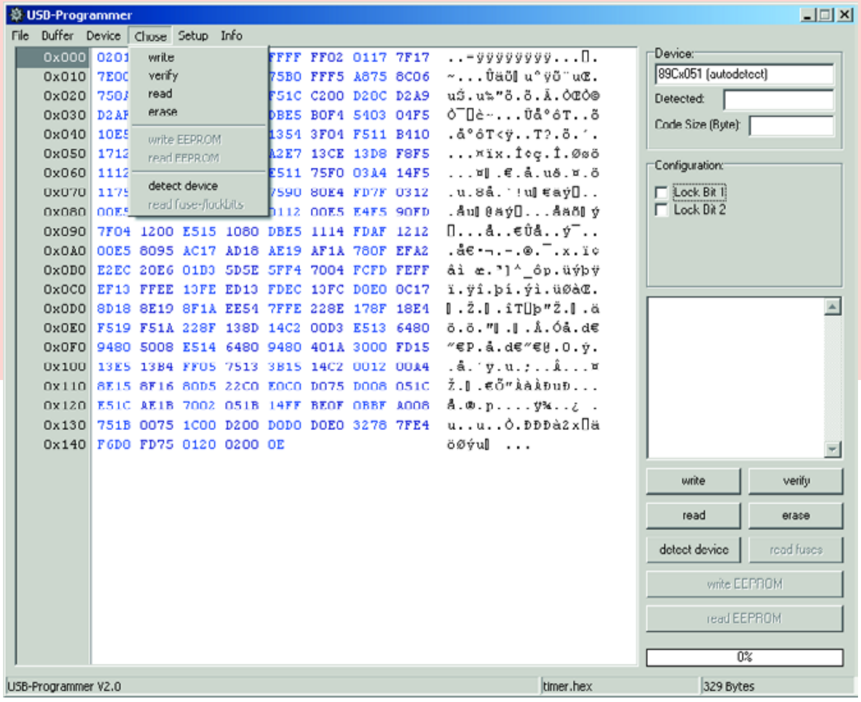
□ Lock Bit 1

□ Lock Bit 2

0%

USB-Programmer V2.0 | timer.hex | 329 Bytes

Figure 3. Device selection



```
0x000 0201                 FFFF FF02 0117 7F17   ..-ÿÿÿÿÿÿÿÿ...▯.
0x010 7E0C                 75B0 FFF5 A875 8C06   ~...ÛäÕ▯ u^ÿÕ¨uŒ.
0x020 750A                 F51C C200 D20C D2A9   uŠ.u‰"õ.õ.Â.ÒŒÒ©
0x030 D2AF                 DBE5 B0F4 5403 04F5   Ò¯▯è-...Ûå°ôT..õ
0x040 10E5                 1354 3F04 F511 B410   .å°ôT<ÿ..T?.õ.´.
0x050 1712                 A2E7 13CE 13D8 F8F5   ...¤ïx.Î¢ç.Î.Øøõ
0x060 1112                 E511 75F0 03A4 14F5   ...¤▯.€.å.uð.¤.õ
0x070 1175                 7590 80E4 FD7F 0312   .u.8å.`!u▯€äý▯..
0x080 00E5                 0112 00E5 E4F5 90FD   .Åu▯@äý▯...ÅäÕ▯ý
0x090 7F04 1200 E515 1080 DBE5 1114 FDAF 1212   ▯...å..€Ûå..ý¯..
0x0A0 00E5 8095 AC17 AD18 AE19 AF1A 780F EFA2   .å€•¬.-.®.¯.x.ï¢
0x0B0 E2EC 20E6 01D3 5D5E 5FF4 7004 FCFD FEFF   âì æ.Ó]^_ôp.üýþÿ
0x0C0 EF13 FFEE 13FE ED13 FDEC 13FC D0E0 0C17   ï.ÿî.þí.ýì.üÐàŒ.
0x0D0 8D18 8E19 8F1A EE54 7FFE 228E 178F 18E4   ▯.Ž.▯.îT▯þ"Ž.▯.ä
0x0E0 F519 F51A 228F 138D 14C2 00D3 E513 6480   õ.õ."▯.▯.Â.Óå.d€
0x0F0 9480 5008 E514 6480 9480 401A 3000 FD15   ”€P.å.d€”€@.0.ý.
0x100 13E5 13B4 FF05 7513 3B15 14C2 0012 00A4   .å.´ÿ.u.;..Â...¤
0x110 8E15 8F16 80D5 22C0 E0C0 D075 D008 051C   Ž.▯.€Õ"ÀàÀÐuÐ...
0x120 E51C AE1B 7002 051B 14FF BE0F 0BBF A008   å.®.p....ÿ¾..¿ .
0x130 751B 0075 1C00 D200 D0D0 D0E0 3278 7FE4   u..u..Ò.ÐÐÐà2x▯ä
0x140 F6D0 FD75 0120 0200 0E                     öÐýu▯  ...
```

Device:
89Cx051 (autodetect)
Detected:
Code Size (Byte):
Configuration:
□ Lock Bit 1
□ Lock Bit 2
0%
USB-Programmer V2.0 | timer.hex | 329 Bytes

Figure 4. Using 'Detect Device' the signature bytes and other data can be displayed in the upper right-hand corner of the window.

used which gives the number of bytes in each packet. A further byte indicates whether the data packet sent is the last one (byte is zero) or whether further packets follow (byte is 1). When reading, a count of the number of bytes to be read is sent to the programmer. Generally the size is specified in kilobytes or kilobits. The software in the programmer can deduce how to interpret the value from the first bytes that were sent.

After each action the programmer sends a number of bytes back to the PC to indicate that it is ready for more data to be sent or to carry out the next action.

Programmer software

The software in the programmer was written using the Keil µVision2 C compiler. The main routine first disables the watchdog timer and then takes all pins on the programming socket to 0 V using the function *ResetProgrammer()*. Next the USB registers are set up.

When the boot loader software in IC1 completes, it disconnects from the USB by taking output pin PUR low so that R1 no longer pulls up to 3.3 V. The downloaded software must set the SDW bit in the MCNFG register to reactivate this output: the programmer will then reappear on the bus.

The PC host then sends a number of SETUP tokens to identify the device and configure its USB interface. These tokens are processed by endpoint 0. The data transferred includes the unique address for the device, which is subsequently used to communicate with the programmer. Various descriptors are also transmitted to the host during the setup phase, providing information about the device and its functions and characteristics. These include the report descriptor, which in this case identifies the programmer as an HID-compatible device.

Once all the descriptors have been sent to the PC, initialisation of the USB interface on the device is complete and it is ready to operate. The direction of data transfer is specified by a token. If the TUSB3210 detects an IN token, then data, such as status information or a data packet, is to be sent from the programmer to the host. If an OUT token is received, then the data packet is unpacked by the programmer and the payload programmed into the device (assuming that that is the selected action).

HID

The advantage of initialising as an HID-compatible device is that no special Windows driver is required to communicate data between the PC and the programmer. Windows versions from 98SE onwards support this standard.

Under the HID standard, data is exchanged in so-called reports. During USB enumeration the PC provides a number of descriptors. The device descriptor includes information such as the Vendor ID (VID), Product ID (PID), and the USB version supported by the connected device.

The configuration descriptor includes information on the current consumption of the hardware and the number of available endpoints. The report descriptor gives the size and number of reports to be exchanged between the PC and the programmer. It specifies how many bytes are to be sent or received and the function of the attached device (mouse, keyboard, joystick, memory stick etc.). More detailed information on USB and HID can be found on the USB homepage at

www.usb.org/home.

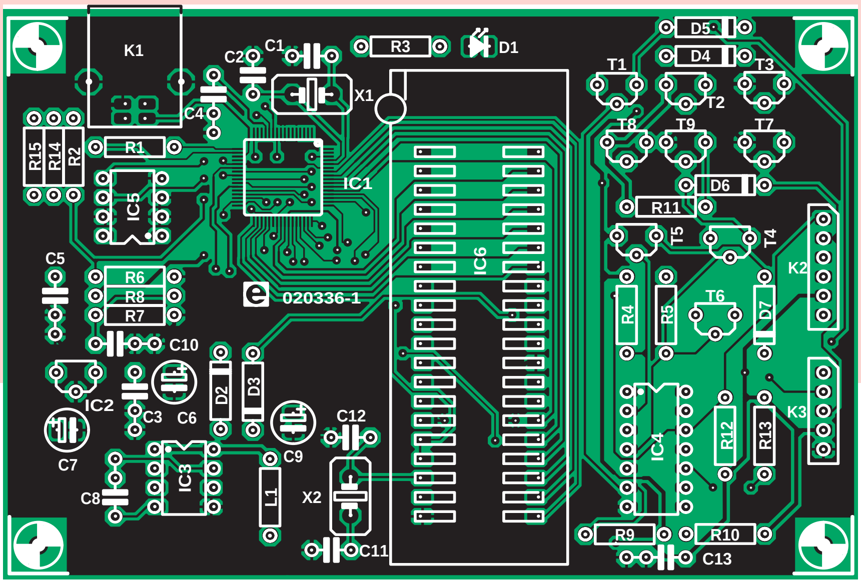


Figure 5. Component mounting plan for the double-sided printed circuit board.

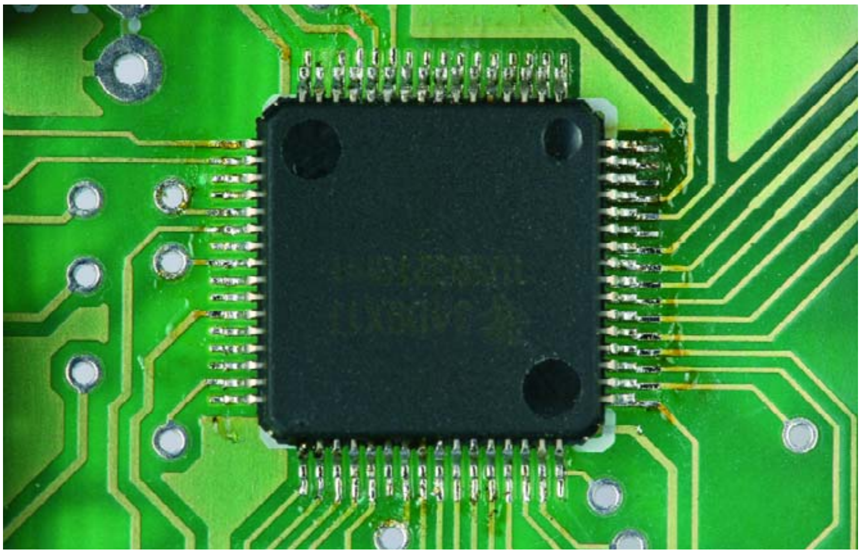


Figure 6. Tiny and tricky to solder: the USB microcontroller comes in a 64-lead SPFP package.

Programming routines

The data received is decoded by the routine *DecodeProgrammerData()* in the file Prog.c. The first byte of the 64-byte report contains the code for the selected microcontroller, while the second gives the desired action. These values are used to call one of a number of different programming algorithms for different devices, as given in the microcontroller data sheets.

Each device series has its own power-up routine which applies power to the correct pins and sets the programming signals used to defined levels. Once an action has been successfully completed, the routine to reset the programmer is called, which sets all the signals on the programming socket back to 0 V. Since in general it is desired to program more than the 64 bytes contained in one report, the PC must send a further data packet to the programmer as soon as the previous one has been processed. The programmer sends a defined message to the software running on the PC to notify it that the next packet can be sent.

The PC then prepares the next report packet and sends it to the hardware. The last packet to the USB programmer contains a zero byte. When the device is read, data is also transferred in reports of 64 bytes each, where the first byte gives the number of valid bytes in the packet.

The above description of the programmer software can give only a broad overview of its operation. More detail can be found in the thoroughly-commented and clearly-structured software itself.

Construction and operation

Populating the printed circuit board would be child's play were it not for the tiny TUSB3210 in an S-PFP-G64 package with many fine leads. Soldering such SMD ICs requires not just a steady hand but also nerves of steel and a good deal of confidence. After fixing the IC in place with a drop of glue, you must take the soldering iron and do what you would normally try to avoid at all costs: rather than soldering the IC's leads to their corresponding pads, solder all the leads together. This should be done as quickly as possible, so the device does not get too hot. When this big short-circuit and the IC have cooled down, lay some unused desoldering wick across the pins and use it (and not a solder pump!) to suck away the excess solder. Again, take care not to get the IC too hot. Finally, check with a magnifying glass under a good light, and using a multimeter, that all the pins are well soldered and no longer shorted to their neighbours. Once the TUSB3210 has been correctly soldered to the printed circuit board, the remaining construction is relatively straightforward. All the ICs (except for the tiny voltage regulator) are provided with sockets. Even the zero insertion force socket should be fitted in a

Devices currently supported

The programmer firmware can easily be updated to the latest version at any time: you simply need to write the new firmware into the EEPROM and, if necessary, replace the software on the PC.

Currently the firmware is capable of programming the microcontrollers and EEPROMs listed below. An update is expected shortly that will support PIC microcontrollers and ATmega devices.

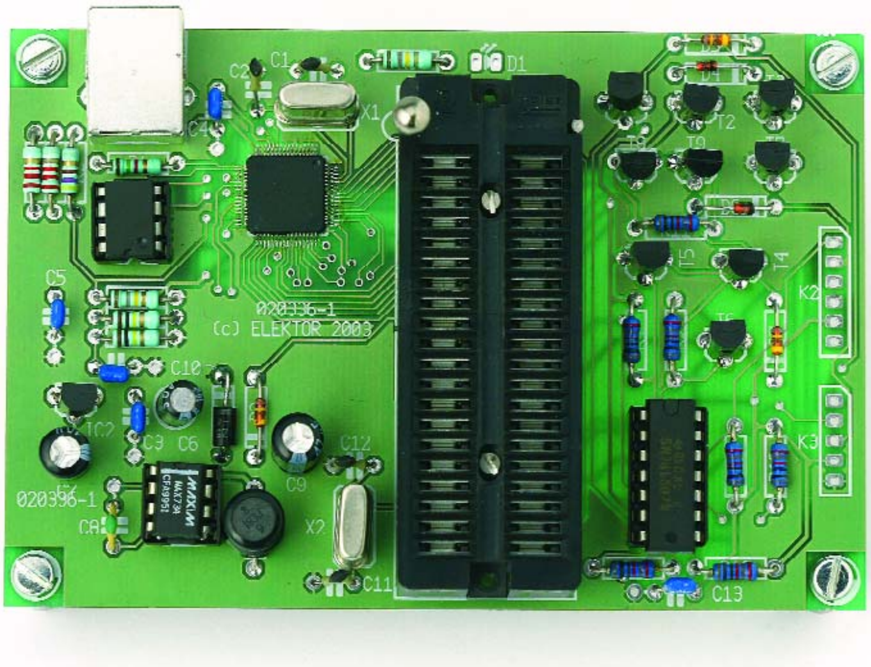
This, as well as all future updates, will be included with the PC software available for free download from the Elektor Electronics website under product number 020336-11, see month of publication.

Atmel microcontrollers:

89C1051, 89C2051, 89C4051, 90S1200, 90S2313, 89C51, 89C52, 89C55, 89LV51, 89LV52, 89LV55, 89S53, 89S8252

EEPROMs:

24xx00, 24xx01, 24xx02, 24xx04, 24xx08, 24xx16, 24xx32, 24xx64, 24xx128, 24xx256, 24xx512



socket rather than being directly soldered to the board.

When construction is complete and you have inspected the board, you can carry out the first test. If a ready-programmed EEPROM is available, no Windows driver will be required. If the programmer is now connected to the USB, it should appear in the Device Manager as an HID-compatible device. You are now ready to program your first microcontroller.

If no serial EEPROM is fitted, the firmware must be downloaded over the USB. The TUSB3210 boot loader registers itself, Windows recognises the new device and will now need the Texas Instruments device driver. This driver (called the *TI Apploader Driver*) is not provided as part of the *Elektor Electronics* disk or download, but can be obtained for free from the TI website at www.ti.com. Select the directory with the file TUSB3210.inf and install the driver and then the file Aploader.sys from the same directory.

Finally you will be asked for the directory containing the firmware: enter the path to the file TUSB3210.bin. This will automatically be copied into the directory /System32/drivers, along with the file Aploader.sys. If the programmer is now reconnected, the driver will send the firmware from the file /System32/drivers/TUSB3210.bin; after a brief delay the code will start executing on the programmer. The programmer will now be re-enumerated as an HID device.

(020336-1)

COMPONENTS LIST

Resistors:

R1 = 1kΩ5
R2 = 470Ω
R3,R6 = 180kΩ
R4,R5,R9-R13 = 10kΩ
R7,R8 = 100kΩ
R14,R15 = 2kΩ2

Capacitors:

C1,C2,C12,C13 = 33pF
C3,C4,C5,C10,C13 = 100nF
C6 = 10µF 16V radial
C7,C9 = 47µF 16V radial
C8 = 1nF

Semiconductors:

D1 = LED, red
D2 = 0A5 or 1N5817 (Farnell # 573-097)
D3,D5,D7 = 1N4148
D4,D6 = BAT43
IC1 = TUSB3210PM
IC2 = LP2950CZ-3.3 or LE33CZ (Farnell # 302-4568)
IC3 = MAX734CP
IC4 = 74LS04
IC5 = 24LC64
IC6 = 40-way ZIF socket with wide slots
T1,T2,T4,T5,T8 = BS250
T3,T6,T7,T9 = BS170

Miscellaneous:

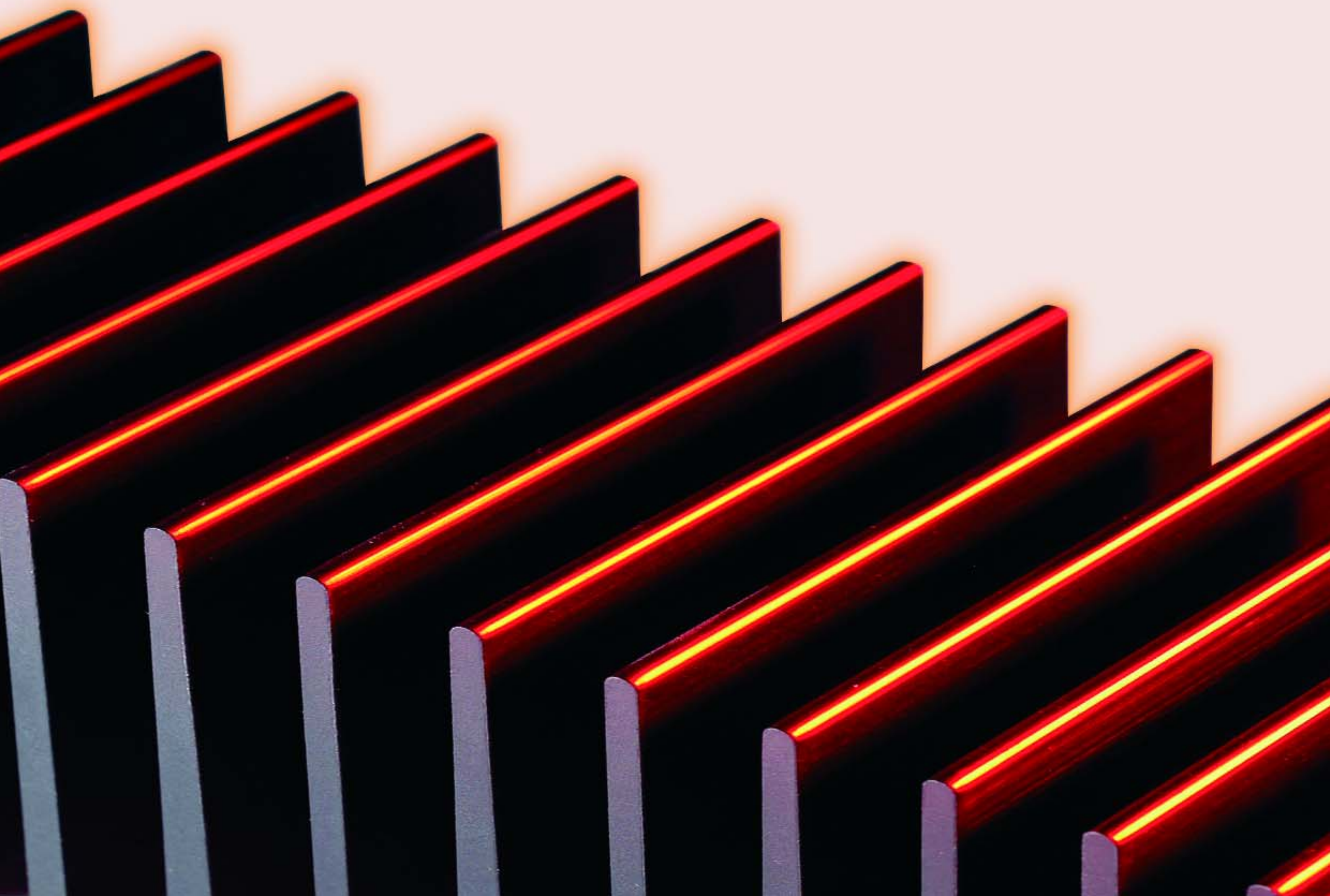
K1 = USB 'B' connector, angled, PCB mount
K2 = 6-way pinheader
K3 = 5-way pinheader
L1 = 18µH miniature choke
X1,X2 = 12MHz quartz crystal
PCB, available from **The PCBShop**
Disk, PC software and firmware / source code files, order code **020336-11** or Free Download

Harry Baggen

THAT'S CLASS...

audio amplifiers from A to T

The final amplifier is the power source of every audio installation. Its job is to convert a small alternating voltage into a powerful signal suitable for driving loudspeakers, with as little distortion as possible. During the years since the invention of electronic audio systems, designers have come up with various approaches to this problem. It all started with Class A...



A little more noise, a lot more power

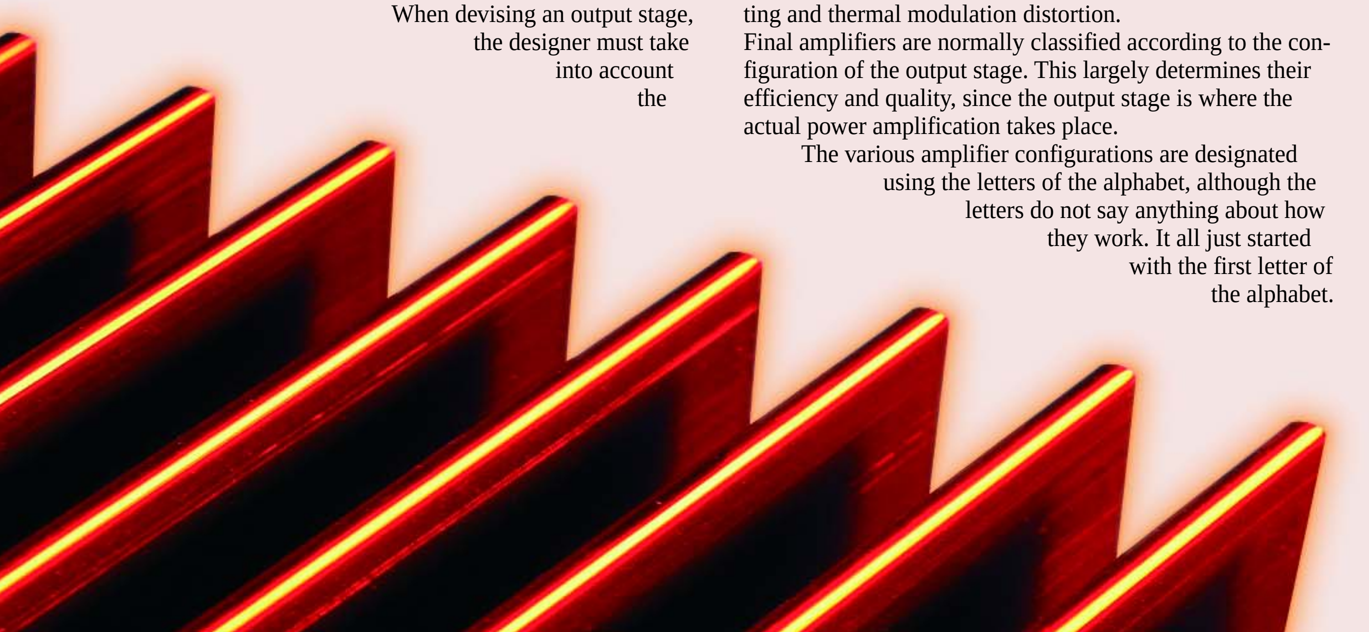
For many people, the amount of power an amplifier can produce is an important factor in judging its characteristics (So your amplifier delivers 2 × 40 watts? Mine does 2 × 70!). But in practice, power plays only a minor role.

You can already make a lot of noise with just a few watts. If you use a set of loudspeakers that can produce a sound pressure level of 86 dB at 1 W (which is a value commonly stated by manufacturers in speaker specifications), you can manage 90 dB with just 2.5 watts. With 25 watts, you have enough for 100 dB. That's already rather loud (and harmful to your ears as well!).

Our ears perceive each 6-dB increase in the sound pressure level as a doubling of the volume level, but this requires increasing the power by a factor of four. This means that if you really want to have a bigger final amplifier with more power than what you presently have, you will need an amplifier with at least four times as much power to actually notice any difference.

Delivering a large amount of power is not a simple task for an amplifier. Voltage amplification and current amplification are both necessary in order to provide sufficient power to the speakers connected to the amplifier. This is because loudspeakers have an efficiency of only a few percent, which means that several watts are certainly necessary to generate an adequate sound pressure level in a living room. In the case of concerts or outdoor events, quite a bit more is required, and the necessary power can easily amount to several kilowatts. To produce power amplification in a final amplifier, various concepts have been developed for using transistors or FETs to generate high-quality output signals and/or improve the efficiency of the output stage. (Here we leave valve amplifiers out of the picture.) When devising an output stage, the designer must take into account the specific properties of the semiconductor devices being used. If we could work with 'ideal' transistors or FETs, it would be much easier to build good amplifiers. Unfortunately, all semiconductor devices suffer from non-linearity in their amplification characteristics, and this causes major problems, especially for processing analogue signals. These problems can be minimised by using properly dimensioned feedback. There are also other nasty side effects that also occur, depending on the selected configuration, such as the notorious problem of crossover distortion.

Especially with large amplifiers, heat generation is another factor that must be taken into account. It can lead to far-reaching thermal effects, such as drift of the quiescent current setting and thermal modulation distortion.
Final amplifiers are normally classified according to the configuration of the output stage. This largely determines their efficiency and quality, since the output stage is where the actual power amplification takes place.
The various amplifier configurations are designated using the letters of the alphabet, although the letters do not say anything about how they work. It all just started with the first letter of the alphabet.



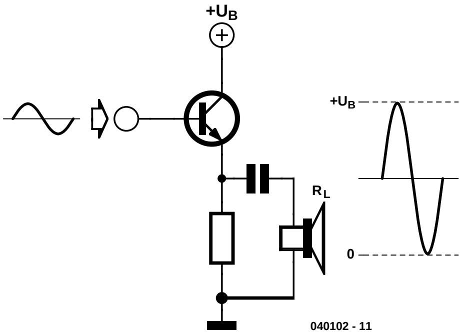


Figure 1. A Class A amplifier has very low efficiency, but it is totally free of crossover distortion.

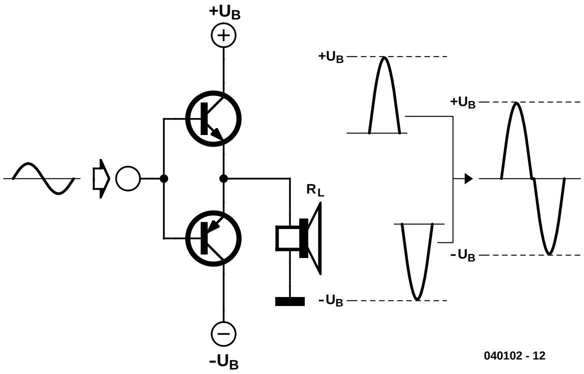


Figure 2. In a Class B configuration, each transistor conducts for half of the sine-wave cycle. Here problems arise around the zero-crossing point.

