ELEKTOR ELECTRONICS

THE ELECTRONICS & COMPUTER MAGAZINE

APRIL 2007
£2.95

www.elektor-electronics.co.uk

96 kHz Sampling Rate Converter

MIDI on RS232 Port

LED Thermometer

Darkroom Timer

PCI Design Basics

Crescendo Millennium Edition

make-over for a legendary amplifier

04

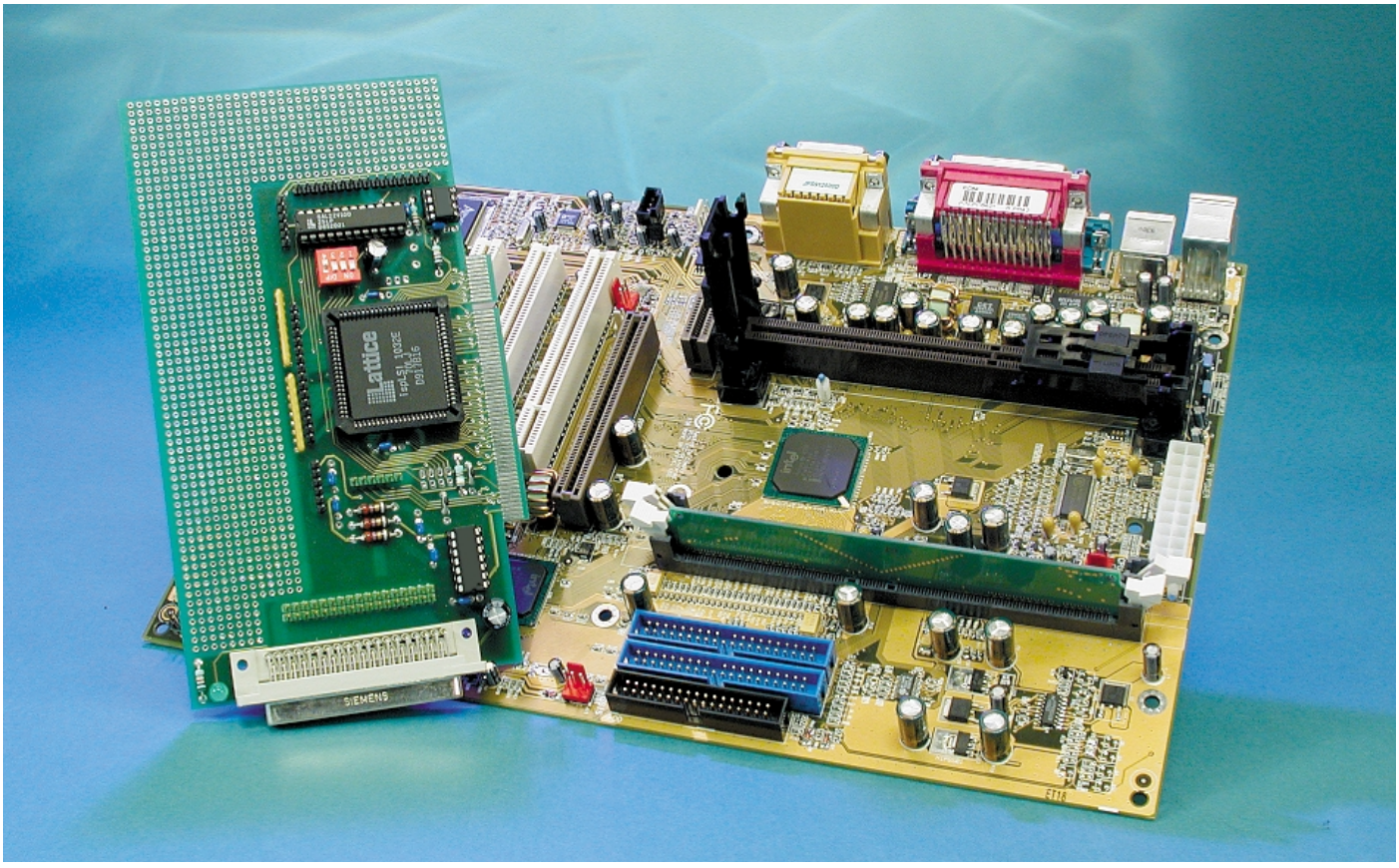
9 770268 451081

PCI Bus Prototyping Card (2)

developments from ISA to PCI

By B.Kluth and C.Kluth (B&C Kluth GbR)

Ever since IBM-compatible PCs have been available, a new bus system has been introduced each time there was a major development step. The PCI bus, which represents the current state of affairs, is not limited to IBM-compatible computers and should survive for several PC generations, thanks to its high level of performance. The price for this is a relatively complex structure, which is thoroughly described in this article.



In 1985, IBM unveiled the first AT computer. The bus system used at that time was the ISA bus. Shortly after this introduction, however, it quickly became evident that the ISA bus no longer represented the current state of technical development. It was clear that '286 processors were already being slowed down by the bus, for example when working with graphics cards or high-speed network technologies. This large loss in performance was due to backwards compatibility with older 8-bit systems. The rather low bus clock rate of 6 MHz, which was later increased to 8 MHz, also did not help matters. The result was a theoretical data transfer rate of 8 MB/s with a bus width of 16 bits. In actual practice, however, the maximum achievable data rate was around 4 to 6 MB/s.

The MCA bus

This problem was the incentive for IBM to develop a new bus system suitable for 32-bit operation. In 1987, the Micro Channel Architecture (MCA) was introduced into the market. This system had a 32-bit data and address bus with multi-master capability, and it promised data transfer rates of up to 16 MB/s. Unfortunately, the new computers did not come with any old-style ISA slots, and consequently this bus was not a commercial success.

The EISA bus

The experience with the failed MCA bus played a major role in the development of the Extended Industry Standard Architecture (EISA) bus. The EISA bus works with an 8-MHz clock and 32-bit technology. With a bus width of 32 bits, the resulting data transfer rate is 16 MB/s in standard mode and 32 MB/s in burst mode. Later on, two other burst modes were added (EMB-66 and EMB-133), which allowed data rates of up to 133 MB/s. However, this level of performance had its price, so it was only used in expensive server systems.

The VLB bus

Various manufacturers of peripheral equipment, along with a now somewhat nervous market, sought a bus system that was inexpensive but still capable of high performance. After a few false starts, this resulted in the VLB bus (VESA Local Bus). It also had a bus width of 32 bits and a clock rate of 25 to 60 MHz. Various versions of this bus appeared, namely VLB-1.0 (clock rate 25–40 MHz), VLB-2.0 (clock rate 25–50 MHz) and VLB-64 bit (clock rate 25–60 MHz). The main problem with this bus technology was non-conformance with the various specifications. The VLB-1.0 specification allowed only two slots to be used up to a maximum CPU clock rate of 40 MHz. However, some manufacturers built systems with up to three slots at a 50 MHz clock rate. This resulted in significant problems with the stability of individual systems.

The PCI bus

The initial design of the Peripheral Component Interface (PCI) bus was made by Intel in 1991. The main reasons for the development of this new bus were:

– to achieve higher data rates than with the 16-bit ISA bus,
– to achieve better electromagnetic compatibility (EMC) than with previous systems,
– to achieve an assured future with following processor generations.

In 1992, the design was complete and the first PCI bus was unveiled. In the course of subsequent development of this bus system, various specifications arose (versions 1.0, 2.0, 2.1 and 2.2, which is the current version). The result of the demanding requirements defined in 1991 is a 32-bit bus system that transfers data and addresses in a time-multiplexed manner and that can execute variable-length burst cycles. Intel was able to win acceptance for a complete, realistic and well-structured set of definitions of all important bus signals as the basis for the PCI bus. One objective in the development of this hierarchical, upwardly open bus system was to avoid repeating the mistakes that were made with the previous bus systems.

PCI bus characteristics

The PCI bus consists of three distinct modules:
– the Data Path Unit,
– the Expansion Bus Interface,
– the Host Bridge, with a cache DRAM controller.

The Data Path Unit is used to set up a 32-bit link to the individual components in the system. Other bus systems can be connected via the Expansion Bus Interface (e.g. ISA or AGP). The Expansion Bus Interface and additional system bridges provide for upwards compatibility with the ISA bus system, for example, or new technologies such as AMR.
The Host Bridge is the central component of the PCI bus system. It can be used to create a link between the PCI bus and the CPU. In addition, it converts PCI cycles into CPU cycles and vice versa. It makes the PCI bus processor-independent, in contrast to other bus systems, since all that is necessary to connect the PCI bus to a particular processor type (Intel, Alpha, AMD etc.) is to use a different Host Bridge. This characteristic makes the PCI bus system relatively independent of future processor generations, as well as extensible.

The PCI specification

The PCI specification allows a total of ten devices on a PCI bus. Since the Host Bridge is seen as a PCI device by the PCI bus, there is basically provision for only nine devices. These can be used for on-board components (SCSI, EIDE, LAN and so on) or for PCI slots for connecting PCI cards. In this regard, you must bear in mind that each slot demands two devices, so that at most four slots can be driven by a PCI bus. However, the number of slots can be increased by connecting a second PCI bus system to the Expansion Bus Interface. In total, up to 256 busses can be strung together, with the first 255 busses being PCI busses and the final bus allowed to be an ISA, EISA or VL bus, or even an MCA bus. Large systems can be implemented in this manner (see **Figure 1**).
Another option for slot extension can be seen on current motherboards (such as the Abit KT-7, Gigabyte ZX and Epox EP-8KTA+). Here up to six slots are driven on one motherboard by means of IRQ sharing. In this scheme, slots 5 and 6 (for example) share an IRQ. However, if more than four PCI cards are used, performance suffers. Also, it is not possible to guarantee in principle that all possible PCI card combinations will function without any problems.

A wide variety of PCI cards can be connected via the slots. The following types of cards are distinguished:
– single-function PCI cards,
– multi-function PCI cards,
– multi-device PCI cards.

Single-function cards can for example be graphics cards or SCSI host cards. A PCI device can have up to eight different I/O functions, which yields the multi-function PCI cards. A multi-device PCI card can consist of several of the above-mentioned types of PCI cards. With a multi-device card, a wide variety of devices can be connected via a PCI-to-PCI bridge. This means that a multi-device PCI card represents a complete PCI bus system.

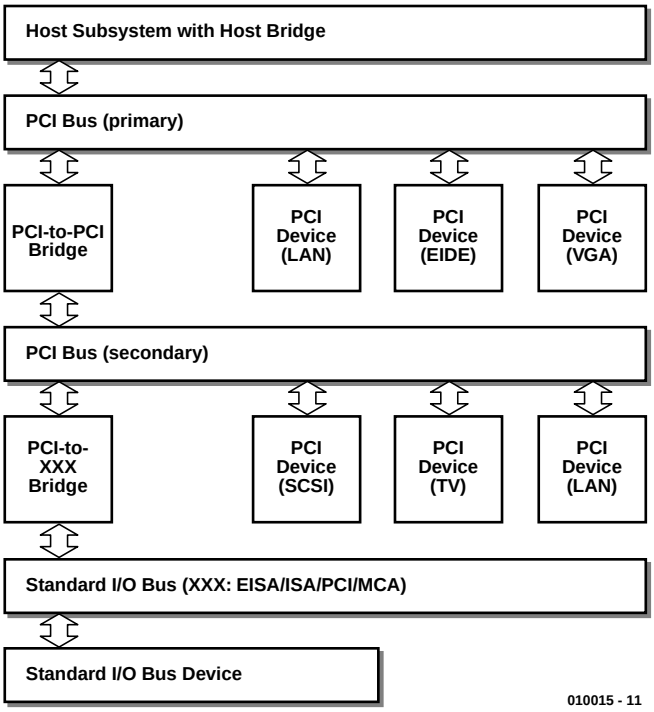


Figure 1. The hierarchical structure of the PCI bus system.

Plug-and-Play function (PnP)

With regard to the PCI bus, Plug-and-Play means the following:
After the computer is switched on, the BIOS finds all PCI devices and queries each device for the resources that it needs. The necessary resources usually consist of I/O addresses, IRQ numbers, DMA channels and memory regions used by the device. If there are overlaps in the resources needed by two different cards, the BIOS attempts to enable both cards to work in the system by reconfiguring one of the cards. If this cannot be done, the BIOS simply disables one of the two cards. However, this happens only very rarely. The resources used by each PCI card are made available to the operating system being used via the ESCD database of the BIOS. Furthermore, this information is also stored in the Configuration Space of the PCI card, which is a memory region of up to 256 bytes (see **Table 1**). The BIOS has also obtained information about the necessary resources from this memory region, and it makes this information available to other applications via special BIOS interrupt calls. Since these BIOS-specific parameter registers are managed dynamically in each system, there is not any standardised, fixed memory location (such as a MEM address) specified for them. The only way to obtain this information is to use a software interrupt, which is prescribed for all motherboard manufacturers.
Using *Status* and *Command*, it is possible to read out information regarding the cards being used, and in some cases to modify the settings. The Vendor ID is the number of the manufacturer of the card being used. This number is assigned by the PCI SIG (PCI Special Interest Group). The PCI SIG is a consortium that supervises all specifications for the PCI bus. The official PCI bus specifications, which contain detailed explanations of the registers, can also be obtained via this organisation.
PCI devices can be divided into two groups, called Initiators (masters) and Targets (slaves). An Initiator causes a data transfer to take place by assuming control of the control signals and specifying the details of the data transfer (addresses, start, etc.). A Target generates a handshake signal for the Initiator after it recognises its address on the PCI bus (initialisation phase). In addition, it can indicate whether data are available or whether it is ready to receive data, and it can signal a request for wait cycles. The collective functions of a PCI bus system can be utilised by employing these two groups.
However, it is possible for all PCI devices on a bus to represent PCI bus masters. In this case, the arbitration logic must decide which mas-

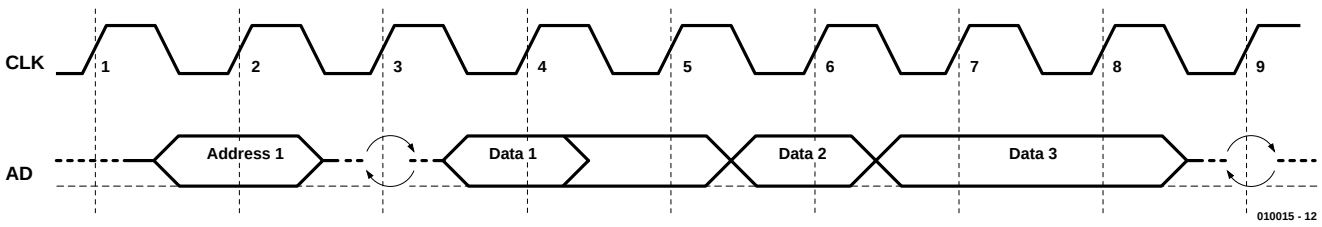


Figure 2. Burst mode timing.

ter is allowed to use the PCI bus (arbitration phase). During this phase, the two PCI devices must determine which mode will be used for the data transfer: burst mode or non-burst mode. Burst mode (**Figure 2**) is normally used to transfer four or more DWORDs (a double word is 32 bits of data), with the address being sent first, followed by the associated data. In non-burst mode (**Figure 3**), a DWORD address is sent before each DWORD. This leads to different data rates for the two modes. In non-burst mode, the resulting data rate is 44 MB/s for reading (3 cycles per DWORD) and 66 MB/s for writing (2 cycles per DWORD). Since one cycle is needed for each subsequent DWORD in burst mode, the data rate approaches 117 MB/s as the burst length (number of DWORDs) increases. With the **PCI Bus** 2.1 specification (32-bit PCI bus at 33 MHz), data transfer rates of up to 117 MB/s can be achieved. The **PCI Bus** 32-bit 2.1 specification (32-bit PCI bus at 66 MHz) allows for data transfer rates up to 234 MB/s, and in the **PCI Bus** 64-bit 2.1 specification (64-bit PCI bus at 66 MHz), the maximum data transfer rate can be as high as 468 MB/s (see **Table 2**).

Table 1 PCI card Configuration Space

31	16	15	0	
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Revision ID	08h
BIST	Header Type	Latency Timer	Cache Line Size	0Ch
Base Address 0				10h
Base Address 1				14h
Base Address 2				18h
Base Address 3				1Ch
Base Address 4				20h
Base Address 5				24h
Cardbus CIS-Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved				34h
Reserved				38h
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line	3Ch

PCI bus timing

PCI is a synchronous bus, which sends and receives all data transfers with reference to a system clock (CLK) signal. The mutual relationships of the individual signals can be illustrated using the Read command as an example (see also the timing diagram shown in **Figure 4**). The individual steps in processing a Read command are the following:

Cycle 1:
The PCI bus is in the idle state.

Cycle 2:
FRAME# is activated by the master device to indicate the start of a transaction.
AD holds an address.
C/BE# holds a bus command.
IRDY#, TRDY# und DEVSEL# are in a turn-around cycle, since a driver changeover occurs.

Cycle 3:
AD performs a turn-around, since control passes from the master to the target.
C/BE# is driven by Byte Enable.
IRDY# is active, because the master is ready to read the data.
TRDY# remains inactive, since no target has been found yet.

Cycle 4:
AD holds the first data.
TRDY# is active, since the first readable data are on the bus.
DEVSEL# is active, since the target has recognised the address from Cycle 2.

Cycle 5:
AD still holds the first data from data phase 1.
TRDY# is inactive, since the target (data source) wants to insert a pause.

Cycle 6:
AD holds the data from data phase 2.
TRDY# indicates that the data can be read.

Cycle 7:
AD carries the data for data phase 3.
IRDY# is deactivated by the master, since it wants to have a wait state.

Cycle 8:
FRAME# is deactivated by the master, since the final data block will now be transferred.

Cycle 9:
FRAME#, AD and C/BE# initiate turn-arounds, so the drivers can change over.
IRDY#, TRDY# and DEVSEL# are inactive, since no transaction is taking place.

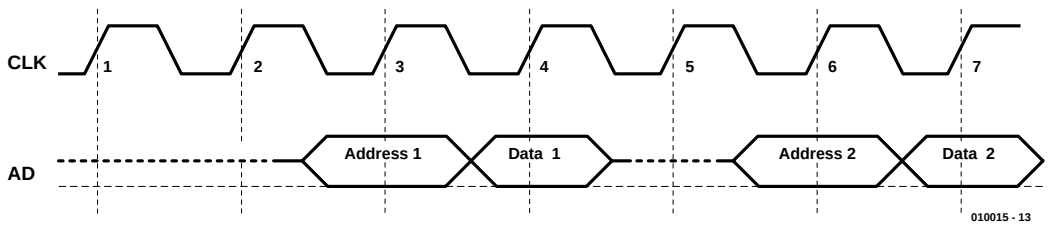


Figure 3. Non-burst-mode timing.

Table 2 The various PCI specifications and associated data transfer rates				
	32 Bit 1.0 PCI-Bus	**64 Bit 2.0 PCI-Bus**	**32 Bit 2.1 PCI-Bus**	**64 Bit 2.1 PCI-Bus**
Bus type	synchronous	synchronous	synchronous	synchronous
Clock rate, MHz	33	33	66	66
with slots	33	33	66	66
Data bus width, bits	32	64	32	64
Address bus width, bits	32	32	32	32
Number of devices	10	10	10	10
Number of slots	4	4	4	4
Max. burst length	unlimited	unlimited	unlimited	unlimited
Data rate at 33 MHz with no extra wait cycles, MB/s				
Non-Burst-Read	44	88	44	88
Non-Burst-Write	66	132	66	132
Burst-Read	106	211	106	211
Burst-Write	117	234	117	234
Data rate at 66 MHz with no extra wait cycles, MB/s				
Non-Burst-Read			88	172
Non-Burst-Write			132	264
Burst-Read			211	423
Burst-Write			234	468
Autoconfiguration			yes	yes
Concurrency			yes	yes
Interrupt Sharing			yes	yes

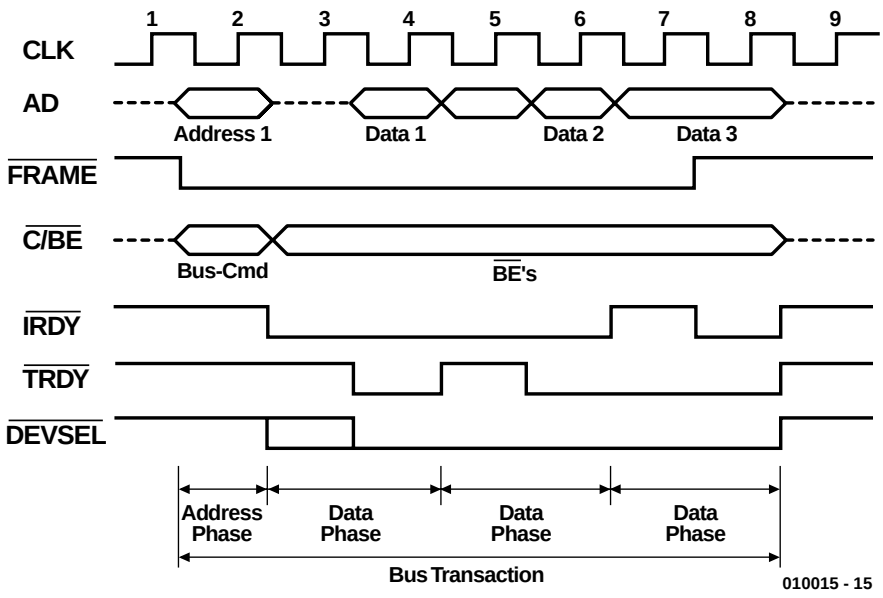


Figure 4. PCI timing diagram for the Read command.

The bus is in the idle state, since FRAME# = 1 and IRDY# = 1.

Figure 5 shows an overview of the individual signal lines.

PCI speed

Thanks to its independence from the speed of the processor, the PCI bus can offer new possibilities for applications such as multimedia, networking, instrumentation and many others. This is primarily due to its very high data transfer rates (especially the PCI Bus 64-bit 2.1 version), which allow data to be processed faster between PCI cards and the processor, in both directions. If you use an 'old' ISA sound card in combination with a PCI 3-D graphics accelerator card, you could be slowing down your system by up to 10–20% under certain conditions if the available performance of the PCI card is fully exploited.

The individual BIOS settings of the computer are also important with regard to stable system performance, since they have considerable influence on system stability. Thanks to the wide data bus and high clock rates of modern motherboards, the PCI bus works at higher speeds than the standard ISA bus, for example. For error-free data traffic, it is therefore often necessary to delay the PCI bus, since the motherboard cannot always match the limits of the hardware plugged into the slots. The following options adjust the length of the delay on the PCI bus for a transaction between the specified PCI slot and the CPU. The value depends on the PCI master unit that is used, among other things.