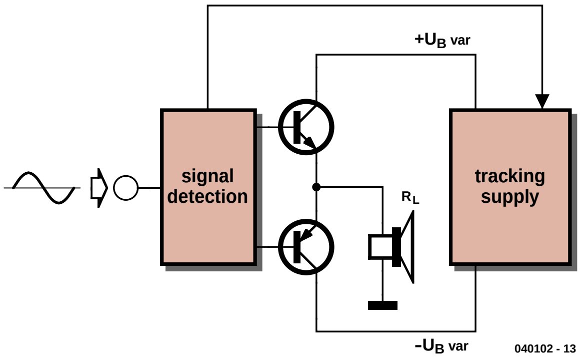


Figure 3. Class G uses a tracking power supply whose voltage is continuously adjusted to match the signal amplitude.

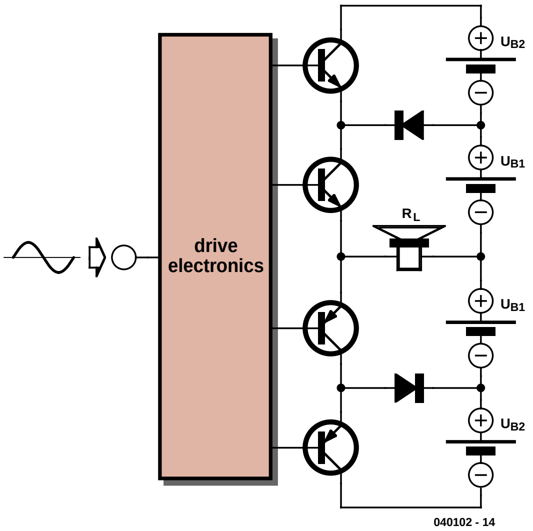


Figure 4. Class H limits itself to switching between several different supply voltages (in this case two).

Class A

Here we begin with the simplest configuration, the Class A final amplifier, which is also one of the best configurations for high-quality audio reproduction. In its basic form, this configuration can be implemented using a standard emitter follower (**Figure 1**). The quiescent current through the transistor is equal to the peak AC output current, which means that the transistor is biased in the middle of its working range and simply conducts more or less current when driven by an alternating voltage. The efficiency is very low: 25 % at maximum output amplitude, and even less at low signal levels. The efficiency can be improved by using a symmetrical design with two transistors, but even then the highest efficiency that can be achieved is 50 %.

Class B

The Class B configuration employs two transistors, each of which conducts for exactly half of the signal cycle (**Figure 2**). In the quiescent state, no current at all flows through the transistors. The efficiency of a Class B output stage is around 78 %, but the primarily disadvantage of this configuration is the 'transfer distortion' that occurs each time the load must be transferred from one transistor to the other. Due to the sharp bend at the bottom end of the transfer characteristic, the two halves of the signal waveform do not properly align with each other. This leads to the notorious problem of crossover distortion, which is a quite audible degradation of the signal waveform.

To solve this problem, Class A and Class B were combined to produce Class AB. This is a Class B configuration in which a small quiescent current flows, causing the output stage to actually work in Class A at low power levels. This approach is presently used in various forms in most final amplifiers. The efficiency remains approximately the same as for Class B.

G and H

Hey, just a minute! Haven't we skipped a few classes? We have indeed, but we did so on purpose. Classes C, E and F also exist, but they are actually only suitable for high-frequency applications, which means they more or less fall outside the scope of what we're talking about here. And the design of Class D amplifiers is so different from Class A and Class B that it we decided to deal with it separately. So, let's first look at classes G and H, which have an important feature in common. This is that in both of these classes, the supply voltage is adjusted according to the magnitude of the output signal. In Class G (**Figure 3**), the supply voltage is continuously adjusted to match the desired amplitude of the output signal. Such a 'tracking' supply voltage can be implemented relatively easily using modern switching power supplies, although it is of course important to have a good regulator circuit to allow the supply voltage to respond sufficiently quickly to changes in the amplitude of the signal generated by the output stage.

In Class H (**Figure 4**), what happens is essentially the same as in Class G, except that here the supply voltage is switched between several distinct levels (usually two) instead of being

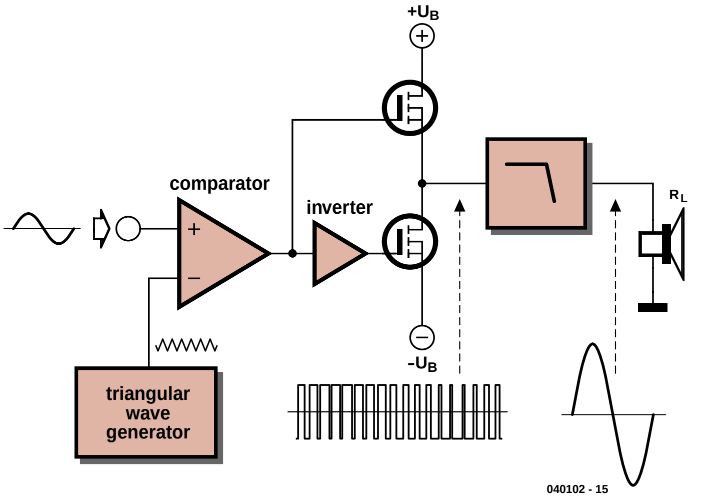


Figure 5. A Class D amplifier consists of a pulse-width modulator with a power output stage and a low-pass filter.

continuously varied. This allows the dissipation in the output stage to be considerably reduced, especially when large amounts of power are involved.

Class D

With the Class D amplifier configuration, the letter ‘D’ doesn’t have anything to do with ‘digital’ (that’s just a coincidence). It refers to a switching amplifier that uses pulse-width modulation (**Figure 5**). The input signal is compared with a triangular waveform, and the signal from the comparator switches the output stage to the positive or negative supply voltage. This is done using a very high switching frequency, which is usually ten times or more higher than the audio bandwidth (which means 200 kHz or above).

With this form of modulation, the pulse width depends on the level of the input signal. If a low-pass filter is placed after the output stage, the pulse-width signal is integrated and what is left is an analogue signal with the same form as the input signal, but of course amplified.

As the output stage only has to switch, its efficiency is very high. However, there are also a number of drawbacks to this approach. It is rather difficult to keep the signal waveform free of distortion, a hefty output filter is required, and drastic measures must be taken to limit radiated interference. For low-distortion amplification, it is always necessary to use negative feedback (analogue or digital).

Classes S and T

Although the working principle of the Class D amplifier is already several decades old, it never managed to become truly established in hi-fi applications. This was primarily due to excessive distortion and a lack of good semiconductor devices (fast power FETs). In the meantime, several manufacturers have devised variations on this theme, and in many cases they have given them their own designations. For instance, Crown came up with the Class I amplifier, Sony developed its S-Master technology, and Tripath has devised its Class T amplifier. Unfortunately, the nice alphabetic sequence has been abandoned in favour of manufacturer-specific abbreviations.

In its S-Master technology, Sony combined several techniques to make the Class D configuration suitable for domestic hi-fi applications. Here the process of converting the incoming signal into a corresponding pulse-width signal is called ‘complementary pulse length modulation’ (C-PLM). Extensive atten-

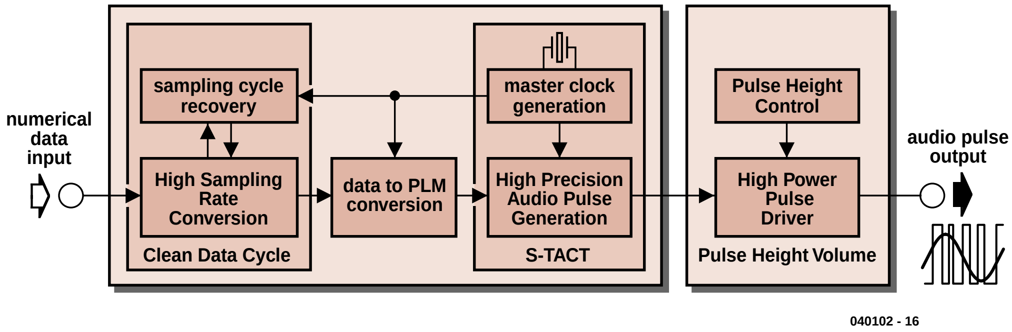


Figure 6. Block diagram of a Sony S-Master amplifier.

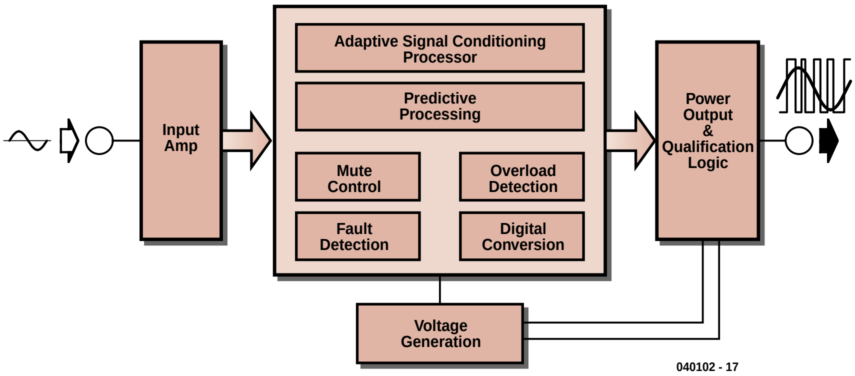


Figure 7. The Tripath Class T amplifier is an 'intelligent' elaboration of the Class D principle. Here a processor continuously monitors the input signal and adjusts the switching signals.

tion was given to suppressing jitter. This was accomplished by using an extremely accurate clock signal and a circuit called 'clean data cycle' that corrects the positioning of the output pulses if necessary (see **Figure 6**).

The method used to implement volume control is certainly an unusual feature of the Sony approach. In a normal Class D design, the full pulse waveform is always present at the output, with an amplitude of 50 to 100 volts peak-to-peak. Particularly with small output signals, it is very difficult to fully eliminate all residual components of the pulse waveform from the filtered signal. In the Sony design, the volume is regulated by adjusting the supply voltage for the output stage. This prevents any information from being lost at low signal levels. This technique has an effective range of 50 dB.

Another company, Tripath, has developed a technique that according to them combines the signal quality of Class A and AB amplifiers with high efficiency (around 80–90 %). This is done using a combination of analogue and digital circuitry, together with digital algorithms that modulate the input signal using a high-frequency switching waveform. The algorithms developed by Tripath are derived from adaptive and predictive algorithms already used in telecommunication systems.
With the Tripath amplifier, the majority of the analogue and digital circuitry is housed in a single IC, which may also include the output transistors (depending on the power). The block diagram of the amplifier is shown in **Figure 7**. The input signal is first buffered by an input stage. From there it passes to the Digital Power Processing block, which contains the signal processor, a digital conversion function, mute switching, overload protection and error detection. The output stage is driven via the qualification logic, and the loudspeaker is connected to a filter following the output stage.
Thanks to its special algorithms, the processor in the Class T amplifier can shape the drive signal for the output stage to match the specific characteristics of the transistors used in that stage. This shaping takes into account non-ideal switching behaviour, mismatching of the complementary output transistors, dead-time distortion and the residual energy of the oscillator in the audio band.

The switching frequency of a Class T amplifier is continuously adapted to the magnitude of the input signal. At low input levels, the switching frequency is quite high (around 1.2 MHz). This has a beneficial effect on signal quality. The switching frequency gradually drops as the input level increases, in order to increase efficiency. The switching frequency ultimately reaches its lowest value (around 200 kHz) when the output is driven to maximum amplitude. Besides this, a form of noise shaping is applied to the peaks of the output signal in order to improve the signal waveform. As a result of all these measures, the Class T amplifier can deliver a sound impression that reminds listeners of audiophile analogue amplifiers.

The future

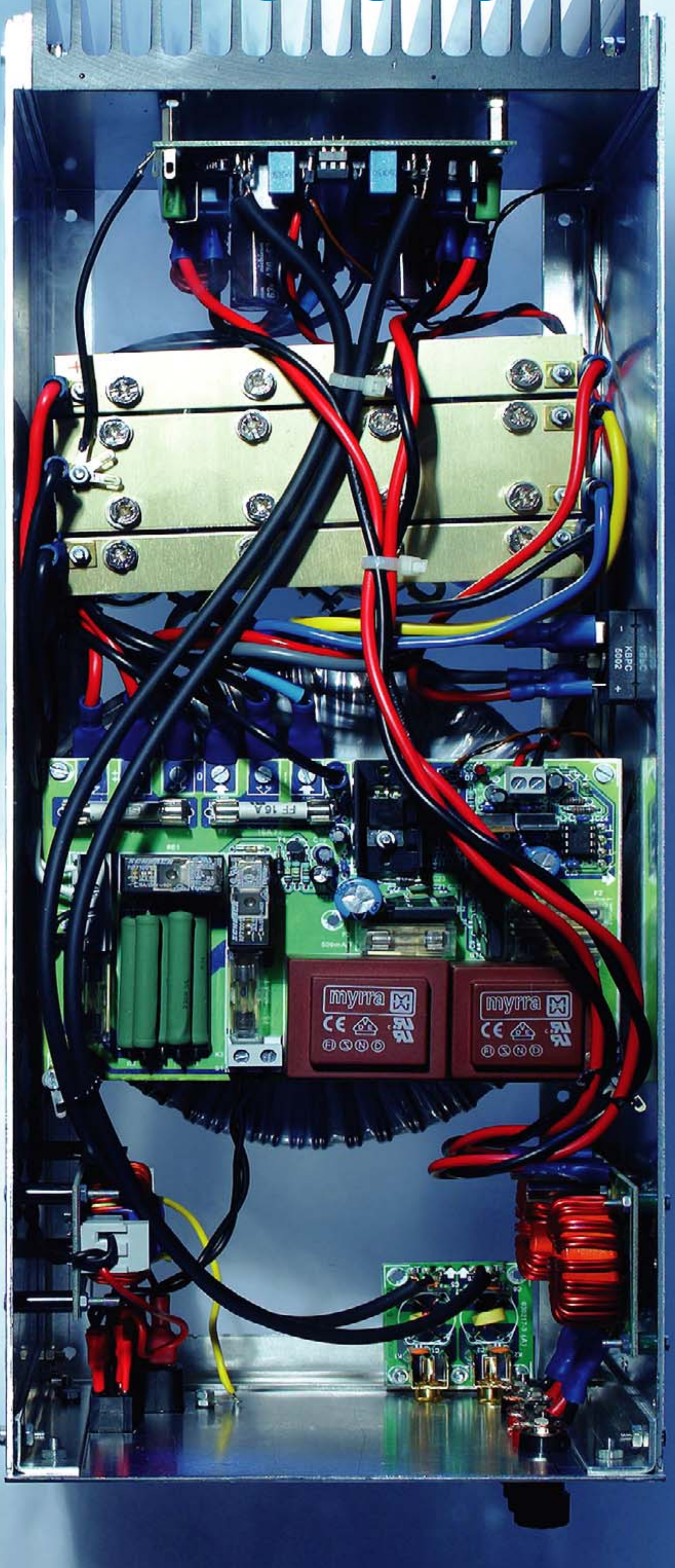
With the steady advance of digital audio, some form of digital output stage will ultimately be found in many consumer amplifiers. The reasons for this are higher efficiency, smaller size and lower manufacturing cost. It's difficult to estimate whether this development will also come to prevail in the high-end realm. There are presently only a few high-quality digital amplifiers on the market. But if you'd like to try it for yourself, you can start by building the Clarity amplifier described in this issue.

(040102-1)

ClariTy 2x300W

Ton Giesberts

This top-end amplifier proves that high power does not have to mean a large, heavy design. Although this amplifier is highly efficient (and thus compact), its specifications easily surpass those of quite a few conventional designs.



Class-T amplifier

If we've given you the idea that the fully assembled amplifier is as light as a feather, perhaps we should qualify our statement somewhat. After all, 2 x 300 watts of *true* power naturally requires a rather substantial power supply. But that's the only aspect of this amplifier that is comparable to other types of amplifiers. Thanks to clever use of pulse-width modulation, this amplifier is so efficient that a heat sink with quite modest dimensions can be used, which means that the enclosure can also be kept relatively small. What's more, this amplifier is not an ordinary pulse-width amplifier. This design, which is based on the Tripath TA3020 Class-T digital audio driver, has excellent specifications and can easily hold its own against other top-end amplifiers. For more information about pulse-width modulation in audio final amplifiers, please see the article 'That's class…' elsewhere in this issue. The design is largely based on the standard application example and the manufacturer's reference PCB layout. This is because the board layout largely determines the quality of the overall amplifier. Besides this, the nature of this design (with high switching frequencies and large currents) imposes severe requirements on various components. That means that special types of electrolytic capacitors and decoupling capacitors are used in most locations. Even for the thermal coupling between the output transistors and the heat sink, ordinary mica or Kapton washers are not satisfactory. Instead, ceramic washers with a thickness of several millimetres must be used. The IC also needs two auxiliary supply voltages, for which a separate printed circuit board has been developed. This board also includes a mains voltage switch-on delay for the main transformer and two hefty fuses for the main supply voltages. To suppress electromagnetic interference (EMC), extra filters are also included at the inputs and outputs. This should already give you an idea of what to expect, but in this first part of the article we primarily concentrate on how the Tripath IC works.

In **Figure 1**, you can clearly see that the IC essentially consists of three sections for each channel: an analogue input stage (inverting amplifier), a processing and modulation unit, and the driver stages (level shifters) for the external power MOSFETs. The IC also provides overcurrent protection, overvoltage and undervoltage protection, and a connection for an external mute signal. All of these collectively determine whether the amplifier outputs are active.

Input stage

The analogue input stage is implemented as an inverting amplifier for convenient dimensioning of the gain and bandwidth. According to the IC specifications, the maximum allowable signal level for fully driving the modulator is 4 V_{pp}. With the dimensioning used here, assuming an input sensitivity of 1.13 V_{eff} for maximum output amplitude, the output of the input stage can be driven to 3.2 V_{pp}. The ratio of R3 and R2 (R24 & R23 for the second channel) determines the gain of the input stage. Here the ratio is 1, as can be seen from the schematic diagram in **Figure 2**. Capacitor C2 (C15) increases the stability of the input amplifier and suppresses RF noise by limiting the bandwidth to approximately 240 kHz. C1 (C14) sets the lower corner frequency, which in this case is around 2.5 Hz. The gain for frequencies in the audio band is thus as flat as possible. C1 and C14 are standard MKT capacitors, since as a matter of principle we try to avoid using electrolytic capacitors in the signal path.

R4, R5 and P1, in combination with decoupling capacitor C3 (R25, R26, P2 & C16), allow the output offset voltage of the amplifier to be adjusted to a minimum.

Modulator

The modulator amplifies the signal from the input stage to the output level. It is the second part of the overall amplification, or better said, the actual gain stage. The processor provides a switching waveform that depends on the level and frequency of the signal. With no input signal, the average value of the switching frequency is approximately 700 kHz. It can vary over a maximum range of 200 kHz to 1.5 MHz. Two complementary MOSFET drivers with

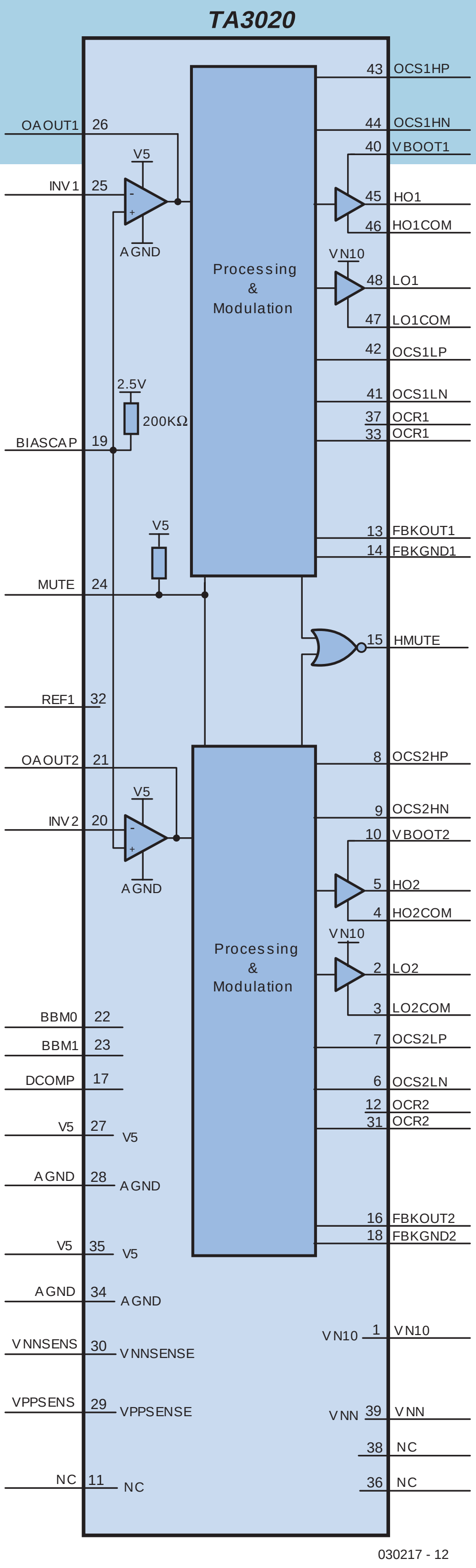


Figure 1. The internal structure of the Tripath TA3020.

level shifters convert this signal to the proper level for driving the gates of the MOSFETs. The supply voltage for the drivers (10 V) is provided via pin VN10. It is first decoupled as much as possible by C38 and C39. These capacitors must therefore be placed as close as possible to the associated pin of the IC. On the negative side, LO1COM (connected to the source of T2) and VN10 provide the supply connections for the driver. On the positive side, bootstrap capacitor C7 (C20) is charged via D5 (D12) to nearly 10 V when the output is negative, and it 'rides up' on top of the output voltage when T1 starts to conduct. This voltage is fed to VBOOT, which together with HO1COM (the source of T1 is also the bridge output) forms the other supply connection for this driver. At the clipping level, C8 (C21) provides an extra buffer, since the switching frequency is lower at the clipping level. R13 (R14) limits the charging current of C8 (C21) when the amplifier is switched on.

MOSFETs

Two n-channel MOSFETs (T1 & T2 or T3 & T4) form a half-bridge circuit for each channel. The level-shifter outputs alternately drive each of the MOSFETs into conduction. A 'dead time' (break-before-make) is maintained to ensure that the two MOSFETs can never both be conducting at the same time (no 'shoot-through' current). This time can be set using two jumpers (JP1 and JP2). We strongly recommend against experimenting with the selected setting. It might be possible to reduce the dead time if you use MOSFETs with considerably smaller gate capacitance (smaller amplifier power), but certainly not here! Gate resistors R78 and R9 (R28 & R30) limit the slew rate, and thus limit the amount of overshoot due to switching. They also somewhat reduce the amount of power that would otherwise be dissipated in the drivers (these are 1-W resistors) for charging and discharging the gate capacitances of the MOSFETs. D1 and D2 (D8 and D9) decrease the gate discharge time. This reduces the fall time of the pulses, and thus reduces the chance that both T1 and T2 may both be conducting at the same time. R8 and R10 (R29 & R31) are added for safety reasons. If no IC is fitted, they insure that the gates of the MOSFETs remain discharged. Without these resistors, leakage currents and noise voltages could occasionally have disastrous consequences. R6 and R11 (R27 and R32) are low-inductance resistors that are necessary for current limiting, which is described later on. R12 and C4 (R33 & C7) form a snubber network that eliminates high-frequency spikes in the output signal. This network is thus placed as close as possible to T2 (T4). Diodes D6 and D7 (D13 & D14) are connected between the source and drain of each of the transistors to suppress overshoots. Such overshoots are primarily caused by the coil in the output filter when large currents flow. These diodes (in SMD packages) are also placed as close as possible to the associated leads, primarily to protect the IC. D3 and D4 (D10 & D11) are additional diodes connected between the sources and drains (respectively) of the MOSFETS to suppress overshoots. All of these diodes (D1–D14) must be ultra-fast-recovery types. C5 and C6 (C18 & C19) decouple the half-bridge circuit and are especially included to suppress spikes on the supply voltage lines. This also has a beneficial effect on the operation of the MOSFETs. These capacitors must also be placed as close as possible to the leads of the MOSFETs. C6 (C19) must be an electrolytic capacitor with extremely low ESR and very good HF characteristics. Here you should not use a substitute for the prescribed type unless its specifications are just as good or better. A normal electrolytic capacitor would probably explode or have a very short life. The pulse-width modulated signal at the output of the half bridge is fed to the output terminals via the LC filter L1/C9 (L2/C22).

Output filter

Thanks to the high switching frequency, here it is only necessary to use a second-order filter with a relatively high corner frequency (resonant frequency 101 kHz). To dampen the Q factor of the filter, which is primarily important if no load is connected, a Zöbel network is placed at the output, since otherwise resonance currents and ringing signals at the output could reduce the reliability of the amplifier. As the corner frequency of the filter is higher than for conventional Class D amplifiers, the connected speaker system has a considerably smaller effect. In light of the large currents involved here, an ordinary noise-suppression choke cannot be used for the output filter. A special core material must be used to maintain low distortion and high efficiency. We have more to say on this subject in Part 2 of this article, which will appear in the September 2004 issue.

Amplifier configuration

The gain of the modulator is set using feedback resistor R15 (R36) and voltage divider R18/R20 (R39/R41). These components must be dimensioned according to the value of the supply voltage that is used. This is necessary to make the amplifier independent of the behaviour of the power supply (such as voltage fluctuations due to the output amplitude, mains voltage variations, etc.). Additional reverse feedback to counter 'ground bounce' is provided by R16 (R37) and voltage divider R17/R19 ((R38/R40). These two networks must be identical! The resistor values can be calculated quite easily. A value of 1 kΩ is generally used for R17 and R18, so the value of the other resistors only depend on the value of the supply voltage VPP (assuming a purely symmetric power supply) and the value selected for R17:

$$R19 = R17 \times VPP \div (VPP - 4)$$

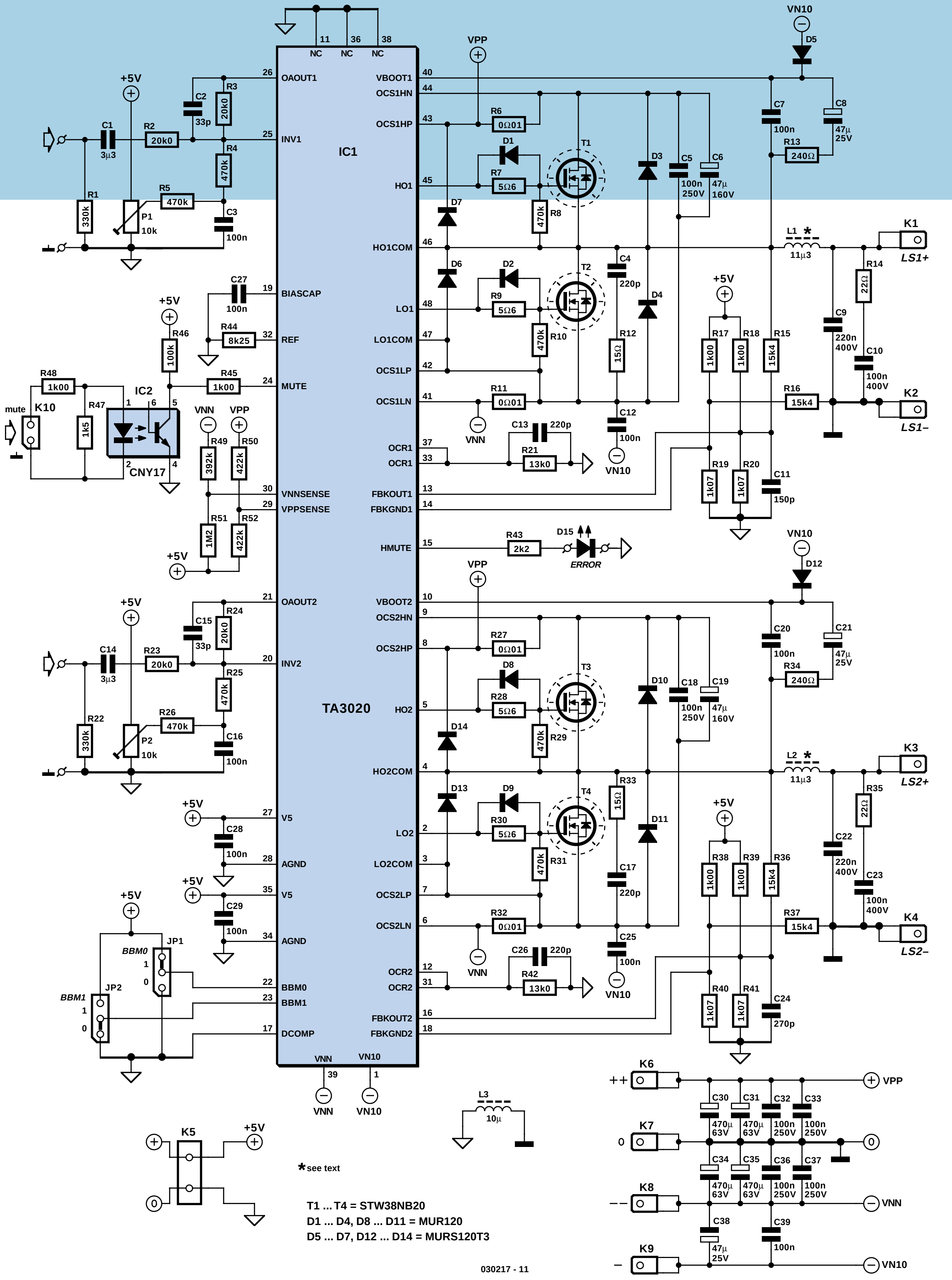
This yields an E96 resistance value of 1.07 kΩ. This value is reasonably independent of the supply voltage. If a maximum supply voltage of 51 V is used, it only increases to 1.10 kΩ.
Finally, the value of R15 determines the gain of the modulator:

$$R15 = R17 \times (VPP \div 4)$$

We have assumed a maximum supply voltage of 62 V (the special decoupling electrolytics on the amplifier board are 63-V types). This yields a value of 15.4 kΩ for R15. The gain of the modulator can then be calculated in the same manner as for a standard non-inverting amplifier:

$$A_{modulator} = (R15 \div Rp) + 1$$

where R_p is the parallel resistance of R18 and R20.
Capacitors C11 and C24 filter and delay the feedback signal to the modulator. They have different values. These capacitors prevent RF noise with very high frequencies from penetrating the feedback network, and using different values causes the modulators to have different switching frequencies. This prevents mutual interference between the modulators. The values are chosen to cause the difference to be greater than 40 kHz.