One of the most important settings is the time value of the PCI Latency Timer, which normally should be set to 32 clock cycles. This option determines how long a PCI card is allowed to reserve the PCI bus as a bus master when another PCI card has also requested access to the bus. If the setting for the duration of the PCI Latency Timer is too short or too long, it may be possible for the CPU to access the hardware faster than is permitted by the processing in the decoder. Although this can handsomely boost the speed of both the computer and the cards (espe-

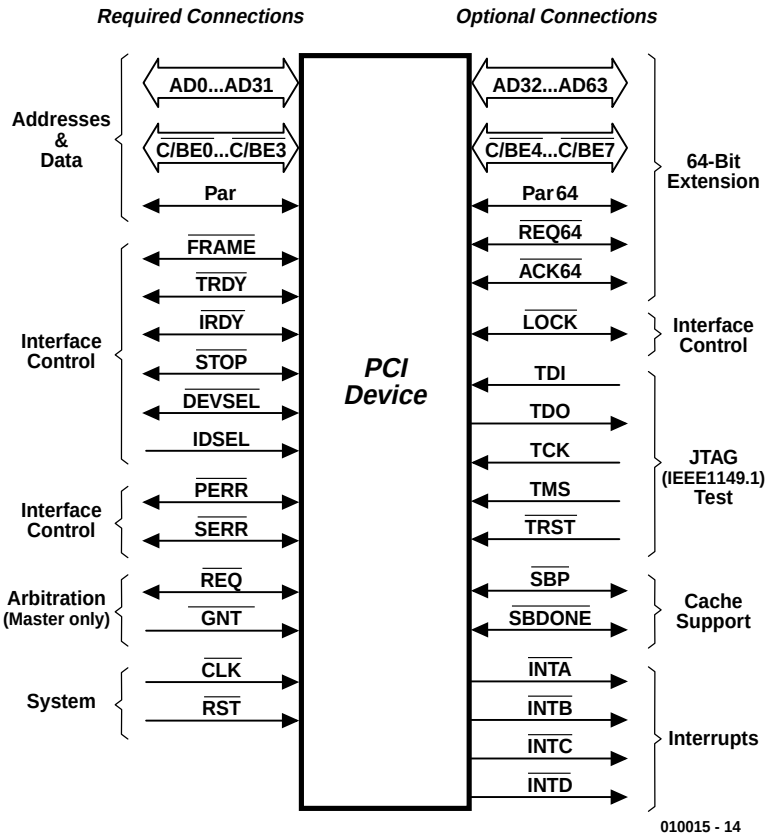


Figure 5. The signal lines of a PCI device.

cially with a value of 0), it often has consequences that may only show up after an extended period of operation (such as overheating, followed by hardware failures). Boot problems, erroneous I/O addresses, incorrect device data and hardware access problems are examples of the types of errors that can occur if the BIOS settings are incorrect (too fast for the hardware installed in the slots).

The second most important BIOS entry is the PCI Delay Transaction setting, which should be set to 'Enable'. All this does is to specify that the hardware used meets at least the PCI 2.1 specification. Internal bus protocols and special data packets will be transferred according to the specification, which makes for increased reliability between the processing unit and the PCI slot hardware. Of course, there are a lot of other factors that come into play with regard to PCI bus transactions, but they cannot be covered within the scope of this article.

Additional BIOS settings, such as PCI Master 0 WS write (= 'Disable'), PCI Dynamic Bursting (= 'Disable') and CPU-to-PCI Buffer (= 'Enable'), are also important for ensuring that the hardware is driven cleanly. The significance of the CPU-to-PCI Buffer setting, in particular, should not be underestimated. If the buffer is enabled, data flows are passed through an 'intelligent' buffer without interrupting the CPU. This buffer also coordinates the read/write cycles to the PCI device in order to avoid data conflicts. With this setting, for example, four DWORDs will be sent to the PCI card in a single process. The previously described PCI Latency Time, which more or less prescribes the separation of the data blocks, is even more important. With the OFF (Disable) setting, the data buffer is not used and processor access is only terminated after the PCI bus sends the processor a signal indicating that the bus is ready to receive data.

(010015-1)

Important PCI, PnP and BIOS information on the Internet:

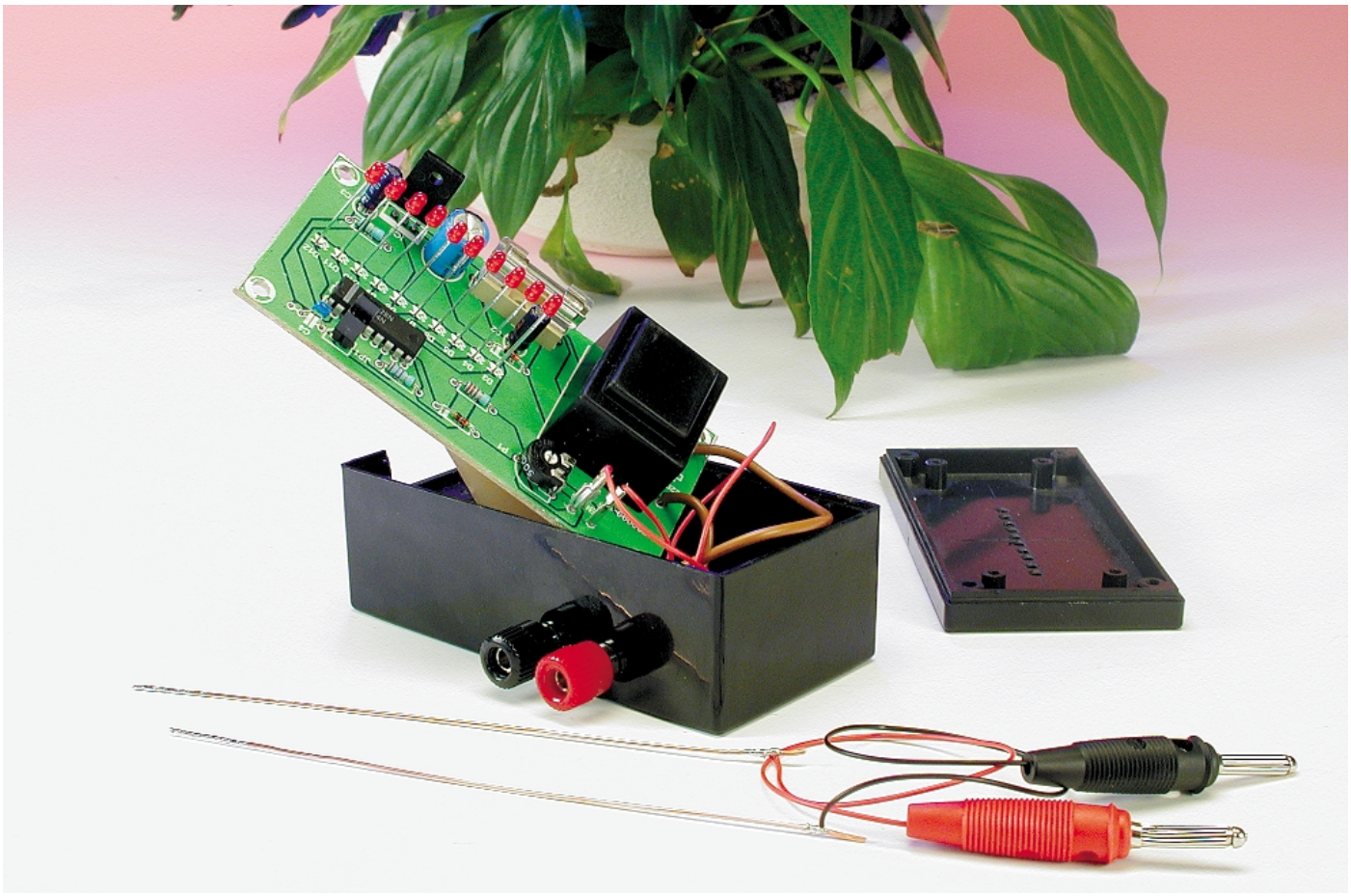
www.pcisig.com
www.PCI-Card.com
www.microsoft.com/hwdev/devdes/pciids.htm
http://members.hyperlink.net.au/~chart/
www.yourvote.com/pci/pciread.asp?sort=venid
www.microsoft.com/hwdev/respec/pnpspecs.htm
www.ping.be/bios/
http://members.aol.com/computerpage/bios.htm
www.bios-info.de/
http://home.t-online.de/home/pqtuning/bios/biosinhalt.html
www.uni-regensburg.de/EDV/PC/bs/w95tech/pnp.httippsm

Soil Moisture Tester

for houseplants

Design by A. Baur

For those of us who don't like to get their hands dirty, this simple tester quickly checks the state of their plants and how much attention they need. Recommended for all plant owners!



Plants seem to have something in common with pets. They are usually acquired or given with the best intentions, but not everybody seems to be able to look after them properly. Of course we do not expect everybody to have green fingers, but when plants are not watered enough they simply die. In any case, too much neglect usually has fatal consequences. Cactuses seem to survive such a careless treatment the longest and we have to admit that these are the only plants that manage to survive at our offices.

So what can we do about it? It's simple really. All it needs is for regular checks to feel if the soil in the pot has become too dry. But what is 'too dry'? Some people just don't seem to have the right fingers for this task. A little electronics can be used to rid us of this problem forever.

We've previously published a Houseplant Buzzer in *Elektor Electronics*, but this was designed to

stay in the pot permanently and it would beep when the soil became too dry. It's a wonderful idea, but it could turn your plants into a group of howling babies, which would get on most people's nerves!

This time we've used a different approach. The circuit described here might be very simple, but it's a very useful soil moisture tester. Two electrodes are stuck in the soil and the moisture level is shown on an LED display. The LEDs have been arranged into three colours: green LEDs indicate that the soil is damp, yellow LEDs that it's getting a bit dry and red LEDs warn that immediate action is required!

Principle

The operation of the tester depends on the property of water to become electrically conductive when it contains dissolved alkalines, oxides, etc. Damp soil can therefore be considered as an electrical conductor. The drier the soil, the less conductive it becomes and the higher its electrical resistance.

Figure 1 outlines the working of our circuit. Two electrodes, which are just lengths of wire, are used to measure the resistance of the soil. A small current is made to flow through the electrodes, which creates a voltage across them which is dependent on the soil resistance and hence on its moisture content. This voltage is rectified and used by an IC to drive a 10-LED display, which represents the voltage level. Since the minimum required moisture level varies between plants, it's been made variable by use of a preset.

The reliability of the electrodes is of the utmost importance for this design. In a damp environment, even a small direct current through the electrodes would cause one to oxidise and the other to dissolve. To prevent this electrolytic breakdown from occurring, we pass an alternating current through the electrodes, rather than a direct current. The continuing change in polarity prevents this breakdown from happening.

Simple schematic

A quick look at **Figure 2** is enough to ascertain that the full circuit is barely more complex than the block diagram. It's only really the supply that is extra. Even this is very simple, consisting of only a small mains transformer rated at 6 V/200 mA, a single rectifier and smoothing capacitor (D1/C1) and a voltage regulator which provides a stable +5 V.

The AC supply fed to the electrodes is obtained in a very simple manner: by taking it from the supply just *before* the rectifier. The preset used to set the sensitivity can be found as P1. D2, R1 and C3 rectify the moisture dependent AC signal, which is then fed to pin 5 of IC2, the heart of the circuit.

This IC used here is an old favourite, the LM3914 bargraph display driver. This 18-pin IC converts an analogue input to drive a 10-LED (linear) display. The IC contains 10 comparators, which each are connected to a reference voltage via a precision resistor network. The inverting inputs of the comparators are connected to the analogue input via a buffer stage. The LEDs are driven directly by the comparator outputs.

Pin 9 is used to set the display to bar-mode or dot-mode. In the first case JP1 should be

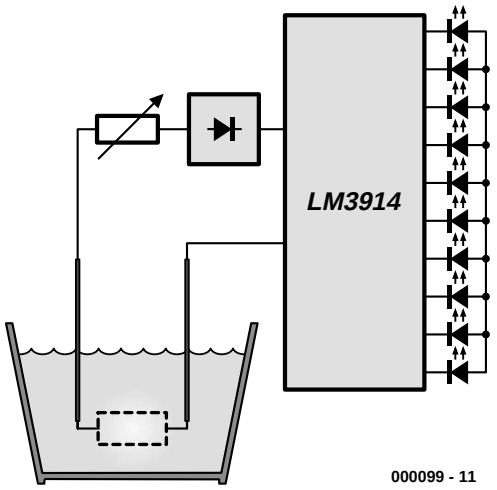


Figure 1. Damp soil is conductive. Some simple electronics are used to give an indication of its conductivity.

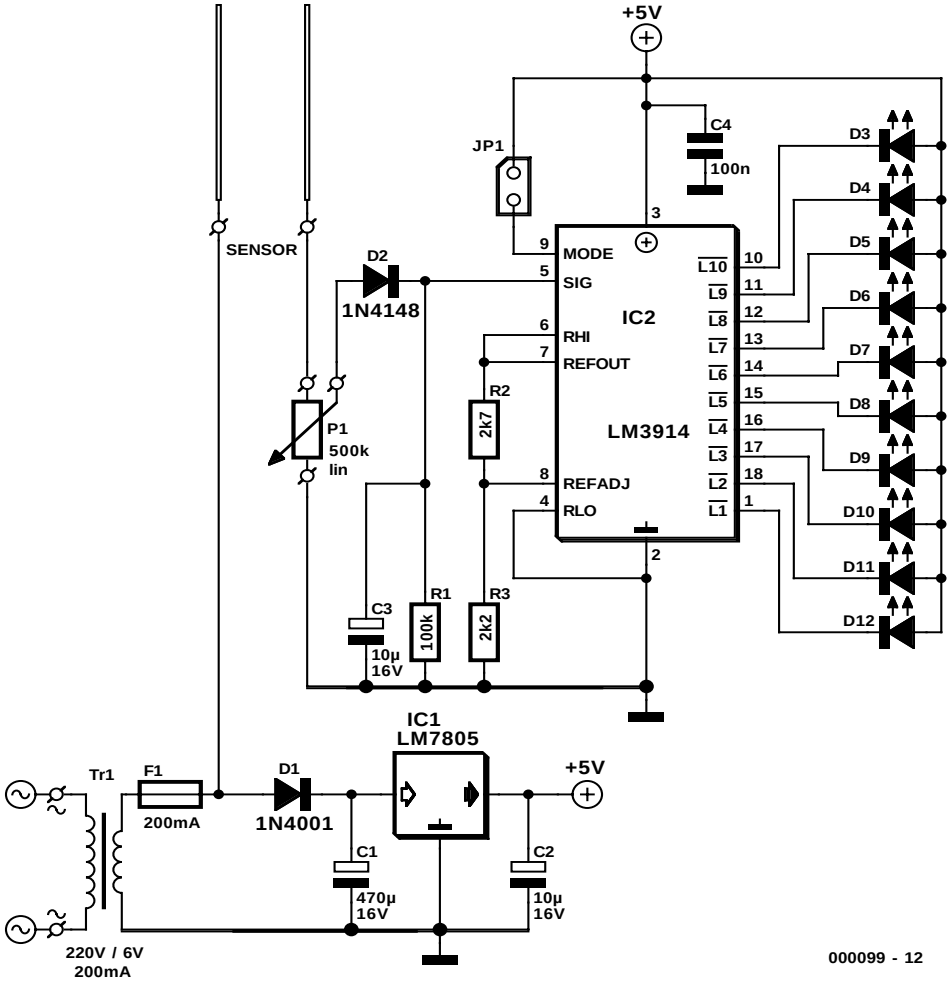


Figure 2. The full circuit is barely more complex than the block diagram.

shorted, in the second it can be left open circuit. Obviously the dot-mode gives the least current consumption of the IC.

Potential divider R2/R3 sets the reference voltage. The total value of both resistors also determines the brightness of the LEDs.

Construction

Due to the small number of components used, it's unlikely that the construction of the tester will give you sleepless nights. The use of the PCB shown in **Figure 3** makes things even easier, but unfortunately the PCB is not available via the Readers Services so you would have to etch it yourself.

Start the construction with the lowest components (resistors); that tends to be easiest. You should preferably use a socket for IC2. Take care that you get the polarity right for the diodes, electrolytics (C1, C2 and C3) and the LEDs (short leg = cathode).

The small mains transformer (Tr1) is mounted onto the PCB last. Make sure that you use a sound and well isolated cable (with a strain-relief) between the mains and the primary of Tr1. Carefully check the finished PCB before applying mains power and never work on the circuit when it's plugged into the mains!

The circuit should be mounted in a safe plastic case, with a label stuck on the bottom, stating the mains voltage and the value of the fuse. A pair of sockets for banana plugs is mounted on the case for the connection to the electrodes.

The electrodes are made from two lengths of stiff, isolated copper wire, about 10 cm long and 1 mm thick. 4 cm of insulation is removed from the ends, which are then tinned. This is to prevent the copper wire from oxidising. The connection between the electrodes and the circuit could be made with two lengths of flexible stranded cable.

Calibration and usage

Once the supply has been switched on and the electrodes have been connected, the tester is as good as ready for use. But first preset P1 needs to be adjusted. All you need for this is a glass of tap water. The electrodes are inserted into the glass of water and should be kept between 1 and 2 cm apart. This corresponds to the maximum moisture level, so we have to adjust P1 until the top green LED (D3) just lights up and D4 just extinguishes. When the electrodes are removed from the water, you should see one of the red LEDs (D10, D11, D12) light up.

Since this absolute maximum level will not occur very often, the tester could be calibrated more practically. A pot plant should be watered liberally, after which the electrodes are inserted into the soil, again keeping them between one and two cm apart. P1 is then adjusted until one of the green LEDs lights up. You will probably find many opportunities to check that one of the three red LEDs lights up when testing a plant that hasn't been watered for three weeks.

And that's it!

After the previous description it should be clear how the tester should be used. The electrodes should always be kept the same distance apart (between one and two cm), perhaps using a spacer, and the tinned ends should always be completely inserted into the soil.

(000099-1)

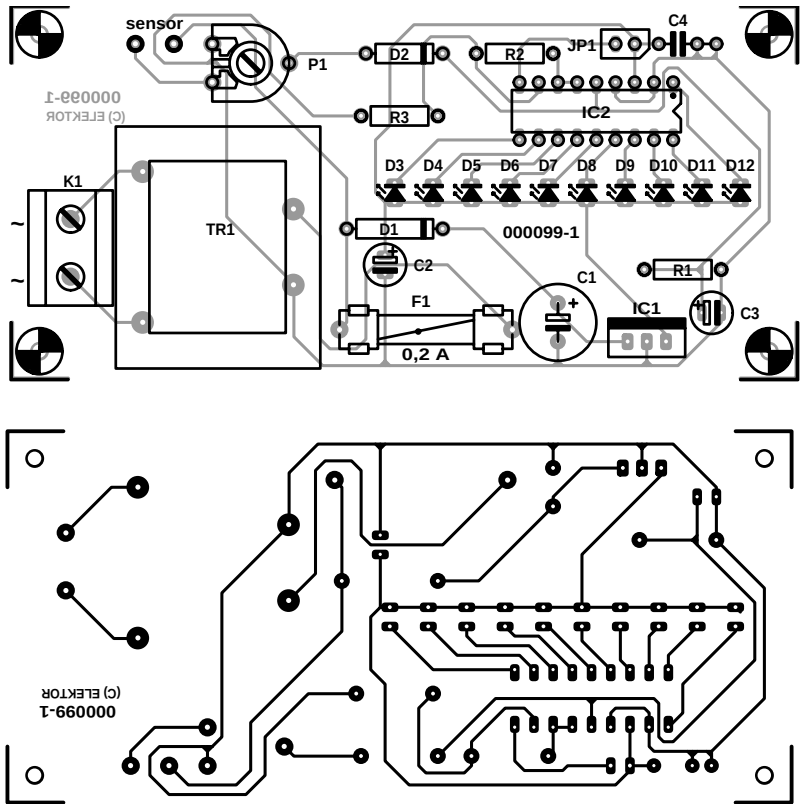


Figure 3. Printed circuit board layout for the moisture tester. There is also room to mount the small mains transformer (PCB not available ready-made).

COMPONENTS LIST

Resistors:
R1 = 100kΩ
R2 = 2kΩ7
R3 = 2kΩ2
P1 = 500kΩ preset H

Capacitors:
C1 = 470µF 16V radial
C2,C3 = 10µF 16V radial
C4 = 100nF

Semiconductors:
D1 = 1N4001
D2 = 1N4148
D3,D4,D5 = LED, green
D6-D9 = LED, yellow
D10,D11,D12 = LED, red
IC1 = LM7805
IC2 = LM3914-N

Miscellaneous:
Tr1 = mains transformer, secondary 6 V 200mA (e.g., Monacor/Monarch VTR-1106)
F1 = fuse, 200 mA, with PCB mount holder
JP1 = short-circuiting jumper
K1 = 2-way PCB terminal block, lead pitch 7.5mm

MIDI on the RS232 Port

music on the serial port!

Design by B. Bouchez

Early 1995, Yamaha introduced a new range of synthesizers with interesting interface capabilities. For the first time, these machines were compatible with three standards: MIDI, Macintosh and PC. To support the PC interface, the MU5, MU10, MU80 synthesizers and the SU10 sampler came with an RS232 serial port, providing connectivity with the PC. This spurred the author into designing an RS232-to-MIDI converter that's supported by a freeware driver from Yamaha.

Initially, these sound generators were designed to work with what Yamaha called 'HOST PORT' supporting software. In fact, only a few sequencers were actually capable of directly supporting this functionality. Fortunately, Yamaha designed a driver that permits the use of a serial port as a standard MIDI interface to almost any sequencer running under Windows.

An interesting spec of the Host Port mode of the MU5 is its ability to retransmit all messages arriving on the MIDI IN connector to the RS232 port. The same ability was available in the other direction, from RS232 to MIDI OUT. In industry-speak, this is called a 'gateway'.

Actually, there are two problems for using an MU or SU sound unit in such a way. First, it is not possible to disconnect the sound generator from the serial port controller. The sound generator always responds to the incoming messages, giving the odd unexpected result. Second, apart the SU10, none of the other synthesizers are manufactured any more.

The interface described in this article allows you to emulate an MU or SU sound unit, as a transparent gateway, using the standard serial port on your PC. If you are using a desktop PC, with a MIDI interface already available, like a MPU or Soundblaster, you can add a new MIDI port for all your applications, particularly under Windows. If you are using a portable computer with no MIDI capabilities, you can access the world of MIDI simply by connecting the interface to a free serial port.

System description

The interface schematic is given in **Figure 1**.

Thanks to a microcontroller, the interface schematic is really simple. Note that a microcontroller is mandatory, since the system needs to store data in a FIFO, due to the different baudrates used on the serial port and the MIDI port.

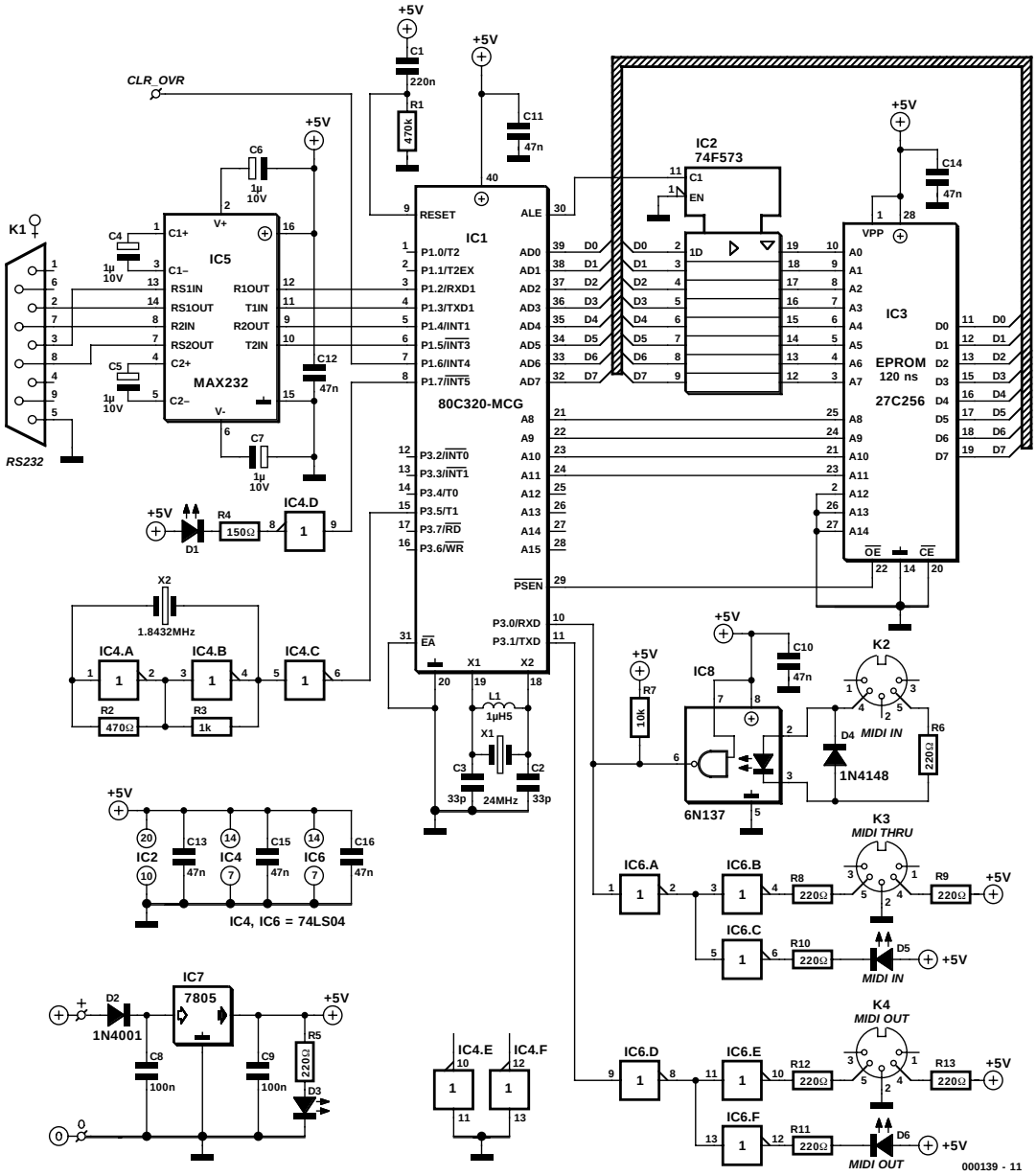


Figure 1. Circuit diagram of the RS232-to-MIDI Converter. At the heart of the circuit you find an 80C320 microcontroller from Dallas Semiconductor.

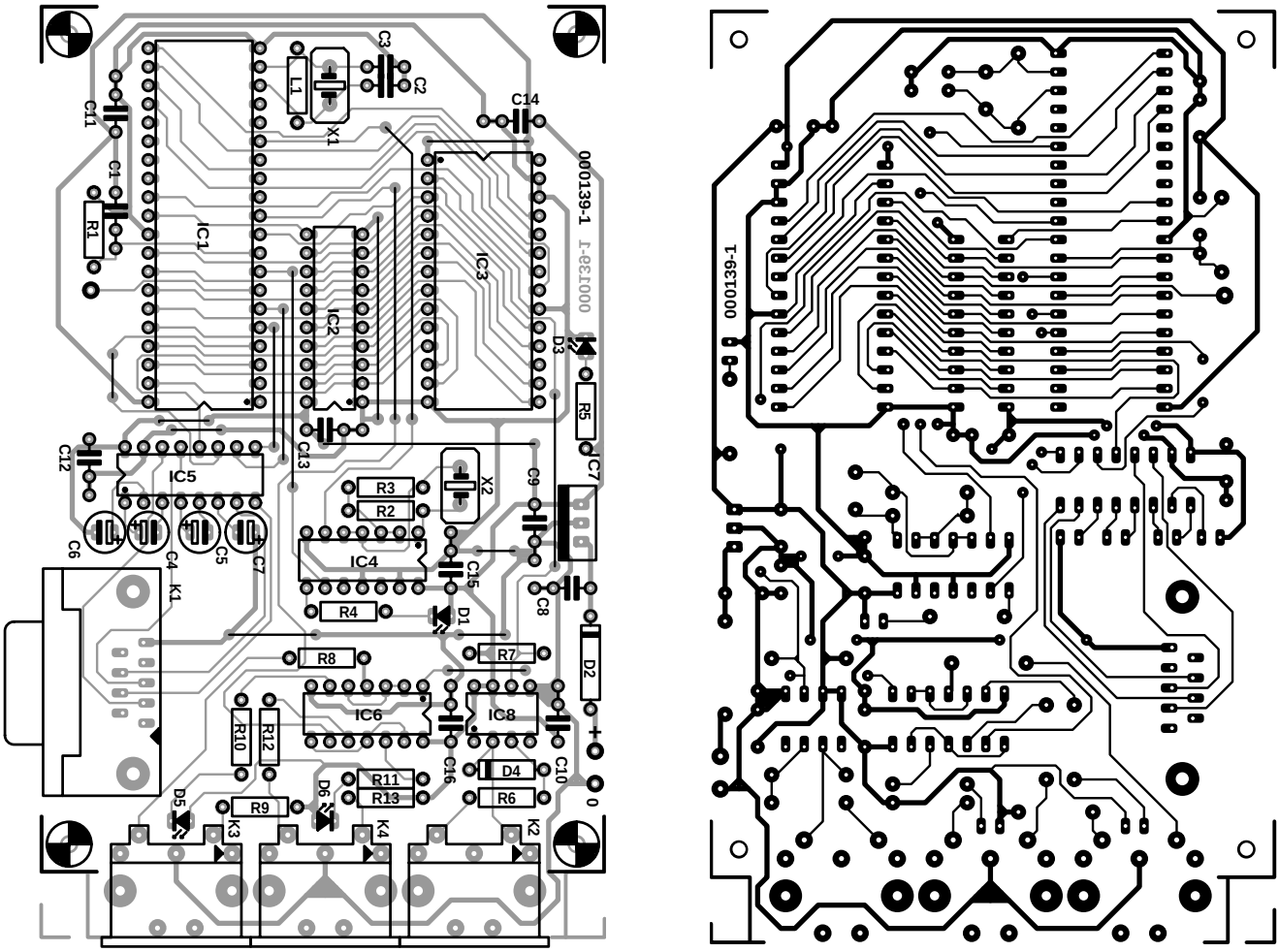


Figure 2. Copper track layout and component mounting plan of the PCB designed for the converter.

The processor is a Dallas type 80C320 microcontroller, which is slightly unusual in that it has two serial ports capable of running at different speeds. Besides, this family of microcontrollers employs a redesigned processor core, running code four times faster than a standard 8032.

Unfortunately, the standard 8032 processor is not compatible with the present application, because it includes only one serial port. The use of a Dallas 80C320 is therefore mandatory.

Here, the C320 does not employ external code memory since the on-chip RAM is sufficient for this application. As always, the address bus is demultiplexed externally by a 573 latch driven by the the processor's ALE signal. You will notice the use of a 27C256 EPROM, with some address lines tied to ground. The application program could easily be stored in a 27C64 EPROM, but these tend to be more expensive then higher capacity models, and they are not easily available in the 120ns speed range, which is mandatory in this case. Please also note that the 573 latch must be an F (fast) TTL device, for the same reasons of speed.

The main system clock is given by a 24 MHz crystal, associated with a small inductor to force it into overtone mode. The MIDI baudrate is derived from the system clock. Since it is not possible to derive the 38,400 baudrate from a 24 MHz clock, an external 1.8432MHz oscillator is used. This oscillator is built around three gates inside IC4. The output of the oscillator drives the T1 input of the processor.

Connected to processor port 1, the MIDI interface is built around IC6 in a classic way. The only difference between the standard MIDI schematic as given in the MIDI specifications is the use of two LEDs as communication indicators. Believe us, these LEDs are really useful in practice, nicely indicating whether a MIDI stream is present or not.

The RS232 interface is connected to the serial port 0 of the processor, via the ubiquitous MAX232 level converter. Thanks to this chip, it is possible to use a single +5 V power supply. The DB9 connector allows the use of any standard serial cable for the link to the PC. Note that the RTS/CTS handshaking signals are used by the interface. A null-modem cable, with only wire links in the connectors, is **not** compatible with this application. It is necessary to use a fully wired-up cable.

The power supply is made around a standard 5 V regulator, powered externally by a small 12 V DC adaptor. Because of the wide range of connectors available on these power supplies, we decided to solder pins only for the supply connections, rather than a complete PCB mount socket which may be impossible to get locally.

The last signal controlled by the processor is just a buffer overflow

COMPONENTS LIST

Resistors:
R1 = 470kΩ
R2 = 470Ω
R3 = 1kΩ
R4 = 150Ω
R5,R6,R8-R13 = 220Ω
R7 = 10kΩ

Capacitors:
C1 = 220nF
C2,C3 = 33pF
C4-C7 = 10µF 10V radial
C8,C9 = 100nF
C10-C16 = 47nF

Semiconductors:
D1,D3,D5,D6 = LED
D2 = 1N4001
D4 = 1N4148
IC1 = 80C320-MCG (Dallas Semiconductor)
IC2 = 74F573
IC3 = EPROM 27C256 100ns (programmed, order code **000139-21**)
IC4,IC6 = 74LS04
IC5 = MAX232 (Maxim Integrated)
IC7 = 7805
IC8 = 6N137 (Atmel-Temic)

Inductors:
L1 = 1µH5

Miscellaneous:
X1 = 24MHz quartz crystal (third overtone)
X2 = 1.843 2MHz quartz crystal
K1 = 9-way sub-D socket (female), PCB mount
K2,K3,K4 = 5-way DIN socket, pins at 180 degrees, PCB mount
PCB, order code **000139-1**
Disk, order code **000139-11** (driver, source code, hex file)

indicator, using the LED D4. If you want to reset this LED by an external command, you may connect a single pushbutton between P1.6 and ground. Note that receiving a SYSTEM RESET MIDI command (0xFF) on the RS232 line also causes the LED to go out.

Building the interface

Building the interface is quite easy, since there is only a handful components to install on the PCB. As usual, we recommend the use of sockets for the ICs, especially for the MIDI current loop inverters and the V24 adapter, since they are directly connected to the external world and may subject to electrostatic discharges when connecting or disconnecting cables.

The only difficulty that could possibly arise when you will build the interface may be encountered around the MIDI connectors. We chose PCB-mounted 5-pin DIN sockets, with a standard shape and PCB footprint. Some models however have slight differences in their housing and pin arrangement. If necessary, 'minor surgery' may be required to install them correctly on the board.

Having installed all components you may power up the board. The only LED that should light is POWER ON. The two other LEDs must remain off. You can verify with an oscilloscope that you have a 1.8432MHz square wave on the T1 input of the processor. A test feature for the internal chip oscillator is provided. On the T2 pin of the processor, you should find a 250 kHz square wave.

If you find the 250 kHz signal, you can be confident that the program is running correctly, and the crystal oscillator runs at 24 MHz. If not, the first thing to check is the value of the inductor near the crystal and check the value. If replacing the inductor does not remedy the fault, you will have to check your board for soldering errors or incorrectly fitted parts.

The interface does not need a lot of current to work. As we said before a small power adaptor (12V/200mA) is sufficient. Due to this, IC7 can manage without a heatsink, since it dissipates only little energy. Unfortunately, some inferior quality adaptors deliver much higher voltages than desired or selected (some even up to 18 V when set to 12 V), resulting in more heat to dissipate by IC7. If you note that the regulator becomes too hot, attach a small aluminium bracket to it (2 to 3 cm^2 is sufficient) to help it to cool.
Your new MIDI interface is now ready for use.

Using and testing the interface

Since the interface is in fact nothing but a gateway, no particular processing is required to send and receive MIDI data. If you are using or creating software that can use the 'Host Port' capability (in the Yamaha jargon), the MIDI datastream just employs the serial port rather than a dedicated MIDI port since there is no information to add or remove. Data are exchanged with the serial interface using the 38,400 bits/s, 8, N, 1 format.

If you want to use Windows (95 or 98) software that does not include the 'Host Port' support, you have to get the Yamaha driver that often goes by the name CBXT3. This driver is available from the download pages at http://www.yamaha.co.uk
This driver is freeware — the only limitation is that it is supplied 'as is', i.e., with no war-

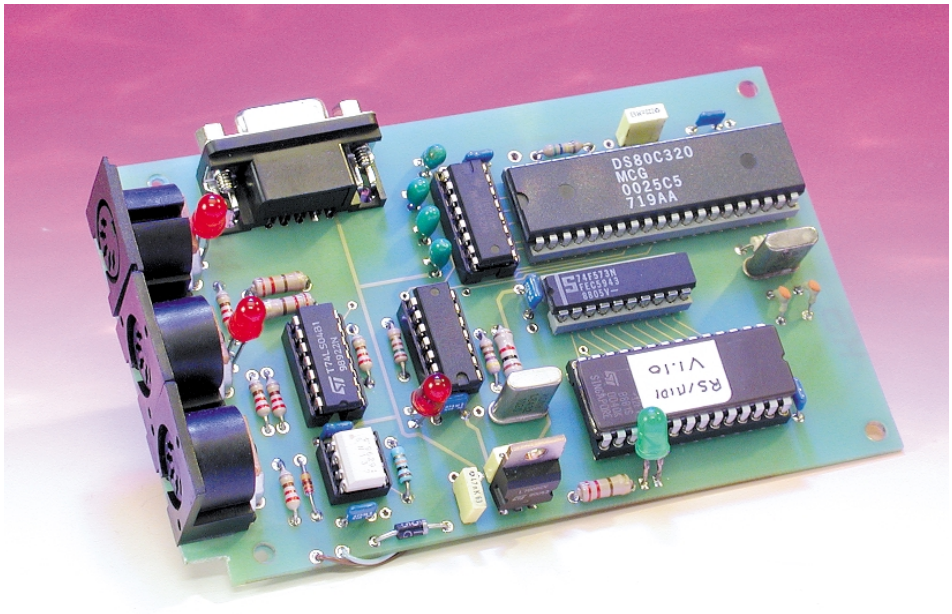


Figure 3. Finished prototype of the RS232-to-MIDI Converter.

ranty or assistance from Yamaha. The driver files are available on a disk with order number **000139-11** through the Publishers' readers Services, and as a ZIP compressed archive from the Free Downloads page, section April 2001, of the *Elektor Electronics* website at www.elektor-electronics.co.uk. The archive includes the source code, hex file and the Yamaha driver, version 1.85.

Under Win95 and Win98 the author tested the CBXT3 driver Version 1.80b2 with full satisfaction. Version 1.85, copied from disk or extracted using Winzip, may be used also. The files will be automatically decompressed into a subdirectory called MIDIDRV.

Launch the Setup program and follow the instructions on the screen. When the software displays a configuration window, choose the serial port you want to use for the converter. When in doubt, don't panic as you will be able to return to this window at all times. Do not check the 'Multiport' box because the present converter does not support this functionality. As a matter of course, you should not check the 'Driver Disable' box because then the driver will not work. This box, by the way, simply serves to free up serial port 'resources' which may be necessary for other applications in heavily loaded PC systems.

After the last window, the software will ask you if you want to reboot the system. Do so to enable the new driver.

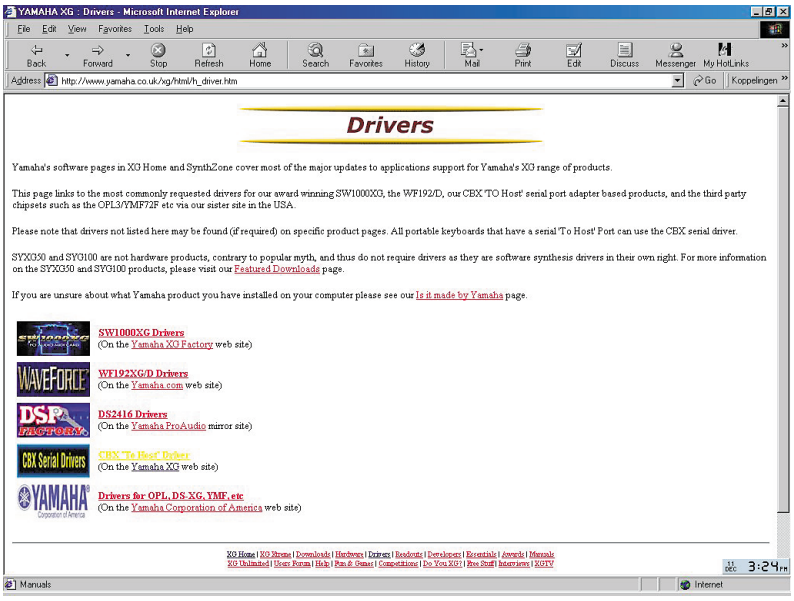
Once the PC is up and running again, you should be able to find a new icon called Yamaha CBX Driver in Control Panels. Clicking on this icon will take you to the configuration window mentioned above.

On some PCs, for reasons unknown to the author, the Yamaha driver will be difficult to install. To be precise: although it will be fully installed, the driver will not be active, an the Yamaha CBX icon will not appear in Control Panels. If that happens, do not panic, just follow the instructions below, literally.

– Install the driver as described above.
– Locate the file CBXT3.DRV installed by the Yamaha program (in principle the file should be in C:\Windows\System).
– Copy the file into a temporary subdirectory created for the occasion.
– Create a text file and name it `OEMSETUP.INF` in this same directory. Include these lies in the `.INF` file (stick to syntax and order):

```
[disks]
1=. ,"YAMAHA CBX Driver for
Windows95",disk1

[Installable.Drivers]
CBXT3 = 1:CBXT3.drv,"MIDI","YAMAHA CBX
Driver for Windows95",
```



– Open the Windows Control Panel
– Launch 'Add New Hardware'
– Click on 'Next' until the system wants to start an automatic search for new hardware. In Windows 98, you'll have to allow the system to do a search for Plug and Play Devices. Next, choose 'select hardware from a list'.
– Select the category 'sound, video and game controllers'.
– Click on 'Have Disk'.
– Browse until you find the files `CTBXT3.INF` and `OEMSETUP.INF`. The `.INF` file should appear automatically in the window.
– Respond 'OK' to the questions that follow, then click on 'Finish'.
– The software should no appear in the Control Panel window as described above. If so, you may restart your PC (if it has not already prompted you to do so).

Normally, at this point, you will have a new MIDI interface listed in the list of available devices under Windows. To test your interface, you can use any sequencer running under Windows, that permits interface selection (the majority of sequencers like Cubase, MIDI Workshop, etc., have this functionality). If you do not know how to set up your sequencer for other MIDI hardware, you can simply use the Media Player, after configuring the driver as the standard MIDI port for Windows. To do so, open the Control Panel and start the Multimedia control. Then select the MIDI folder and pick 'Yamaha serial driver' from the driver list.

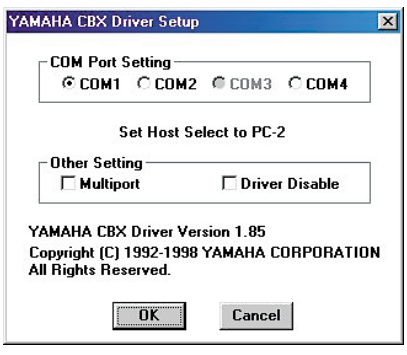
Note that it is possible to adapt the driver behaviour (in some respects), especially for SYSEX messages. This functionality is not available for all drivers, but if you experience some problems with System Exclusive messages, take a moment to look at text files to know if adaptations should be necessary.

If you notice that your serial port seems not to work correctly for some other Windows applications (it happens mostly with some portable computers), you just need to open the 'YAMAHA CBX Driver' icon in the Control Panel, and then check the 'Driver Disable' box.

A final interesting point to note: the Yamaha driver frees the serial port when there is no multimedia application using it. This port is then available for other applications when MIDI is not useful.

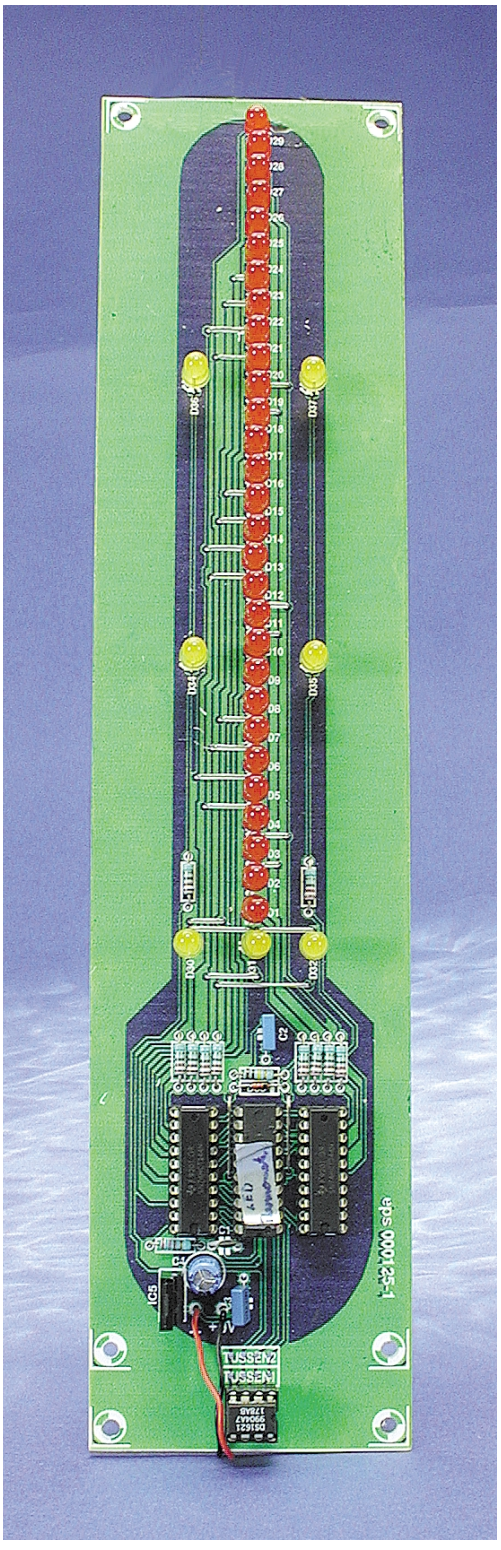
And now, have fun with your new interface…

(000139-1)



LED Thermometer

Dallas sensor IC and COP8 microcontroller



By F. Wohlrabe

The project presented here provides a novel form of temperature display. Its appearance makes it an ideal add-on to the 'Rhine Tower Clock' and will certainly attract attention.