*see text

T1 ... T4 = STW38NB20
D1 ... D4, D8 ... D11 = MUR120
D5 ... D7, D12 ... D14 = MURS120T3

Figure 2. The circuitry around the amplifier IC.

Protection

To protect the amplifier, the driver IC monitors the supply voltages and the currents through the transistors. The VPPSENSE input is used to monitor the main positive supply voltage for overvoltage and undervoltage; the VNNSENSE input is used in the same way for the main negative voltage. If the supply voltage is outside the allowable limits, the output stage is disabled (mute mode). If the supply voltage returns to within the allowable limits, the output is again enabled. For the calculation of the associated component values, please refer to the data sheet. Theoretically, the amplifier could become 'stuck' in situation in which it constantly detects an overvoltage. However, that is very unlikely, since both detection inputs need more than roughly 68 V before they respond. This thus primarily amounts to protection for the IC itself, since several of the power supply capacitors are only rated for 63 V.

The calculations for overcurrent protection are certainly more interesting that those for voltage protection, since they determine the minimum load impedance the amplifier can handle at maximum output power. Since the output stage operates in switch mode, the MOSFETs used in the circuit determine the maximum load capacity of the amplifier. Here we have selected a relatively heavy-duty ST Microelectronics type, the STW38NB20. This transistor, which is housed in a TO-247 package, can handle up to 38 A and has a maximum drain–source voltage of 200 V. The maximum channel resistance with a gate–source voltage (U_{GS}) of 10 V is 0.065 Ω (I_D = 19 A). A disadvantage of MOSFETs with this sort of specifications is that their input capacitance (C_{iss}) is rather large, in this case as much as 3800 pF. That explains why the drivers in the IC must be able to deliver rather substantial currents in order to switch the MOSFETs sufficiently quickly. We primarily chose these transistors in order to reduce the risk of unpleasant surprises when using speaker systems with unknown impedances. Naturally, the break-before-make time could be made shorter if transistors with significantly smaller gate capacitance are used, which would reduce the distortion level. However, our choice was in favour of a design that can tolerate low impedances.

Overcurrent detection is provided by the two low-inductance resistors R6 and R11 (R27 & R32), which are connected in series with the transistors as sense resistors. R6 is used for positive half-cycles in series with the drain of T1, while R11 is used for negative half-cycles in series with the source of T2. The response threshold of the protection circuit is set in combination with R21. The IC directly measures the voltages across the sense resistors and uses these voltages to generate a current through R21. The maximum output is determined by comparing the voltage across R21 with the overcurrent threshold voltage V_{TOC}. C13 (C36) filters the voltage from the rectifier. The relationships between these components are given by the following two equations:

I_{max} = 3580 x (VTOC – (Ibias x R21)) ÷ (R21 x R6)

R21 = (3580 x VTOC) ÷ (Imax x R6 + 3580 x I_{bias})

Here VTOC is the threshold voltage for overcurrent detection (typically 0.97 V) and I_{bias} is 20 μA.

The first equation can easily be rearranged to allow the component values to be calculated. The second equation can be used to determine the value of R21 (R42). We have chosen a maximum output current of nearly 20 A, so that a load of somewhat less than 3 Ω just avoids triggering the mute mode.

The mute mode can only be reset by briefly switching the level at the Mute input or briefly switching off the amplifier. When the mute mode is active, the HMUTE output is High, and this signal drives a LED that can be fitted to the front panel if desired. A red high-efficiency LED should be used for this purpose, since reducing the value of R43 would overload the output.

Power supply

The supply voltages for the amplifier board are provided by a second printed circuit board. This board includes, among other things, the +5-V and VN10 supplies, as well as fuses for the main supply voltages. It also supplies a delayed 'un-mute' signal that prevents switch-on 'plopping'. To avoid creating an earth loop and prevent ripple currents from flowing though the input stage ground path, the mute signal is fed to the IC via an optocoupler. It is located on the amplifier board. The input of the optocoupler is thus fully isolated from the amplifier, but an active signal is required to switch the amplifier Mute input.

The main supply voltages (VPP and VNN) for the TA3020 are decoupled as well as possible using special electrolytic capacitors (C30, C31, C34 and C35) and MKT capacitors (C32, C33, C36 and C37). A simple decoupling network is used for the 5-V supply voltage for the input amplifiers.

To suppress possible interference from the output circuit as well as possible, analogue ground and modulator ground (which is also the ground for the rest of the circuit) are kept separate and coupled on the solder side of the board at a single location using an SMD inductor.

Layout

As already mentioned at the beginning of the article, the amplifier relies on a carefully designed layout. The layout thus forms an essential part of the overall amplifier. Tripath emphatically recommends copying the reference layout, since otherwise the large high-frequency currents could give rise to unexpected effects. Naturally, some of the components we have selected differ from those used on the reference board, primarily with regard to their dimensions. This is because we have given special attention to the availability of the components (preferably in single quantities). Some of the tracks have been shifted slightly in some places, and a few components have been added, but by and large we still succeeded in maintaining the recommended layout. If you take the trouble to look at the photo of the reference board in the data sheet, you will see the resemblance to the photos of the prototype. For some of the components shown in the schematic that are fitted on the solder side of the board in the Tripath layout, we have put them on the component side instead. This is why the circuit board is placed parallel to the heat sink in our version, with the transistors mounted below the board. This produces an attractively compact and robust module, but we'll save further comments for the construction description in the second part of the article. What we can tell you already is that although the module looks very simple at first glance, on closer examination you will notice that compared with the schematic, a few things seem to be missing. Many of the components are SMD types, and they are fitted on the solder side of the board. This yields the lowest likelihood of interference problems and results in an amplifier board with very modest dimensions for a 2 ¥ 300 W amplifier. Most of these SMD devices come in 0603 'shapes', which in all honesty are nasty little things to work with. To make things easier for you, we will try to supply the circuit board in the near future with the SMD components already fitted.

In the second part of this article (September 2004 issue), we give detailed attention to the construction of this unusual amplifier.

(030217-1)

Web pointers

TA3020 data sheets & application note:
www.tripath.com/downloads/TA3020.pdf

TA3020 reference board --
www.tripath.com/downloads/RB-TA3020.pdf

Paul Goossens

Design Your Own IC

Part 2: CPLDs in practice

Following the description of the hardware in last month's issue (Part 1), it's now time to start working with the experimenter's board. We assume that you have already installed the Altera software and read the tutorial.

Designing digital circuits usually amounts to repeatedly breaking down the problem into subproblems until you finally arrive at a design consisting of variety of basic logic functions. Based on this, you can develop the electronic circuit and, if necessary, a printed circuit board.

Descriptive languages

Designing digital logic circuitry is easier if you use a descriptive language. The purpose of such a language is to allow specific functions to be described (hence the name). A descriptive language allows intelligent software to be used to design an electronic circuit that meets the description specified by the designer.

There are presently several different descriptive languages. Two of them are manufacturer-independent and are supported by a large number of manufacturers: Verilog and VHDL. In this article, we use Verilog as our descriptive language.

Verilog

We chose Verilog because it is somewhat clearer than VHDL. However, the two languages are the same in many aspects. The biggest difference between them is in how the descriptions are formulated. This means that many of the considerations, pitfalls and the like described in this article are also directly applicable to VHDL.

It's possible to generate hierarchical designs using Verilog. This means that the design can be divided into smaller designs. These smaller 'subdesigns' can in turn be further divided into various subdesigns as necessary. In Verilog parlance, such subdesigns are called 'modules'.

Dividing a design into several modules has the beneficial side effect that it may be possible to reuse the modules in other designs. A counter is an example of a type of module that is used relatively often, so it is definitely a good idea to put a counter into a separate module.

Example 1

The best way to learn something is by actually doing it. Consequently, we can begin right away with an example. Before you can get started, you must download the examples from the Elektor Electronics Internet site (www.elektor-electronics.co.uk). The examples are located under item number **030385-11** for the June articles. All you have to do is unpack the Zip file.

Example 1 can be found in the *Ex1* folder. Just double-click on the *ex1.quartus* file, and the design software will start up automatically. In the schematic diagram that is displayed, you can see that the various I/O pins of the IC are connected to a block in which several signals are listed. These signals are the inputs and outputs of this block.

As you have already seen in the tutorial (you did look at the tutorial, didn't you?), you can view the associated source code by double-clicking on the block. In this case, the source code is written in the Verilog language. The text shown in green is all comments as far as Verilog is concerned, so it has no effect on the ultimate result. However, Quartus uses these lines to store information, so it's a good idea to leave them as they are.

Structure

Verilog source code is always organised using the same structure. It starts with the module declaration. This part of the code begins with the word 'module', followed by a name. This is accompanied by a collection of inputs and outputs in brackets, separated by commas. The whole thing is terminated by a semicolon. This can be seen in lines 30–35 of our example.

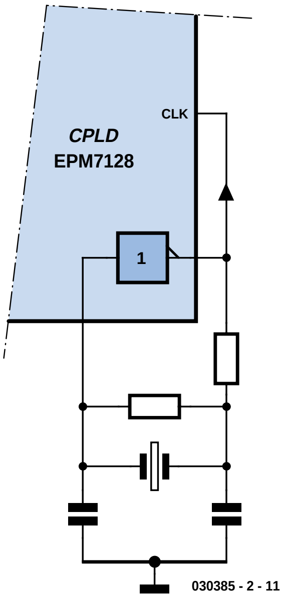


Figure 1. Schematic diagram of a standard crystal oscillator.

The next thing you have to do is to define the directions of the signals (ports) identified in the module declaration. You can see how this is done in lines 39–45 of our example. There are three options for each signal: *input, output,* or *inout* (bi-directional). Here we have only used 'input' and 'output'. Each line is terminated by a semicolon (;). In the first line, you can see that several signals can be defined in a single line if commas are used to separate the definitions.

The outputs require an additional specification. If the function of an output is described in a procedural statement (don't worry, we'll explain what

Table 1. Boolean functions

&	=	AND
~&	=	NAND
|	=	OR
~|	=	NOR
~	=	NOT
^	=	XOR
~^	=	XNOR

Table 3.
Arithmetic operators

+	=	add
-	=	subtract
*	=	multiply
/	=	divide
%	=	modulo

Table 3.
Relational operators

>	greater than
> =	greater than or equal to
<	less than
< =	less than or equal to
= =	equal to
! =	not equal to

this means further on), it must have the type *reg* (register = output of a flip-flop). In line 47, signals D2, D3 and D4 are defined as registers.

Now that we've taken care of the administrative duties, we can start with the actual design. This example, as befits every initial example, is very simple. Here we demonstrate the ways in which signals can be described using Boolean algebra.

This can be done in two manners in Verilog: either by using an *assign* statement, or in what is called a 'procedural statement'. **Listing 1** shows an example of each of these methods.

The first method is demonstrated in line 50. Here the description says that signal D1 is the result of an AND operation on signals S1–S4. That's another way of saying that D1 is only active if S1–S4 are also active ('1'). In all other cases, D1 is inactive ('0').

The symbol '&' thus stands for the AND function. The Boolean functions in Verilog are summarised in **Table 1**.

Procedural statements

The remaining outputs (D2–D4) are described in a procedural statement. Procedural statements are always preceded by the word *always*. This keyword is described in more detail in one of our later examples.

Just as in the Pascal programming language, you can combine a group of statements into a unit by using the keywords *begin* and *end*. All of the statements between these two words are collectively regarded as being a single statement.

If you look at line 54, you will see that signal D4 is described as a signal that becomes active if S1 or S2 or S3 or S4 is active. Here we intentionally used the word 'becomes' instead of 'is'. The symbol <= means 'becomes' or 'assumes the value'. As a general rule, we can say that this symbol is used in a procedural statement instead of the = sign.

It's not difficult to figure out the functions of signals D3 and D4 if you use **Table 1** for a bit of help.

The advantage of putting signals in procedural statements instead of working with *assign* statements will become clear in a later example.

Finally, the keyword *endmodule* indicates that the description of the module is finished.

Compiling

Now we're getting close to the point where you have to roll up your sleeves and get to work. First, the design has to be compiled. The compiler already knows exactly which signal must be connected to each pin of the CPLD. That's because we already did this for you. This makes compiling child's play; just click on *Start Compilation* in the *Processing* menu and the software go into action.

Various messages will be shown on the screen, and several progress bars will move along. After a while, the program will report that the compilation was successful. That means it has created a programming file that you can use to program your IC.

Programming

As already mentioned in Part 1 of this article, to program the CPLD you will need the JTAG programmer described in the September 2002 issue of *Elektor Electronics.* Of course, an original Altera ByteBlaster is also OK. We assume that your programmer is connected to the printer port of the PC and the JTAG connector is attached to connector K2 of the experimenter's board.

Now switch on the power for the experimenter's board.

In Quartus, first select the *Tools* menu and then *Programmer.* A new window will appear. Check that the programmer is set to 'JTAG' and the correct interface (ByteBlaster) is selected.

In this window, there is a line that has 'EPM7128SLCM' in the *Device* column. On the same line, the programming file *ex1.pof* is shown in the *File* column.

Everything is now ready for programming the CPLD. You must tell the program you want to program this IC by placing a tick mark under the *Program/Configure* column.

Finally, click on the *Start Programming* button, which is located at the very top and looks like a sort of 'Play' button.

Testing

After being programmed, the CPLD is immediately active, which means that the programmed design can be used right away. Make sure that jumpers JP1 and JP2 are fitted.

Checking the design is easy. LED D1 should only be on if all of the switches are in the '1' position. In all other cases, the LED must be dark. By contrast, LED D4 should do just the opposite. That means that if D1 is on, D4 must be off, and vice versa.

We have described D2 as an OR function, which means that this LED must be on if one or more switches are in the '1' position.

LED D3 must light up whenever S1 and S2 are both in the '1' state OR S3 and S4 are both in the '1' state. These functions can be easily checked using the switches.

Now try to modify the design in Quartus to cause LED D1 to be on whenever S1 is in the '1' state and S2 is in the '0' state. The states of the other switches don't matter. Good luck with your design!

Listing 1. Boolean algebra

```
assign D1 = S1 & S2 & S3 & S4;

always
 begin
   D2 <= S1 | S2 | S3 | S4;
   D3 <= ( S1 & S2) | (S3 & S4);
   D4 <= ~(S1 & S2 & S3 & S4);
 end
```

Example 2

As already mentioned, Example 1 is very simple. The special power of Verilog is that it allows designers to develop designs in a more descriptive manner. Boolean algebra can occasionally be handy in Verilog, but it is certainly not the intention that relatively complex designs must be entered entirely in Boolean algebra.

This can be demonstrated using Example 2, in which we set about designing two flip-flops and a latch.

The files for Example 2 can be found in the *ex2* folder. In this folder, open *ex2.quartus,* and the program will automatically open all of the other necessary files.

In the schematic diagram *(ex2.bdf),* you can see that switches S1–S4 are connected to a functional block named *flipflop.* S1 is connected to the CLK input, etc. Double-click on the block to open the associated Verilog source code.

Always @

Up to line 46, there's nothing new to be seen. But in line 46 you can see something added to the keyword *always:* an @ character followed by a comparison. This code segment can also be seen in **Listing 2**. The @ sign indicates that the procedural statements belonging to this *always* statement are only allowed to be evaluated (but not 'executed'; only processors execute statements) if the following comparison is satisfied In this case, that means that the following statements are only applicable at the moment when the clock signal (CLK) OR the reset signal OR the SET signal has a rising edge *(posedge).* Just to avoid any confusion, a rising edge is the transition from a Low level to a High level.

Verilog also has the modifier 'negedge', which in normal English means 'negative edge' or 'falling edge'.

If any one of these conditions is satisfied, this section of the code is evaluated. First, a check is made to see whether the RESET signal is '1'. If this is the case, OUT becomes inactive ('0') and this code segment is done. Otherwise, a check is made to see whether the SET signal is '1'. If it is, the OUT output goes to '1' and the code is done. Beside the values '1' and '0', each signal can also assume the values 'x' (unknown) or 'z' (high impedance).

If neither RESET nor SET is '1', CLK must have a rising flank, since otherwise this code would not have been evaluated. The intention is that on the rising edge of the clock signal, the output of the flip-flop assumes the value present at the input.

But what happens to the output whenever there isn't a rising edge on CLK, RESET or SET? The answer is very simple: nothing. In line 45, the OUT signal is defined to be a register, which means that the value most recently assigned to this signal must be held. Whenever the code is not active, the value of this register will not change.

By adding the '@' character to the *always* statement, we can thus indicate the conditions under which a portion of the code is allowed to be evaluated. During the rest of the time, the output that is controlled by this block must remain the same.

From the code, it can clearly be seen that the RESET input has higher priority than the SET input. However, that doesn't mean that he CPLD evaluates these two signals one after the other when it is operating. The CPLD will respond just as fast to the RESET signal as to the SET signal. The sequence is only important for the compiler. It evaluates a section of code and determines what must happen to the output for every imaginable combination of input signals. Based on this evaluation, the compiler 'designs' a bit of digital logic that respond in exactly the manner described in the code.

Variations on a theme

A second flip-flop (appropriately named 'flipflop2') is also shown in the schematic diagram *(ex2.bdf).* The associated Verilog file is very similar to the file for the first flip-flop. The only difference with respect to the first flip-flop is that the state of the SET signal is checked before the state of the RESET signal. That means that for this flip-flop, the SET input has a higher priority than the RESET input. The output will thus go to '1' if the RESET and SET inputs both have a value of '1'. With the first flip-flop, the output will go to '0' in this situation.

Latch

The final block in the schematic is a latch. A latch is also a frequently used type of component in digital designs. The operation of a latch is actually quite simple. As long as the clock input is '1', the latch's output must be the same as its input. If the input state changes, the output must immediately follow the change. By contrast, if the clock signal is inactive ('0'), the last known state of the output must be retained, regardless of any changes to the input state.

The Verilog file *latchexample.v* shows how this can be described in the Verilog language. The output signal **can** change if the state of the clock signal changes OR the state of the data input changes. This can happen on the rising edge as well as on the falling edge. After the @ sign you can see '(posedge CLK or D)'. What's special about this is

Listing 2. A clocked flip-flop

```
46  always @ (posedge CLK or posedge RESET or posedge SET)
47  begin
48    if (RESET)
49      OUT <= 1’b0;
50    else if (SET)
51       OUT <= 1’b1;
52    else
53      OUT <= D;
54  end
```

that signal D is named without ‘posedge’ in front. This means that the code must be evaluated for every change in the state of signal D.

In the code belonging to this *always* statement, you can see that the compiler first looks at the state of the clock signal. If the clock signal is active ('1'), the output is the same as the input. Otherwise nothing happens, and the current state of the output remains unchanged.

You can compile this example and program it into the CPLD in exactly the same manner as for the previous example. After doing so, use the experimenter’s board to verify that the design actually does what you expect it to do. After this, as an exercise you can see whether you can provide the latch with SET and RESET inputs. As the saying goes, practice makes perfect!

Arithmetic

The previous examples demonstrate how to describe functions without having to worry about logic gates, Boolean algebra and so on. After working with Verilog with a while, every designer will certainly be able to appreciate this. The relatively trivial tasks are handled by the compiler instead of the designer. Our third example shows that arithmetic is also not difficult in Verilog. In this example we use counters. Counters need clock signals, and that’s where we start.

Figure 1 shows the schematic diagram of a standard crystal oscillator. Except for the inverter, all of the components are present on the experimenter’s board. If we now place an inverter between pins 71 and 81 of the CPLD, we have a crystal oscillator.

If you open Example 3, you will see this inverter drawn between two leads of the IC at the top of the schematic. After the CPLD has been programmed, the result is thus an oscillator whose output (pin 81 on the circuit board) is connected to pin 83 of the CPLD. This input is specially intended to act as a clock input.

Arrays

We assume that you are familiar with doing arithmetic in the binary number system. If you are, you are certainly aware that numbers are usually represented by a group of signals. In Verilog, several signals can be conveniently grouped into a structure called an ‘array’.

In the Verilog file for the *Count* block, you will thus see the following in line 40: 'output [7:0] D;'. This specifies an array of eight signals (D[7]...D[0]). This group of signals can be collectively written as ‘D’.

To avoid possible confusion, we have to explain line 47. Here a new signal is declared. This signal is not present in the module declaration, which means that it is not externally visible (outside the module). Such a signal is purely for internal use.

The next interesting line is line 51: 'temp=temp+1;'. This shows that we can count by simply using the + sign. **Table 2** lists additional arithmetic operators that can be used with arrays in Verilog.

In line 52 you can see a comparison (‘==’ means to check whether the left-hand term is equal to the right-hand term). All of the relational operators (such as ‘==’) are listed in **Table 3**. The number 24'd4000000 may appear a bit strange at first glance. This is the notation for writing numbers in Verilog. The first number indicates how many signals are involved (in this case, 24). The ‘d’ indicates that the constant is stated in decimal notation. Finally, ‘4000000’ is the actual constant.

In this case we must use 24 signals, since the register *temp* consists of 24 signals. Note that in Verilog, 24 zeros is not the same as 23 zeros! That means that you must make sure that the same number of signals are present on each side of the ‘==’ symbol.

Counter

If you analyse the Verilog code, you will see that the value of register *temp* is incremented on each clock pulse. As soon as the register reaches the value 4,000,000, register D is incremented by the value ‘1’, *temp* becomes ‘0’ and output *SLOW* becomes ‘1’. If register *temp* has not yet reached the value 4,000,000, *SLOW* receives the value ‘0’. With a 4.000-MHz clock signal, this means that the value of register D is incremented once per second, with output *SLOW* briefly going to ‘1’. *SLOW* is thus a 1-Hz clock signal.

BCD counter

The 1-Hz clock signal goes to the clock input of a BCD counter. This is labelled ‘BCD_counter’ in the schematic diagram.

The Verilog code for the BCD counter has three separate sections, each of which begins with an *always* statement. In addition, we should point out that two registers are declared for internal use: *SEG* and *COUNT.*

The first function (starting at line 52) is a counter that causes the *COUNT* register to count from 0 to 9. When it reaches the value 10, *COUNT* is reloaded with the value 0.

The second function is traversed each time *COUNT* changes. In line 60 you will find a new statement with the name *case.* An example of this can be found in **Listing 3**. C programmers will

Listing 3.
The case statement, and grouping signals into an array

```
59 always @ (COUNT)
60   begin
61   case (COUNT)
62     4’d0 : SEG=7’b1111110;
63     4’d1 : SEG=7’b0110000;
...................................................
71     4’d9 : SEG=7’b1111011;
72     default : SEG=7’b0000001;
73    endcase
74  end
75
76  always @(SEG)
77    {SEGA,SEGB,SEGC,SEGD,SEGE,SEGF,SEGG} = SEG;
```

find this a familiar concept. In this instance, the case statement has one argument *(COUNT)*. This means that the following lines:

"4'd0 : SEG=7'b1111110;
4'd1 : SEG=7'b0110000;"

can be translated as:

if (COUNT==4'd0) SEG=7'b...
else if (COUNT ==4'd0 SEG=..."

and so on.

The line starting with *default* is processed if the current value of *COUNT* does not appear in the list.
The last section of the Verilog code starts at line 75. As you can see, it is evaluated if the value of register *SEG* changes.
An interesting feature of Verilog can be seen in line 76: several signals can be grouped into an array by using the '{' and '}' characters. As *SEG* is an array and the outputs for the seven-segment display have been declared as individual signals, they must be combined into an array.
Another possible solution would be to couple each signal to an element of the array, for example by using:

"SEGA = SEG[6];"

This method works just as well, but it would make the code quite a bit longer and thus more difficult to read.

Testing

With regard to testing this example, we must mention a shortcoming of the experimenter's board.
When the CPLD is being programmed, all of its outputs assume the non-active state. Immediately after being programmed, the CPLD is active. As a result, the oscillator does not start properly. This means that after programming the CPLD, you have to briefly reset it. You can do this by fitting a pushbutton switch with a make contact between pins 2 and 20 of connector K6. Alternatively, you can briefly switch the power off and then on again.

Conclusion

You can do a lot more with the Verilog language than what we've been able to describe in this article. Here we have limited ourselves to the most commonly used features of Verilog. Still, these features allow users to design quite complex digital functions. Various sites dealing with a wide variety of designs using Verilog can be found on the Internet. By studying and simulating these designs, you can quickly accumulate experience with this interesting language.
An important aspect that we were unable to discuss in this article (due to lack of space) is simulating designs in Quartus. The Quartus tutorial should help you quite a bit in this regard, and there is also always the Help function. Incidentally, it's a good idea to develop a design in small parts and simulate these small 'subdesigns' one by one in order to determine whether they work the way they should.
We would appreciate hearing from readers who have created their own interesting applications using the experimenter's board. Good luck!

(030385-2)

Web pointers

http://www.altera.com/support/examples/verilog/verilog.html
http://www.asic-world.com/verilog/index.html

Tips

- A module can consist of several code segments, each of which is proceeded by the statement 'always'. Here the limitation is that an output can only be defined in a single code segment. To get around this limitation, a designer can define two signals (such as wires T1 and T1) and control these two signals using separate code segments. The actual output can then be defined as a Boolean function of these two signals.

- Ensure that all flip-flops in the design respond to the same signal edge (rising or falling). This yields a more efficient (faster) design.