At the heart of the circuit is a National Semiconductor COP8782 microcontroller, as has already been used in *Elektor Electronics* projects like the Running Text Display (PC Topics Supplement, February 2000). The microcontroller reads temperature data from a sensor IC made by Dallas Semiconductor, which has also been seen before in *Elektor Electronics* projects ('Temperature Measurements with the DS1621' and 'DS1621 Datasheet', both in the March 2000 issue). A DS1621 programmer was published in the February 2001 issue.

The microcontroller also has the job of displaying the measured value on 32 multiplexed LEDs.

Temperature sensor

Since the principles of operation of the DS1621 have already been described in Elektor Electronics, we can concentrate here on the important features. The sensor incorporates a complete temperature measurement system, including a thermostat function. All data are communicated over an I^2C bus. The sensor operates from –55 °C to +125 °C, with a resolution of 0.5 °C. In this project, the display is limited to the range –29.5 to +59.5 °C. The thermostat function of the DS1621 (output T_{out}) is, as can be seen in the circuit diagram in **Figure 1**, not used in this application.

Technical Data

Microprocessor controlled	
Temperature range	–29.5 to +59.5 °C
Resolution	0.5 °C
Error over temperature range	±0.5 °C
Supply voltage	9 V
Supply current	approx. 100 mA

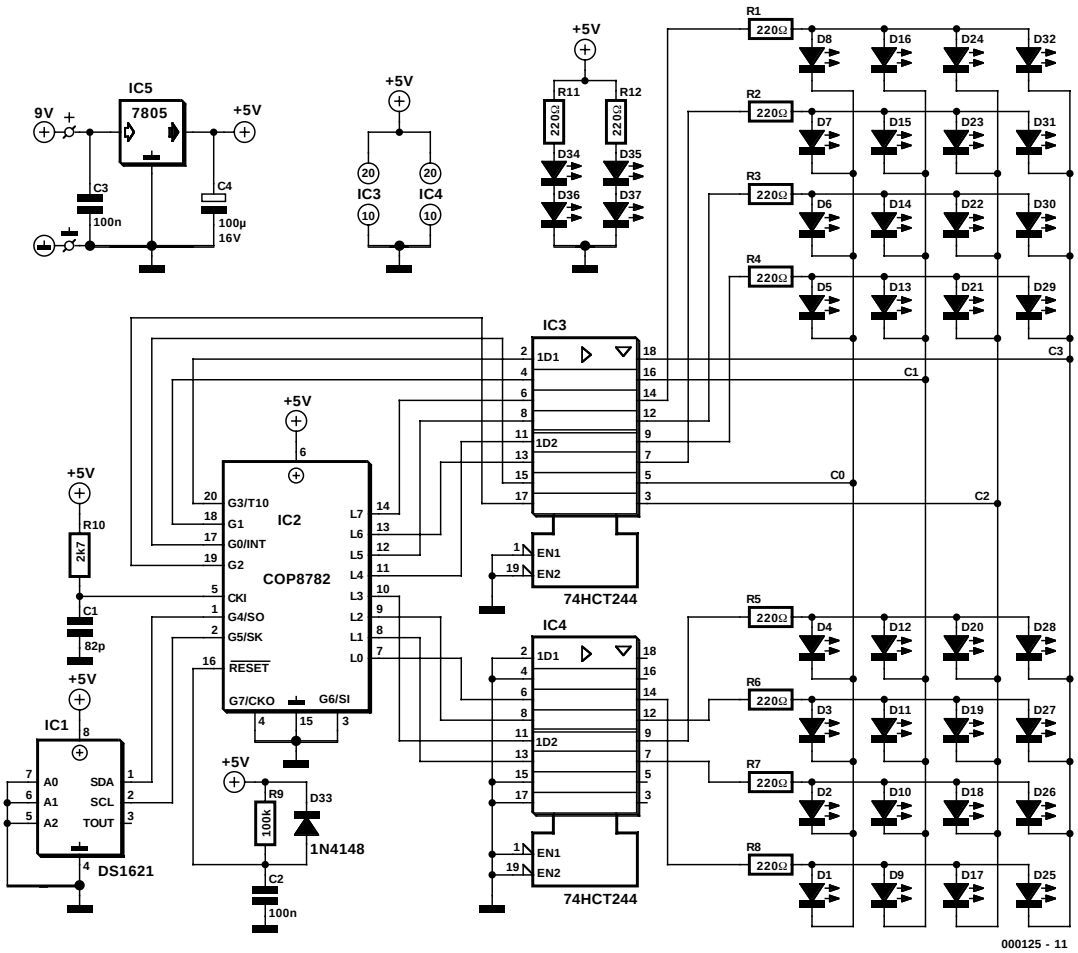


Figure 1. Circuit diagram of the LED thermometer: 32 LEDs driven from 12 I/O ports of a COP microcontroller.

In theory up to eight sensors can be connected to the same bus, each device being assigned a different address using the address inputs A0, A1 and A2. For this LED thermometer we do not need this feature, so all the address inputs are tied to ground and the sensor is thereby assigned address zero.

The temperature is read out in a 9-bit format, arranged as follows:

Temp. in °C	Output data binary	hex
+125	01111101 00000000	7B00
+25	00011001 00000000	1900
+0,5	00000000 10000000	0080
0	00000000 00000000	0000
- 0,5	11111111 10000000	FF80
- 25	11100111 00000000	E700
- 55	11001001 00000000	C900

Byte 1 contains the integer part of the temperature, with a negative value indicated by the MSB being set. The MSB of byte 2 being set indicates an extra +0.5 °C. The remaining bits of byte 2 are always zero.

According to the data sheet, each measurement takes at most 1 s. In order to read the temperature value out of the sensor, the microcontroller issues the following commands:

Microcontroller commands

Once only, to configure sensor:

1001 0000 4 bit control code plus device address, write mode
1010 1100 write to configuration register
0000 0010 continuous measurements

Before each measurement:

1001 0000 4 bit control code plus device address, write mode
1110 1110 'start measurement' command

After a 1 s delay:

1001 0000 4 bit control code plus device address, write mode
1010 1010 'read temperature' command
1001 0001 4 bit control code plus device address, read mode

Read in data as shown in data sheet timing diagram

The typical error in the reading from the DS1621 is shown in **Figure 2**. In the range

of interest to us it is at most 0.5 °C.

Sensor and COP

The COP8782 microcontroller from National Semiconductor has many features that make it particularly well suited to this project:
– 4096 x 8 (OTP) EPROM
– 128 bytes RAM
– 1 µs cycle time at 10 MHz
– 16 bit timer with automatic reload, external event counter and capture functions
– 16 I/O pins, 14 of which can be individually programmed as inputs or outputs
– Selectable tri-state, push-pull or pull-up I/O pin configuration
– Microwire interface
– Timer, software, and external interrupt sources, with programmable polarity

The COP8AC7 is a broadly pin- and functionally-compatible successor device to the COP8782. The starter kit unfortunately does not offer real-time emulation, but it does allow programming of OTP devices and offers a comprehensive introduction to this economical and technically interesting family of microcontrollers.

For more demanding projects, where the real-time performance of the microcontroller must be verified, the purchase of an emulator is essential or else the project will degenerate into continuous experimentation.

Driver and display

In order to minimise the number of connections for a large number of display elements, the 32 LEDs are driven in a multiplexed fashion. Four LEDs (D34-D37) are permanently lit (reminiscent of the Rhine Tower Clock) and indicate the tens of degrees, making the display easier to read. Since pins G6 and G7 can only be used as inputs, the other twelve ports are all used as outputs. With the use of clever software we can drive 32 LEDs through these 12 ports. The LED matrix is divided into four groups (A-D), each consisting of eight LEDs. These groups are driven from port pins G0-G3 in sequence, so that at most eight LEDs are ever lit simultaneously. The LEDs are turned on by setting port pins L0-L7 high. With a low level on one of G0-G3 the appropriate group of LEDs is enabled. Although only at most eight LEDs are on at any one time, the multiplexing happens so fast that the visual appearance is that up to 32 LEDs are lit.

The overall temperature range from –29.5 °C to +59.5 °C is divided into three ranges (-29 to 1 °C, 0 to+29 °C and 30 to 59 °C), as indicated by the three range LEDs (D30, D31 and D32). The odd 0.5 °C is indicated by the range LED flashing at about 1 Hz. So, for example, at 20.5 °C the middle range LED D31 flashes, indicating a temperature between 0 °C and 29 °C.

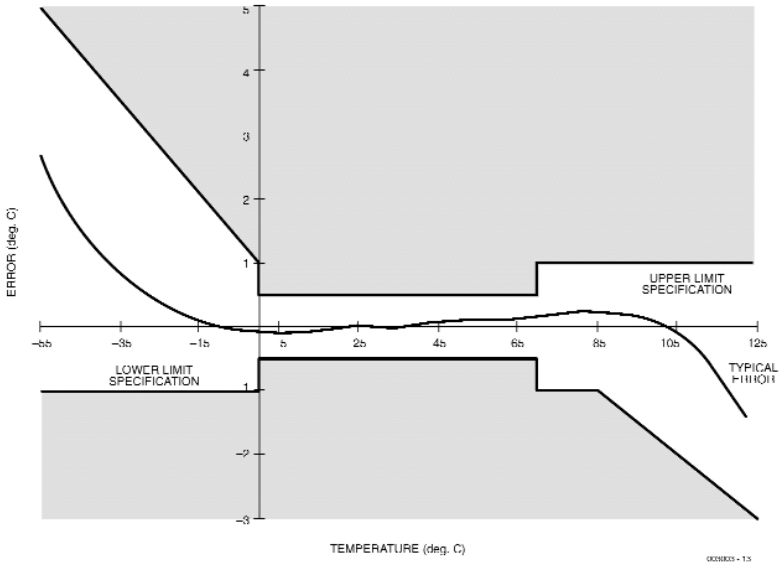
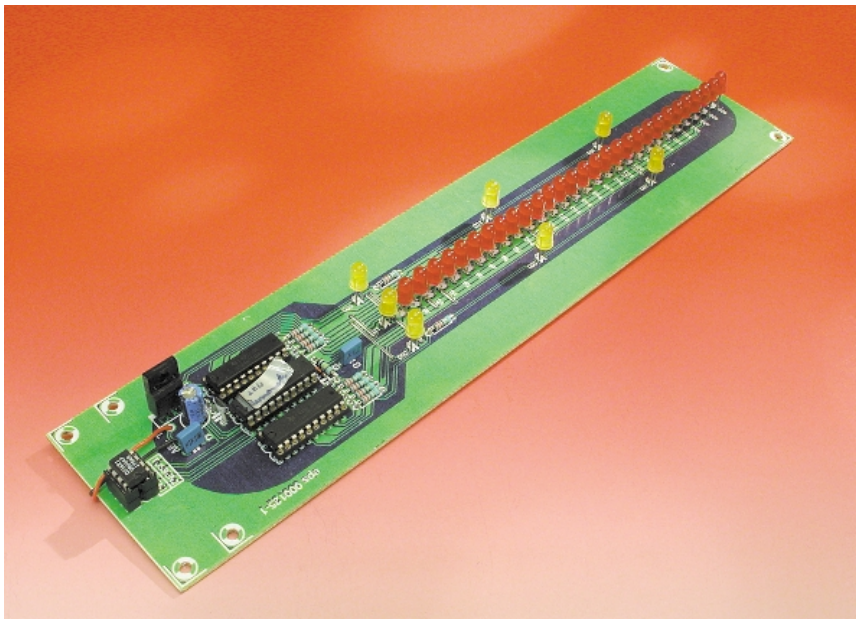


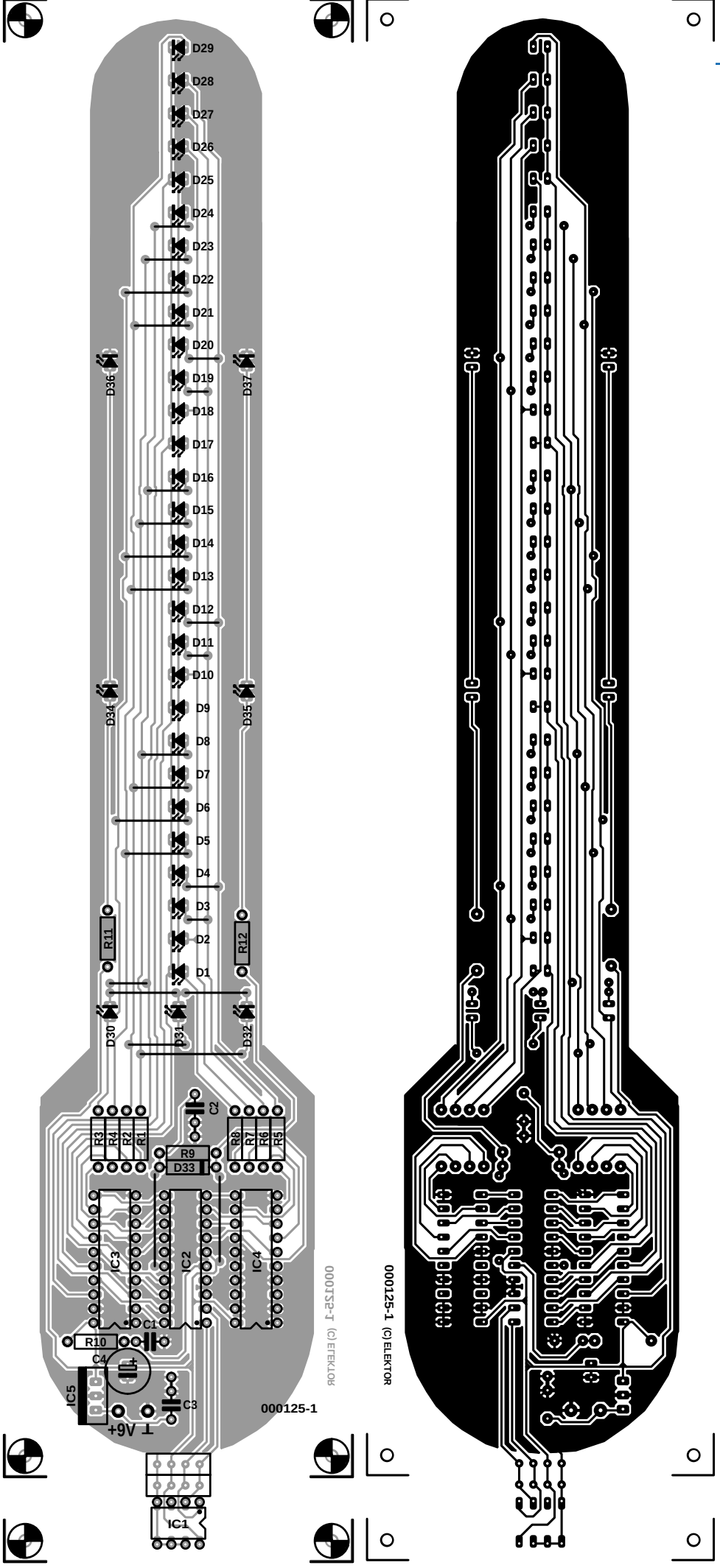
Figure 2. Typical error characteristic of the DS1621.

Construction

Since the outputs of the microcontroller cannot provide enough current to drive the high efficiency LEDs directly, the twelve drive signals are buffered using a 74HCT244 driver. Because of the high multiplex ratio and the drive impedance of the ports, current-limiting resistors of only 220 Ω can be used. Maximum brightness (although with a current consumption of 180 mA!) can be obtained by dispensing with current-limiting resistors altogether, substituting wire links. An ordinary (unstabilised) mains adaptor, with an output voltage of at least 9 V, can be used as a power supply.

Assembling the circuit board (**Figure 3**) should present no difficulties. The sensor device can be separated from the rest of the circuit, allowing remote mounting if for example it is desired to measure the temperature outside. The I^2C bus and power supply can be connected using a cable at most 3 m long.





Software

Circuit board layout, source and object code can be downloaded from the Free Downloads page of the Elektor Electronics website. Source and object code are also available from Readers' Services on disk number 000125-11, and a ready-programmed microcontroller (000125-41) is also available.

For a tidy appearance a template should be used when bending the resistors and wire links for fitting to the circuit board. When soldering, use a spacer to ensure that the LEDs are accurately lined up. Check the polarity of the LEDs: an incorrectly fitted LED may be hard to track down.

(000125-1)

COMPONENTS LIST

Resistors:
R1-R8,R11,R12 = 220Ω
R9 = 100kΩ
R10 = 2kΩ7

Capacitors:
C1 = 82pF
C2,C3 = 100nF
C4 = 100µF 16V radial

Semiconductors:
D1-D29 = LED, high efficiency, red
D30,D31,D32 = LED, high efficiency, yellow
D33 = 1N4148
IC1 = DS1621 (Dallas Semiconductor) (Farnell # 760-704)
IC2 = COP8782-CN, programmed, order code **000125-41**
IC3,IC4 = 74HCT244
IC5 = 7805

Miscellaneous:
PC1,PC2 = solder pins
PCB, order code **000125-1**
Disk, COP8 source code file, order code **000125-11**

Figure 3. Circuit board, shaped similarly to that for the Rhine Tower Clock.

Darkroom Timer

simple… with a PIC!

Design by S. Müller

stefan-mueller@onlinehome.de

A darkroom timer is not such a complicated device: you can easily build one yourself using a microcontroller.

Although previous darkroom timer designs have contained vast expanses of TTL or CMOS devices, that has all radically changed now, as can be seen in **Figure 1**. The circuit is controlled by a microcontroller from the PIC range. This drives an opto-triac (IC3) which switches the mains voltage from the input connector K1 to the output connector K2, and thereby turns on the enlarger lamp. The desired exposure time can be set to a resolution of 0.1 s using three thumbwheel switches. The setting is read directly from the switches in BCD format: their four outputs A-D represent the four bits of the BCD digit.

Since the PIC16F84 device chosen is only equipped with 13 I/O pins, the corresponding bits of the three switches are connected together in the so-called wired-OR configuration (D5/D9/D13, D6/D10/D14 and so on) and then connected to the controller. The four resistors pull the port inputs low when not pulled high through a switch. This ensures that clean

logic levels are present at the inputs to the PIC.

The three thumbwheel switches are driven in sequence via ports RB0, RB1 and RB2, and the switch positions are read consecutively by the control software. A further port pin (RB4) is used for the 'Start' switch. On pressing switch S4 the darkroom timer starts, and after the preset time the software switches the output RB3 low again. During this period, LED D17 lights. The enlarger lamp can also be turned on using switch S5, overriding the timer.

A standard power supply, consisting of a small 1.5 VA mains transformer, a full-wave bridge rectifier and a 7805 fixed voltage regulator, provides the supply for the PIC. The microcontroller is clocked at 4 MHz.

Software in JAL

The software was developed using *Just Another Language (JAL)*, which is freeware. DOS/Windows and Linux versions can be downloaded from the site http://come.to/jal. JAL is very easy to use and especially suited for those new to programming PIC16C/F84 and certain SX microcontrollers. It is handy for small projects where optimal use of program memory is less of a concern.

The structure of the software is very simple. As soon as power is applied to the darkroom timer, the

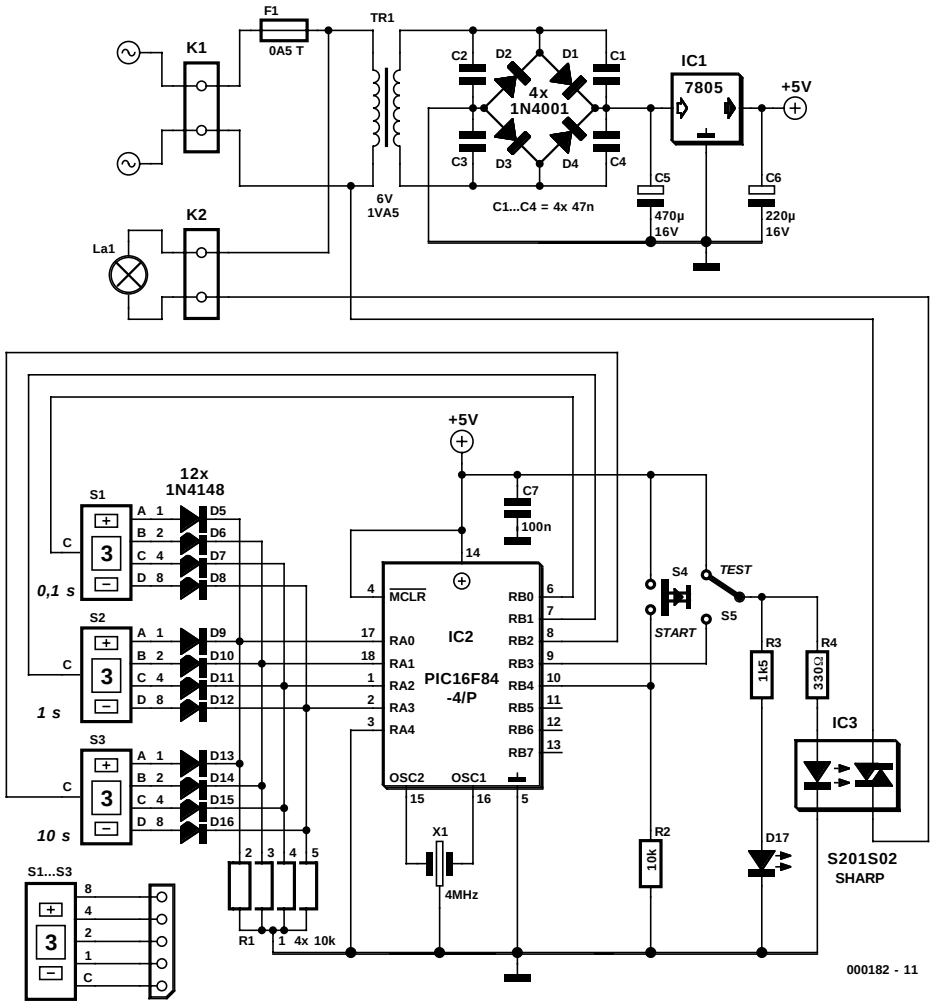


Figure 1. The darkroom timer is controlled by a PIC microcontroller.

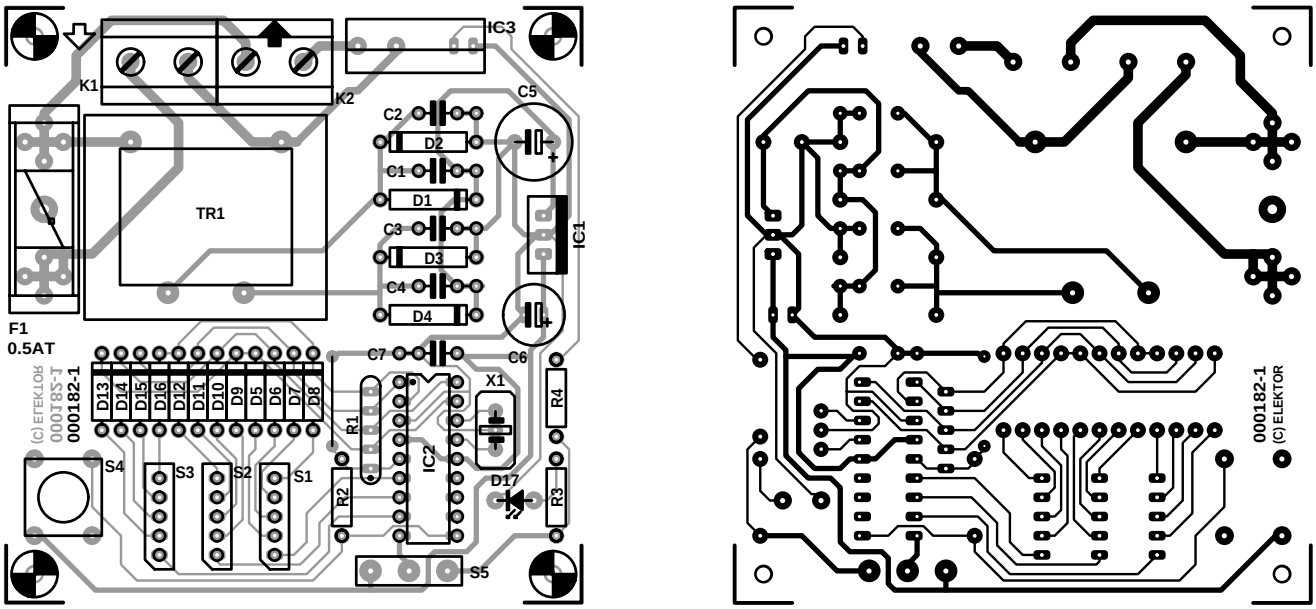


Figure 2. The circuit board carries all the components.

software starts running in a loop, continuously reading the switch settings. As soon as a falling edge is detected on RB4 (i.e., when the switch is released), the loop is exited, and the lamp is turned on. A delay is executed, and then the lamp is turned off again. Then the software returns to the beginning. You may encounter one peculiarity if you study the source code: the value read from the switch in the 10 s decade (S3) is doubled, and then the delay executed in the 5 s loop. This is because the JAL *delay* command only allows a maximum time period of 5 s.

You can obtain the JAL source code, as well as the object code, from the *Elektor Electronics* download site via
http://www.elektor-electronics.co.uk.

For non-Internet users, it is also available on diskette, order number **000182-11**. It is easy to modify the code or to add extra functions.

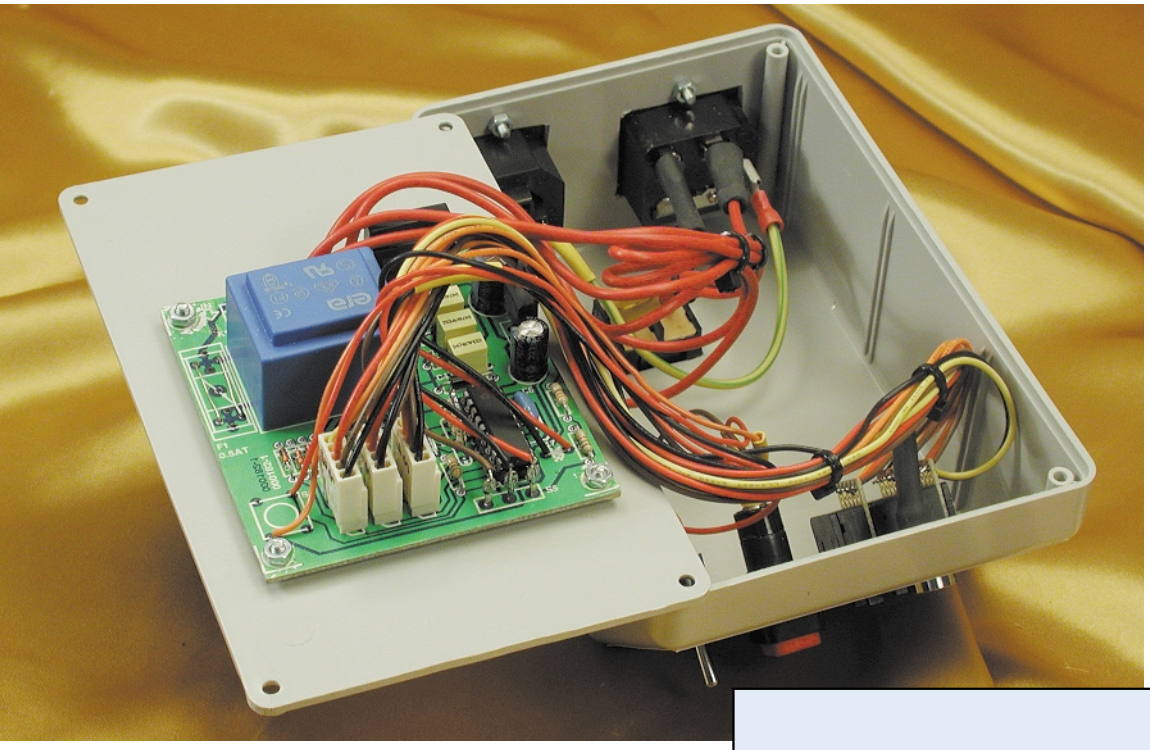
COMPONENTS LIST

Resistors:
R1 = 4 x◊10kW SIL array
R2 = 10kW
R3 = 1kW5
R4 = 330W

Capacitors:
C1-C4 = 47nF
C5 = 470mF 16V radial
C6 = 220mF 16V radial
C7 = 100nF

Semiconductors:
D1-D4 = 1N4001
D5-D16 = 1N4148
D17 = LED, red, high efficiency
IC1 = 7805
IC2 = PIC16F84-04/P (order code 000182-41)
IC3 = S201S01 (Sharp) (Conrad # 16 81 65)

Miscellaneous:
K1,K2 = 2-way PCB terminal block, 7.5 mm lead pitch
F1 = fuse, 0.5 AT (time lag), with PCB-mount holder
TR1 = Mains transformer 6 V, 1.5 VA (Conrad Electronics # 50 60 44)
S1,S2,S3 = BCD-encoded thumbwheel switch type PICO (Conrad 70 10 84)
2 spacers (Conrad Electronics # 70 11 06)
1 pair end cheeks (Conrad Electronics #70 11 33)
X1 = 4 MHz resonator (Conrad Electronics # 50 31 69-55)
S4 = pushbutton, 1 make contact
S5 = SPDT changeover switch
Case: Teko 362 (Conrad Electronics # 52 39 68)
Disk, source and object code, order code **000182-11**



If you do not wish to be bothered with programming a PIC yourself, ready-programmed devices are also available under order code **000182-41**.

The circuit board

Unfortunately, the printed circuit board for the darkroom timer, is not available ready-made. Populating the board should present no surprises as long as the specified components are used and correct polarities are observed. This goes not just for semiconductors and electrolytics: the resistor array and the thumbwheel switches must also be fitted with the correct orientation. Don't forget the single wire link, next to resistor array R1. The PIC microcontroller is worth a socket, as well as visual and supply voltage checks before fitting it and applying power.

(000182-1)

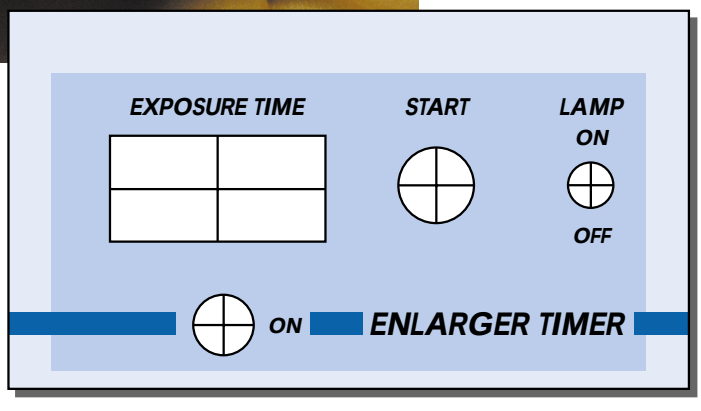


Figure 3. Suggested front panel layout for the darkroom timer.

Crescendo *Millennium Edition*

remake of a famous amplifier

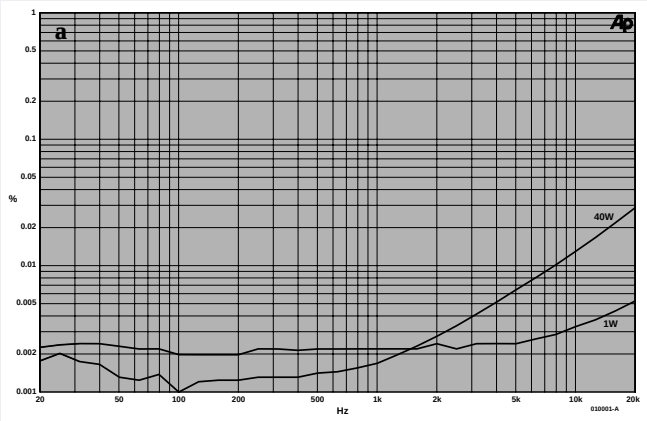
Design by T. Giesberts

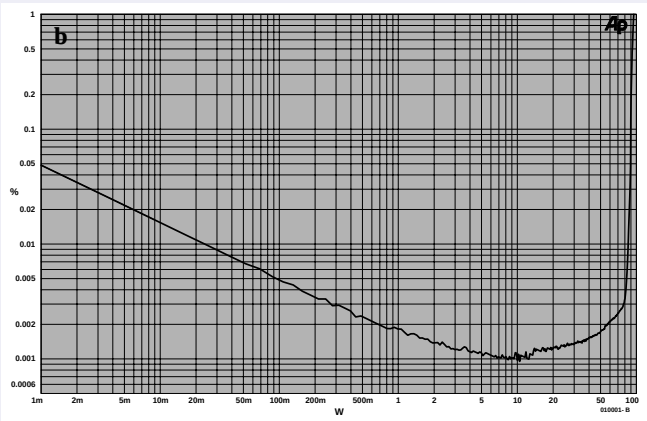
In early 1984, *Elektor* magazine beat the competition hands-down by publishing the design of a then-revolutionary MOSFET amplifier. Even now, this amplifier enjoys an enthusiastic following. In response to many requests, we have brought the design up to date and given some attention to improved reliability and operating safety. The output power is 90 watts into 8 ohms or 135 watts into 4 ohms, which should leave little to be desired for most users.

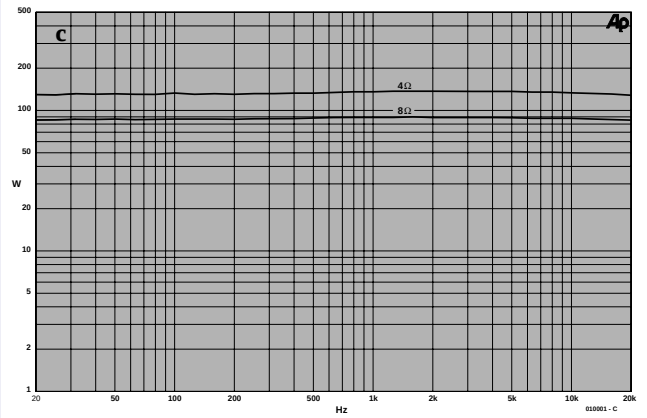
Measured results

(power supply as shown in Figure 3; quiescent current 200 mA)

– input sensitivity:		1 V_{rms}	
– input impedance:		45 kΩ	
– sine-wave power (0.1 % THD):		90 W/8 Ω, 137 W/4 Ω	
– power bandwidth (80 W/8 Ω):		1.5 Hz – 300 kHz	
– slew rate:		60 V/µs (rise time = 1µs)	
– signal/noise ratio:		104 dB (A–weighted)	
(with respect to 1 W/8 Ω)		96 dB (BW = 22 kHz, linear)	
– harmonic distortion		at 8 Ω:	at 4 Ω:
(bandwidth 80 kHz):	at 1 kHz:	0.002 % (1 W)	0.0026 % (1 W)
		0.0017 (40 W)	0.004 % (80 W)
	at 20 kHz:	0.028 % (40 W)	0.04 % (80 W)
– intermodulation distortion:		0.0017 % (1 W)	0.003 % (1 W)
(50 Hz : 7 kHz = 4 : 1)		0.004 % (40 W)	0.007 % (80 W)
– dynamic IM distortion:		0.0026 % (1 W)	0.003 % (1 W)
(3.15 kHz square wave with 15 kHz sine wave)		0.0014 % (40 W)	0.0023 % (80 W)
– damping factor (at 8 Ω):		460 (1 kHz)	
		330 (20 kHz)	
– open–loop parameters:			
gain:		4,000	
bandwidth:		25 kHz	
output impedance:		0.5 Ω	
protection:			
DC:		+ 4.7 V / – 4.3 V	
overload (0 V out):		+ 5.8 A / –5.4 A	
switch-on delay:		8 to 10 s	
bias compensation:		± 4.5 µA	







From the number of zeros after the decimal point, you can see in a single glance that this is an exemplary set of results. You will not often come across a better set of figures. The distortion is very low, the damping factor is very good and the slew rate can even be said to be remarkably good.

As you may expect, we have also measured a number of curves using the Audio Precision analyser in order to complement the performance figures, which always have a somewhat 'dry' taste.

Figure A shows the harmonic distortion (THD + N) over the range of 20 Hz to 20 kHz with an 8-Ω load, using a measurement bandwidth of 80 kHz. At 1 W the increase in the distortion level at 20 kHz is minimal, but at the 50% power level (40 W is equivalent to 70% of the maximum output amplitude) the effect of the non-linear input capacitance of the MOSFETs can be recognised.

Figure B shows the distortion of a 1-kHz signal into an 8-Ω load as a function of the output level in watts, measured with a bandwidth of 22 kHz. The behaviour of the amplifier is more readily visible with this narrower measurement bandwidth. Up to 10 W, the THD + N is predominantly due to supply ripple and noise. A slight increase in the distortion can be seen above 10 W, but a level of 0.1% is reached only at 90 W.

Figure C shows the maximum output power into 4-Ω and 8-Ω loads at a distortion level of 0.1% for frequencies between 20 Hz and 20 kHz (80 kHz measurement bandwidth). Both of these curves can be said to be practically straight.

Finally, Figure D shows the results of a Fourier analysis of a 1-kHz signal (1 W into 8 Ω) with the fundamental suppressed. At this power level, the THD is clearly lower than the supply ripple, whose harmonics lie below –100 dB. The 2nd and 3rd harmonics lie at negligibly low levels (–118 dB and –115 dB, respectively).

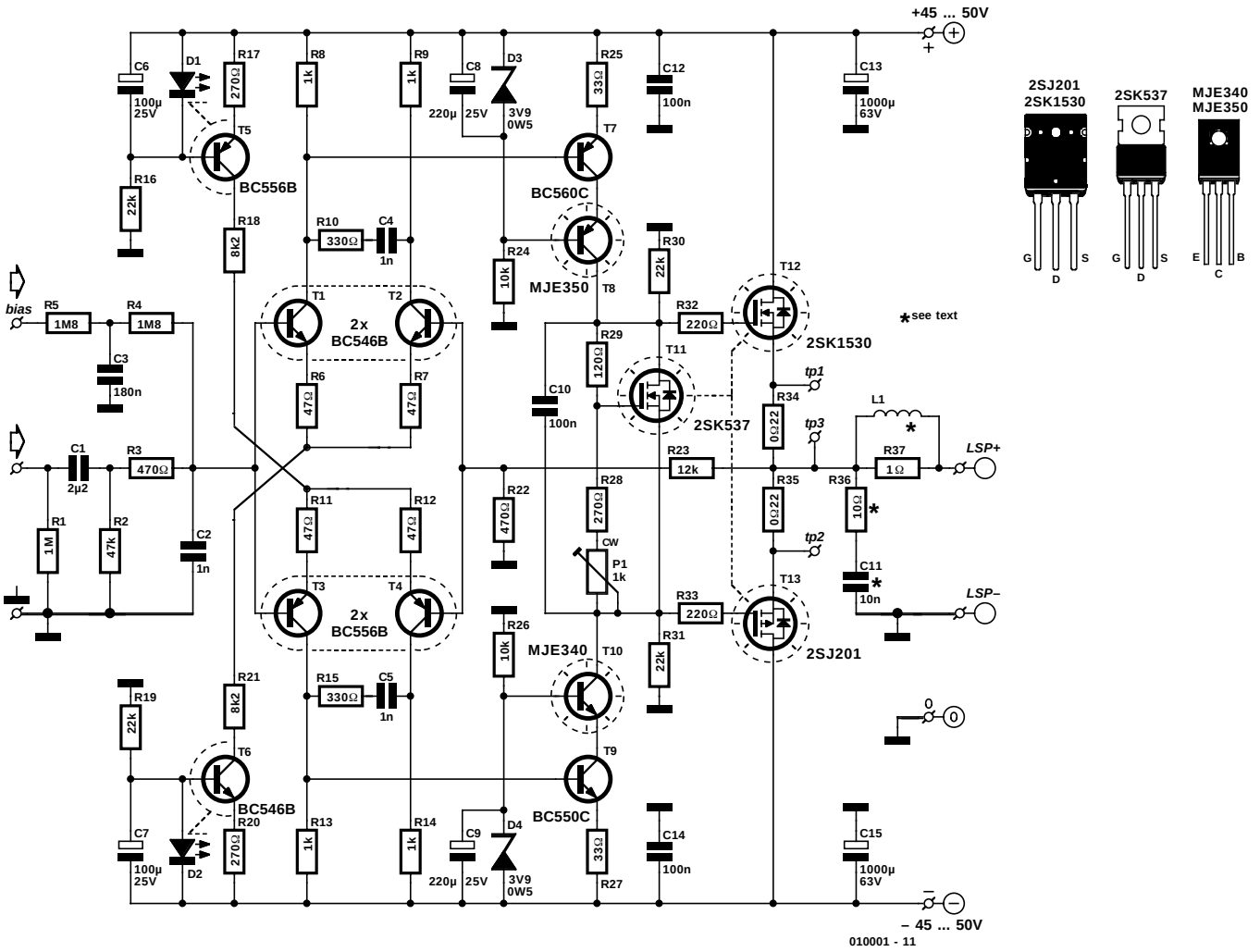


Figure 1. In the schematic diagram, the changes from the original version are hardly noticeable at first glance.

There are surely not very many circuit designs that continue to enjoy such a high level of interest more than ten years after their original appearance, as does the Crescendo power amplifier from 1984. In part, this is due to its completely symmetrical design, which was in fact an unusual feature at that time, but unquestionably it is mainly due to the use of power MOSFETs in the output stage. A lot of people happen to be fervent fans of these devices. Even people who swear by valve amplifiers and are allergic to anything with 'semiconductor' in its name often have a weakness for MOSFETs, and are thus prepared to make an exception for them. Sadly enough, most of the problems with the amplifier in question had to do with the MOSFETs. The original types have long since gone obsolete and become unavailable, and suitable replacements are hard to find. However, there were also other difficulties. The stability of the amplifier sometimes gave cause for concern, and users considered the absence of protective circuitry to be a major weakness.

Consequently, in honour of our anniversary, we decided to take another look at the original design. Our objective was to update the design of the amplifier in a way that would eliminate the sources of criticism without sacrificing the good characteristics of the original design. This objective has been quite successfully achieved. In addition, we were able to obtain such a generous level of output power using a new pair of MOSFETs that it is not necessary to split the new Crescendo into 'light' and 'heavy' versions.

The same concept

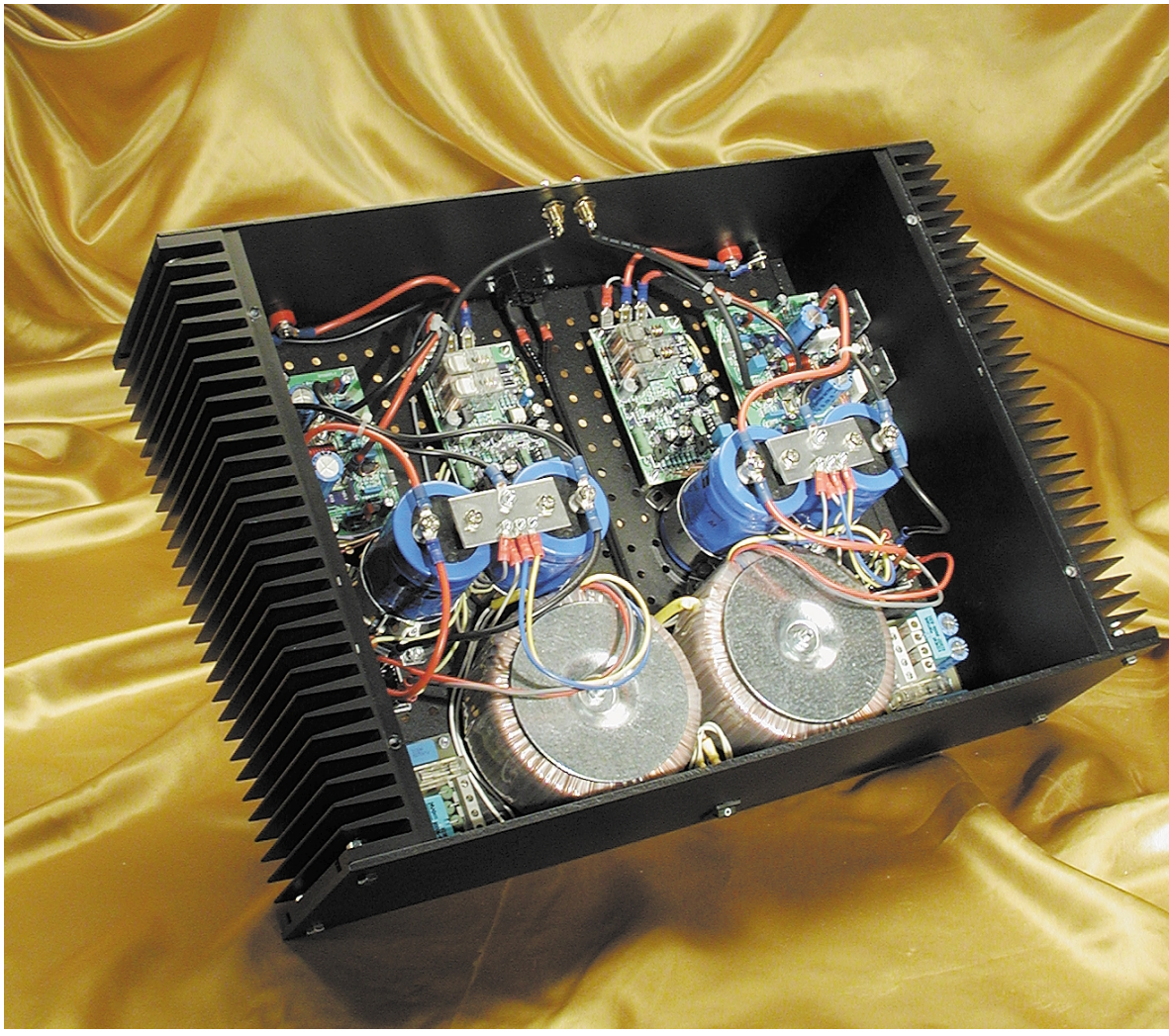
Since we have intentionally tried to change the old amplifier design as little as possible, the differences between the schematic diagrams of the old and new versions are minimal. The design still consists of an input stage with dual differential amplifiers and current sources, a cascode driver stage and a MOSFET output stage. That may have been a rather sophisticated design in 1984, but nowadays it would more likely be described as a 'minimal design'. There's nothing wrong with this, by the way, since attempting to keep the signal path as short as possible is certainly not a mistaken endeavour in an amplifier design — but we don't need to dwell on this point.