- Bear in mind that the compiler evaluates the code. This is fundamentally different from what happens when a processor executes a segment of code. If a design contains several segments using the 'always' keyword, these segments will actually be executed simultaneously

Smooth Ope

for model railway turnouts and semaphores

Ray King

In many model railway layouts, electromagnets instantly snatch turnouts and semaphores from one position to another with an associated 'clunk' noise. Armed with a PIC micro and a small servo, the circuit described in this article provides much slower, smoother, quieter and hence more realistic operation of these mechanical devices.

rator

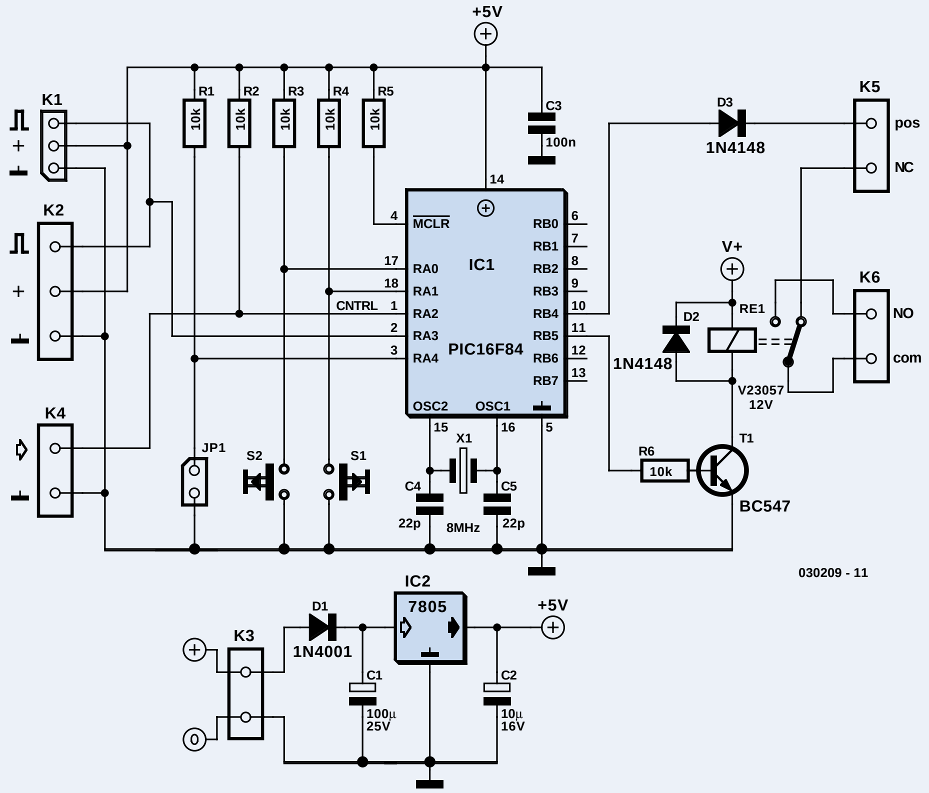


Figure 1. At the heart of the circuit we find the trusty PIC16F84 microcontroller.

There are a large number of model railway builders using proprietary track and points to realise their layouts. When adding remote controls for points (turnouts) there is often disappointment with the unrealistic and noisy movement of the mechanism. There are commercially available motorised units but these are expensive as well as difficult to install. A further disadvantage of commercial units is that the force they use is often detrimental to fine scale turnouts in that it can easily cause damage if not precisely adjusted and maintained.

On a different track

The proposed design uses standard radio-control (R/C) servos to drive the turnouts. A PIC microcontroller is used to generate the necessary PWM pulses under the control of a simple on/off switch. The circuit includes a facility to switch the 'frog' polarity depending on the position of the turnout. In railway terminology, a frog is 'a grooved piece of iron at place in railways where tracks cross'. There is also a signal returned to the operator to indicate correct operation of the unit for display on a track diagram, for example. The turnout will move slowly and smoothly from one position to the other with no excess travel or brute force that will damage the turnout. If you are interested in making your own pair of points (also known as a 'turnout') then visit the webpage listed at the end of this article — it also shows a 'frog'. Since control is by a single on/off switch using standard TTL levels, it is also simple to interface the unit to an automatic or computer-controlled system (like the one described elsewhere in this issue).

So why use a PIC when the design could be accommodated with some simple hardware? The present design uses very few components, is easy to set up and the speed of movement can be adjusted by changing parameters in the code.

Circuit description

Figure 1 shows the circuit diagram of the smooth control for model railway turnouts. Basically, a PIC microcontroller sits between input and output connectors. The inputs include the 'turnout change' control signal supplied by the master control unit or a simple switch on your control panel. This signal arrives at PIC input RA2 via connector K4. The two other input devices are pushbuttons S1 and S2 which are read via PIC port lines RA1 and RA0 respectively. They are used to set the servo action required to make the turnout reach its two extreme positions. Looking at output connectivity, we find the servo control pulses being supplied by PIC port line RA3 and fed to the servo by way of connector K1 and /or K2. Port line RA4 is read to detect the presence or absence of jumper JP1 which selects between two memory settings (more about this further on). Port line RB4 supplies control information about the turnouts status, for use

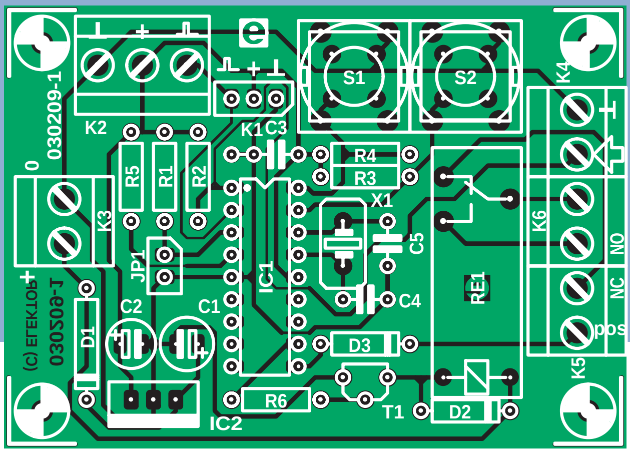


Figure 2. Component overlay of the PCB designed for the turnouts control. The board is available ready-made through The PCBShop.

Free Downloads

PIC microcontroller software.
File number: 030209-11.zip

PCB layout in PDF format.
File number: 030209-1.zip

www.elektor-electronics.co.uk/dl/dl.htm, select month of publication.

Web pointer

How to build a pair of points (turnouts) & pictures of a model 'frog': www.worldrail-fans.org/GardenRailway/SLRPointBuilding.shtml

COMPONENTS LIST

Resistors:
R1-R6 = 10kΩ

Capacitors:
C1 = 100µF 25V radial
C2 = 10µF 16 V radial
C3 = 100nF
C4,C5 = 22pF

Semiconductors:
D1 = 1N4001
D2,D3 = 1N4148
T1 = BC547
IC1 = PIC16F84-10P, programmed, order code **030209-41**
IC2 = 7805

Miscellaneous:
JP1 = 2-way pinheader with jumper
K1 = 3-way pinheader
K2 = 3-way PCB terminal block, lead pitch 5mm
K3-K6 = 2-way PCB terminal block, lead pitch 5mm
S1,S2 = pushbutton with 1 make contact, e.g., type D6-R
X1 = 8MHz quartz crystal
RE1 = relay, PCB-mount, 12V SPDT, e.g., Siemens V23057
PCB, available through **The PCBShop**
Disk, Proton PIC Basic Plus source code, order code **030209-11** or **Free Download**

as a feedback signal to the master control system. RB5 effectively controls a relay for use with a 'frog'. Depending on its mechanical structure, the 'frog' employs the normally open (n.o.), normally closed (n.c.) contact, or both.

The circuit has an on-board 5-V supply regulator, IC2. The input voltage range should not exceed about 12 VDC.

The PIC has a standard quartz complement in its oscillator circuit consisting of an inexpensive 8-MHz quartz crystal X1 and the two usual small loading capacitors, here C4 and C5.

Printed circuit board

The printed circuit board (PCB) for the turnouts control has been spaciously laid out. What's more, it contains standard size components only. The component overlay is given in **Figure 2**. Easy to use PCB terminal blocks with 5 mm pin spacing are used for the connectors, except K1 which is a 3-pin pinheader for use with ready-made servo cables. However, in all cases where you are not certain about the servo connections, use terminal block K2 instead. The completed and tested board should be mounted out of sight, which in nearly all cases will mean securing it to the underside of the model railway tabletop. If you do not need 'frog' control, then components R6, T1, D2, K6 and Re1 may be omitted.

Setting up

There are two calibration pushbuttons, S1 and S2. Press them simultaneously and the servo will adopt a central position. The turnout is held in its central position and attached to the servo's operating arm. Make the control input (RA2) logic High and press S1 and S2 individually to adjust the servo's extreme CCW (counter clockwise) position. Next, make RA2 logic Low and again press S1 and S2 individually until the desired extreme CW (clockwise) position is reached.

If necessary, repeat the adjustments for each level of the control signal, until the servo drives accurately and smoothly from one position to the other. The settings are automatically stored in the PIC's internal EEPROM. Jumper JP1 directs the PIC to use an alternative memory location permitting two settings to be used. Its use is optional.

Software

The source file (.BAS) was written in Proton PIC Basic Plus and should not be too difficult to convert to other PIC compilers. The Proton environment also generates an assembler file which, together with the Basic listing should provide enough clues to adapt the program and assemble it with your favourite assembler for the PIC16F84.

The circuit could also be used to operate semaphore signals. By adjusting the parameters for operating speed and/or modifying the program code it should even be possible to mimic the 'bounce' of the semaphore arm as it rises and falls.

(030209-1)

Location photograph courtesy South Limburg Steam Railway Foundation (www.zlsm.nl)

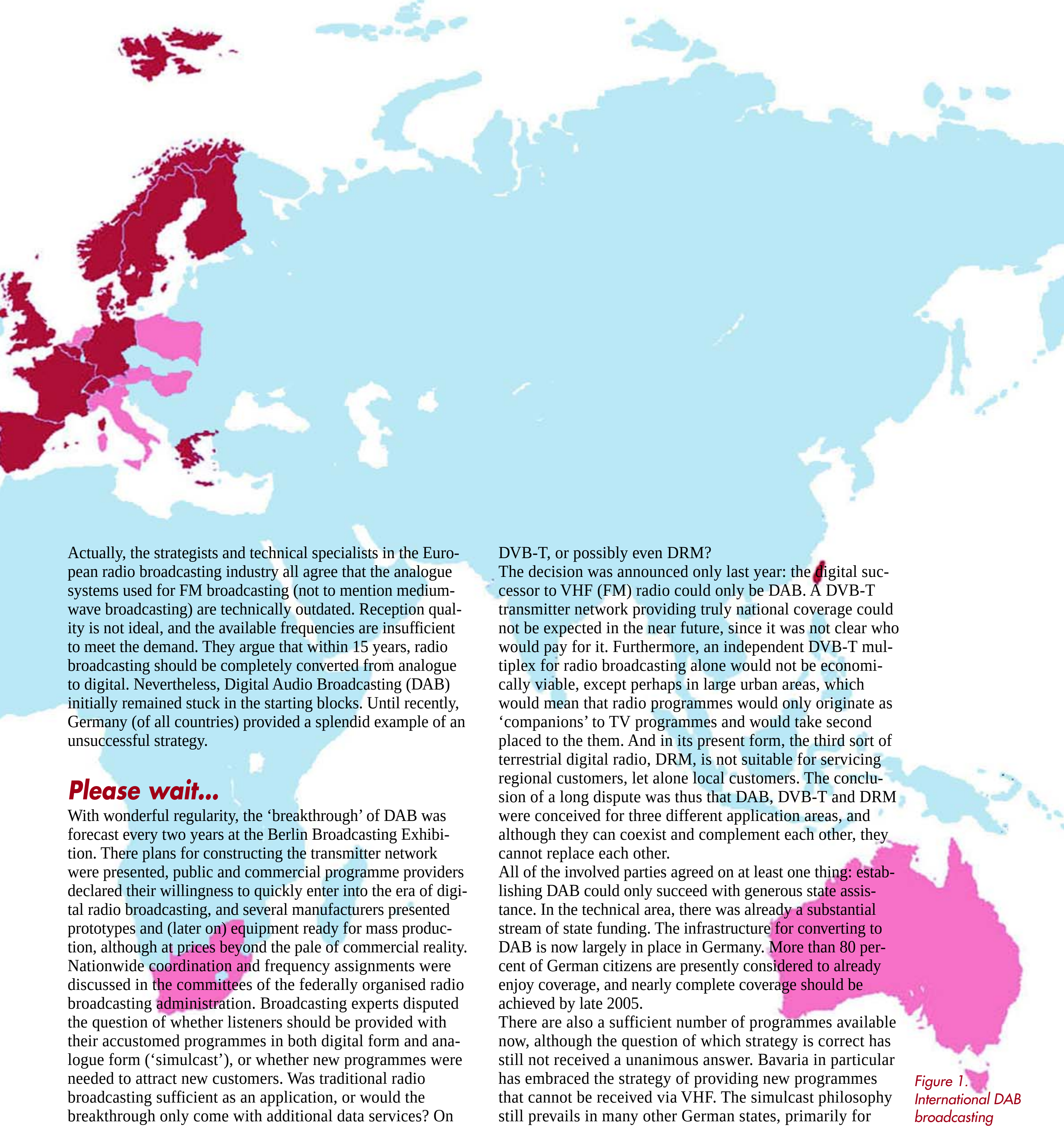
TEN YEARS AFTER

DAB in Europe

🟥 Operational Services
▨ Pre-Operational Services

Hans Weber

Preparations for the market introduction of DAB digital radio have been underway in Europe for more than ten years. Although setting up the transmitter network proceeded rapidly in most countries of western and central Europe, sales figures in the receiver market were initially modest. In the UK, sales came up to speed starting in late 2002, and recently there are increasing signs that an upturn could also occur on the Continent.



Actually, the strategists and technical specialists in the European radio broadcasting industry all agree that the analogue systems used for FM broadcasting (not to mention medium-wave broadcasting) are technically outdated. Reception quality is not ideal, and the available frequencies are insufficient to meet the demand. They argue that within 15 years, radio broadcasting should be completely converted from analogue to digital. Nevertheless, Digital Audio Broadcasting (DAB) initially remained stuck in the starting blocks. Until recently, Germany (of all countries) provided a splendid example of an unsuccessful strategy.

Please wait...

With wonderful regularity, the 'breakthrough' of DAB was forecast every two years at the Berlin Broadcasting Exhibition. There plans for constructing the transmitter network were presented, public and commercial programme providers declared their willingness to quickly enter into the era of digital radio broadcasting, and several manufacturers presented prototypes and (later on) equipment ready for mass production, although at prices beyond the pale of commercial reality. Nationwide coordination and frequency assignments were discussed in the committees of the federally organised radio broadcasting administration. Broadcasting experts disputed the question of whether listeners should be provided with their accustomed programmes in both digital form and analogue form ('simulcast'), or whether new programmes were needed to attract new customers. Was traditional radio broadcasting sufficient as an application, or would the breakthrough only come with additional data services? On top of this came the great 'religious war': was it to be DAB, DVB-T, or possibly even DRM?

The decision was announced only last year: the digital successor to VHF (FM) radio could only be DAB. A DVB-T transmitter network providing truly national coverage could not be expected in the near future, since it was not clear who would pay for it. Furthermore, an independent DVB-T multiplex for radio broadcasting alone would not be economically viable, except perhaps in large urban areas, which would mean that radio programmes would only originate as 'companions' to TV programmes and would take second placed to the them. And in its present form, the third sort of terrestrial digital radio, DRM, is not suitable for servicing regional customers, let alone local customers. The conclusion of a long dispute was thus that DAB, DVB-T and DRM were conceived for three different application areas, and although they can coexist and complement each other, they cannot replace each other.

All of the involved parties agreed on at least one thing: establishing DAB could only succeed with generous state assistance. In the technical area, there was already a substantial stream of state funding. The infrastructure for converting to DAB is now largely in place in Germany. More than 80 percent of German citizens are presently considered to already enjoy coverage, and nearly complete coverage should be achieved by late 2005.

There are also a sufficient number of programmes available now, although the question of which strategy is correct has still not received a unanimous answer. Bavaria in particular has embraced the strategy of providing new programmes that cannot be received via VHF. The simulcast philosophy still prevails in many other German states, primarily for cost reasons.

Figure 1. International DAB broadcasting (source: World DAB Forum).

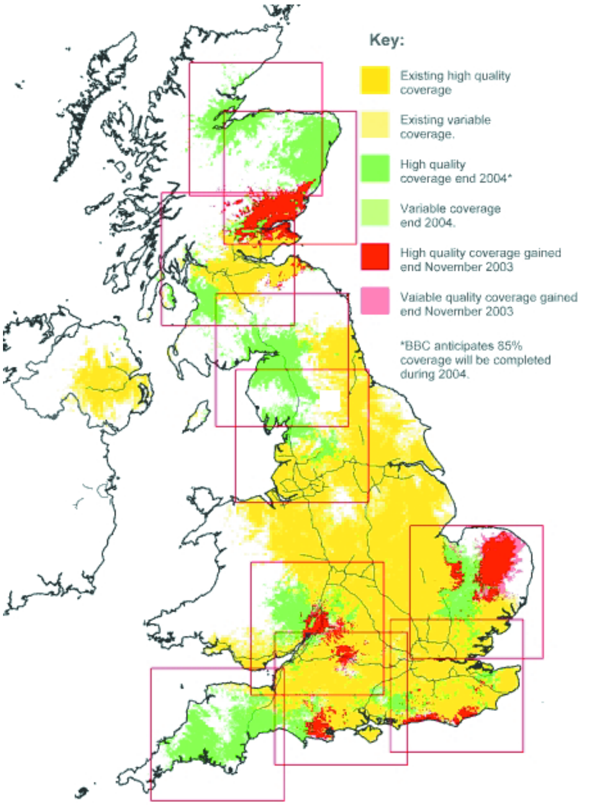

Key:
Existing high quality coverage
Existing variable coverage.
High quality coverage end 2004*
Variable coverage end 2004.
High quality coverage gained end November 2003
Vaiable quality coverage gained end November 2003
*BBC anticipates 85% coverage will be completed during 2004.

Figure 2. DAB coverage in the UK.

Figure 3. The Restek EDAB high-end DAB tuner (www.restek.de).

Figure 4. Compact DAB receivers for the 'German' L band are available at less than 200 euros (www.thiecom.de). The receiver on the right can also receive VHF broadcasts.

Consumers have apparently not gained very much from all of this: the number of receivers sold up to now is estimated to be around 100,000. That is presumably because there is not yet sufficient awareness of the product. Most Germans regard DAB as a brand of beer, and the trademark currently being used, 'Digital Radio', does not provide adequate differentiation from other types of digital radio broadcasting. As the marketing organisations to be found in almost all of the German states explain, 'First we invested a lot of money in the technology, and now we have to invest even more in promotion'. Success will still take a while.

The 99-quid radio

The situation in Great Britain is quite different. Here the market started to move in late 2002. It was recognised that the only way to achieve widespread market penetration of DAB receivers was via the price. The 'magic boundary' was considered to be a receiver under the 100-pound level. The initiative for producing a '99-quid kitchen radio' did not come from the manufacturers or a similar group, but instead from the programme providers. The BBC and commercial broadcasters, in particular Digital One, provides funding for developing an IC named 'Chorus'. The manufacturing cost of this IC was so low that the receiver price could be brought below the magic boundary, although at first this was only possible with additional subsidies. Shortly before Christmas 2002, the Pure Digital Evoke-1 went on sale in selected shops following an extensive advertising campaign, and it was sold out after only a few hours. This 'initial spark' encouraged other manufacturers to join the competition with attractively priced offerings. Not only did this drive down prices (presently, the cheapest receiver costs around 70 pounds, equivalent to 105 euros), it also increased awareness and attracted other customers who were willing to pay somewhat higher prices.

The number of DAB receivers sold in the UK has risen from approximately 50,000 in mid-2002 to the present figure of more than half a million — and that with a degree of coverage originally comparable to that in Germany, and now even somewhat lower.

Of course, the initial situation in Great Britain was also different. Medium-wave broadcasting is much more common here than elsewhere, so the difference in reception quality is more evident than with FM broadcasting. In addition, greater emphasis was placed on 'new' programmes (which cannot be received using analogue receivers), and they were promoted in the media.

Another advantage relative to Germany is that in the UK, DAB is available in only one frequency band (Band III, 174–240 MHz). In Germany, DAB is also transmitted in the L band (1452–1492 MHz), which unfortunately increases costs and reduces transmitter range.

West Europe: a varied situation

Nearly complete coverage has been established in Belgium, which has the highest level of coverage (95 %) in all of Europe.

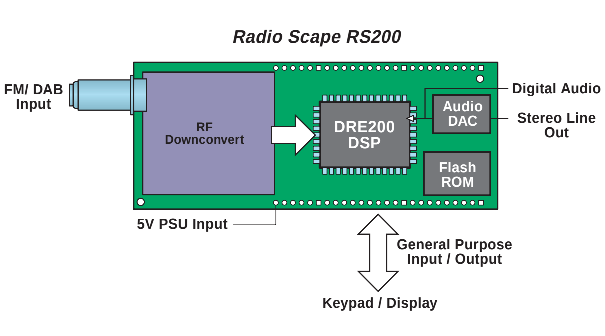
In France, the focus was on large metropolitan areas from the very beginning. DAB programming is available in Paris, Lyon, Marseille, Toulouse and Nantes, and in total 25 to 30 percent of the population can be reached. In the Nether-

DAB inside

If you look for components for DAB receivers, you will quickly find two British manufacturers: FrontierSilicon (www.frontier-silicon.com) and RadioScape (www.radioscape.com). FrontierSilicon's product line is based on the Chorus FS1010. This IC is a highly integrated multimedia processor with ADC input, a DSP processor core, DAB coprocessors, on-chip RAM and cache memory, and a wealth of on-board peripheral. The DAC is not integrated, so users are free to spend as much as they wish on this component (or as little as they wish, if cost is an objective). For equipment manufacturers, FrontierSilicon also provides convenience products in the form of complete modules that include the RF front end (tuner) and several peripheral components. To produce a complete receiver, the manufacturer only has to add his own user interface (control processor, display and buttons), audio portion (DAC, buffer, sockets or Toslink, etc.), and an enclosure.

RadioScape goes a step further with its tiny RS200 Module Board, which uses the Texas Instruments DRE200 DAB IC and incorporates all of the functions of a DAB/FM digital radio. As you can see from the illustration, a complete DAB/FM tuner can be made by simply adding a few pushbuttons, a rotary encoder, a standard LCD module (2 × 20 characters) and a power supply. With its compact dimensions, the RadioScape module can also be used to build pocket radios.

It's certainly not much of a secret that most DAB receivers are fitted with such modules. As an example, the photos show what's inside the DAB tuner from Restek's Audio Mini Module series (www.restek.de).



Almost plug-and-play: the RadioScape RS200 DAB/FM tuner module (www.radioscape.com).

The module used in the DAB tuner in Restek's Audio Mini Module series.

lands, regular DAB operation was only started on 27 February 2004 after a five-year test phase, which surprisingly made the Netherlands the last of the pre-expansion EC countries to introduce regular DAB service. There as well, availability is concentrated in regions with high population density, and according to official statements, approximately 40 percent of the population is currently covered.

All of these countries have in common that the number of receivers sold lags behind the technical investments in the transmitting network. This may in part be due to the fact that in the beginning, relatively small markets are not especially attractive for large manufacturers.

The receiver market

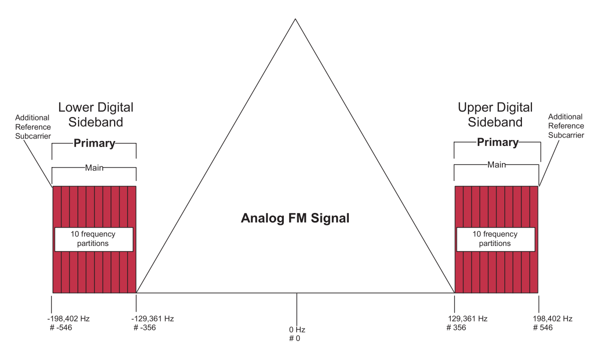
The end-user market is divided into four major segments. The first of these is car radios, which in Germany was initially regarded as the most important market segment. Prices in this segment are relatively stable. The market leader, Blaupunkt, offers its Woodstock 53 (!) model for 579 euros. The Grundig Allixx is somewhat less expensive at 399 euros. Prices do not vary much from one country to the next. Only the UK is again slightly ahead of the curve, with a model (Goodmans) available for less than 200 pounds.

The situation with mobile and portable receivers is different. In the UK, the market is dominated by small, inexpensive receivers that can only receive Band III. Portable receivers are available starting at around 70 pounds. By contrast, prices in Germany maintained a lower limit of approximately 200 euros until recently. This situation changed at the CeBIT exhibition, where TechniSat presented a pair of lower-priced receivers: the DAB-Man for 169.99 euros and, for home use, the Digit-Radio DAB for 159.99 euros. Visitors to the CeBIT could also admire the first combined DAB/DRM receiver, the Starwaves Prelude. This receiver, which is made by a small manufacturer in Hanover, will not become commercially available until late this year. The price is said to be 'in the high three-figure' euro range.

In Germany, DAB tuners for hi-fi systems are still the domain of the high-end manufacturers, such as Restek

Things are different in the USA:

HD radio instead of DAB



The hybrid HD Radio signal consists of a frequency-modulated main carrier (for FM programmes) and digitally modulated sidebands for HD Radio.

The DAB article in the April 1998 issue of Elektor Electronics reported that 'in the USA, the National Association of Broadcasters (NAB) opposes the introduction of DAB and has come out in favour of In-Band Digital Radio for operation in the VHF and medium-wave bands'. At that time, no practical method had been found to implement such a solution. The situation has changed since then: under the name 'HD Radio', iBiquity Digital Corporation (www.ibiquity.com) developed a technique that allows digital programmes to be broadcast on existing AM and FM frequencies using existing transmitters. This can be done in combination with conventional AM and FM programmes, which can continue to be broadcast on the same frequencies. All that is necessary to enjoy the improved sound quality and data services of HD Radio is to buy a new receiver with HD Radio capability. CD quality can be achieved with FM reception, while AM programmes have the same quality as current FM broadcasts. In both cases, noise-free reception is provided along with supplemental data functions. Most HD Radio receivers can also receive normal AM and FM broadcasts.

Before even a single receiver was commercially available, iBiquity managed to grant licenses to more than 280 radio stations in 37 US states. Component manufacturers, such as Alps (front-end tuners, A/D and D/A converters, and signal processors) and Philips (SAF3350 HD Radio Processor) plan to start mass production this year.

(www.restek.de). Particularly demanding listeners can spend as much as 2700 euros for such a DAB tuner, while in the UK an Acoustic Solutions DAB/FM tuner for stereo systems can be obtained for as little as 115 pounds. Computer-based DAB radios, by contrast, are scarce. The only model available in all countries is the TerraTec DR Box 1 (300 euros or about 210 pounds).

Summaries of available equipment and prices can easily be found on the Internet, such as at www.digitalradio.de for Germany and www.digitalradionow.com for the UK.

What about the future?

In Germany, DAB has been pronounced dead once already. Presently, the situation can be regarded as being more positive. It appears that prices could again drop in the near future: the Anglo-Israeli company Sonarics is offering their CSM DAB module for US$ 25, and the Analog Devices Blackfin DSP, which allows DAB functionality to be implemented in software, is available at 5 dollars in large quantities. At last, the large manufacturers (in particular Sony) are indicating their willingness to enter the market. In parallel with this, promotional efforts are being increased and coordinated under the direction of Initiative Marketing Digital Radio (IMDR). In any case, in Germany hopes are now being pinned on a 'soft transition' instead of a 'breakthrough'. The demand that VHF radio broadcasting simply be stopped by 2015 (or even earlier) has now been dropped.

(040101-1)

References:

'Digital Audio Broadcasting (DAB)', Part 1 / Part 2, Elektor Electronics, March / April 1998.

Web pointers:

www.worlddab.org

www.digitalradio.de

www.radionumerique.be

www.pure-digital.com

www.drdb.org

www.digitalradionow.com

www.restek.de

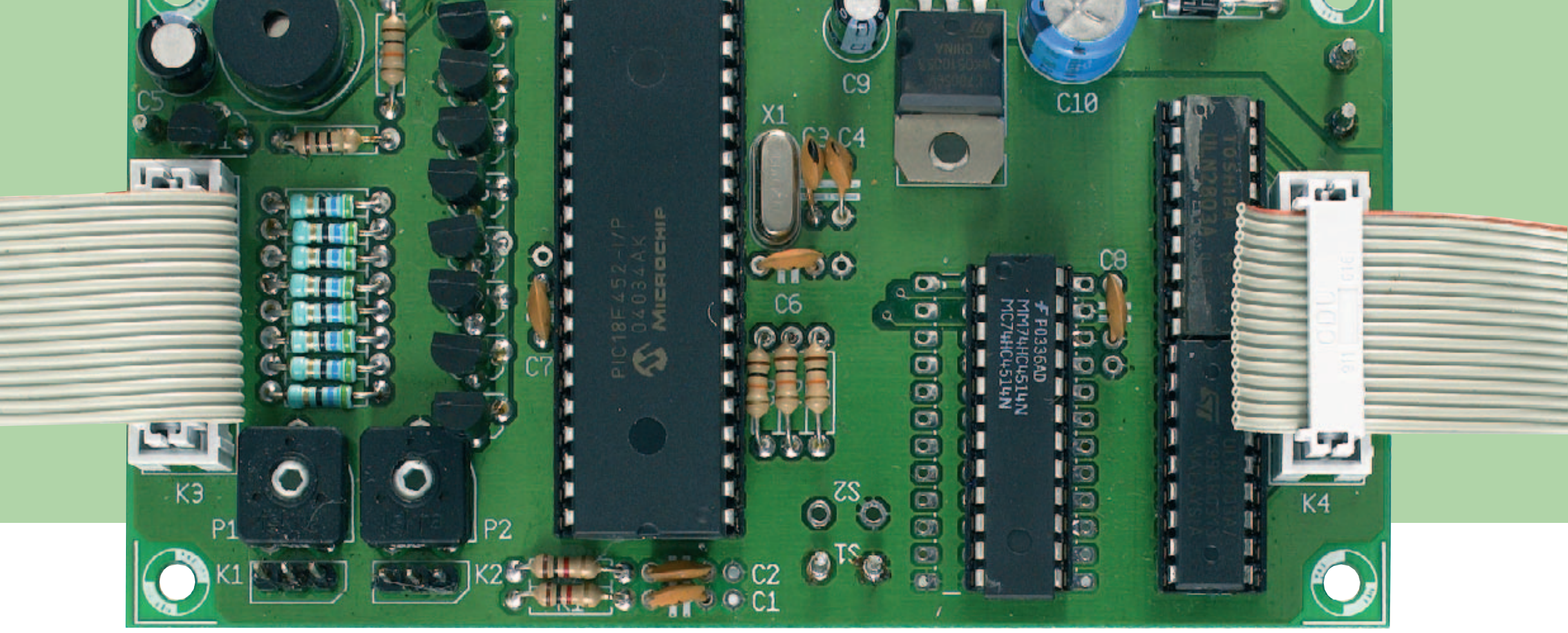
www.sonarics.com

www.thiecom.de

www.technisat.de

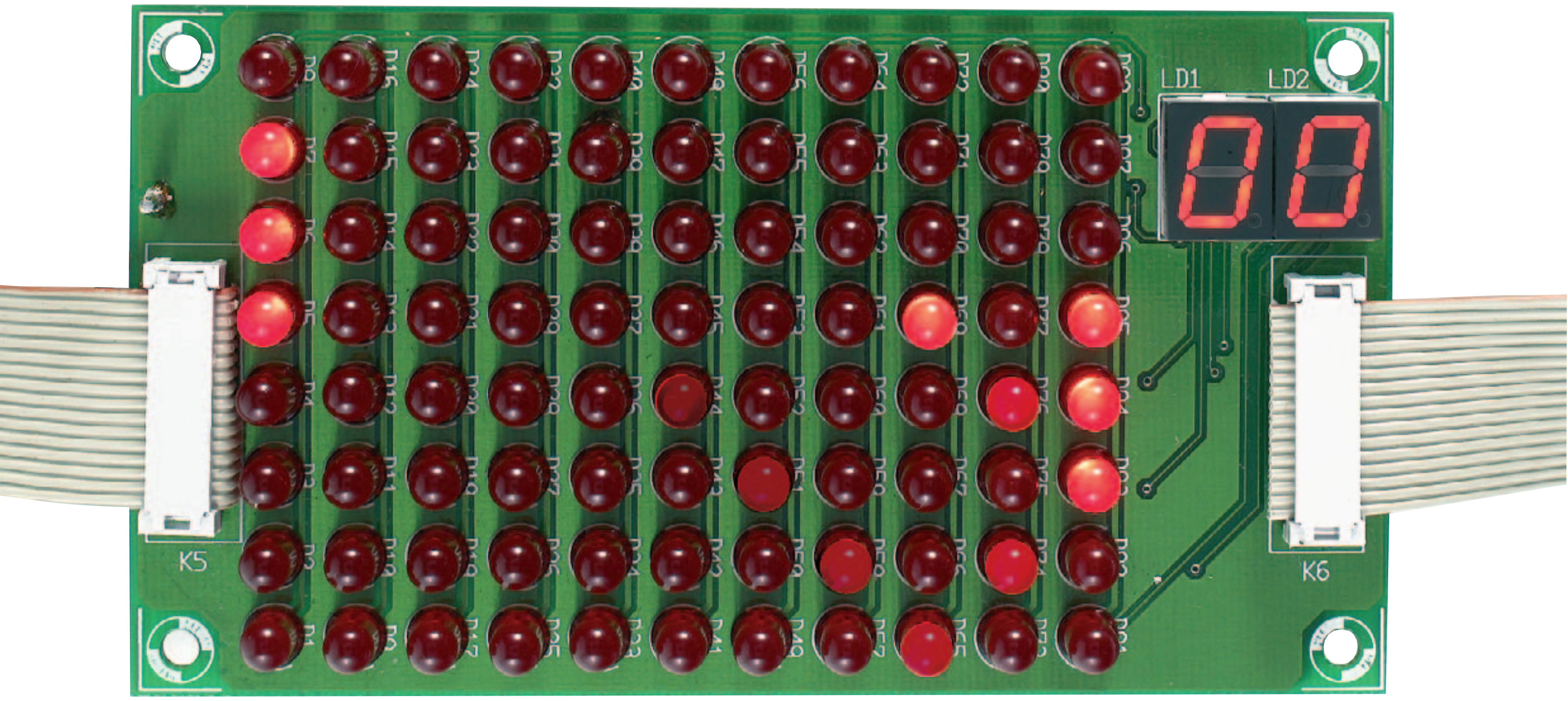
www.radioscape.com

www.frontier-silicon.com



Pocket Pong

a primeval game cast in modern hardware



Provided they manage to recognise them in the first place, youngsters will label classics like Pacman and Pong as video games although historically they are 'video games', the concept behind them dating back to the 1950's. in this article you'll find a modern (computer) version of such a prehistoric game that — as far as we are concerned — has not lost any of its compulsive character.

It is easily forgotten that the first electronic games were played on a TV set. In fact, technology at the time nearly did not make it to TV altogether. In 1951, TV technician Ralph Baer thought it would be nice to use the telly as a screen for an electronic game. His boss however did not see the promise and the idea was quickly abandoned. Years later, however, it started to surface again and in 1966 Baer started to build one of his early prototypes. The video game was born. The game covered by this article is Atari's 'Pong' which is actually a derivate of one of Baer's original concepts. The first versions for use at home were designed around 1974. As opposed to other manufacturers, Atari found the pot of gold: an ASIC (application specific integrated circuit) was designed for Pong. The chip allowed the production costs to be kept low while the game functionality (including a digital on-screen scoreboard and sound effects) was excellent compared to competitive products. The home version of Pong was launched in 1976. Today, 28 years later, we have another go at casting Pong in electronics. This time, we will not be using a TV set for the 'screen' but a LED matrix.

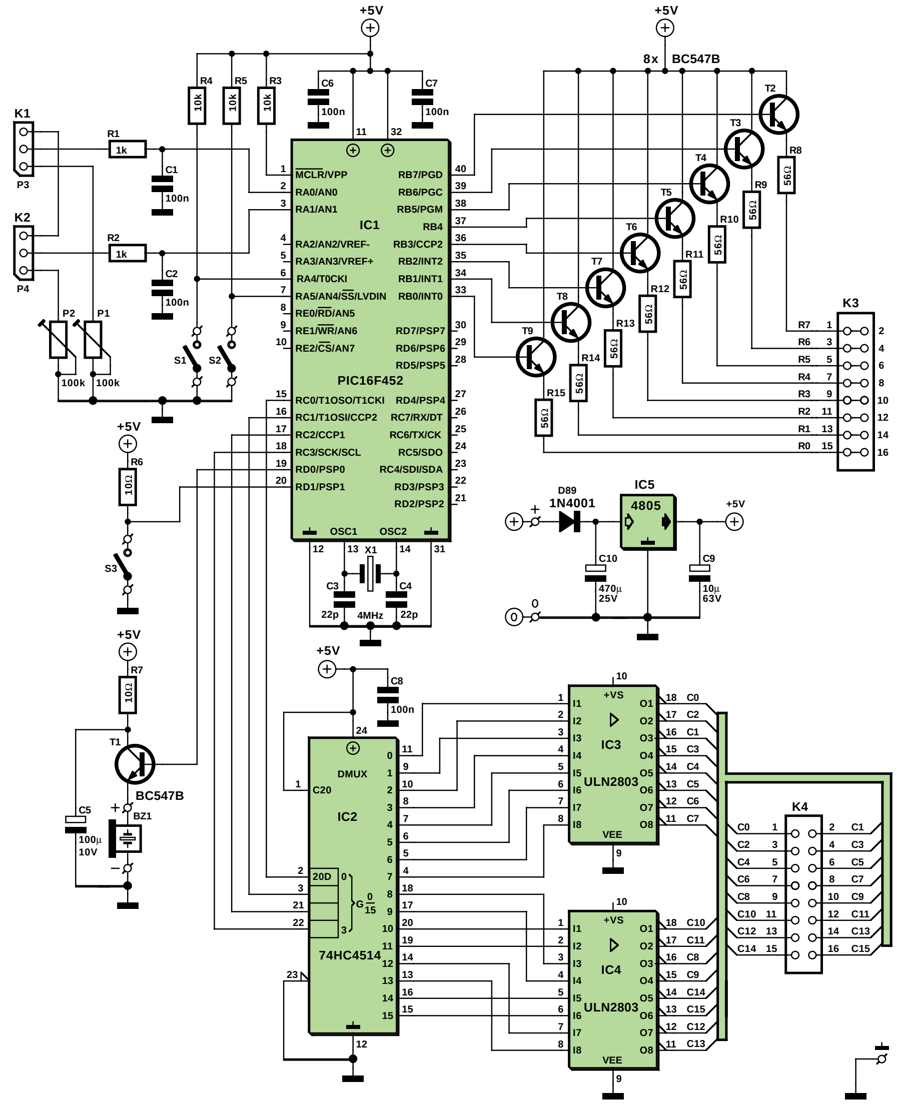


Bild 1a. Der Mikrocontroller als Schiedsrichter.

The circuit

Just like Atari did many years ago, we will be designing a chip tailored to the game only. Fortunately, that no longer means you have to design a completely new circuit and burn it into a chip. Today we simply use a microcontroller running software that tells it exactly what to do. We chose the PIC18F452, a 40-pin MCU containing, among others, 32 kBytes of program memory and a 10-bit A-D converter. As you can see from **Figure 1**, the PIC is not the only IC in the circuit. IC2, a 4-to-16 line decoder, together with IC3 and IC4 arranges the display control. The display actually consists of two parts: the 7-segment displays LD1 and LD2 showing the 'score' and a LED matrix (D1-D88) that mimics the playing field. Virtual rackets or bats move at the left and right side of the court, allowing the ball to be bounced back and forth. Buzzer Bz1 provides the sound effects. You are looking at a dc (or 'active') piezo buzzer that's driven by transistor T1. C5 and R7 afford sufficient decoupling of the supply voltage.

The power supply around IC5 is dead standard. Diode D89 affords a degree of protection against an accidentally reverse polarized mains adapter (with 9-12 VDC output). With the PIC drawing just a few milliamps, it is fair to say that the current consumption of our electronic game goes on account on the LEDs. However, thanks to the multiplexed drive

Components list

Resistors:
R1,R2 = 1 k
R3...R6 = 10 k
R7 = 10 Ω
R8...R15 = 56 Ω
P1,P2 = 100 k Trimmpotentiometer

Kondensatoren:
C1,C2 = 100 n
C3,C4 = 22 p
C5 = 100 µ/10 V stehend

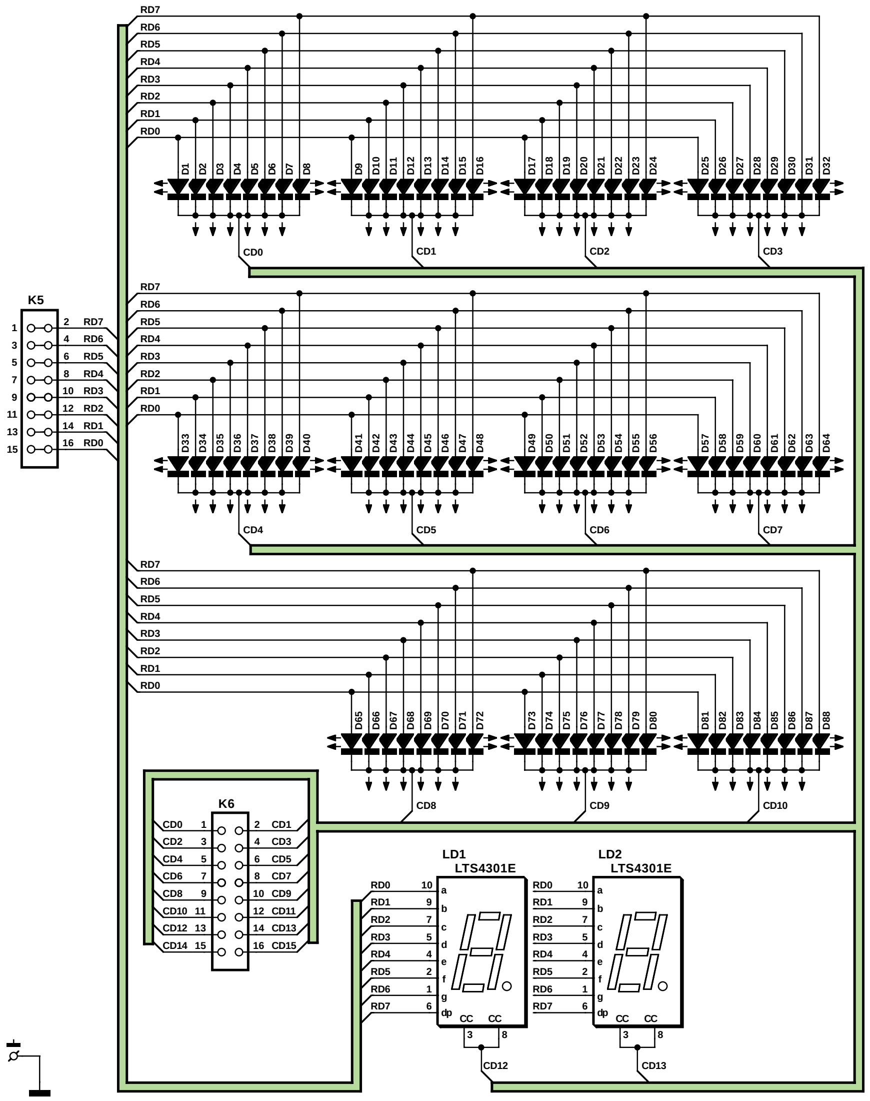


Figure 1. In this game, the 'screen' is formed by a huge number of LEDs.

scheme used here, the average current consumption remains limited to a modest 35 mA or so.

Display

Both the LEDs in the 7-segment displays and the LEDs in the matrix have their cathodes connected via ULN2803 driver ICs (IC3 and IC4). The anodes are connect to the MCU port lines via transistors T2-T9. It would appear that the transistors are not strictly necessary as the PIC port lines are specified at 25 mA each. This may well be sufficient for high-efficiency LEDs, but it isn't with regular LEDs which at such a low current light dimly, reducing the 'playability' of the game.

Using the indicated component values (i.e. with transistors and R8-R15 = 56 ohm) a LED current of about 27 mA is obtained. By the way, the value of R8-R15 may be changed without problems using Ohm's law. Assume a supply voltage of 5 V, then subtract the following: collector-emitter drop (0.7 V); LED 'on' voltage (approx. 1.8 V for red LEDs); voltage drop across Darlington drivers in the ULN2803 (approx. 1 V). Tha leaves about 1.5 V across the resistor. If the desired current is 10 mA, V = I x R tells you that 1.5 = 0.01 x R, or 1.5 / 0.01 = 150 ohms.

Operation

The game is played using two potentiometers and two switches. S1 serves to serve a ball. S3 is the speed selector. Whenit is closed, the ball moves faster making the gamne more difficulkt to play. There is a connection for a thirs switch (S2) but hthis is not used in the Pong game. P3 and P4 may be ordinary roraytry potentiometers but slide pots will of course give a more realistic control of the bats on the field. A real joystick is of coursed the ultimate.

'Analogue' PC joysticks in general contain two potentiometers, one for each direction (horizontal and vertical). In most cases 470-kohm pots are used of which the 0-120 kohm resistance range is actually used. For our circuit, a resistance range of 0-4.7 kohm is required, so if a joystick is connected, a resistor has to be connected in parallel with the input (between +5 V and pin 2 / 3 of the PIC) to make sure a much lower resistance is obtained. The equivalent resistance of the parallel network is calculated from

$$1/_{\text{Req}} = 1/R1 + 1/R2$$

So, if we want 4.7 kohms and the joystick R1 = 120 kohms then

C6...C8 = 100 n
C9 = 10 µ/63 V stehend
C10 = 470 µ/25 V stehend

Halbleiter:
D1...D88 = rot; High-efficiency-LED, 5 mm, z.B. HLMP-D101 von HP (Farnell-Nr. 323-044)
D89 = 1N4001
LD1, LD2 = LTS4301E (LiteOn)
T1...T9 = BC547B
IC1 = PIC18F452-I/P (programmiert, siehe Text)
IC2 = 74HC4514 (auch 74HCT4514 oder 4514)
IC3, IC4 = ULN2803
IC5 = 4805

Außerdem:
K1 = P3 = 4k7 Potentiometer, linear, Mono, + 3-polige Stiftleiste
K2 = P4 = 4k7 Potentiometer, linear, Mono, + 3-polige Stiftleiste
K3, K4 = 16-polige Stiftleiste, zweireihig, mit Schutzkragen
K5, K6 = 16-poliger Flachkabel-Konnektor für Platinenmontage
S1 = Drucktaster 1 x Schließer
S2 = nicht vorhanden
S3 = Schalter 1 x Schließer
X1 = 4-MHz-Quarz
BZ1 = DC-Buzzer 5 V
Platine 030320-1 (Layout-Download von www.elektor.de)
Software 030320-11 (Download von www.elektor.de)

1/R2 = 1/120k – 1/4.7k
R2 = 4.89 k

In practice no problems will occur if you use a resistor of 4.7 kohms.
Pins 1 and 3 on the 15-way joystick connector (a sub-D type) are for the horizontal direction and pins 1 and 6 for the vertical direction.
Finally, R1/C1 and R2/C2 suppress noise generated by the potentiometers as they are operated.

Construction

The printed circuit board (**Figure 2**) consists of two parts interconnected with a piece of flatcable. Building the LED matrix is sure to take some time. Although the job itself is straightforward, we should emphasize the importance of checking the LED polarity because it is hard to think of anything more annoying than 88 LEDs fitted the wrong way around. Usually, the cathode is the largest surface inside the LED as well as the shorter pin. Usually… not always, so make sure of the polarity and in case of doubt use a conductance tester.
The orientation of the 7-segment displays may appear to be wrong but if you follow the indications on the component overlay shown in Figure 2 the circuit will work as expected.
The polarity of the electrolytic capacitors and transistors in the circuit also deserve your attention. Also, run a double check on the orientation of the PIC micro before inserting it into its socket — after all, the PIC is the most expensive component.
The circuit board is best mounted into a case that will also accommodate the switches, mains adapter socket, pots and LEDs. A red bezel on top of the matrix clearance and the score displays will provide the finishing touch to the game.
If necessary the game may be powered by four AA batteries. In that case the current through the LEDs has to be reduced, however, by increasing the value of R8-R15. This is necessary to save battery power.

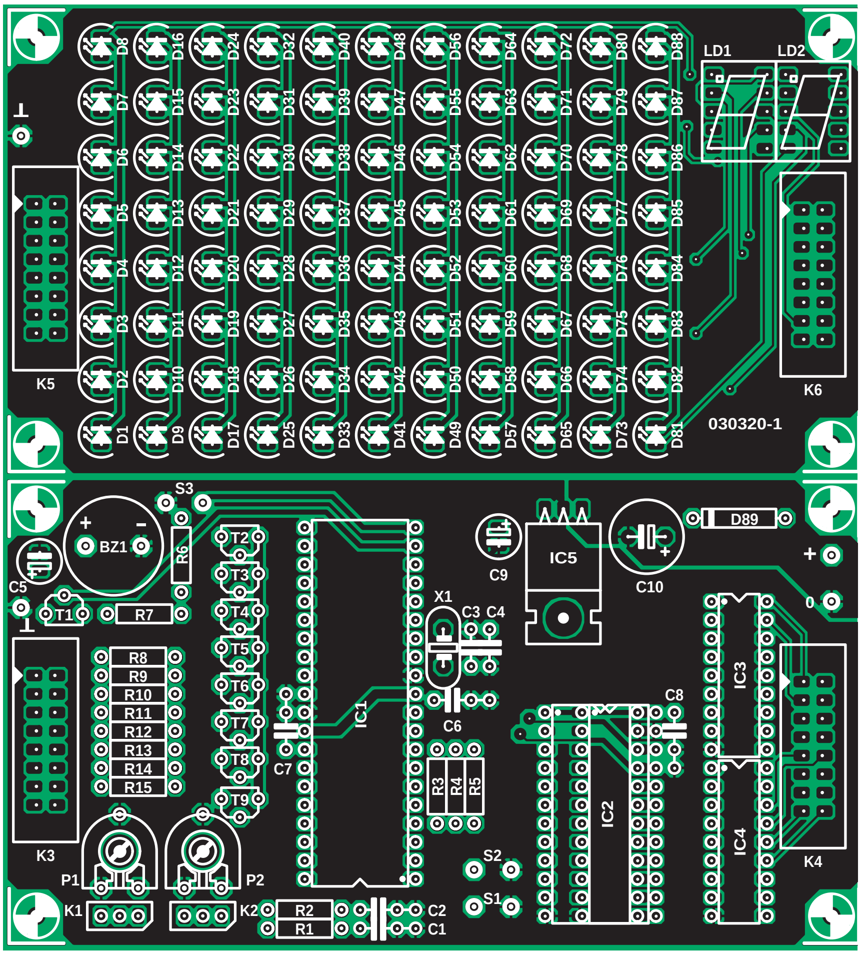


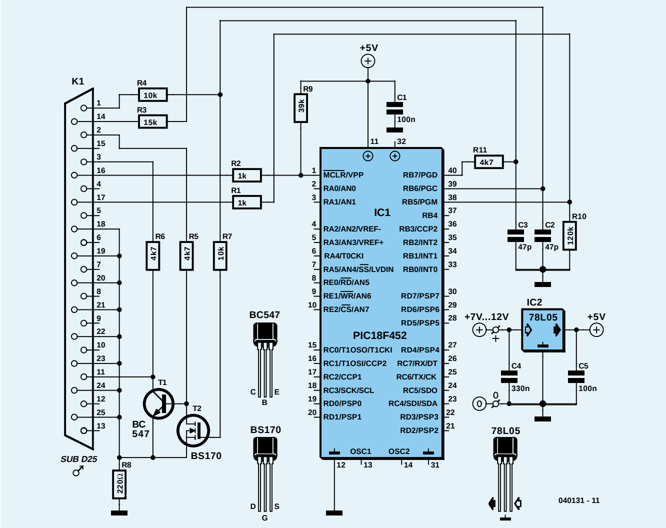
Figure 2. The PCB consists of two parts to be connected with flatcable.

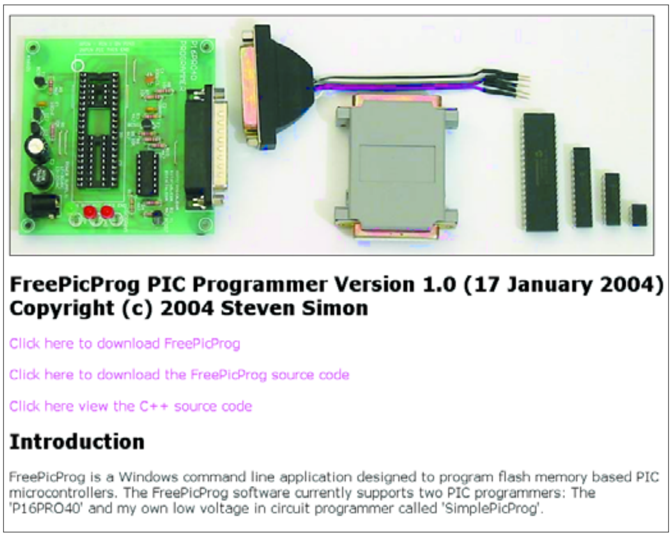
Software

That leaves us with the software burned into the PIC micro, here, a PIC18F452. Since we are looking at a game that should not cost too much, we looked for an ultra-simple programmer that allows anyone to program the chip using his/her PC. The article 'Free PICProg' elsewhere in this issue comes up with the goods. AS usual, the PIC software for his project may be obtained free of charge from our website as file number **030320-11** (see Free Downloads, month of publication) The PIC assembly code file is well commented.

(030320-1

py review copy review copy review copy review c





FreePicProg PIC Programmer Version 1.0 (17 January 2004)
Copyright (c) 2004 Steven Simon

Click here to download FreePicProg

Click here to download the FreePicProg source code

Click here view the C++ source code

Introduction

FreePicProg is a Windows command line application designed to program flash memory based PIC microcontrollers. The FreePicProg software currently supports two PIC programmers: The 'P16PRO40' and my own low voltage in circuit programmer called 'SimplePicProg'.

FreePICProg

cheap 'n easy PIC programming

David Daamen

The PocketPong game found elsewhere in this issue was designed around a PIC18F452 for which, we have to admit, no suitable programmer was ever published in this magazine. Sure, investing in a commercially produced programmer is an option, but making one yourself is cheaper and much more fun, too!

We reckon that's what Steven Simon must have thought when he needed a programmer for one of the newer PIC micros. On his website, Steven describes the software he produced for the purpose — both the executable and the source code files may be downloaded free of charge. The program can work with various hardware variants which are also described on the site. The circuit shown in **Figure 1** is a related item, albeit a very rudimentary one. No LEDs or other frills — just connect & program!

Construction and operation

The circuit consists mainly of passive parts, the majority of them serving to shape and tidy up various switching signals. Of the active parts, the transistors ensure that the edge steepness of the signal found at the BUSY line (pin 11) is improved. That leaves us with voltage regulator IC1, which we take it is so familiar as to defy further description.

The circuit should be easy to build on a small piece of Veroboard. For K1 it is wise to use an angled connector and solder it directly on to the board. The programmer may be plugged directly into the printer port on the PC.

The power supply is uncritical and may be realised using a simple mains adaptor with an output voltage of 7-12 VDC.

Software

The little program called 'fpp.exe' allows a HEX file to be burned into the PIC micro. The software is compatible with all versions of Windows. However, because the printer port is accessed, Windows NT, 2000 and XP do require an extra driver to be installed — in this case, 'DLPortIO' from Scientific Software Tools. Incidentally, only LPT1 (0x378h) is supported.

The hardware may be tested by typing this command line:

```
fpp lvp –t
```

where the 'lvp' option specifies the programmer type. You may also choose 'lvp_fast' which speeds up the programming sequence. It may not work in all cases, though, so first give it a try.

Once the hardware has been recognised, the PIC may be installed with the programmer supply switched off. The actual programming operation is launched with

```
fpp lvp <PIC type> -p
<HEX file>
```

There are also options for reading a PIC device ID, blank checking, erasing, etc. Type 'fpp' only for an overview of all available parameters.