Since the basic concept of the original design has been retained, anyone who compares the schematic diagram of the new version (**see Figure 1**) with that of the old version (May 1984) will first have to try to find the differences. Of course, there

are indeed differences, and it seems like a good idea to list the most important changes before diving into a detailed description of the schematic diagram.
The most evident change is naturally the new pair of MOSFETs in the output stage. The Toshiba 2SK1530 and 2SJ201 are readily available, and furthermore they can dissipate so much more power than the original devices that we were able to boost the output power of the old 'Mini-Crescendo' by a factor of nearly two (90 W into 8 Ω in place of 50 W) using only a single pair of transistors.
As a consequence of the increased power level, the bias currents of the various stages must be modified and different transistors must be used in the cascode stage, as will be seen later on.
The next change is the addition of the networks R10/C4, R15/C5 and R30/R31, which represent the results of measures that have been taken to optimise the stability of the amplifier. A very important final item is that the amplifier has been provided with reliable protection circuitry and automatic offset compensation, by means of an extra printed circuit board.
This pretty well covers the most important changes.

Schematic details

Now that we've seen the global picture, it's time to take a more detailed look at the circuit diagram. Let's start at the beginning, which is of course the input stage.
The design of the input filter is more or less standard. R2 (with R1 in parallel) determines the input impedance, and in combination with C1 it forms a high-pass filter that blocks frequencies below around 1.5 Hz. C1 is also needed to isolate the DC bias of the input stage. The combination of R3 and C2 forms a low-pass filter that is dimensioned for a frequency of more than 300 kHz. This helps prevent TIM (transient intermodulation) distortion and eliminates possible RF interference.
The dual differential amplifier (T1–T4) has been designed to work with a bias current that is approximately three times a great as that of the original design, on account of the increased output power. The current sources that regulate this setting, T5 and T6, now use LEDs as references (D1 and D2), since this results in less noise than using Zener diodes. In the interest of the thermal stability of the DC setting, D1/T5 and D2/T6 are thermally coupled, as are the transistor pairs T1/T2 and T3/T4.
The bias currents of the cascode stages T7/T8 and T9/T10 are also significantly greater than in the original design. Since this would be a bit too much for the transistor types used for T8 and T10 in the old version, they have been replaced by the somewhat more robust types MJE340 and MJE350.
Now we come to the output stage. In contrast to the MOSFETs used in the old version, the 2SK1530 and 2SJ201 devices used here have a positive temperature coefficient. This means that with a constant gate-source voltage, the drain current increases with increasing temperature. This made it necessary to use a different design for the quiescent- current circuit. Here the MOSFET T11, which is mounted on the same heat sink as T8/T10 and T12/T13, provides the necessary compensation.
Finally, there are a couple of other significant items.
Insiders will notice that the none-too-attractive bipolar electrolytic capacitor has been eliminated from the reverse feedback network (R22/R23), which means that DC coupling is used here. To get rid of the resulting output offset, we have provided an automatic compensation circuit that is located on the protection circuit board. We anyhow intended to



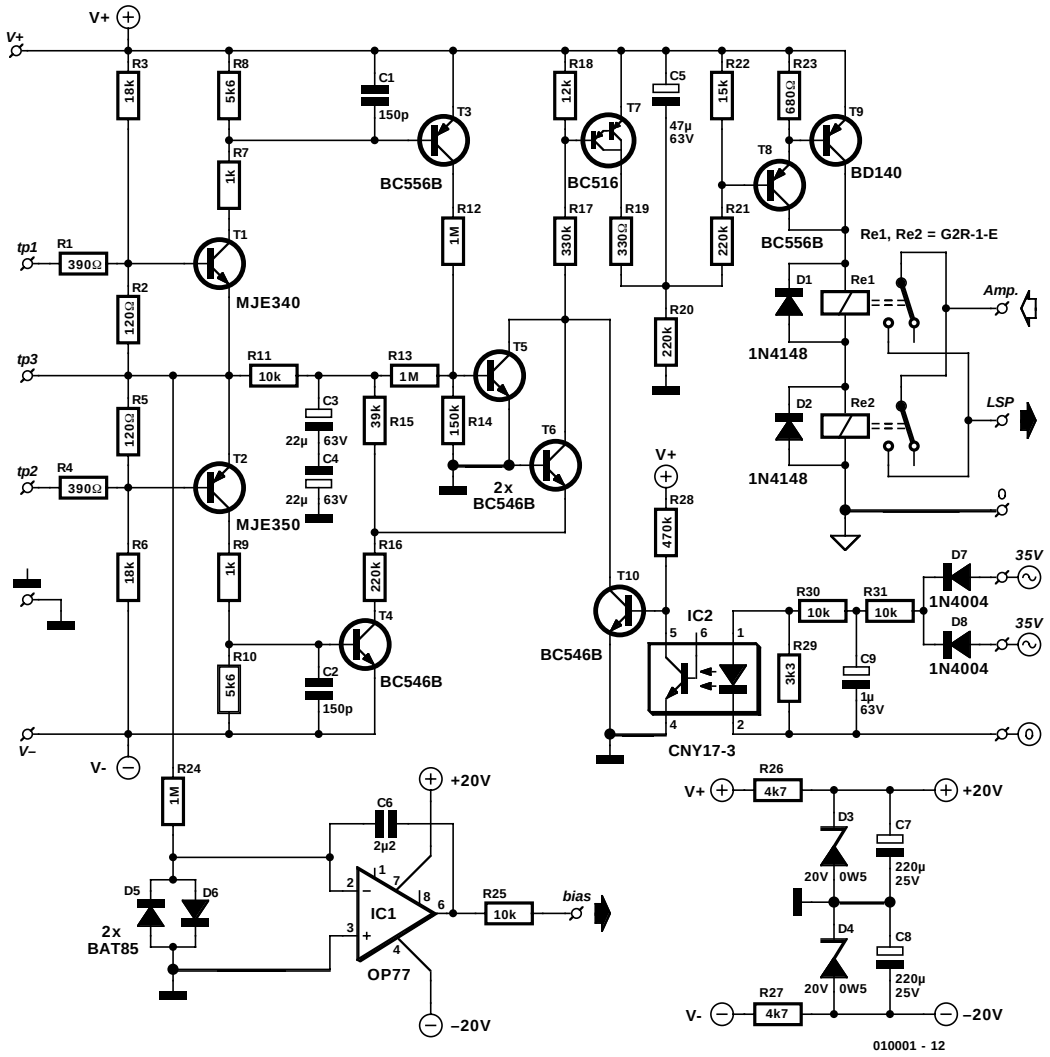


Figure 2. Schematic diagram of the added protection circuitry and offset compensation circuit.

use the compensation circuit to correct for the offset caused by the unavoidable asymmetry of the input stage. The necessary compensation circuit consists of nothing more than an opamp wired as an integrator, which measures the output voltage of the amplifier and provides the proper amount of reverse current feedback to the (bias) input. Thanks to the very high values of R4 and R5 and the decoupling provided by C3, this correction has absolutely no effect on the audio signal.

Another essential detail is that the open-loop gain has been made independent of the load by the addition of R30 and R31. These resistors together determine the output impedance of the voltage amplifier, and as a result the source followers T12 and T13 now operate purely as buffers in the audio range. Without these resistors, the behaviour of the amplifier is directly dependent on the connected load, which is not the way things are supposed to be.

Together with the compensation networks R10/C4 and R15/C5, the modification made using R30/R31 ensures that the amplifier is unconditionally stable, so much so that the standard Boucherot network (R36/C11) can even be omitted.

Protection

The protection circuitry (**Figure 2**), which is located on a separate printed circuit board, includes overload protection, DC protection, a switch-on delay for the output relay and a voltage detector that directly disables the output relay when the power is switched off or any of the transformer voltages is absent. The integrator for the offset compensation is also located on this circuit board.

There are three terminals on the power amplifier board that provide information to the protection circuitry: tp1 and tp2 convey the voltage across the emitter resistors, while tp3 conveys the output voltage. The actual protection takes place with the help of two relays (Re1 and Re2), whose switching contacts are connected in parallel in order to keep the insertion resistance as low as possible. The relay contacts are wired in series with the amplifier output via the terminals 'Amp' and 'LSP'.

The supply voltage for the protection circuitry is tapped off from the supply points on the amplifier board. The supply voltage for the integrator is simply derived from the amplifier supply voltages using a pair of Zener diodes (D3 and D4).

The **overload protection** circuit is constructed in a 'classic' manner using a voltage divider and a tran-

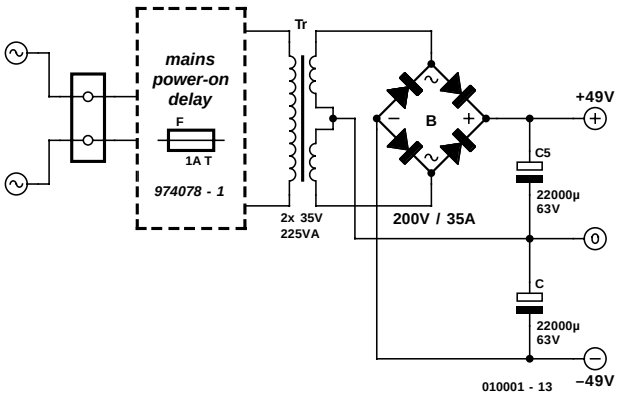


Figure 3. The power supplies of mass-produced amplifiers are rather skimpy. This one has no such problems.

sistor. T1 measures the voltage across R34 in the power amplifier circuit via the network R1-R3 and can thus determine whether the 2SK1530 has exceeded its safe operating area. The combination of T2 and R4-R6 performs the same service for the 2SJ201 by measuring the voltage across R35. The maximum allowable current through the output transistor is linearly dependent on the voltage across the transistor, up to the point that the maximum allowable voltage or current has been reached. The maximum current limit is set using the voltage divider R1/R2 (or R4/R5 for the other half), and this limit is decreased via R3 (or R6) as the voltage across the output transistor increases. Since we can assume a music signal, we have stuck to the 100-ms limit, so that the limiting value for the load impedance can be set lower without causing problems for the output transistors. If T1 (or T2) starts to conduct, the transistors of the DC protection circuit are utilised via T3 (or T4) to disengage the relay. C1 and C2 reduce the circuit's sensitivity to HF interference. R7 and R9 are 5-W types, since their power dissipation can be significant in certain fault situations.

The **DC protection** circuitry employs a commonly used principle. Any DC voltage that is present is received via the low-pass filter R11/C3/C4 (roll-off frequency 1.5 Hz). C3 and C4 together form a bipolar electrolytic capacitor. If a sufficiently large positive voltage is present, T5 is brought into conduction via the voltage divider R13/R14, and T7 is then brought into conduction via R17. With a sufficiently large negative voltage, the current through T6 will be large enough to cause T7 to conduct. The voltage divider R13/R14, in combination with R15/R17/R18, ensures that the positive and negative threshold voltages are nearly the same. T7 can thus be brought into conduction via R12 /R16 and T5/T6. When the supply voltage comes up and no fault is present, the electrolytic capacitor C5 will be charged to approximately half of the supply voltage level via voltage divider R20/R21. The time delay before the relay engages thus amounts to around 8 to 10 seconds. Darlington T8/T10 connects the relay coils to the supply voltage. If T7 starts to conduct, C5 is immediately discharged and the relays disengage.

An optocoupler is used for the **voltage detection** circuit in order to prevent ground loops between the transformer ground and the signal ground, as well as other possible types of interference. The current for the optocoupler diode is provided by R29–R31, and the time constant determined by C9 has been chosen such that the transistor in IC2 remains continuously conducting only as long sufficient voltage is present on both transformer windings. If the voltage drops, T10 starts to conduct and the relays are disengaged.

The **offset compensation** circuit consists of only two resistors, one capacitor, an opamp (IC1) and two diodes, in addition to the supply components. Since the correction current is coupled into the non-inverting input of the power amplifier, this integrator must invert the signal. D5/D6 and R25 provide additional protection for the opamp. With an eye on protection we have chosen an OP77 (ultra-low offset) opamp, which already has internal input protection and is short-circuit proof.

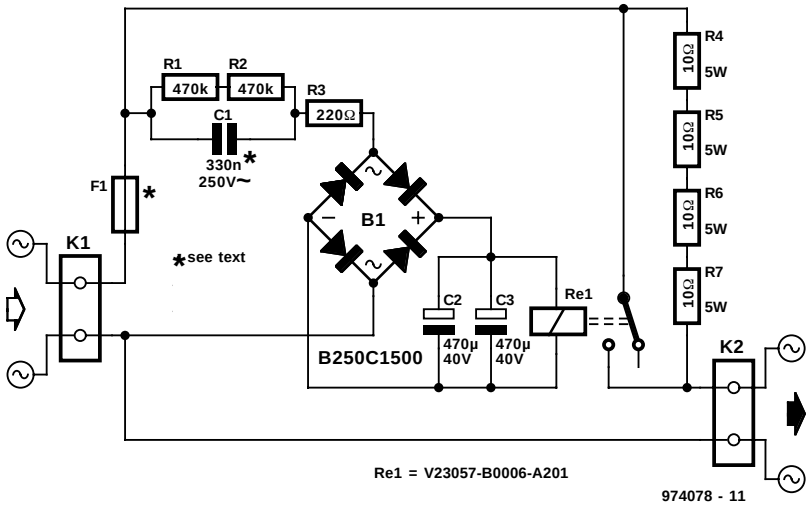


Figure 4. A mains switch-on delay circuit, such as the one shown here, prevents the fuse from blowing when the amplifier is switched on.

A robust power supply

In the description of the original Crescendo, it was already noted that the power supply is one of the most important components of a power amplifier. In fact, the ultimate sound quality depends on the power supply. The design of a good power supply does not have to be difficult, since the well-known and commonly used formula of a transformer, bridge rectifier and electrolytic filter capacitors is fully adequate. However, you should not try to cut corners here, which is why two electrolytic capacitors of no less than 22,000 µF (22 mF) are used in the power supply shown in **Figure 3**. In order to avoid misunderstand-

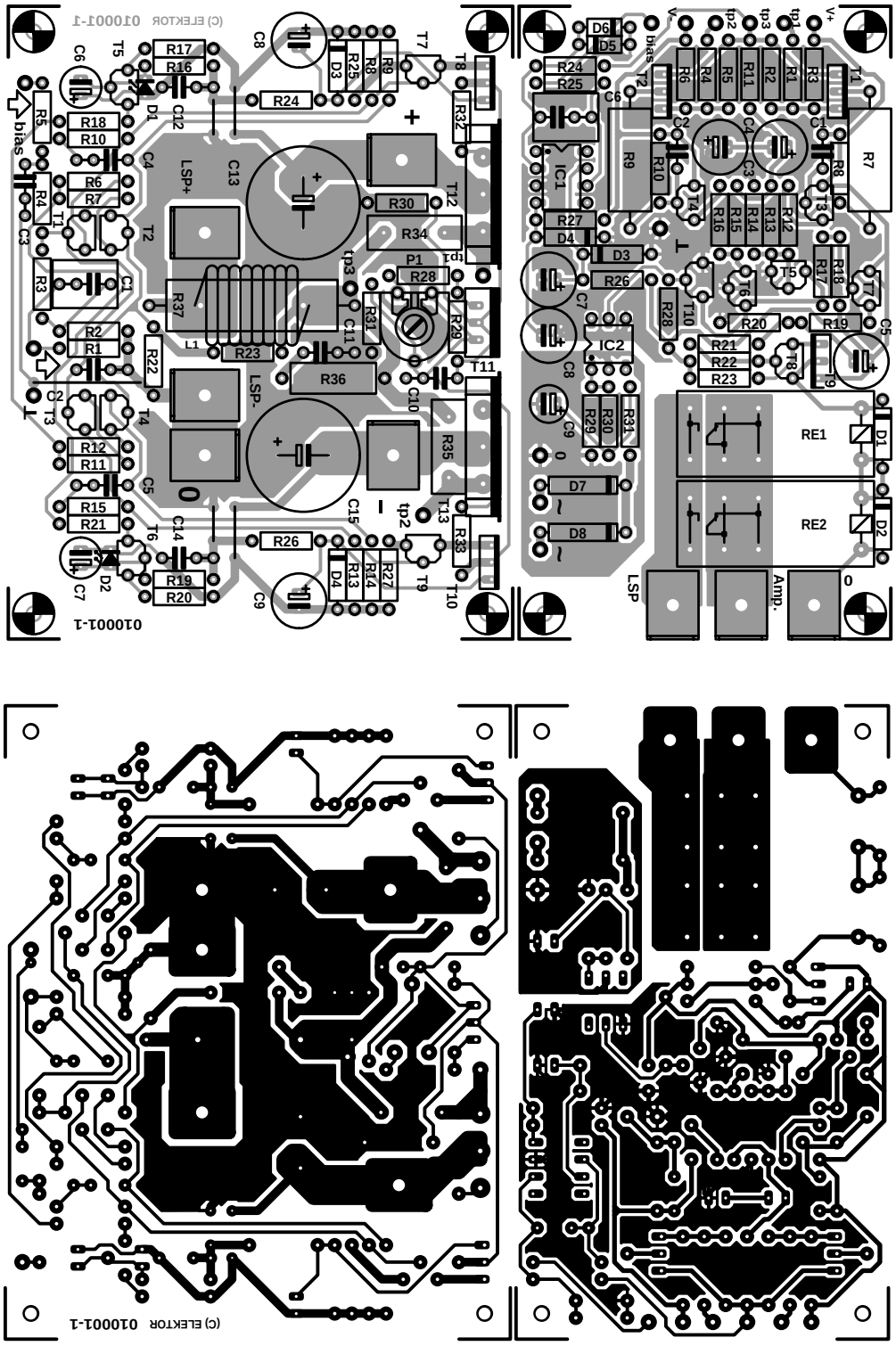


Figure 5. The printed circuit boards for the amplifier and the protection circuitry are delivered as a single board and must be sawn apart.

COMPONENTS LIST
Amplifier board

Resistors:
R1 = 1MΩ
R2 = 47kΩ
R3,R22 = 470Ω
R4,R5 = 1MΩ8
R6,R7,R11,R12 = 47Ω
R8,R9,R13,R14 = 1kΩ
R10,R15 = 330Ω
R16,R19,R30,R31 = 22kΩ
R17,R20,R28 = 270Ω
R18,R21 = 8kΩ2
R23 = 12kΩ
R24,R26 = 10kΩ
R25,R27 = 33Ω
R29 = 120Ω
R32,R33 = 220Ω
R34,R35 = 0Ω22 / 5W low-inductance, e.g., MPC71 series
R36 = 10Ω / 1W *
R37 = 1Ω / 5W
P1 = 1kΩ preset H

Capacitors:
C1 = 2µF2, MKT (Siemens), lead pitch 5mm or 7.5mm
C2,C4,C5 = 1nF
C3 = 180nF
C6,C7 = 100µF 25V radial
C8,C9 = 220µF 25V radial
C10,C12,C14 = 100nF
C11 = 10nF *
C13,C15 = 1000µF 63V radial

Inductors:
L1 = 9 turns 1.5 mm dia. ECW around R37, inside diameter 8 mm

Semiconductors:
D1,D2 = rectangular face, red
D3,D4 = zener diode 3V9 / 0.5W
T1,T2,T6 = BC546B
T3,T4,T5 = BC556B
T7 = BC560C
T8 = MJE350
T9 = BC550C
T10 = MJE340
T11 = 2SK537 (Toshiba)
T12 = 2SK1530 (Toshiba)
T13 = 2SJ201 (Toshiba)

Miscellaneous:
5 off M3 spade terminals, PCB mount
3 off ceramic (or mica) isolating washer for voor T8/T10/T11
2 off mica isolating washer for

ings, we hasten to point out that we are talking about a monaural version here, so for a stereo amplifier you will have to build two of these supplies!
The 'mains switch-on delay' shown inside the dotted box in Figure 3 is not mandatory, but it is highly recommended — especially if a toroidal transformer is used. This circuit does exactly what its name suggests, and it ensures that excessive current surges do not occur when the mains voltage is switched on. Such circuits have frequently been described in *Elektor Electronics*; the most recent one can be found in the Summer Circuits issue of 1997, and we have reproduced its

T12/T13 (e.g., TO-218 sheets size 21 x 24 mm)
Heatsink: <0.5°K/W (e.g., Fischer type SK47/100 mm, Dau Components)
PCB, order code **010001-1**
Mainsd power-on delay PCB , order code **974078-1**
Enclosre, e.g., Monacor (Monarch) type UC113/SW

*) may be omitted

Protection board

Resistors:
R1,R4 = 390Ω
R2,R5 = 120Ω
R3,R6 = 18kΩ
R7,R9 = 1kΩ / 5W
R8,R10 = 5kΩ6
R11,R25,R30,R31 = 10kΩ
R12,R13,R24 = 1MΩ
R14 = 150kΩ
R15 = 39kΩ
R16,R20,R21 = 220kΩ
R17 = 330kΩ
R18 = 12kΩ
R19 = 330Ω
R22 = 15kΩ
R23 = 680Ω
R26,R27 = 4kΩ7
R28 = 470kΩ
R29 = 3kΩ3

Capacitors:
C1,C2 = 150pF
C3,C4 = 22µF 63V radial
C5 = 47µF 63V radial
C6 = 2µF2 MKT (Siemens), lead pitch 5mm or 7.5mm
C7,C8 = 220µF 25V radial
C9 = 1µF 63V radial

Semiconductors:
D1,D2 = 1N4148
D3,D4 = zender diode 20V / 0.5W
D5,D6 = BAT85
D7,D8 = 1N4004
T1 = MJE340
T2 = MJE350
T3,T8 = BC556B
T4,T5,T6,T10 = BC546B
T7 = BC516
T9 = BD140
IC1 = OP77GP (Analog Devices)
IC2 = CNY17-3

Miscellaneous:
Re1,Re2 = relay, type G2R-1-E (Omron), 16A / 24V / 1100 ohm)
3 off M3 spade terminal, PCB mount

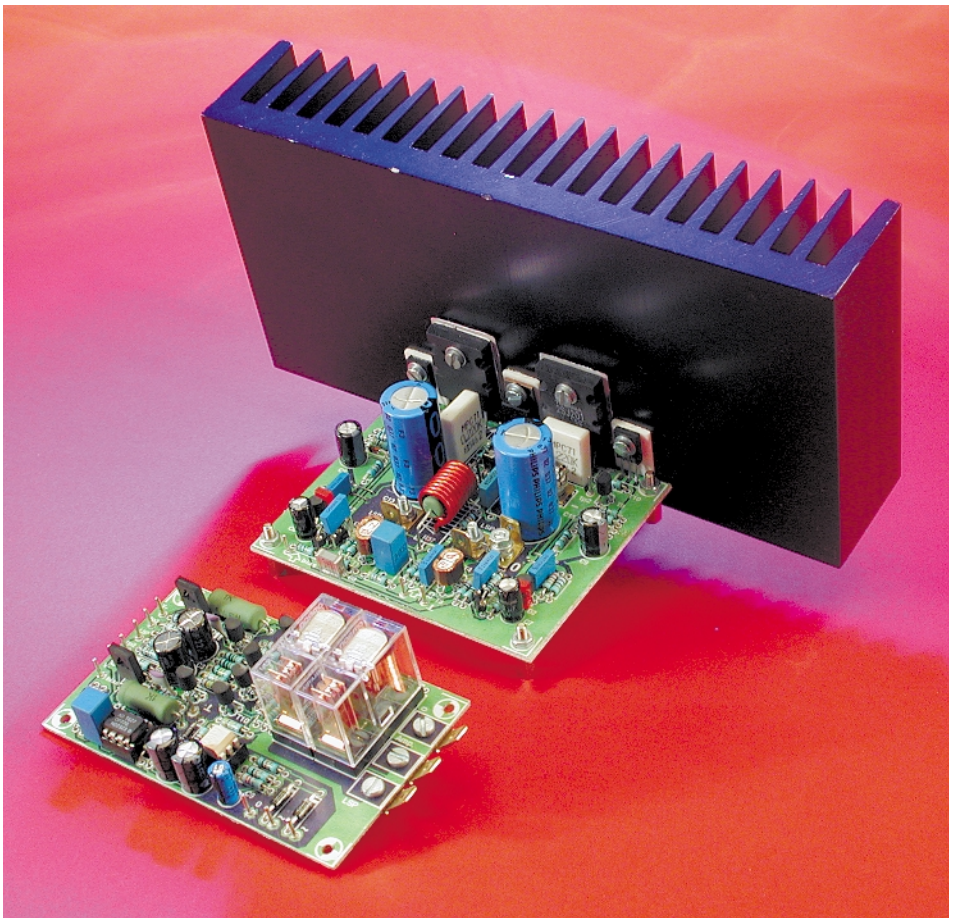


Figure 6. This is how the finished circuit board should appear. Don't forget the insulating washers for transistors T8 and T10–T13!

schematic diagram in **Figure 4**. Its operation is simple, and is based on the fact that the current is initially limited by R4-R7 immediately after switch-on. After the expiry of a time delay determined by C2 and C3, these resistors are bridged over by the relay and the full current flows between K1 and K2. The relay used here is a type that can switch 2000 VA. The supply voltage for the relay is taken directly from the mains circuit via C1, R3 and B1, so this circuit is dangerous to the touch!