PICs supported

FPP V1.00 supports the following PIC types:

- 12F629
- 16F876, 16F877, 16F628 (-A suffix versions are not supported!)
- 18F252, 18F452

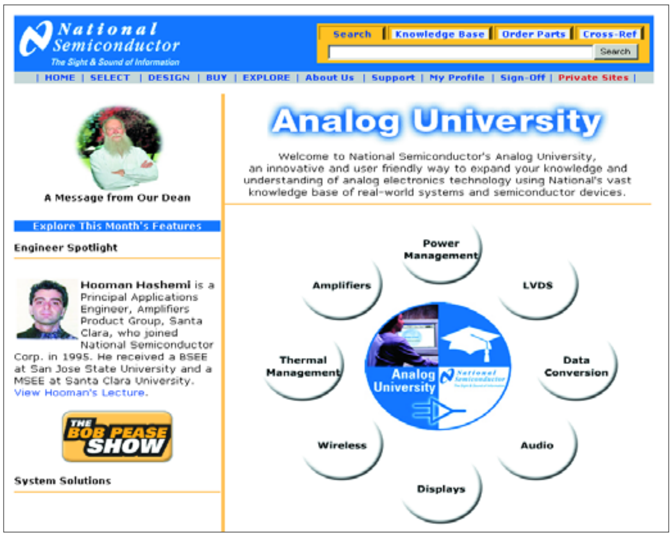
Although the 12F675 is in principle also supported by the software, the hardware shown here is unsuitable for this particular PIC device. For more information, have a look at the web pages covering FPP.

(040131-1)

FPP website:
www.geocities.com/SiliconValley/Hills/1924/freepicprog.html

FPP download from Elektor Electronics:
file # 040131-11,
www.elektor-electronics.co.uk/dl/dl.htm,
select month of publication.

DLPortIO
www.driverlinx.com/DownLoad/DlPortIO.htm


National Semiconductor
The Sight & Sound of Information
Search | Knowledge Base | Order Parts | Cross-Ref
Search
| HOME | SELECT | DESIGN | BUY | EXPLORE | About Us | Support | My Profile | Sign-Off | Private Sites |
Power Management
Amplifiers
LVDS
Thermal Management
Analog University
Data Conversion
Wireless
Audio
Displays
THE BOB PEASE SHOW

Analog University

Welcome to National Semiconductor's Analog University, an innovative and user friendly way to expand your knowledge and understanding of analog electronics technology using National's vast knowledge base of real-world systems and semiconductor devices.

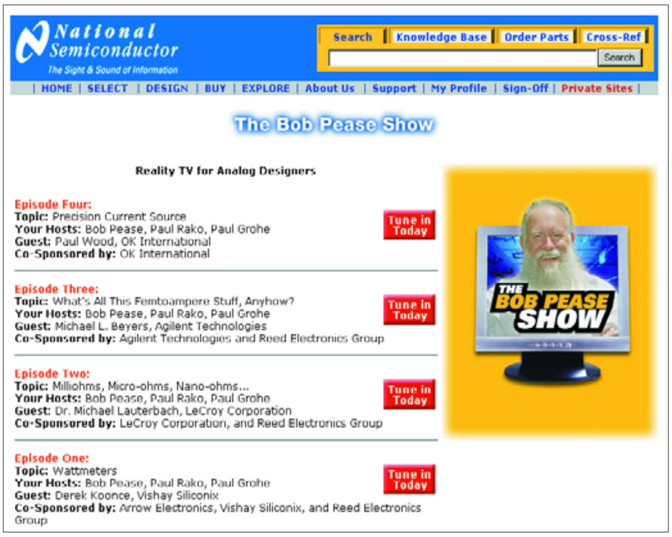
A Message from Our Dean

Explore This Month's Features

Engineer Spotlight

Hooman Hashemi is a Principal Applications Engineer, Amplifiers Product Group, Santa Clara, who joined National Semiconductor Corp. in 1995. He received a BSEE at San Jose State University and a MSEE at Santa Clara University. View Hooman's Lecture.

System Solutions


National Semiconductor
The Sight & Sound of Information
Search | Knowledge Base | Order Parts | Cross-Ref
Search
| HOME | SELECT | DESIGN | BUY | EXPLORE | About Us | Support | My Profile | Sign-Off | Private Sites |
Tune in Today
Tune in Today
Tune in Today
Tune in Today
THE BOB PEASE SHOW

The Bob Pease Show

Reality TV for Analog Designers

Episode Four:
Topic: Precision Current Source
Your Hosts: Bob Pease, Paul Rako, Paul Grohe
Guest: Paul Wood, OK International
Co-Sponsored by: OK International

Episode Three:
Topic: What's All This Femtoampere Stuff, Anyhow?
Your Hosts: Bob Pease, Paul Rako, Paul Grohe
Guest: Michael L. Beyers, Agilent Technologies
Co-Sponsored by: Agilent Technologies and Reed Electronics Group

Episode Two:
Topic: Milliohms, Micro-ohms, Nano-ohms...
Your Hosts: Bob Pease, Paul Rako, Paul Grohe
Guest: Dr. Michael Lauterbach, LeCroy Corporation
Co-Sponsored by: LeCroy Corporation, and Reed Electronics Group

Episode One:
Topic: Wattmeters
Your Hosts: Bob Pease, Paul Rako, Paul Grohe
Guest: Derek Koonce, Vishay Siliconix
Co-Sponsored by: Arrow Electronics, Vishay Siliconix, and Reed Electronics Group

Analog University

attend National Semiconductor lectures

David Daamen

Sure, the Internet contains thousands of electronic circuits covering equally vast numbers of applications. That's all very well, but King Shallow & Sgt Superficial often rule and it may take a lot of searching if you really want to know all the ins and outs of a certain design aspect. Of course, enough theory may be found in datasheets and application notes published but various renowned manufacturers, but somehow the method does not appear very efficient.

In this month's instalment of *Review Copy* we'd like to draw your attention to National Semiconductor's 'Analog University', a virtual academy that's existed for about a year now and has grown into a sizeable collection of mini courses covering a wide range of subjects. The curriculum [1] is divided into the following categories:

- *Power Management,* about linear and switch-mode power supplies and converters;
- *LVDS,* about low voltage differential signalling and bus configurations, as well as transmission lines for these technologies;

Data Conversion, about analogue-to-digital converters;

- *Audio,* about thermal properties and cost cutting;
- *Displays,* about the principles behind the design of TFT and CRT screens;
- *Wireless,* basics of RF technology and information on various PLL techniques;
- Thermal Management, about thermal control systems, analogue and digital temperature sensors;
- *Amplifiers,* about (high-speed) opamps, low-power and low-voltage amplifiers;

The courseware comprises, among others, articles, application notes and pointers to relevant contributions in the Knowledge Base. Moreover quite a few subjects are supported by On-line Seminars. Using a Java presentation — complete with sheets and sound — your lecturer will explain the subject matter in great detail. Eventually, the knowledge acquired in this way may be verified with the aid of a short research assignment or a quiz.

The lectures are also accessible via a separate overview [2].

National's website offers a lot more, including a massive Knowledge Base [3] with a search engine. There's also a section called Design Tools [4] comprising WEBENCH, an on-line design, simulation and construction assistant!

Finally, we should not forget to mention "The Bob Pease Show" aptly called "Reality TV for Analog Designers" by National Semiconductor themselves.

(045043-1)

Web references

[1] analoguni.national.com
[2] www.national.com/onlineseminar
[3] knowledgebase.national.com
[4] www.national.com/design
[5] www.national.com/rap

All links are also accessible from the National Semiconductor homepage: www.national.com

Precision Function Generator

MAX038 goes SMD

Klaus-Jürgen Thiesler

Although surface-mount parts are not the easiest to handle, the reward in test and measurement equipment is increased accuracy.

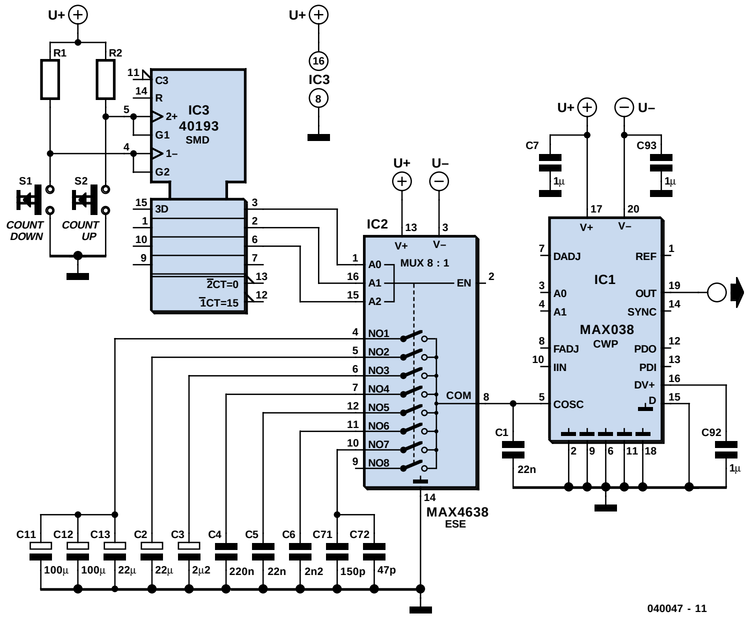


Figure 1. Suggested circuit diagram for a precision function generator based on the MAX038 (now in an SMD case.

Integrated precision function generator type MAX038 is now an old faithful with a history of ten years. In the June 1995 issue we described a full-blown function generator built from standard size parts and using what is now considered a large PCB. The project was hugely successful and PCB sales reached several thousands. In 1995, the use of SMDs in Elektor circuits was more or less prohibited because the advantages of a smaller board did not outweigh the disadvantages of problematic supply and mounting methods.

Today there's no way you can avoid SMDs anymore, however we generally do not use them unless they serve a good purpose, or if the component we wish to use is simply not or no longer available in 'leaded' form. Other compelling reasons to use SMS are the ongoing miniaturisation of circuits and near elimination of stray capacitance and inductance (particularly in compact RF gear).

In some case, however, SMDs simply serve advancing technology — if you build the function generator from SMD instead of leaded parts and ensure low-inductance (i.e., *short*) PCB tracks then the instruments' technical specifications as well as its frequency range can be improved dramatically.

A digital frequency range switch in combination with multilayer ceramic capacitors (MLCC) in SMD technology not only improve the general performance of the MAX038 chip but also extend its frequency downwards to below 0.1 Hz and upwards to over 20 MHz. Such improvements we feel are worth just that extra effort in building up the circuit.

Two easily realised design aspects will help to unlock the full potential of the MAX038:

- an optimised PCB layout with the shortest possible copper tracks to keep stray inductance and capacitance to a minimum;
- SMD parts are marked by lower temperature coefficients and closer tolerances.

The circuit and PCB layout presented here do not represent a complete construction project. Rather, the publication aims at providing a few useful tips when you really want to juice the MAX038 for maximum performance. Hence Mr. Thiesler's contribution fills this month's Start Here pages.

Range switch

Instead of the mechanical range switch we propose an electronic CMOS analogue switch type MAX4638ESE, an 8-1 multiplexer that, depending on its control, connects the various frequency-determining capacitors to the internal oscillator. In the highest frequency range (20 MHz), it is essential for capacitor C8 to be connected directly to the MAX038 — and not by way of the multiplexer. Using the SMD stuffing in SO16 and 0815 enclosures this can be realised without problems. The ground side of C8 has to be connected directly to pin 6 (GND) — the other ground pins are unsuitable for that purpose.

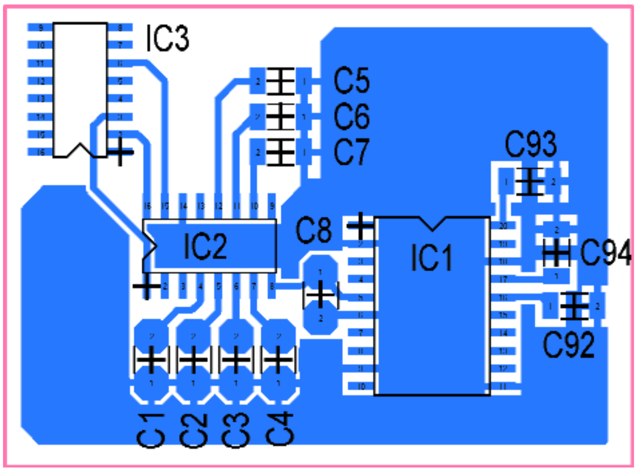
The switching currents through the frequency determining capacitors are really low and the oscillator voltage (COSC) is just –1 V (at a supply of ±5 V). This makes the selection of the CMOS switch rather easy. Much more critical however is the PCB layout; 1 cm of 1-mm wide copper track representing an inductance of 7 nH. Long wires cause large stray capacitances and that is where SMDs are clearly superior to their leaded counterparts. As a bonus, SMDs are manufactured to closer tolerance specifications.

MLC capacitors

Solid, leaded, capacitors in the microfarads range are every expensive, inaccurate and bulky components and it is not surprising to see they are rarely used these days. SMD multilayer ceramic chip capacitors (MLCCs) are much more accurate, tiny and cheap mainly because they are manufactured by the millions for use in mobile phones. Mind you, MLCCs are not a new development — in fact they have been around for at least ten years. The codes stated in the parts list indicate the dielectric (X5R...) and the case shape (1210).

For the low end of the frequency range a 100-µF MLCC with a working voltage of 6.3 V is used with excellent results. Its ESR (equivalent series resistance) is stunningly low at less than 10 mΩ. A 100-µF, 68-µF and 22-µF MLCC stacked to make a neat pile was measured and found to have an equivalent capacitance of 216 µF.

Large value ceramic capacitors in X5R technology have a typical tolerance of ±20%, as opposed to +80/–20% for Y5V types. RF ceramic capacitors like the 22-pF MLCC with C0G structure and a maximum deviation of 0.5 pF are manufactured by, among others,



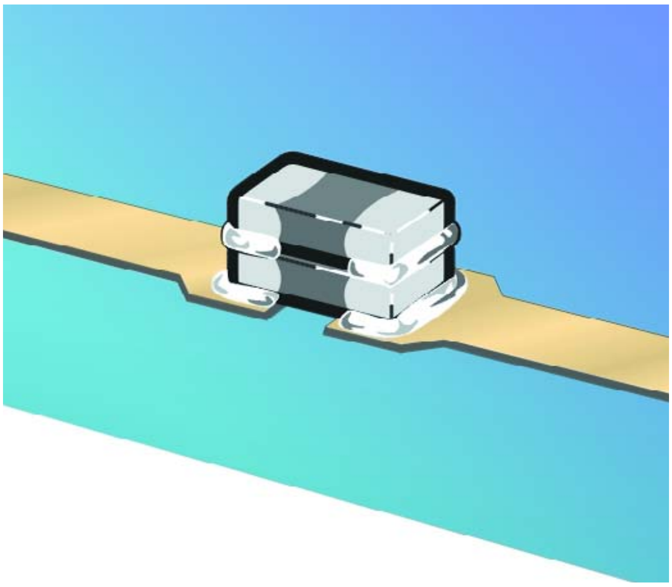


Figure 2. Proposed PCB layout featuring ultra-short tracks and an 8-way all-electronic frequency range switch.

Figure 3. Capacitor stacking — no problem with MLCCs.

Taiyo Yuden. Although the temperature coefficient of just ±30 ppm (Class 1) is a bit over the top for our application, it does demonstrate the technological advances made by passive component designers. The author found that Reichelt (www.reichelt.de) supplies an assortment of 0805 style C0G ceramic chips at a reasonable price.

Adjustment requires a capacitance meter to be able to select the frequency determining capacitors for desired value, or combine a nmef of them to arrive at the desired values. These chip capacitors may be stacked onto each other and occupying just one pair of PCB solder pads. As you can see from Figure 3, the 'tower block' method helps to keep copper tracks as short as possible. For a change, you'll find that the trusty solder iron has the edge over industrial mounting methods like flow soldering or hot air! Although SMD components in 0805 and 1210 cases are really small, they can still be soldered manually if you have a steady hand.

(040047-1)

Components list:

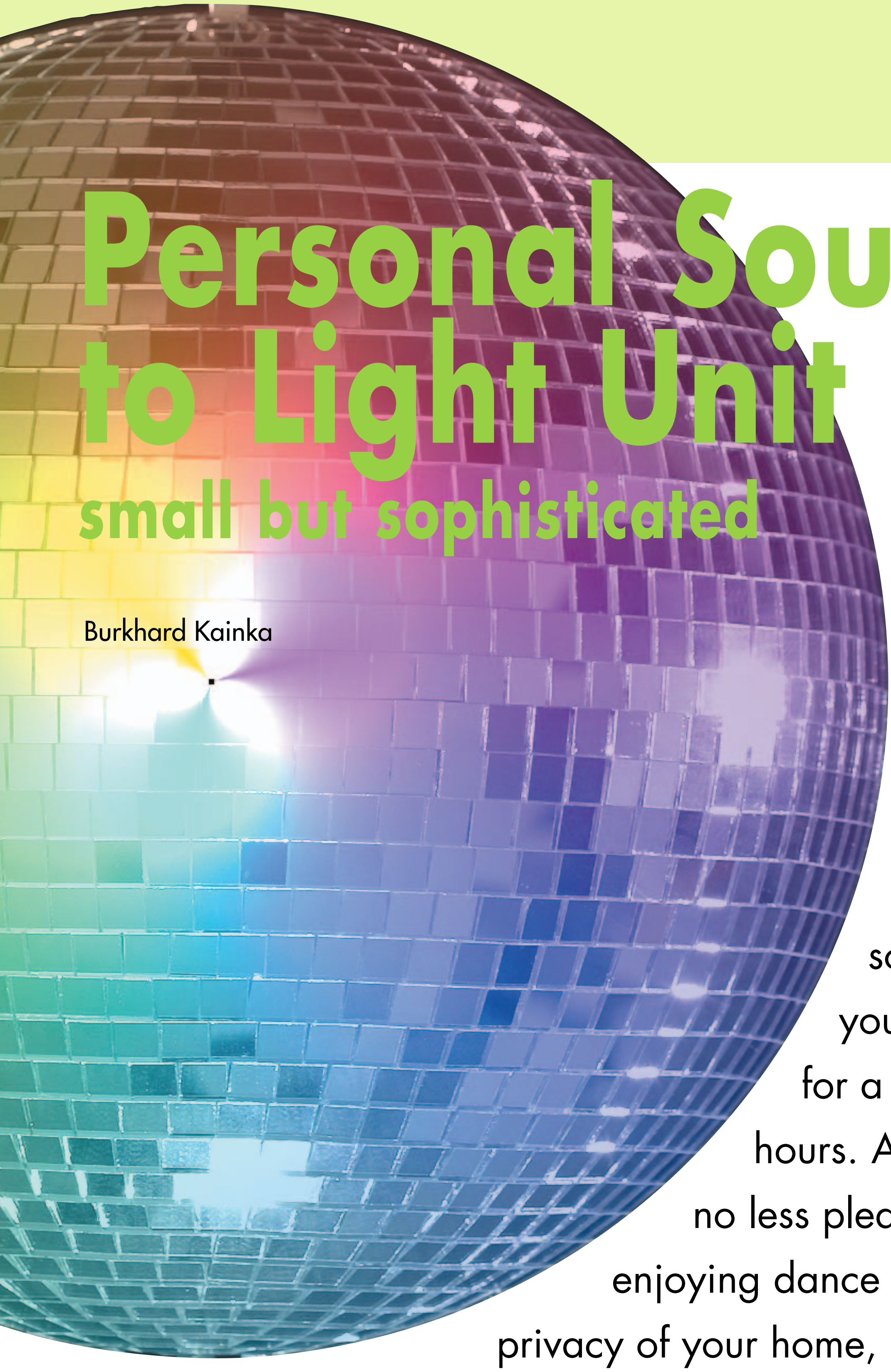
Capacitors:

C1a,C1b = 100µF ceramic MLCC	Y5V	1210
C1c,C2 = 22µF ceramic MLCC	X5R	1210
C3 = 2µF2 ceramic MLCC	X7R	1210
C4 = 220nF ceramic MLCC	X5R	0805
C5 = 22nF ceramic MLCC	X7R	0805
C6 = 2nF2 ceramic MLCC	COG	0805
C7a = 150pF ceramic MLCC	COG	0805
C7b = 47Fp ceramic MLCC	COG	0805
C8 = 22pF ceramic MLCC	COG	0805

Semiconductors:

IC1 = MAX038CWP (SO-20)
IC2 = MAX4638ESE (SO-16)
IC3 = 40193 (SMD)

n	Frequency ranges		
	@ 2.2 x 10^n pF	**@ 4.7 x 10^n pF**	**@ 1.0 x 10^n pF**
8	0.1 - 1 Hz	0.04 - 0.4 Hz	0.2 - 2 Hz
7	1 - 10 Hz	0.4 - 4 Hz	2 - 20 Hz
6	10 - 100 Hz	4 - 40 Hz	20 - 200 Hz
5	100 Hz - 1 kHz	40 - 400 Hz	200 Hz - 2 kHz
4	1 - 10 kHz	400 Hz - 4 kHz	2 - 20 kHz
3	10 - 100 kHz	4 - 40 kHz	20 - 200 kHz
2	100 kHz - 1 MHz	40 - 400 kHz	200 kHz - 2 MHz
1	1 MHz - 10 MHz	400 kHz - 4 MHz	2 - 20 MHz



Personal Sound to Light Unit

small but sophisticated

Burkhard Kainka

Disco nights are great fun from time to time but you don't want to overdo it in regard of the sound volumes you're exposed to for a couple of hours. Arguably there's no less pleasure in enjoying dance music in the privacy of your home, study or student digs. However, the true disco feeling is not obtained without a matching sound to light unit, so here's a really small version.

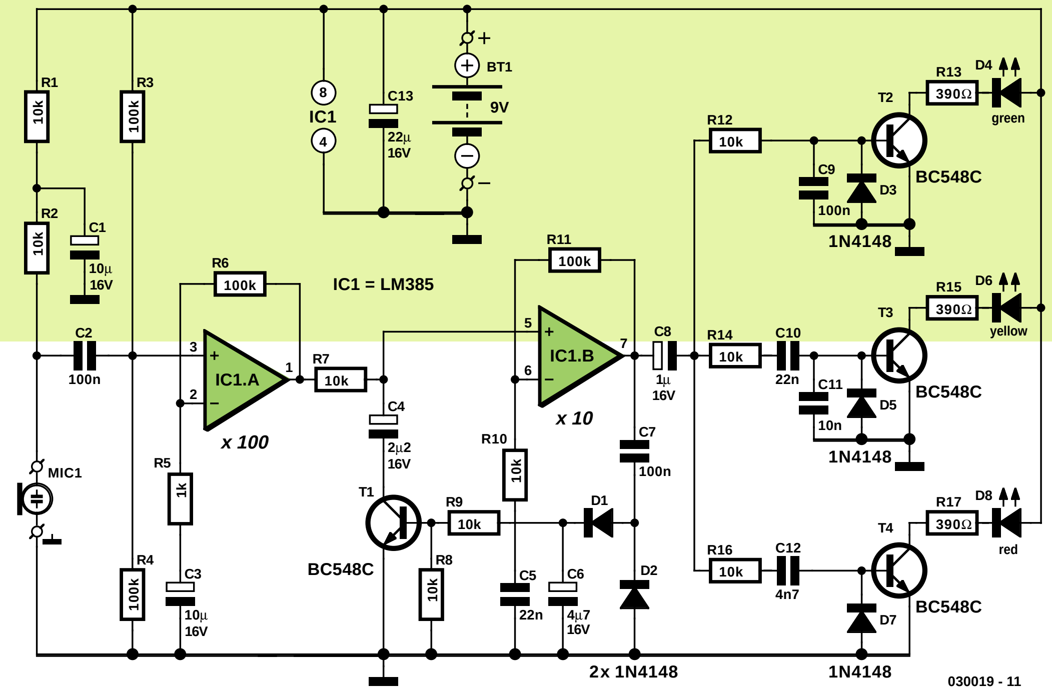


Figure 1. The circuit of the sound to light unit consists of a controlled-gain amplifier followed by a 3-way frequency splitter.

A sound to light unit converts music signals into light pulses. In most cases, three channels are used to cover different frequency ranges. The bass channel will then indicate the 'beat' of the music by a more or less rhythmical flash, while the two other channels represent the higher frequency ranges of the music channel.

The 'private' version of a sound to light unit discussed in this article employs three coloured LEDs instead of powerful flashing lights or floodlights as used in most discos. For the rest, it has practically the same functionality as the 'real thing' experienced on Saturday nights. However, a direct connection to the music amplifier is not necessary as the unit we've in mind has been designed to pick up the sound information through a microphone. Another peculiarity of the circuit is the automatic sensitivity adaptation to the music volume detected. In this way the circuit can work without any controls.

The circuit draws an average current of just 20 mA and works happily off a 9-volt PP3 block battery. However, the low current requires super bright LEDs to be used to ensure sufficient brightness.

Amplify it

High amplification is required to enable the tiny signals produced by an electret microphone to be turned into bright flashes from LEDs. The circuit diagram in **Figure 1** shows that we employ two operational amplifiers with a total gain of about 1000 times. The input of the first opamp (IC1.A) is connected to the electret microphone capsule via coupling capacitor C2. The microphone is given a certain DC bias level obtained from the 9-V supply rail by resistors R1 and R2. R1 together with electrolytic capacitor C1 decouple the supply voltage for the sensitive microphone while R2 equals the microphone impedance. Capacitor C2, then, ensures that the microphone bias level does not appear at the opamp input. In other words, it will only pass the alternating component, which is caused by sound picked up by the microphone. The + input of the opamp has its own bias voltage supplied by potential divider R3-R4. Because the two resistors have the same value, the supply voltage is effectively halved, i.e., 4.5 V exists at the junction of R3 and R4 (assuming a nominal 9 V supply). This bias voltage will also exist at the output of the first opamp and, because of R7, at the input of the second one (IC1.B) whose output will also copy this dc level. In this way, R3 and R4 keep both opamps biased at half the supply voltage. Both opamps are used in the non-inverting configuration hence provide unity (_1) dc gain.

For alternating signals, however, the gain is much greater. In the case of the first opamp, the gain is determined by the ratio between resistors R6 and R5, or R11/R10 for the second opamp. Just look at the relevant resistor values and you'll discover that IC1.A is configured for a gain of 100 and IC1.B for a gain of 10.

The signal level at the output of IC1.B is rectified by D1 and D2, smoothed by C6 and then used to drive n-p-n transistor T1. To the signal voltage at the output of IC1.A, the combination of R7 and T1 looks like a voltage divider. With rising signal levels, the rectified voltage on C6 also rises and the transistor is driven harder because of the larger base current supplied by R9. The result is a lower resistance in the transistor and consequently a lower AF signal behind R7.

You may wonder why we did not use an n-p-n transistor without direct current in the collector circuit. Alternatively you might have expected to see a FET at this position, its drain-source junction acting as a controlled resistance. It is less known that virtually the same function may be obtained from a regular switching/AF transistor like the ubiquitous BC548C. An n-p-n transistor, too, represents a variable resistance that can be controlled wit-

Components list

Resistors:
R1,R2,R7-R10,R12,R14,R16 = 10kΩ
R3,R4,R6,R11= 100kΩ
R5 = 1kΩ
R13,R15,R17 = 390Ω

Capacitors:
C1,C3 = 10μF 16V radial
C2,C7,C9 = 100nF
C4 = 2μF2 16V radial
C5,C10 = 22nF
C6 = 4μF7 16V radial
C8 = 1μF 16V radial
C11 = 10nF
C12 =4nF7
C13 = 22μF 16V radial

Semiconductors:
D1,D2,D3,D5,D7 = 1N4148
D4 = LED, green (see text)
D6 = LED, yellow (see text)
D8 = LED, red (see text)
IC1 = LM385N (with socket)
T1-T4 = BC548C

Miscelllaneous:
BT1 = 9V PP3 (6F22) battery with clip-on lead
MIC1 = electret microphone
PCB, available from The PCBShop

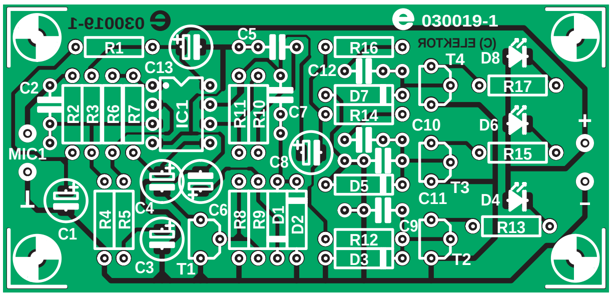
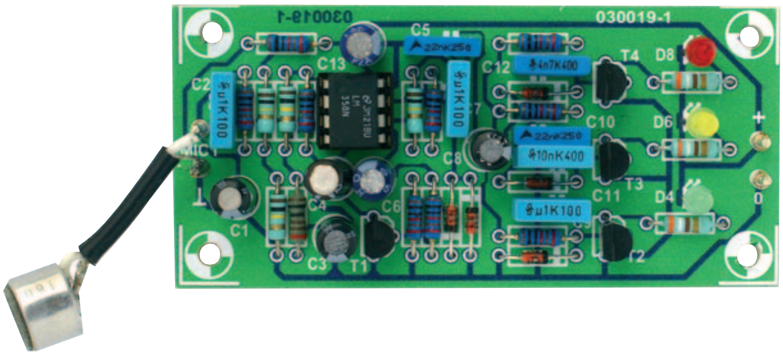


Figure 2. The single-sided PCB has no wire links.



hin a certain range. However, for a low-distortion volume control, only a tiny signal level (of the order of millivolts) may be applied to the collector. This condition is not satisfied here as the output voltage is regulated to about 1 V_{pp}. If the second stage has a gain of 10, about 100 mV_{pp} can be found at the collector. At such a level, distortion occurs that will not be acceptable in other applications. No problem for the sound to light unit, however, because the output signal is used to control LEDs than drive an audio amplifier. If you do want to use such a 'volume control' for audio applications, you should make sure a much smaller signal level is handled, which is probably easiest realised by moving the automatic volume control towards the circuit input.

Filter

The output signal supplied by the second opamp drives the LED controls by way of simple filters. Each LED driver stage consists of a transistor (T2, T3 and T4). To prevent the transistor bases from being charged with negative levels, anti-parallel diodes (D3, D5, D7) are used on each base terminal. Each transistor is only actuated on the positive half cycle of the drive voltage. Overall, however, the higher frequencies do cause an impression of average brightness.

The filters consist of simple RC networks broadly dimensioned for a cut-off frequency using the formula

$$f_c = 1 / (2 \pi R C) \quad [Hz]$$

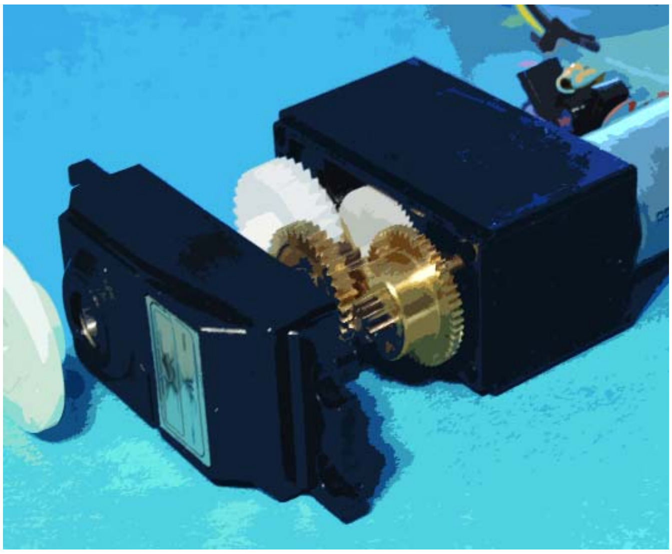
For example, the low-pass section R12/C9 is dimensioned for about 160 Hz using 10 kΩ (R12) and 100 nF (C9). All lower frequencies in the music signal will therefore pass through this 'channel'. The mid-tone channels contains a combined high-pass / low-pass filter R14/C10+C11. The treble channel is driven by a simple high-pass R16/C12. If necessary the cut-off frequencies may be changed to suit individual requirements and that's easiest done by making small changes to the capacitor values.

Printed circuit board

If the circuit of a relatively simple design, the actual construction of the sound to light unit is made even simpler by a printed circuit board (**Figure 2**). The two opamps we've discussed are contained in a single IC type LM385, which is best fitted in an IC socket (look at the notch in the IC body). The single-sided PCB has no wire links. No problems are expected to arise if you watch the polarity of the diodes, electrolytic capacitors and LEDs and work carefully all the way. The electret (or 'condenser') microphone insert can be almost any available type as long as it has two terminals.

The maximum LED current amounts to about 18 mA so the circuit is suitable for LEDs described as 'standard', 'bright' and 'superbright'. Using the latter the effect of the 3-way filtering is more pronounced, while for a relatively small room with not too much light ordinary LEDs will be found to be perfectly suitable. If you decide to use low-current LEDs, the value of series resistors R13, R15 and R17 should be increased to 3.3 k or even 3.9 k. The resulting reduction in current consumption will enable your 9-V battery to last much longer.

(030019-1)



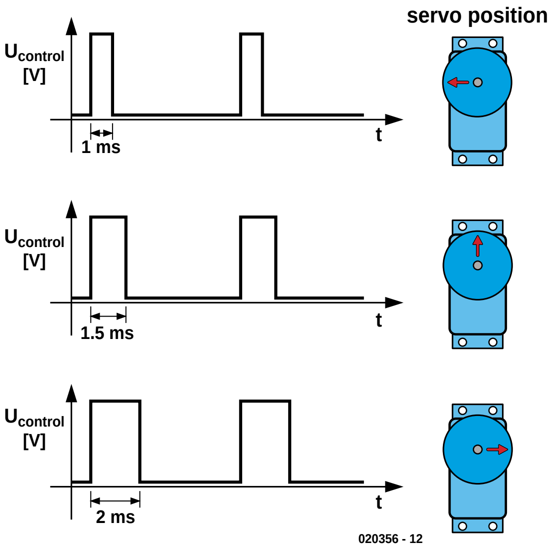


Figure 1. Spindle (actuator) position as a function of pulsewidth.

Digital Motors

intelligent drive for servos

Giel Dols

Analogue servos are really ingenious bits of hardware. But there's an even better alternative: the digital servo! Based on the operation of the 'normal' servo we tell you what's so much better about the latest digital ones.

A servo is a combination of a dc, ac or brushless motor with a position detection circuit. In modelling and robotics, we usually find three-wire dc servos. Such a servo then consists of dc motor, a reduction gearbox and two mechanical 'stops' preventing the spindle from turning beyond extreme positions. Usually, the servo also contains a potentiometer that enables the position of the spindle to be communicated to a small piece of electronics. The circuit, then, is the link between the motor, the measured position and the real world.

The three wires carry the supply voltage, ground and the drive signal to the servo. The drive signal — in RC modelling usually supplied by a receiver — determines the target position of the servo spindle. If external forces (like air pressure on a model plane rudder) try to change the spindle position, the feedback in the control circuitry will counteract the movement and ensure the rudder remains in the desired position. As long as its drive signal remains unchanged, a servo will maintain the current spindle position, correcting it if necessary. Only if there's a change in the drive signal will the spindle move to a different position.

Operation

The drive signal consists of rectangular pulses with a swing of 5 V. The pulse period time is constant and the width of individual pulses determi-

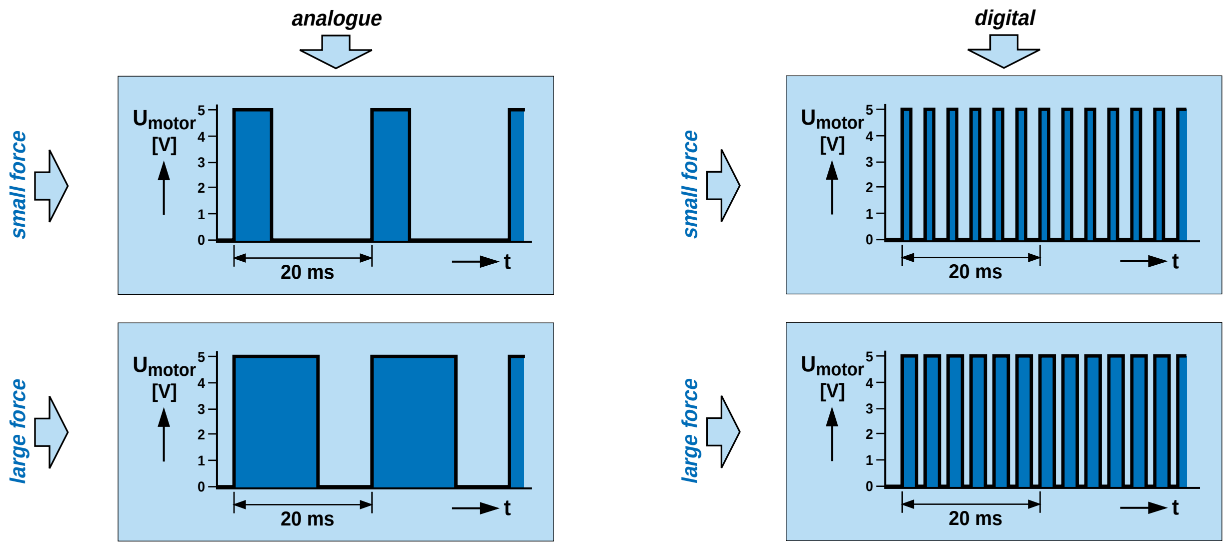


Figure 2. Motor voltage when a certain spindle position is maintained (left: analogue — right: digital).

nes the spindle position. The zero or home position of a servo corresponds t a pulse width of about 1.5 ms. Depending on the desired amount of spindle rotation, the control pulses are shortened or lengthened. However, the period of 20 ms is maintained al the time.

The servo actuator spindle is coupled to a potentiometer. Consequently, the potentiometer's wiper position not only indicates the spindle position but also equates to a certain pulsewidth. The internal electronics looks after this fixed relationship, maintaining the spindle position until a change in the drive signal pulsewidth occurs.

The digital way

In recent years, servos have seen great improvements — size, responses time and actuator force have been subject to constant improvements. The latest development called 'digital servo' is an important step forward.

In principle, the digital servo is equal to its analogue ancestor. The only difference exists in the activities of a microcontroller constantly analysing the drive signal and if necessary driving the motor. The mechanical construction is the same as that of the analogue counterpart.

The use of a microcontroller offers significant advantages. As already mentioned, an analogue servo is capable of correcting its spindle position to counteract external forces. Not much will happen if the external 'disruption' is small, as small changes equate to narrow pulses. In many cases, the effective power fed to the motor under these conditions is too low to get the motor to turn so a certain amount of hysteresis is created: below a certain limit, the servo will respond in a non-linear fashion to disturbances in the spindle position. With digital servos, the microcontroller has been programmed to 'know' the motor's specific response and account for it in the way it is driven. For example, the width of the pulses sent to the motor in response to a relatively small spindle movement will be larger than with an analogue servo. Because the software allows the processor to calculate the optimum amount and length of the drive pulses for a certain disturbance or desired position, a digital servo will show much faster accurate responses.

It's not just the 'intelligence' that makes the servo behave the way we want. A further difference with analogue servos is the use of much higher switching frequencies of the motor drive pulses. This not only benefits the accuracy, but also allows more power to be delivered as the motor can be 'on' more often. Extra power is not only practical in maintaining the existing spindle position, but also when moving to a new one because the sensor can accelerate much faster and reach the target position well before its analogue counterpart.

Disadvantage?

Increased energy consumption is a direct consequence of higher motor pulse speeds. This disadvantage should be duly considered when applying the new digital servos in any battery-powered equipment like model craft. Fortunately, the problem is far from insurmountable thanks to recent advances in battery technology.

(040041-1)

Rail

Ray King

‘Smooth Operator’ elsewhere in this issue employs model control servos to drive the turnouts on a model railway. One of its advantages is that it can be activated over a single wire making it ideal for computer control described... here! Rail Router is a hardware/software combination capable of controlling up to 127 turnouts.

Router

model train routing with a PC

The Rail Router hardware comes in two flavours: a **master** router board capable of directly controlling up to 15 devices and a **slave** board connected up via a ribbon cable and adding a further 16 turnout controls. The general layout of the system is illustrated in **Figure 1**. The master and slave circuits use the same printed circuit board stuffed to reflect the desired function. Slave routers are optional — if you are satisfied with 'just' 15 turnouts and/or semaphores then you're fine with just the master router.

A dual-purpose circuit...

The circuit diagram shown in **Figure 2** is unusual in that it shows the master as well as the slave circuit. Dashed outlines and connections are used to indicate the difference between the two circuits, which can be built on one and the same board. Electrically the difference between the two circuits exists in the presence or absence of jumpers and circuit parts. The MAX232 two-way RS232/TTL level converter, for example, is only required for the master function, which (as you may have guessed already) requires a connection with a PC running the specially written Rail Router control software (more about this further on). The RS232 port on the PC is connected up to the Rail Router master board via sub-D socket K17. Only Tx/Rx traffic is used, without handshaking.

A PIC16F877 microcontroller is found at the heart of the master as well as the slave circuit. Although the micro is loaded with the one that the same software for the master or slave function, it actually selects between two different code chunks by looking at the logic level you've defined at port line RC5 by means of jumper JP1. The 16F877 ticks at 8 MHz as determined by quartz crystal X1 and its loading capacitors C1 and C2.

The master and slave router boards require a power supply of 8 V to 15 V DC which can be provided by a small mains adaptor or from a DC outlet on one of the railway speed governors.

Master operation

The PIC micro continuously monitors the incoming serial information, determining whether the device specified in the command is comprised in the first 15 turnouts. If so, it changes the state of the turnout control (via K2-K16). If not, it passes the information over to buffer IC3 and from there to connector K18 for slave units to test. Each output connector (K1-K16) on the router board comprises an unregulated supply (V+) ground and the control lead as required by the 'Smooth Operator' servo control circuits. Note that Output #1 (K1) is not used by the master configuration. It is planned to employ it at a later stage for enhanced facilities.

Slave operation

The operation of the slave board is similar to that of the master but simpler because of the absence of the RX/TX serial interface with the PC. The unique address of each slave board is determined by the settings of DIP switch S1. Setting the address to 001, for example, allows the slave board to operate turnouts 16 through 31, where code 001 is RE2 = 0; RE1 = 0 and RE0 = 1 on the PIC.

... and a dual-purpose circuit board

As already suggested by the circuit diagram, the circuit board designed for the Rail Router system can act as a master or a slave, depending on how it is populated. The two different component stuffing plans are given in **Figures 3a** (Master) and **3b** (Slave). Carefully study the parts list and the component overlays to avoid hard to find problems. If necessary, refer back to the circuit diagram. All components on the two boards are regular-size devices and construction is therefore not expected to cause any difficulty if you take your time and pay attention to polarised components in particular (ICs, transistors, electrolytic caps). We recommend using a good quality socket for the PIC device(s) as it (they) will the most expensive part(s) in the circuit.

The PIC software

We can be very brief about the software run by the PIC micro(s) used in this project. Do-it-yourself programmers among you will be pleased to know that the complete annotated source code files supplied by Ray King are available free of charge from our website under ref. **030403-11**. Simply download them, compile and then program your own 16F877 chip. Alternatively use the hex code directly.

Those of you without the means or wherewithal to burn your own PIC chip may resort to our Readers Services who supply ready-programmed PIC chips for this project under order code **030403-41**.

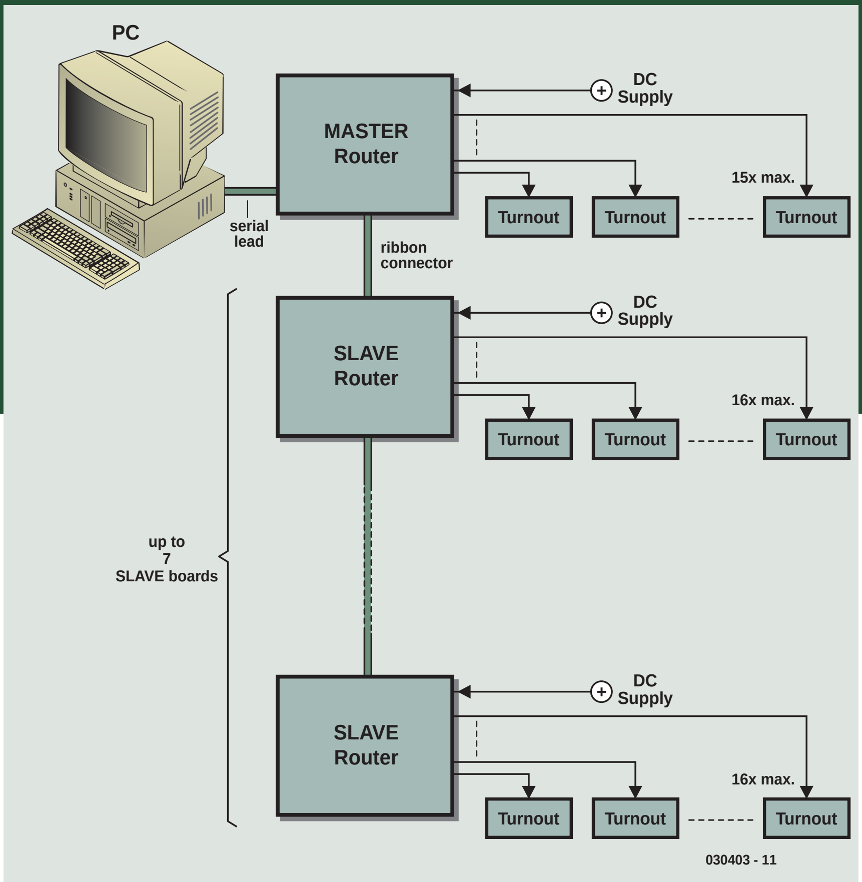


Figure 1. A complete system would be set up like this. The boxes marked 'turnout' represent a 'Smooth Operator' unit.

Rail Router for the PC

A screenshot of the Rail Router software running on a PC is shown in **Figure 4**. Any number of track layout designs may be stored on the PC and loaded when the Rail Router program is launched, or at any time during a session. The software written by Ray offers the facility to design and alter the railway layouts before saving them to disk. The baud rate to the master router is 9600 bits/s. The PC software was written in Visual Basic 6 (VB6). The source code (.VBP and components) as well as the executable version are included in the 'software package' for the project, ref. **030403-11**. Simply run the railrout.exe file and the program will be installed on your computer.

The latest updates and hardware additions may be found at Ray King's website.

Design a track layout...

Click on the 'Change Layout Design' button and the design screen will appear. Click on any square in the layout area, then click on the required symbol to be put in this square. Continue adding symbols until your layout is complete. Make sure all sidings as well as roads in and out of the layout end

COMPONENTS LISTS

Master Router

Resistors:
R1 = 4kΩ7
R2,R3,R4 = 10kΩ
R5 = 47kΩ

Capacitors:
C1,C2 = 22pF
C3-C7,C9 = 10µF 25V radial
C8,C11,C12 = 100nF
C10 = 1µF 16V radial

Semiconductors:
IC1 = PIC16F877-20/P, programmed, order code **030403-41**
IC2 = MAX232
IC3* = 74HCT241
IC4 = 7805

Miscellaneous:
JP1,JP2 = jumper
K2-K16 = 3-way SIL pinheader
K17 = 9-way sub-D socket (female) angled pins, PCB mount
K18* = 10-way boxheader
K19 = 2-way PCB terminal block, lead pitch 5mm
X1 = 8MHz quartz crystal
PCB, order code **030403-1** (see Readers Services page)
Disk, all project software (PIC & PC), order code **030403-11** or Free Download

* only required when a Slave Router is connected

Slave Router

Resistors:
R2,R3 = 10kΩ
R5-R8 = 47kΩ

Capacitors:
C1,C2 = 22pF
C9 = 10µF 25V radial
C12 = 100nF
C10 = 1µF 16V radial

Semiconductors:
T1 = BC550
IC1 = PIC16F877-20/P, programmed, order code **030403-41**
IC4 = 7805

Miscellaneous:
K1-K16 = 3-way SIL pinheader
K18 = 10-way boxheader
K19 = 2-way PCB terminal block, lead pitch 5mm
S1 = 3- or 4-way DIP switch
X1 = 8MHz quartz crystal
PCB, order code **030403-1** (see Readers Services page)

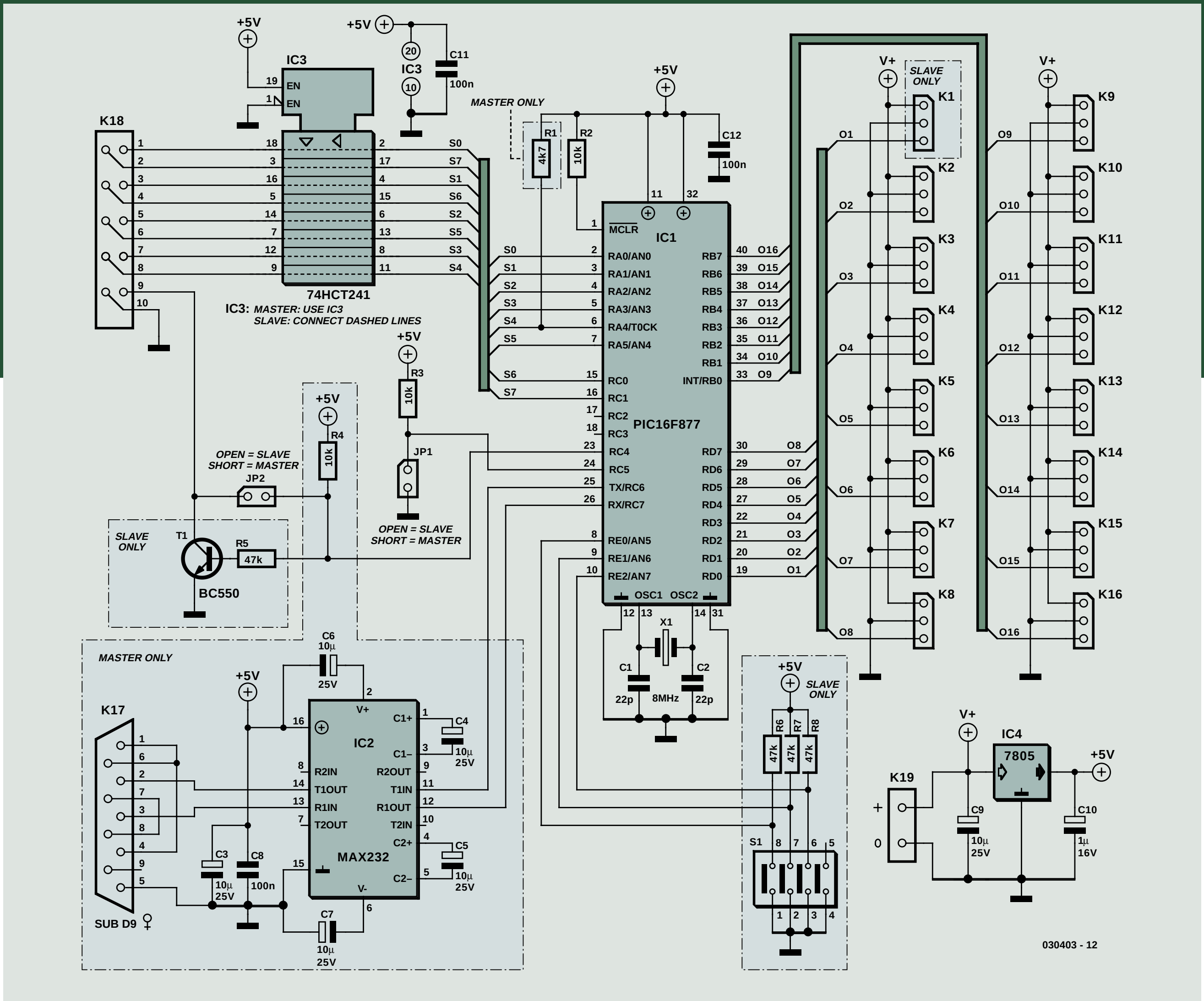


Figure 2. Combined circuit diagram for the Master and Slave Router configurations.

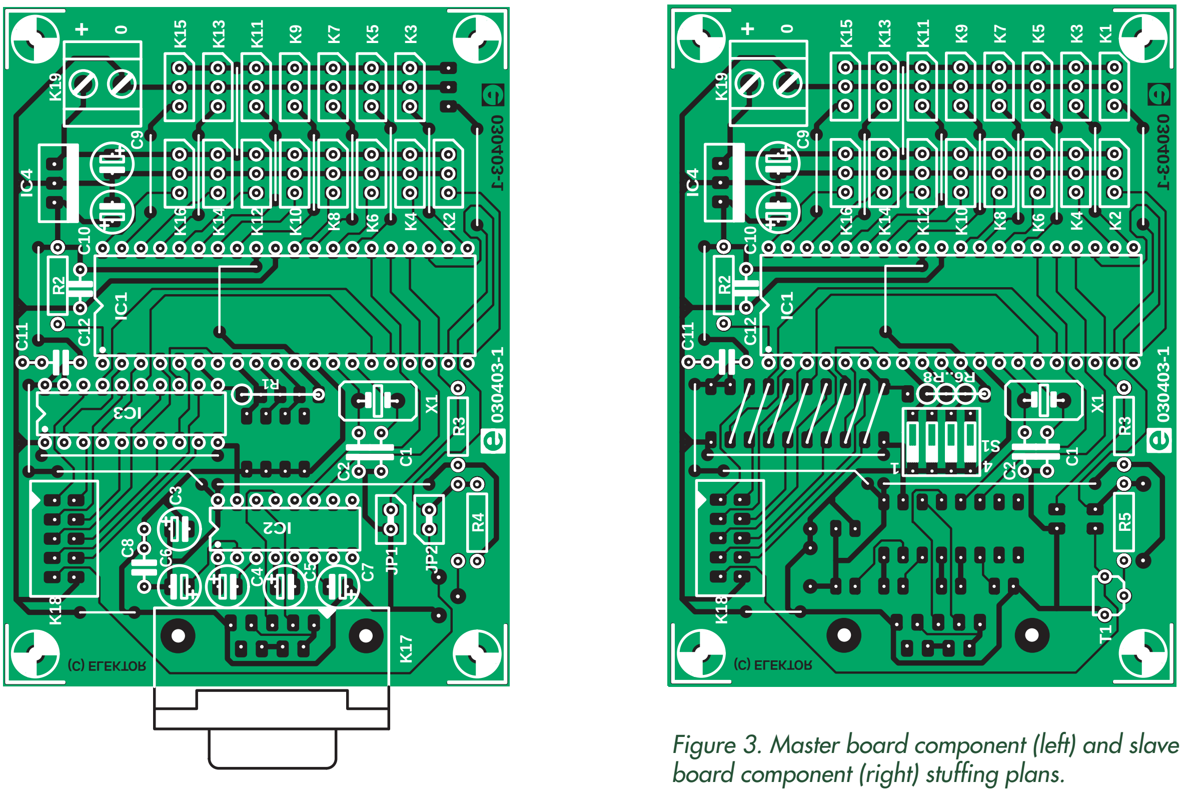
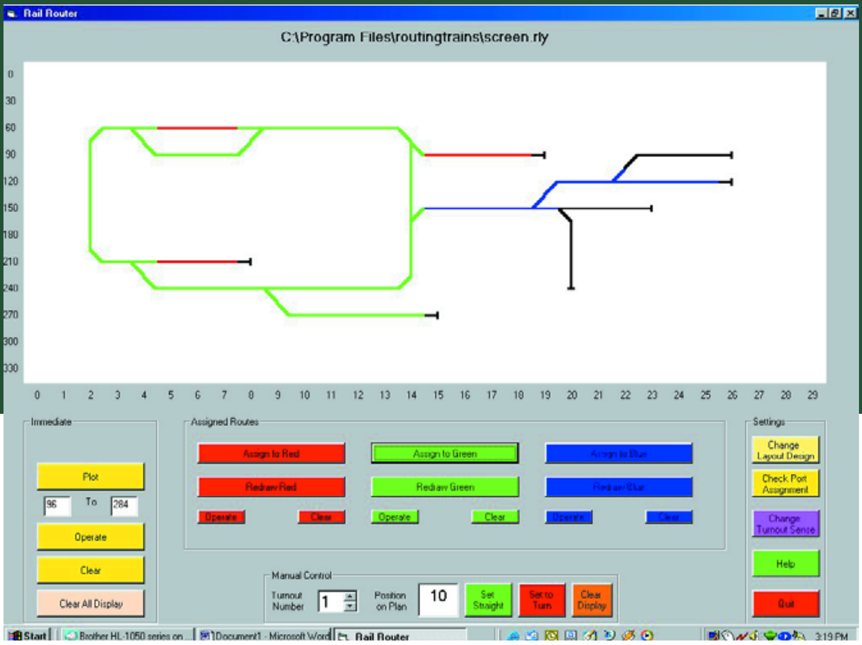


Figure 3. Master board component (left) and slave board component (right) stuffing plans.