Soldering

The printed circuit board layouts for the amplifier and protection circuitry are shown in **Figure 5**. These circuit boards are supplied as a single piece, so the must be (carefully) sawn apart. Experienced electronics types will not need very much advise with regard to the construction of the circuit boards, since the component layout overlay and the components list speak for themselves. Still, we would like to make a few practical remarks.

On the amplifier board, five wire bridges must be inserted, and it is a good idea to do this at the beginning. In addition, there are two items on the amplifier board that could be considered to be somewhat difficult: the thermal coupling and the output coil L1.
For the thermal coupling between the D1/T5 and D2/T6 pairs, it is sufficient to mount the LED so that it is in contact with the flat face of the transistor. In the case of the T1/T2 and T3/T4 transistor pairs, it is recommended to clamp a small metal ring around each pair. Incidentally, we have discovered that suitable rings can be made by sawing them from a piece of copper water pipe and then bending them into a suitable shape.
Coil L1 can be easily wound on an 8-mm drill bit. After this you can insert R37 into the coil and then solder both components to the circuit board, after having first removed the lacquer from the two ends of the coil with the aid of a knife.
Transistors T8 and T10–T13 are intentionally placed along one edge of the circuit board so that they can easily be screwed to a single common heat sink. Naturally, the transistors must be mounted using insulating washers,

and as usual it is recommended to smear a thin layer of thermal grease on each side of the insulator before mounting the transistor. The thermal resistance of the heat sink should be less than 0.5 K/W. **Figure 6** shows one of the fully assembled prototype amplifier circuit boards with attached heat sink.
There isn't much to say about the protection circuit board. You should pay attention to the diameter of the electrolytic capacitor C5, which must be no more than 8 mm. If you cannot obtain a suitable type, a 40-V type can also be used.
For the sake of completeness, the printed circuit board layout of the previously mentioned mains switch-on delay circuit is shown in **Figure 7**. This circuit board was never included in the Readers Services list in the past, but since this 'two-stage' delay can be especially useful for a variety of applications, we have now added it to the list.

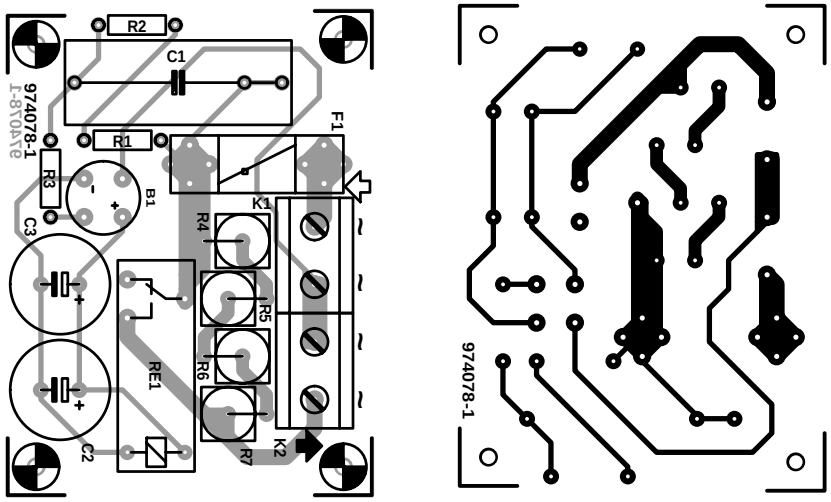


Figure 7. The printed circuit board layout for the mains switch-on delay circuit shown in Figure 4.

Wiring and set-up

Once you have finished building the amplifier and protection logic boards (or sets of boards) and have carefully checked them against the components list, it is time to start looking for a suitable enclosure. The first decision to be made is whether you want to build the amplifier as a monophonic building block or as a stereo version. We chose the latter option for our prototype, which means that what we actually did was to build two mono blocks into a single enclosure, each with its own power supply and mains switch-on delay. The only shared item is the mains switch. For the enclosure, we chose a Monacor (in some countries: Monarch) box that provides a generous amount of room for everything, and then mounted hefty heat sinks (bigger than actually required) on opposite sides of the box.
Since there are several circuit boards involved, the wiring of the complete amplifier includes quite a few interconnections – which is why we have made a separate wiring diagram, as shown in **Figure 8**. Connect the V+, V–, earth, tp1, tp2, tp3 and bias points on the protection board to the corresponding points on the amplifier board using ordinary insulated stranded wire. The '~35 V' points should be connected directly to the outer ends of the transformer windings, and point '0' should be connected to the junction of the filter capacitors in the power supply.
Use lengths of screened audio cable to make the connections between the input sockets (Cinch sockets) and the input points on the amplifier boards.
Flat tab connectors (automotive connectors) are used for the output and supply connections on the circuit boards. The connections between these points must naturally be made using heavy-gauge wiring. We used 2.5-mm^2 electrical wire for this purpose. The contacts of relays Re1 and Re2 on the protection board are simply connected in series with the amplifier output by connecting the output terminal 'LSP+' to the relay input terminal 'Amp' and the 'LSP' terminal of the protection board to the positive output socket (banana socket). The other (negative) banana socket is connected directly to the 'LSP–' terminal.
The necessary connection between the circuit ground of the amplifier and the metallic enclosure can best be realised by fitting the Cinch (a.k.a. RCA or 'line') input sockets in a 'normal' (non-insulated) manner. Take care that there is not any other unintentional connection between the signal ground and the enclosure ground, since this will create an earth loop that can cause stubborn hum problems.
It goes without saying that a well-insulated cable, a robust mains switch and an equally robust mains entrance must be used for the connection to the 230-V mains circuit. Pay attention to the electrical safety of the overall assembly, and attach an identification label that lists the specified values of the supply voltage (230 V) and fuse to the outside of the enclosure.

Once you have again thoroughly checked everything and re-measured the supply voltages, it's nearly time to power up the amplifier. Before doing this, however, you must turn trimpot P1 *fully to the left* (counter clockwise). Otherwise you run the risk that the quiescent current will immediately rise to a very high level, which is not what we want.
After switching on the unit, first check the amplifier output (test point tp3) to verify that the voltage is zero. An offset of a few millivolts is acceptable, but if you measure 0.1 V or more you will have to carefully reinspect the whole assembly, since there is something wrong.
Following this, you can set the quiescent current to the proper value. The ideal value for this amplifier is 200 to 250 mA. To adjust the quiescent current, connect a voltmeter across R34 (test points tp1 and tp3) and turn P1 slowly until the measured voltage is between 0.044 and 0.055 V. Then let the amplifier warm up for half an hour, and again adjust the current to the same value using P1.

Listening

Readers who have already taken a peek at the measurement results shown in the separate box will have quickly concluded that the Crescendo scores very well as far as the numbers are concerned. However, we know from experience that amplifiers with practically identical specifications can sound quite different.

We thus come to the crucial question: how good is the sound of the new amplifier (or, if you will, the 'refurbished old amplifier')?

The first thing that struck us in listening sessions is that the Crescendo can produce a nicely spacious and open sound image with all different types of music. Of course, the relative differences between good amplifiers are always very subtle, but the Crescendo clearly revealed itself to be an amplifier with a pleasantly warm-blooded character.

After listening to the amplifier for a while, we developed a certain understanding of the preferences of fervent MOSFET fans, since the sound produced by the amplifier is just a bit less reserved and clinical than that produced by a typical amplifier with bipolar transistors in the output stage. An amplifier such as the 'Compact AF Power Amplifier', which was published in May 1997 (and which is one of our favourites), offers reproduction that (according to our convictions) can hardly be surpassed in terms of natural fidelity and detailing, but it still misses that slight trace of warmth that is so typical of the Crescendo. Can we say that one of the two is the better amplifier? No, that would be going far. The differences are too small for such a pronouncement, and anyhow such a judgement is always very subjective. 'Better' and 'worse' are qualifications that do not have a place here; at most we can say 'different'.

What well can be considered to be no less than amazing is that this Crescendo, in spite of (or thanks to) its simple concept and the age of the original design, can easily hold its own against many more modern examples of the breed. This amplifier can be highly recommended, and not only for MOSFET fans!

(010001-1)

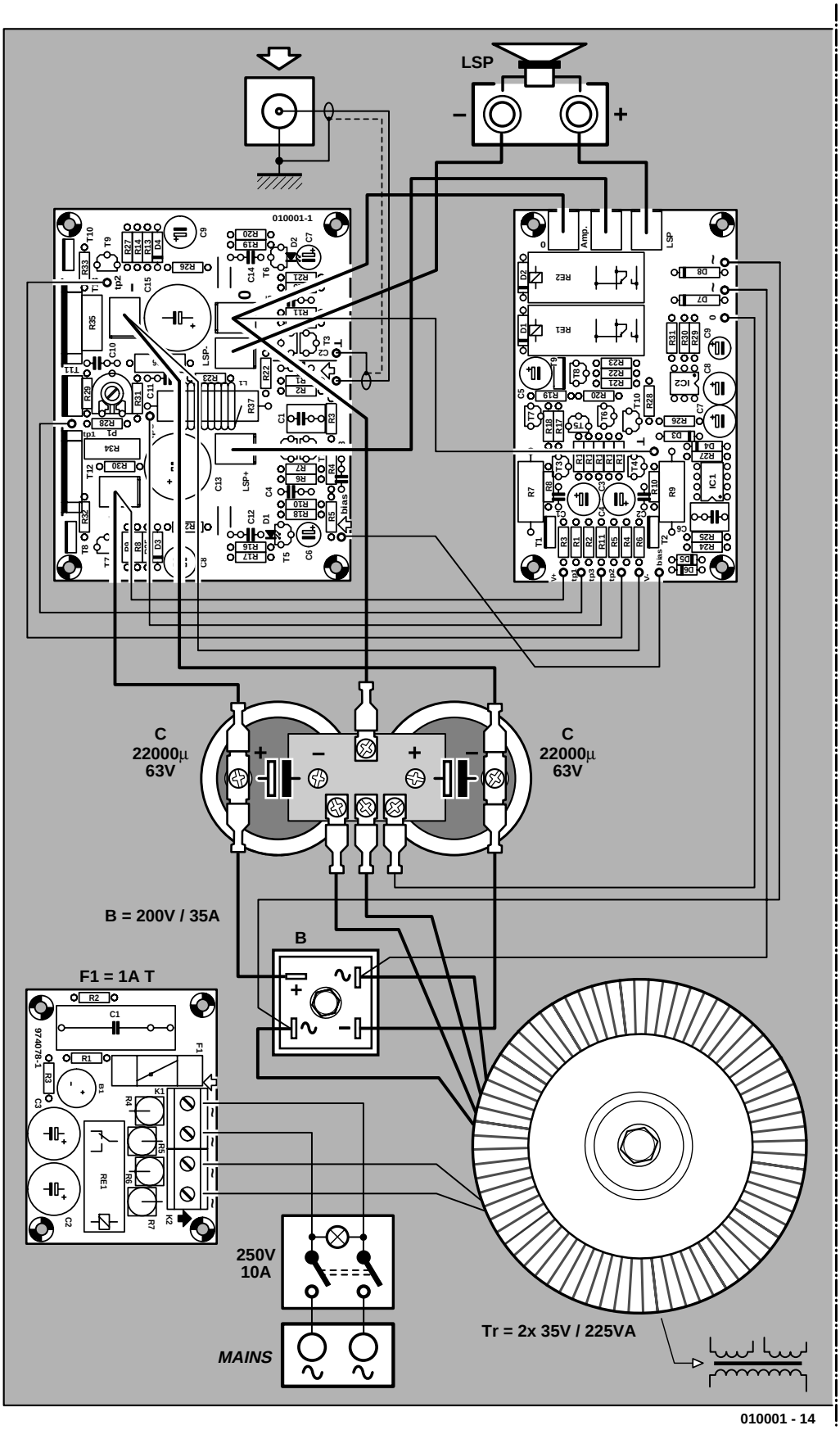


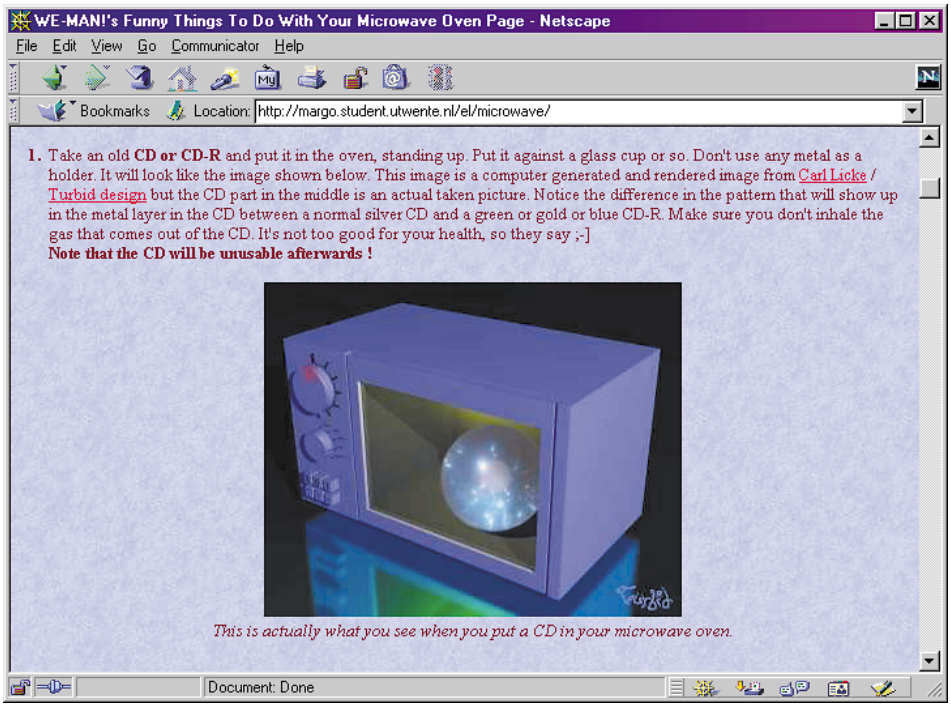
Figure 8. Wiring diagram. Thick wires must be used for the power supply and amplifier output wiring!

Microwave Oven Experiments

all about cooking unusual ingredients…

By Harry Baggen

The experiments referred to in this article should be complemented with a stark American-style warning: Don't Try This at Home! The good news is you don't need to, because the effects of 'cooking' various non-food-stuffs in a microwave oven may be seen on a variety of exciting websites specialised in the subject.



A microwave oven may be compared to a powerful RF transmitter. Functionally, the microwave oven employs a transmitter running some 1 kilowatts of RF power at about 2.45 GHz, in an RF-tight 'cage', with the purpose of heating foodstuffs. Since experimenting with a microwave oven may be dangerous, we would urge you to stick to the use as outlined in the user's manual.

In spite of this (fairly obvious) warning, we came across quite a few instances of 'non-recommended use' of a microwave oven when trailing the Internet.

Radio amateurs have been using microwave ovens, or rather the magnetron components inside, for EME (earth-moon-earth) links on the 13 cm band. Be warned though, this communication mode requires a radio amateur licence, highly specialized ancillary equipment and years of experience in designing SHF circuits. Fortunately, all other experiments we came across were limited to 'popping it in the oven and se what happens'. It is these experiments that we found described in great detail on a number of websites. Please note, however, that this article is by no means to be understood as encouragement to replicate any of the experiments described. Be careful — a microwave oven is not a toy.

The title of the first website we hit upon pretty much says it all: **Funny things to do with your microwave oven** [1]. The website proves beyond doubt that a microwave is suitable for a lot more than just cooking foodstuffs. Because it provides a clear list experiments carried out, this website is an excellent introduction into microwave

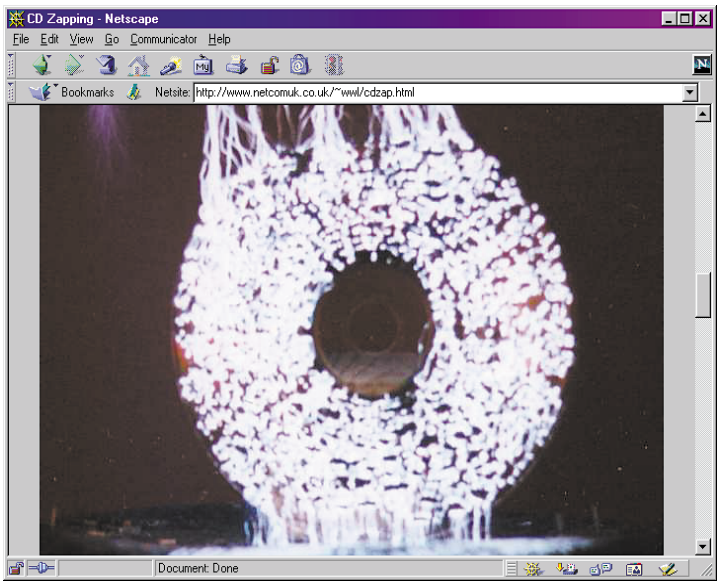
oven 'abuse'. Objects and materials popped in the oven for the fun of it include a CD, a toothpick, a light bulb, aluminium foil, a Christmas ball and a small fluorescent tube.

The CD appears to be one of the most popular victims to die in the oven. The thin metal layer which is present in the polycarbonate carrier is absorbed (and consequently heated) by the RF energy emitted by the microwave oven. As a result of the sudden heating, cracks appears in the metal layer, whereupon the CD starts to light up with a spectacular pattern. If you want to know more about the effects produced by cooking a CD in a microwave oven, surf to the website called **CD-ROMs in the Microwave** built by Paul Haas [2]. Paul has experimented with different types of CD and describes the surprising effects. He also comes up with an explanation of the effects that occur when a CD is subjected to an extremely high dose of RF energy.

The next site is again fully covered by its title: **Unwise Microwave Oven Experiments** [3]. Because it contains a couple of really hazardous things like exploding coffee, the opening page rightly warns of the dangers of such experiments.

At **Powerlab Microwave Oven Experiments** [4] you may study some rather unusual experiments like producing a plasma ball in the microwave. There is also a page describing how a microwave oven is taken apart to turn it into a magnetron gun. Fortunately, this is not used for shooting but to enable the plasma ball to be made outside the magnetron cage. Extremely dangerous for the non-technical and not recommended to try for yourself. On the other hand, an interesting phenomenon to study from a technical point of view.

A rather more serious approach was found at **Physics Inside a Microwave Oven** [5] built by Maarten Rutgers of the University of Ohio. The website, in a popular-scientific tone and aimed at students, describes a number of experiments



with objects including a light bulb, a CD, fax paper and a burning candle.

To some people, none of the above experiments goes far enough. Take for instance Patrick Michaud. On his website called **Fun with Grapes — a Case Study** [6] we found an in-depth study of his experiments with different kinds of grape in the microwave oven — with and without skin!

Fun Things to do with Microwave Ovens [7] also has high amusement value. 'All things exploding' seem to enjoy great popularity, including beans and eggs.

A useful overview of websites covering microwave oven experiments may be found on the **Microwave Experiment Page** [8]. In addition to a number of try-outs using different objects and links to other microwave oven websites, these pages also provide a short history of the microwave oven.

If, after viewing these amusing (but dangerous) experiments you want to know how a microwave oven works, visit **How things work — Microwave Ovens** [9] built by Louis Bloomfield of the University of Virginia. In an easy to follow manner Professor Bloomfield explains the underlying physics of the effects that occur in a microwave oven. Still more links to technical backgrounders on microwave cookers may be found on the link page for **Microwave Ovens** [10].

Finally, we should mention a website that is not strictly aimed at microwave experiments. Rather, it discusses the microwave oven as a kind of tool. After bombarded a CD with microwaves, the disc is mounted on a Tesla transformer, which is a kind of high-voltage generator capable of creating stunning light effects. De combination of the 'cooked' CD and the Tesla coil is good for breathtaking photographs which you can not afford to miss — see **CD Zapping** [11]. (010031)

Website url's

(also available as hyperlinks at www.elektor-electroncis.co.uk):

[1] Funny things to do with your microwave oven:
http://margo.student.utwente.nl/el/microwave/

[2] CD-ROM's in the Microwave:
http://www.hamjudo.com/notes/cdrom.html

[3] Unwise Microwave Oven Experiments:
http://www.amasci.com/weird/microexp.html

[4] Powerlab's Microwave Oven Experiments:
http://www.powerlabs.org/uwavexp.htm

[5] Physics Inside a Microwave Oven:
http://www.physics.ohio-state.edu/~maarten/microwave/microwave.html

[6] Fun with Grapes - A Case Study:
http://www.sci.tamucc.edu/~pmichaud/grape/

[7] Fun Things to Do with Microwave Ovens:
http://www.everist.org/special/mw_oven/index.htm#cocky

[8] Microwave Experiment Page:
http://members.tripod.com/~hochwald/microwave/micro.html

[9] How things work — Microwave Ovens:
http://rabi.phys.virginia.edu/HTW//microwave_ovens.html

[10] Microwave Ovens:
http://www.physics.udel.edu/wwwusers/watson/scen103/less-muwave.html

[11] CD Zapping:
http://www.netcomuk.co.uk/~wwl/cdzap.html

Fuzzy Logic

Part 1 – fuzzy essentials

By Owen Bishop

OandA.Bishop@bigpond.com

In this two-part article Owen Bishop looks at designing fuzzy control systems with the aid of software.