Rail Router

C:\Program Files\routingtrains\screen.rly

Immediate: 96 To 284

Manual Control: Turnout Number 1 Position on Plan 10

Figure 4. The Rail Router PC software in action.

Free Downloads

PC and microcontroller software. File number: 030403-11.zip

PCB layout in PDF format. File number: 030403-1.zip

www.elektor-electronics.co.uk/dl/dl.htm, select month of publication.

with an 'end' symbol. When done, click on 'Save Design and Exit' and give the layout a name when prompted.

When the layout is saved, the program records details of all turnouts and allocates a hardware 'port' to each one. To check the assignments, click on 'Check Port Assignment' to produce a list of the turnout number from the diagram and its associated hardware port. This hardware port (i.e., K2-K16 on a master board, or K1-K16 on a slave board) should be wired to the 'Smooth Operator' control for that particular turnout.

On installation it is possible that the turnout mechanism works in the opposite sense to the software, that is, if 'ahead' is selected the turnout moves to the turn. This can be resolved by using the 'Change Turnout Sense' button. This option will ask for a port number and will invert the sense (digital polarity) of the port. This information is displayed on the port assignment screen.

... change it...

Click on the 'Change Layout Design' and the current layout will be displayed. You can add or delete symbols to change the layout as necessary. Next, save the modified design by clicking on the 'Save and Exit Design' button. Alternatively you can abandon the changes by clicking on 'Exit Design without Saving'. There is also an option to load another layout, enabling the same software to be used for a number of different layouts.

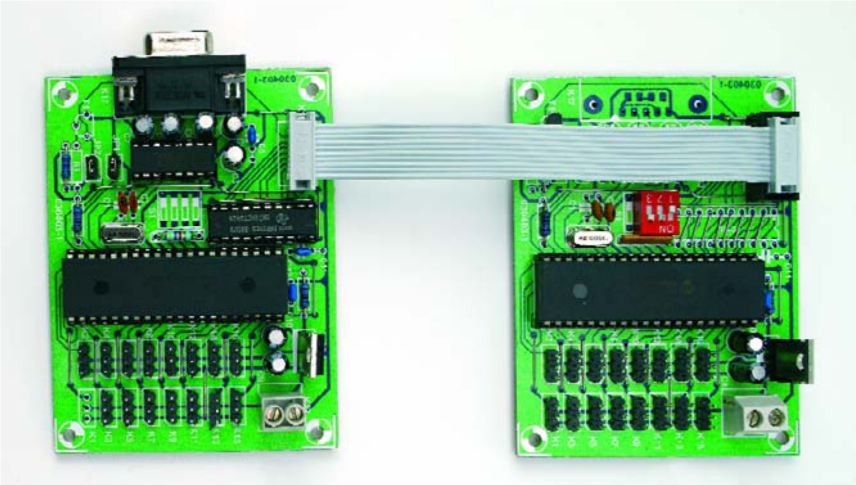


Figure 5. Master Router coupled to a Slave Router by a piece of flatcable.

... and be a train controller

Click on the section of track which you intend the route to start on and then click on the track section where you want to finish. Click on 'Plot' and the program will attempt to find a path from start to finish. If it succeeds, the route will be highlighted in yellow. If not, 'No Route Found' will be visible. If the routing is successful but you feel there is a better way through then simply click on 'Plot' again until your preferred route is selected.

The yellow route can be operated directly using the appropriate button, alternatively it can be stored against one of the three coloured routes and operated at any time. Up to three different routes can be stored in this way.

Avoid starting or ending your route from a crossover, turnout or end sign as these can cause the software to produce unusual results — normally false 'No Route' messages, which, come to think of it, is not as bad as 'No Train Services Today'.

(030309-1)

Web pointer

www.king.ray.btinternet.co.uk/index.htm

Location photograph courtesy South Limburg Steam Railway Foundation (www.zlsm.nl)

T. Fondrat

Programming with Word

write Visual Basic programs using MS Word

Not many PC users are aware that the well-known Microsoft Word includes a programming environment for Visual Basic. It allows you to get to grips with the fundamentals of this programming language.

This article will be of particular interest to those of you who don’t have the complete Visual Basic package, but who would still like to play with a programming language under Windows. The trick here is: use **Visual Basic for Applications**, which is available in Word as well as Excel.

Step 1

Start Word. All the screendumps and programs shown here were made using Word 2000, but other versions such as Word 97 work just as well!

Step 2

From the *View* menu click on *Toolbars*. Then select *Visual Basic*, as shown in figure 1.

Step 3

The *Visual Basic* toolbar appears (figures 2 and 3).

Step 4

Now we’ll make a button called **CommandButton** (figure 4). To look at a list of the properties of the button you have to position the mouse cursor over the button, then right-click on it and select *Properties*.

A new window pops up containing all the properties of the object (figure 5).
The two most important properties are *name* and *caption*.

Give a name to the button, which in our example is **ButtonTestMessage**.
Replace the text on the button with **Click on this button!**.
You can also change other settings, such as the size, font, background colour, background picture, etc.

As an aside, Visual Basic has many windows that are clearly laid out, or not! That is left to the user, but try not to get confused by all those open windows.

Step 5

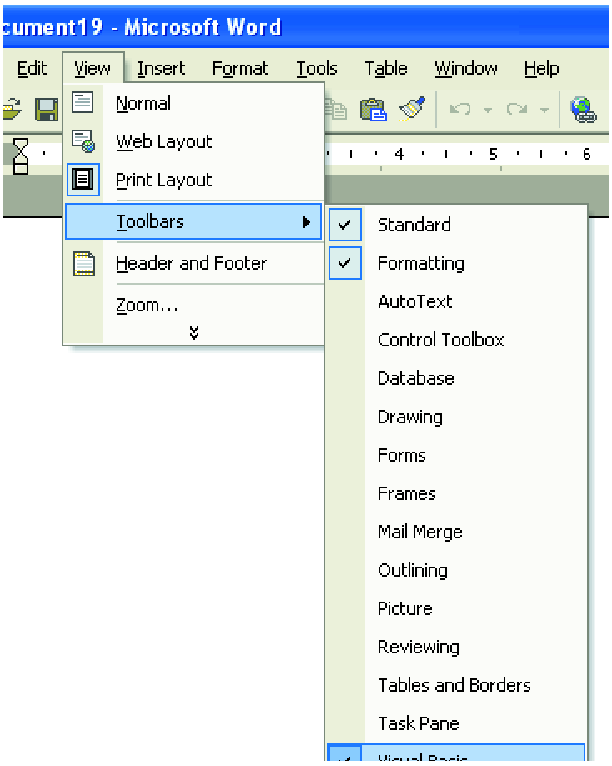
Double-click on the button.
The Visual Basic editor opens. The procedure *ButtonTestMessage_Click* is created, if it didn’t exist already, and the cursor appears HERE, where you will write your program (figure 6)!

This procedure is run with every click on the **ButtonTestMessage** button.

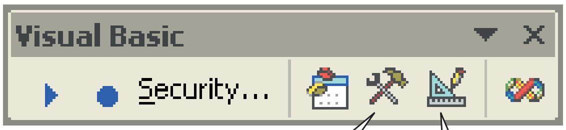
Or more precisely, the procedure *Click* is linked to the object **ButtonTestMessage**, of type *CommandButton* (also described as: belongs to the CommandButton class).
The code for this procedure is entered here.

Step 6

Enter the line: *MsgBox (“You’ve clicked here!”)*
(figure 7a).



1

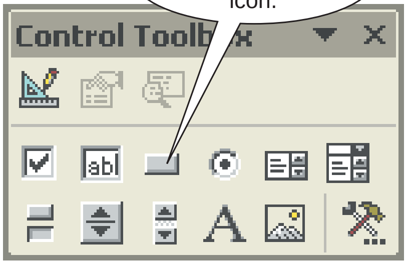


Click on the ‘Control Box’ icon.

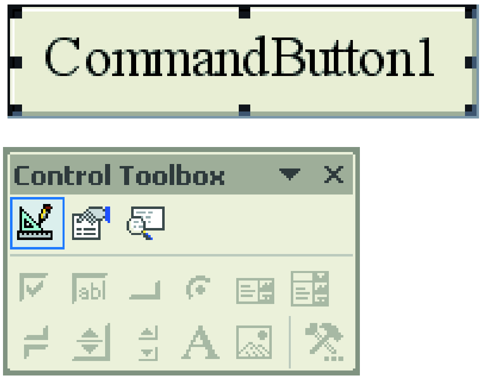
Activate ‘Design Mode’.

2

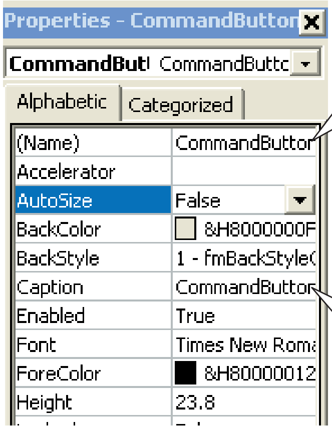
Click on the ‘Command Button’ icon.



3



4

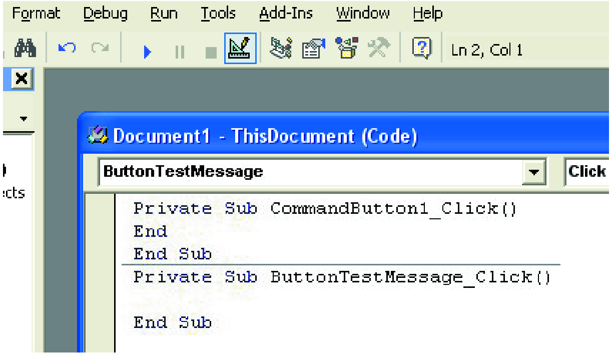


(Name)	CommandButtor
Accelerator	
AutoSize	False
BackColor	&H8000000F
BackStyle	1 - fmBackStyle(
Caption	CommandButtor
Enabled	True
Font	Times New Roma
ForeColor	&H80000012
Height	23.8

Name of the button you’re about to create. Choose an unambiguous and clear name stating its function. For example: ‘LightSwitch’.

Text appearing on the button. Do no confuse it with its functional name.

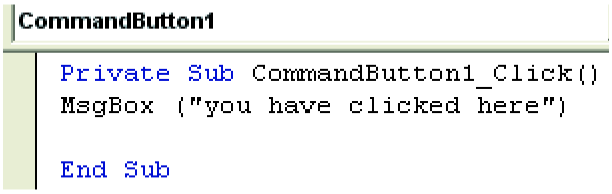
5


Format Debug Run Tools Add-Ins Window Help
Ln 2, Col 1
Document1 - ThisDocument (Code)
ButtonTestMessage
Click

```
Private Sub CommandButton1_Click()
End
End Sub
Private Sub ButtonTestMessage_Click()

End Sub
```

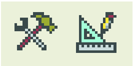
6



```
Private Sub CommandButton1_Click()
MsgBox ("you have clicked here")

End Sub
```

7a



7b

Step 7

Return to Word, using Alt-Tab for example, and return to Run-mode by deselecting the two icons on the Visual Basic toolbar (figure 7b).

You can also minimise the *Visual Basic* toolbar if you want to have a clear desktop, but you will have to enable it again if you want to continue programming.

Step 8

Save the file in Word format and test how it runs.

Summary

A small program can be written in seconds using MS Word. You should now have an idea of all the possibilities it offers. The same method can also be used in MS Excel to program in Visual Basic: create windows with buttons, input fields for text or data, pictures, etc. Show the information in graphs, personalise the Word menu with user-defined commands, run other programs. It is also possible (although that really needs a separate article) to use API's under Windows, to program the parallel port for example. Complex programs that hide the standard menus can be written this way. The users will think that they're running a dedicated program, whereas the reality is that it runs under Word or Excel.

(040046-1)

Help, my program doesn't work!

Three possible solutions and some advice.

1.Security level.
If a button doesn't work you should first check that the security level has been set to 'Medium'. The setting of this level determines how Word deals with macros and Visual Basic code that are embedded in documents. Word always asks this question when a document is opened. A high security level gives you no choice and programs written in Visual Basic cannot be opened.

The security level in Word can be changed as follows:

– From the Tools menu click on **Macro** and then on **Security**.

– Select the **Medium** *security level.*

– Now save your work and close Word.

– Re-open the document.

When the document is opened, Word gives a warning that it contains macros. In this instance, we click on **Enable Macros**.

2. Problems when running a program.
Your program could contain a bug, which has to be tracked down.

The debugger is a utility that stops your program at the point where the error occurs! The screendump below shows a typical window that you could see.

The Visual Basic editor then opens automatically.

3. It still doesn't work.
It's possible that the cause of the problem lies elsewhere. Try the following: Save the program and close Word. Open the document again. Amazingly, this trick is often enough to get a program to work.

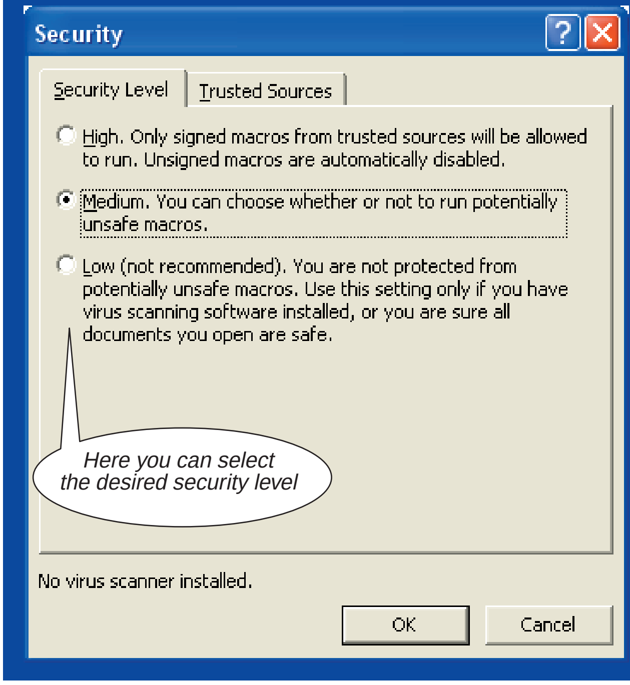
If none of these help, there is a large amount of information on various Internet sites and in discussion forums.

Internet links:

Further information on Visual Basic
http://msdn.microsoft.com/vbasic/

Visual Studio
http://msdn.microsoft.com/vstudio/

Source codes in VB
http://www.vbcode.com/



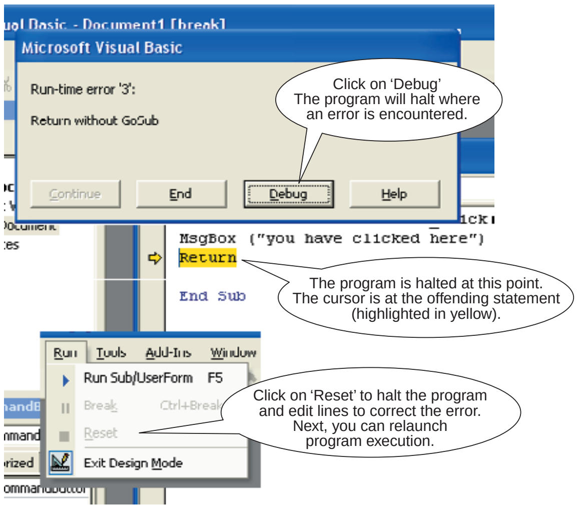
○ High. Only signed macros from trusted sources will be allowed to run. Unsigned macros are automatically disabled.

● Medium. You can choose whether or not to run potentially unsafe macros.

○ Low (not recommended). You are not protected from potentially unsafe macros. Use this setting only if you have virus scanning software installed, or you are sure all documents you open are safe.

No virus scanner installed.

Here you can select the desired security level



Run-time error '3':

Return without GoSub

```
MsgBox ("you have clicked here")
Return

End Sub
```

Click on 'Debug'
The program will halt where an error is encountered.

The program is halted at this point.
The cursor is at the offending statement (highlighted in yellow).

Click on 'Reset' to halt the program and edit lines to correct the error.
Next, you can relaunch program execution.

New Layout (1) I have been an Elektor reader since 1977. Your magazine has always kept up with times and this new format is no exception. Graphics/icon based format is the way forward and you could not have done better. One picture tells a thousand words. Elektor is simply the best.
Cemal Ozturk

New Layout (2) Hi, you have asked for feedback on the restyled Elektor magazine. I like the new style, it is reminiscent of the style used in the French and Italian electronics mags which I often buy when on holiday abroad. I would like to make one adverse comment, I do not like white characters on black background. The readability is very poor. If one must have a black background then yellow or magenta is quite good. I know this applies to the monitor and may well apply to the printed page. Perhaps you could do an experiment in a future issue and invite readers' comments. I have been a reader for very long time and have some of the earliest issues.
Mike Baker

New Layout (3) I'm afraid the 'new look' is a bit of a disappointment. The use of a full page image preceding major articles is clear attempt to fill the magazine without having to create content. The April 2004 issue has at least seven pages wasted in this manner (nearly 10% of the issue!).
Magazine subscribers such as me will have to think long and hard when it comes to renewal time if this practice is maintained. I would rather see reprints of manufacturers' datasheets or even adverts than this. The new logo also instantly made me think of other ill-conceived re-branding exercises, such as BP's daft flower thing and the 'Royal Mail' madness during their brief 'Consignia' phase. Your old logo was instantly recognisable and yet not dated, why any company would wish to abandon years of branding is beyond me. I can only assume that there are people in the company spending too much time 'in meetings'.
If all this seems negative, it is due in no small part to the high regard I have had for your magazine in the past. I would not wish to see *Elektor Electronics* go down the same path that others have, such as *Wireless World*. Change for the sake of change is madness.
Owen Grantham.

New Layout (4) As we say on our side of the pond, it's OK. Though sometimes, as in the car industry much of the time, one prefers evolution to revolution. Still, that's a matter of taste.
However, there is a physically large flaw with the cover — none other than that great big 'e'. Maybe you had your reasons, but really, it's too clever by half, no, by more than half. It's over the top, no, beneath the bottom. I can't even begin to imagine what you were thinking. Could therapeutic quantities of Guinness Stout have been involved somehow? Rather than connoting sophistication, it calls to mind every oh-so-revolutionary but lame and hackneyed e-this, e-that, and e-the-other-thing that has ever wended its way into an e-business e-advert e-cliché during the entire course of e-history. And — e-eek — it also reminds one of Microsoft's Internet e-xplorer, which … let's not go there.
It would be a considerable kindness for you to lose it just as soon as the e-novelty wears off.
Paul Schick

New Layout (5) I was very surprised to receive a copy of your restyled Elektor Electronics magazine. I am 54 years old and wear glasses, which were barely adequate to resolve the printed words on pages 5 onwards. When it came to reading the circuit diagram, I had to resort to using a magnifying glass to read the component values! I consider my eyesight to be average for my age and therefore assume that your team are much younger than me and therefore have better eyesight? My advice to you is to not proceed with this mini version if you wish to retain your older readers and indeed younger readers with less than perfect eyesight.
C. Sinclair

New Layout (6) I think it's great. Keep the good articles coming. Anything of educational basics for amateurs like me would be welcome.
Brian Moore

New Layout (7) The magazine appears to have grown up. Please keep it in your new format.
Edward Williams

Project c+ (1) Dear Jan, referring to the 'time inversion' article, The oscillograms in the article fail to use a time reference in order to *locate* ourselves in *time*.
At the time of pressing the trigger (to initiate the approximate 1-second pulse to drive the pulse shaper) we should establish the reference on the oscilloscope 1 and 2 (channels B).
At the moment it seems that the very long transmission line is a momentary short circuit to the poor overloaded opamp... the wave travels to the receiver shows itself on scope 2 and LED, then travels back to the source and at last makes the led light up.
Of course there is a delay as the signal has travelled twice the distance. Nice try....
As I was looking for your email address I suddenly thought of the month… I hope some people did not trim too much. I like the touch with the high precision parts. Thank you for a fantastic magazine.
G. Brennet (by email)

Project c+ (2) Dear Editor, I am going to buy shares in Project c+. You are on a winner!
Don Phelps (by email)

Project c+ (3) Hi Jan, you seem to have overlooked an important part when analysing your circuit. The connection between the sending and receiving end is a *distributed network*. If you include this in your transfer-function, you will find that at a given time *t*, the solution for the total network is 040104, and the solution for

the sending part alone is 040204. It is quite obvious that 040104 is just one time unit ahead of 040204. I hope this has clarified the obvious.
Knut Bakke (by email)

Your projects, my projects

Dear Sirs, I have been a long time reader of Elektor, and would like to thank all Elector (*sic*) staff for their superb circuits.
Your April 2004 magazine is a success and with each new issue you prove that electronics can be conveyed in interesting ways.
My favourite pastimes are audio and music and I've actually built many of your circuits. Some of these may be seen on the photographs on the enclosed CD-ROM. Equipment cases, screen prints and loudspeaker cases are all home made.
As I enjoy every good circuit you provide and build lots of them, I thought I'd show you a few examples of finished equipments.
D. Bozanic

The photographs on the CD-ROM Mr. Bozanic kindly sent us certainly prove his skills at building high-end audio equipment. The photograph reproduced here shows his version of the Prelude stereo preamp, a classic we published decades ago.

Stealing current

Dear Editor, I am constructing the project 'Burglar Alarm' published in the January 2004 issue of Elektor. I have a few doubts about the power supply section. Apparently you have not provided overcharge protection for the 12-V SLA battery. Will this not reduce battery life or destroy it due to over charging?
Also, the battery charge current limiting resistor used is 1 k (R1) and the series diode (D6) adds around 500 ohms to it. Will this combination provide enough current for efficient charging? I would like to use a 12 V/7 Ah SLA battery.
Kinjal (by email)

With lead-acid batteries (which includes sealed SLAs), charging with a certain maximum voltage such as 13.2 V or even 13.8 V also serves as an effective overcharge protection. Regarding the charge current, this is intended for continuous charging. There's nothing wrong with the current of about 0.3 mA used in this case (battery capacity 1.2 Ah) as the circuit is continuously connected to the mains. In your case, you intend to use a much larger battery and the current may be increased by changing the 1-kΩ resistor into 680 Ω. Do not use lower values as that may cause R1 to burn out.

Contact those CPLDs

Dear Jan, your 'Hands-On CPLDs' project is supported by a beautiful double-sided board. Unfortunately, the length of the PLCC socket pins and DIP switch pins is such that they can not be soldered at both sides of the board although it is clear that signals have to be routed to a number of these pins. Do you have a solution to this problem or am I barking up the wrong tree?
Philip Hyams

Double-sided boards supplied through our Readers Services are through-contacted unless otherwise stated. You need to solder at the underside of the board only. The through contact will then establish an electrical connection to any tracks that may run at the component side.

Exotic opamp

Dear Editor, I'd like to build the Audio Level Check for Line Input project published in the November 2002 issue as I find that signals from my stereo system are causing overdriven recordings. Unfortunately the TS924IN is unavailable locally, can I use a TL074 instead?
Bernard Hook (by email)

Because of the minimum supply voltage it is not possible to use a 'regular' opamp o the type you suggest. If you need to find an alternative, go for a 'rail to rail' opamp.

Corrections & Updates

MIDI Lights & Slide Control

March 2001, p. 26-33, 000179-1.
A few readers have reported lamp flicker when the control is operated. No problems are observed with constant brightness. Flicker may be suppressed by lengthening the synchronisation pulse. This is easiest done by increasing the value of C1 and C2 to 3.3 nF.

MailBox Terms

- Publication of reader's correspondence is at the discretion of the Editor.
- Viewpoints expressed by correspondents are not necessarily those of the Editor or Publisher.
- Correspondence may be translated or edited for length, clarity and style.
- When replying to Mailbox correspondence, please quote Issue number.
- Please send your MailBox correspondence to:
editor@elektor-electronics.co.uk or
Elektor Electronics, The Editor, P.O. Box 190,
Tunbridge Wells TN5 7WY, England.

the sending part alone is 040204. It is quite obvious that 040104 is just one time unit ahead of 040204. I hope this has clarified the obvious.
Knut Bakke (by email)

Your projects, my projects

Dear Sirs, I have been a long time reader of Elektor, and would like to thank all Elector (*sic*) staff for their superb circuits.
Your April 2004 magazine is a success and with each new issue you prove that electronics can be conveyed in interesting ways.
My favourite pastimes are audio and music and I've actually built many of your circuits. Some of these may be seen on the photographs on the enclosed CD-ROM. Equipment cases, screen prints and loudspeaker cases are all home made.
As I enjoy every good circuit you provide and build lots of them, I thought I'd show you a few examples of finished equipments.
D. Bozanic

The photographs on the CD-ROM Mr. Bozanic kindly sent us certainly prove his skills at building high-end audio equipment. The photograph reproduced here shows his version of the Prelude stereo preamp, a classic we published decades ago.

Stealing current

Dear Editor, I am constructing the project 'Burglar Alarm' published in the January 2004 issue of Elektor. I have a few doubts about the power supply section. Apparently you have not provided overcharge protection for the 12-V SLA battery. Will this not reduce battery life or destroy it due to over charging?
Also, the battery charge current limiting resistor used is 1 k (R1) and the series diode (D6) adds around 500 ohms to it. Will this combination provide enough current for efficient charging? I would like to use a 12 V/7 Ah SLA battery.
Kinjal (by email)

With lead-acid batteries (which includes sealed SLAs), charging with a certain maximum voltage such as 13.2 V or even 13.8 V also serves as an effective overcharge protection. Regarding the charge current, this is intended for continuous charging. There's nothing wrong with the current of about 0.3 mA used in this case (battery capacity 1.2 Ah) as the circuit is continuously connected to the mains. In your case, you intend to use a much larger battery and the current may be increased by changing the 1-kΩ resistor into 680 Ω. Do not use lower values as that may cause R1 to burn out.

Contact those CPLDs

Dear Jan, your 'Hands-On CPLDs' project is supported by a beautiful double-sided board. Unfortunately, the length of the PLCC socket pins and DIP switch pins is such that they can not be soldered at both sides of the board although it is clear that signals have to be routed to a number of these pins. Do you have a solution to this problem or am I barking up the wrong tree?
Philip Hyams

Double-sided boards supplied through our Readers Services are through-contacted unless otherwise stated. You need to solder at the underside of the board only. The through contact will then establish an electrical connection to any tracks that may run at the component side.

Exotic opamp

Dear Editor, I'd like to build the Audio Level Check for Line Input project published in the November 2002 issue as I find that signals from my stereo system are causing overdriven recordings. Unfortunately the TS924IN is unavailable locally, can I use a TL074 instead?
Bernard Hook (by email)

Because of the minimum supply voltage it is not possible to use a 'regular' opamp o the type you suggest. If you need to find an alternative, go for a 'rail to rail' opamp.

Corrections & Updates

MIDI Lights & Slide Control

March 2001, p. 26-33, 000179-1.

A few readers have reported lamp flicker when the control is operated. No problems are observed with constant brightness. Flicker may be suppressed by lengthening the synchronisation pulse. This is easiest done by increasing the value of C1 and C2 to 3.3 nF.

MailBox Terms

- Publication of reader's correspondence is at the discretion of the Editor.
- Viewpoints expressed by correspondents are not necessarily those of the Editor or Publisher.
- Correspondence may be translated or edited for length, clarity and style.
- When replying to Mailbox correspondence, please quote Issue number.
- Please send your MailBox correspondence to: editor@elektor-electronics.co.uk or Elektor Electronics, The Editor, P.O. Box 190, Tunbridge Wells TN5 7WY, England.