What is your size in shirts? There are two ways of answering this. Many shirtmakers quote one of several dimensions such as size of collar or circumference of chest. Run a tape measure around your neck, read off the size in centimetres, and select a shirt of the appropriate size. Wearer's necks are allocated to different size categories, often at 2 cm intervals. Thus, the category '41 cm' may refer to necks that are 40 cm of more in circumference but less than 42 cm. Note that the sizes of customer's necks, not the sizes of the necks of the shirts. The customers in a store could be measured and sorted out into groups on this basis. The manufacturers classify the **customers**, then make shirts intended to fit them. Each group of customers can be sold shirts that fit their necks with sufficient degree of comfort. Whether the shirts also fit the customers on the chest or waist in another matter! Putting this in logical terms, customers are classified into **sets** based on neck circumference.

For a given set (say, the '41 cm' set), a customer is either a member of that set or is not a member. This 'is'/'is not' feature is a **binary** one, a characteristic of Boolean logic. Because the neck size system leaves no room for doubt at the boundaries between one set and another, we say that its sets are **crisp** sets.

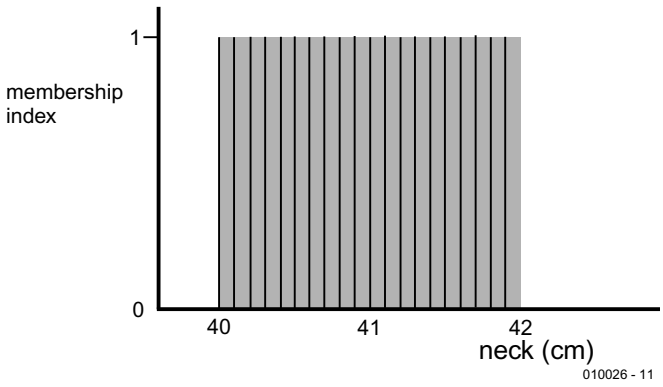


Figure 1. In a crisp (Boolean, binary) set all members have a membership index of 1.

Fuzzy sets

Another way of defining size categories is by verbal description. The popular system has a range of values such as S, M, L, XL, and XXL.

A shirt marked 'L', for example, is intended to fit a 'typical' large man. Occasional shirtmakers produce shirts for really huge men in sizes from 3XL up to 8XL. At the other end of the scale, there may be an XS size, and sometimes SM comes between S and M. In words, the most common categories are small, medium, large, extra-large, and extra-extra-large.

The essential feature of this system is that a customer may be classed as a member of more than one set. The customer may find that shirts of two adjacent categories fit reasonably well. For example, the author falls into both the L and the XL set and, when he buys a shirt, he usually takes an L shirt and an XL shirt along to the fitting room and buys the one that fits better. Much depends on the maker, the fabric and the style of the shirt and there are no clear-cut boundaries between the sets. These are **fuzzy sets**. In general, fuzzy sets correspond to real-life situations and are a more satisfactory and practicable way of characterising shirt sizes. Humans come in a wide variety of shapes and sizes and it can not be expected for everyone to slot neatly into the crisp sets based on neck circumference.

Set membership

Figure 1 shows the **membership function** of the '41 cm' crisp set. Each of the various sizes of human neck included in this set are indicated by vertical lines spaced, for convenience, at 1-mm intervals. Neck sizes vary continuously over the range, to the actual membership function is the whole of the shaded

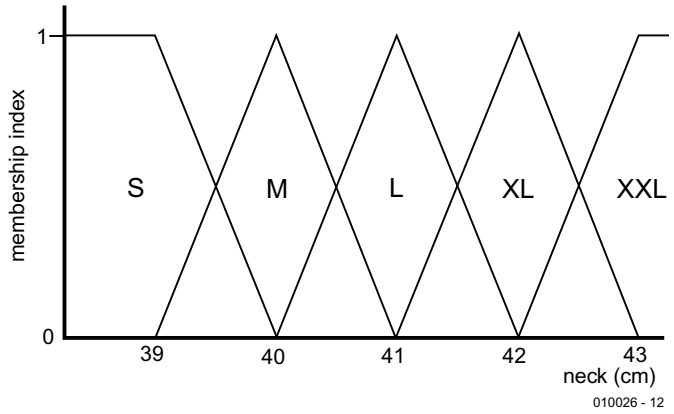


Figure 2. Fuzzy sets such as shirt sizes each cover a range of neck sizes.

area. Each of the sizes within the set has a **membership degree**, expressing the extent to which that size is considered to be a member of that set. In a crisp set, every member of the set has a membership degree of 1. They all take collars of one particular size, possibly made 42.5 cm in circumference to provide a comfortable fit and to allow for shrinking. The necks all belong fully to the '41-cm' set and to no other set. Conversely, all necks that do not belong to the set have a membership degree of zero. The 0 or 1 situation reflects the binary (two-valued) nature of a crisp set.

Figure 2 shows typical membership functions of fuzzy sets. Only necks toward the average of the set have membership degrees in the region of 1. Other members belong to the set with degrees between 0 and 1. Membership is not binary as in a crisp set, but is **multi-valued**. The horizontal scale may be collar size as in Figure 1, but there is now a bigger spread of neck sizes in each set. In addition, a person with a neck of a given size may be a member of two (or even more) adjacent sets.

Because of their shape, the sets illustrated in Figure 2 are referred to as **triangular** sets. There may also be **trapezoidal** sets, in which members in the central region of the range all have a membership degree of 1. Trapezoidal sets are seen at the limits of the size ranges. In certain applications, there may be sets with Gaussian or other functions.

Linguistic variables

A very important feature of fuzzy logic is that it does not have to be quantitative. Sets are defined with fuzzy boundaries where the members have low membership degrees and therefore do not contribute fully to the set. The exact cut-off point is not particularly important. Instead of defining a set in numerical terms, we describe it by using one or two appropriate words or abbreviations. We call these **linguistic variables**. The size variables S, M, L, XL, and XXL are examples. In the case of a dishwasher, the size of the load could be described by variables such as: 'very small', 'small', 'medium', 'large', and 'full'.

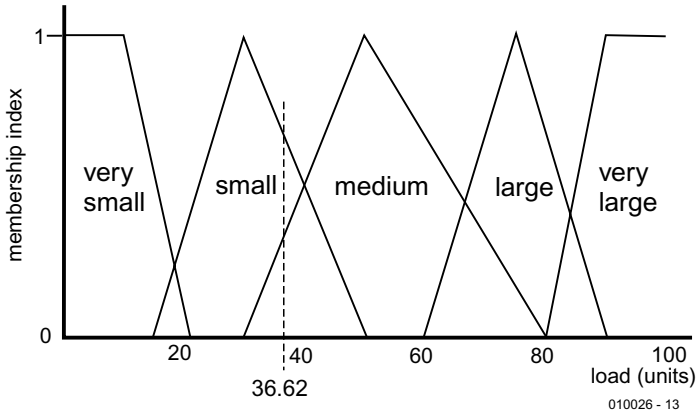


Figure 3. A washing machine load rated at 36.62 units is in both the 'small' and 'medium' fuzzy sets.

Fuzzy controller

Taking the example of a dishwasher in more detail, we will show how to build a fuzzy logic controller for a hypothetical dishwasher, which uses fuzzy logic to control the wash cycle. The input to this system is the size of the load, which may be categorised under the five linguistic variables listed in the previous paragraph. **Figure 3** defines the five fuzzy input sets on an arbitrary scale from 0 (empty) to 100 (full). The size of the load could be measured electronically by load cells on the wire baskets. The output variable is the wash time, which might range from 5 minutes to 40 minutes. **Figure 4** shows five fuzzy output sets and variable names. Note that the number of output sets does not have to be equal to the number of input sets.

Like any logic system, a fuzzy logic program processes a set of logical rules. The rules for this dishwasher might be:

1. IF load is 'very small',
   THEN washtime is 'minimum'.

2. IF load is 'small',
   THEN washtime is 'reduced'.

3. IF load is 'medium',
   THEN washtime is 'medium'.

4. IF load is 'large',
   THEN washtime is 'extended'.

5. IF load is 'full',
   THEN washtime is 'maximum'.

These rules have the same format as those in crisp logic but there is an essential difference. In crisp logic, the rules are applied in sequence. As each rule is processed, the computer moves on to the next line the program. The rules are processed one at a time, in the order in which they appear in the program. In fuzzy logic, the rules are applied in parallel. A fuzzy logic processor chip is specially designed for parallel processing of the rules. However, this does not prevent us writing fuzzy logic programs for ordinary com-

puters that process rules sequentially. The results of applying each rule in turn can be kept on hold, until all are processed. This is not as efficient as parallel processing but makes it possible to write useful fuzzy programs for PCs and other computers as described below.

Fuzzy processing

We take the example of the fuzzy dishwasher. The first stage in operating the system is to measure the size of the load electronically. We might measure weight or volume, or perhaps both and combine the two by using a formula. Whatever kind of measurement we make, it will have a definite value. It will be **crisp**, even if it is measured on an arbitrary scale. For example, the load size may be 36.62, measured on an arbitrary scale from 0 to 100.

The next stage in processing is **fuzzification**. In **Figure 3**, the input value 36.62 is seen to belong to two input sets, 'small' and 'medium'. When the rules are applied, the conditions of both Rule 2 and Rule 3 apply. We say that rules 2 and 3 are **fired**. Rules 1, 4 and 5 are not fired.

Fuzzy inference

Next comes the stage known as **fuzzy inference**, estimating the result of the firings. Note that the two rules are not fired to the same extent. They have different **degrees of applicability**. The value 36.62 has a membership index of approximately 0.67 in the 'small' set, and an index of 0.33 in the 'medium' set. The corresponding outputs have correspondingly different degrees of applicability. Output 'reduced' has a DOA of 0.67 and output 'medium' has a DOA of 0.33. We need to combine the two outputs in some way to give a single output value.

Defuzzifying

There are various ways of obtaining crisp output, of which the following was developed by Mamdani and is one of the most popular. Its mathematical basis is beyond the scope of this description but, fortunately, the computer does the mathematics. We will look at some software in Part 2 of this article.

The first step is to truncate the two membership functions at levels equal to the two membership indices (**Figure 5**). The functions are not only clipped but also scaled, as the figure shows. The two membership functions are now regarded as a single function. Usually the area of overlap is ignored. Finally, we produce a single crisp value to represent this combined function. There are several ways of doing this. The most favoured way is known as the 'centre of area' or 'centre of gravity' technique. This produces a weighted mean of the wash times in proportion to their membership indices. Imagine the combined shape to be cut from thin card, then balanced on a pencil that lies parallel to the vertical (index) axis. The card balances when the pencil is placed under the card at 21.6550 on the horizontal (time) axis.

There are other ways of defuzzifying, which give better results in certain types of application, and are easier to calculate. For example, where two areas do not overlap we may take the 'centre of the largest area' as the defuzzified value.

Summing up: the action of a fuzzy controller (as realised on a fuzzy computer or on a PC) is:

1. Crisp input
2. Fuzzification — fire all rules
3. Fuzzy inference — combining the results from those rules that fire.
4. Defuzzification.
5. Crisp output.

In Part 2 we'll look at the software and carry the development of the fuzzy controller a stage further.

(010026-1)

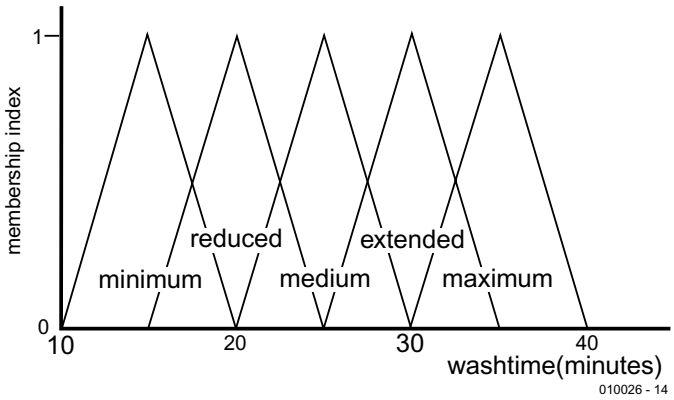


Figure 4. Washtimes are classified into five fuzzy sets covering washtimes from 10 to 40 minutes.

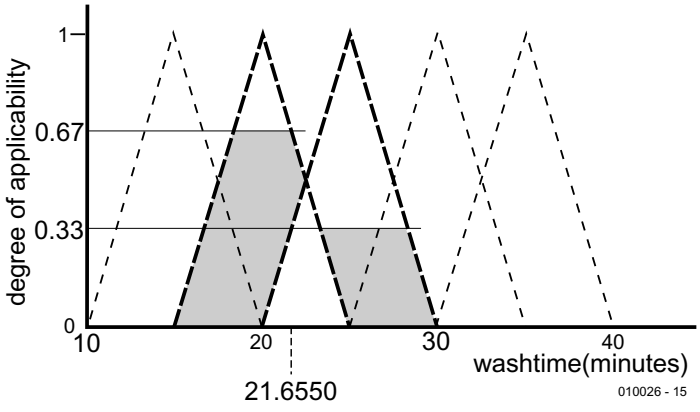


Figure 5. If Rules 2 and 3 fire with applicability 0.67 and 0.33, the resulting crisp output is the centre of the shaded areas, at 21.6550.

IR Remote Control Codes (2)

Part 2

By A.N. Other

In this second and concluding instalment we look at the structure of the infra-red remote control code systems proposed by Denon, NEC, Motorola, Samsung and Daewoo, as well as a format which is generally referred to as 'Japanese Code'.

Correction to Part 1 (March 2001)

In last month's instalment, the illustrations with the RECS80 format description show the Daewoo format. The correct drawings are shown below.

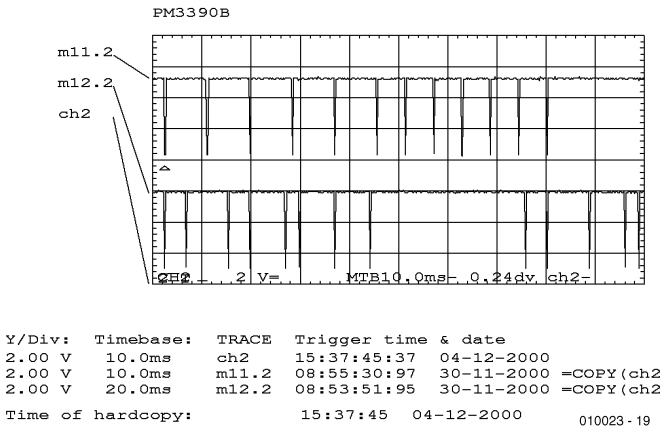


Figure 5. RECS80 code at the output of the TFMS5360 receiver IC.

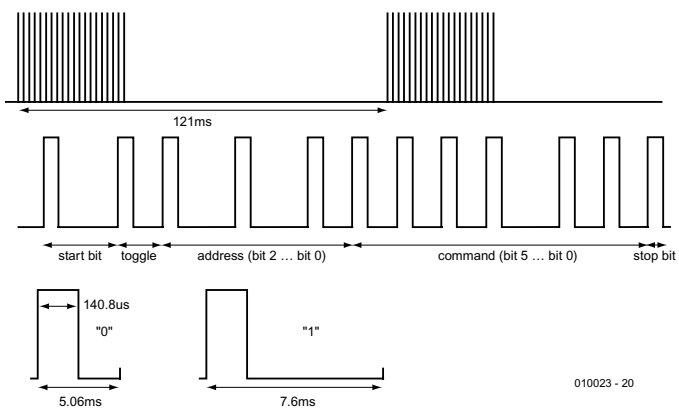


Figure 6. RECS80 code message format

Denon Code

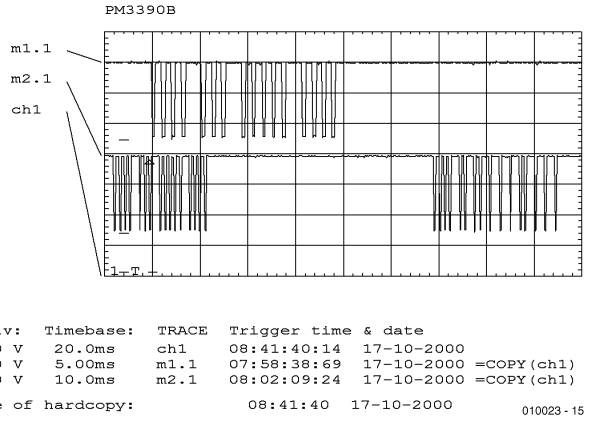


Figure 7. Denon code at the output of the TFMS5360 receiver IC (CD Player)

A Denon coded message consists of 15 information bits. The first five bits are the equipment address field while the remaining ten bits contain the command.
The modulation frequency is 32 kHz and the bit coding is:

1: 275 µs mark, 1900 µs space
0: 275 µs mark, 775 µs space

To reduce the effects of interference the message is sent twice, the second time 65 ms after the first. During the second time the command field bits are inverted. The receiver will only accept commands when the second message is identical to the first after the command field bits are inverted. The address field is always sent uninverted. Bit 16 is a stop bit.

There are currently no dedicated chips to implement this code. Transmitters and receivers can be built from mask programmable micro controllers e.g. the Mitsubishi M50560.

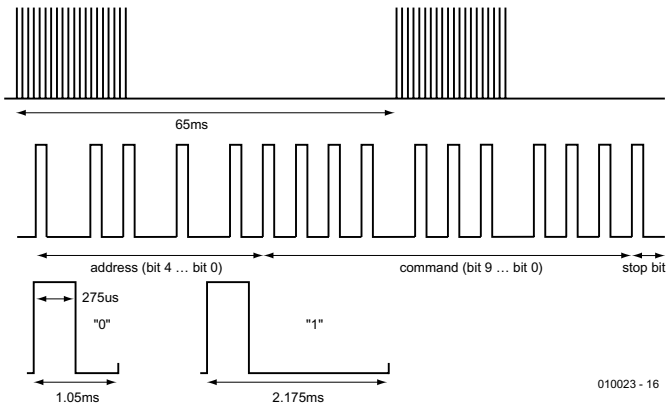


Figure 8. Denon code message format

NEC Code

The NEC code operates with a carrier frequency of 38 kHz and uses pulse position modulation (PPM). Transmission begins with a 9 ms long start bit, followed by a 4.5 ms space. The message information is contained in the following 32 bits which consists of a 16 bit manufacturer field and a 16 bit command field. The 8 bit wide data is sent twice, the second time inverted. A complete message is 67.5 ms long. Each bit is sent using the following format:
1: 0.56 ms Pulse, 1.69 ms space
0: 0.56 ms Pulse, 0.565 ms space
A new message is sent 108 ms after the start of the preceding message. A special current saving feature is implemented if a key is held down on the controller. In this case the message consists of a 9 ms start bit followed by a 2.25 ms space and a 0.56 ms pulse.

Sanyo supply ICs that generate codes using this format but have a 13 bit manufacturers code.

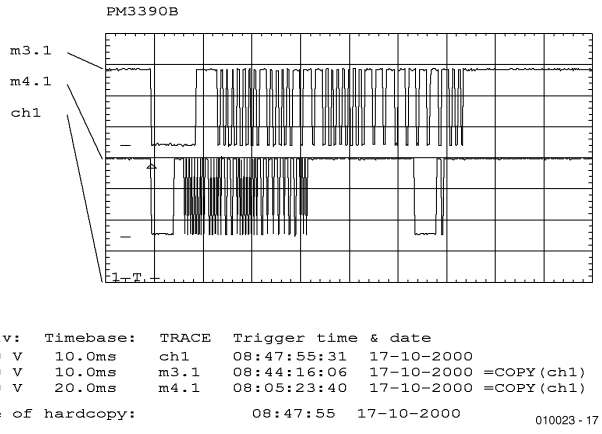


Figure 9. NEC code at the output of the TFMS5360 receiver IC

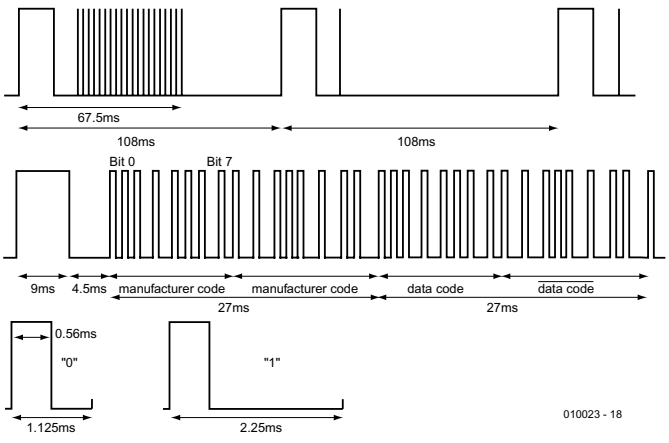


Figure 10. NEC code message format

Typical transmitter ICs remote controls:
PTPT2221, PT2222 (Princeton)

uPD6120, uPD6121 (NEC)
LC7461M, LC7462M (Sanyo)

Motorola Code

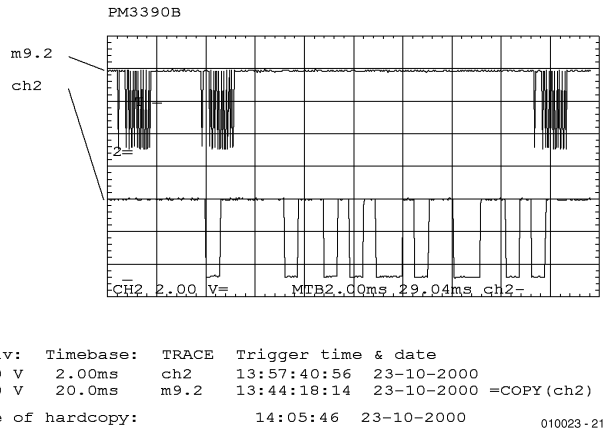


Figure 11. Motorola code at the output of the TFMS5360 receiver IC.

The Motorola code consists of a 9 bit data word. Biphase modulation is used similar to the RC5 standard but with Motorola a '0' is represented by a 512 µs pause followed by a 512 µs long high and a '1' is represented by a 512 µs high followed by a 512 µs long pause. This is opposite to the data representation found in the RC5 code. A 32 KHz carrier frequency is used.

A typical telegram has a message start header consisting of nine consecutive '1's followed by the key code of the pressed key (repeated for as long as the key is pressed) and ended by sending nine consecutive '1's. A brief key press at the controller will cause three messages to be transmitted.

Each message consists of a pre-bit, a pre-bit-pause, a start-bit and nine data bits. The pre-bit and the start-bit are always a logical 1. The pre-bit is used by the receiver to set the AGC gain level of the IR receiver.

The IC MC144105 chip is typically used to implement a remote control system using the Motorola code.

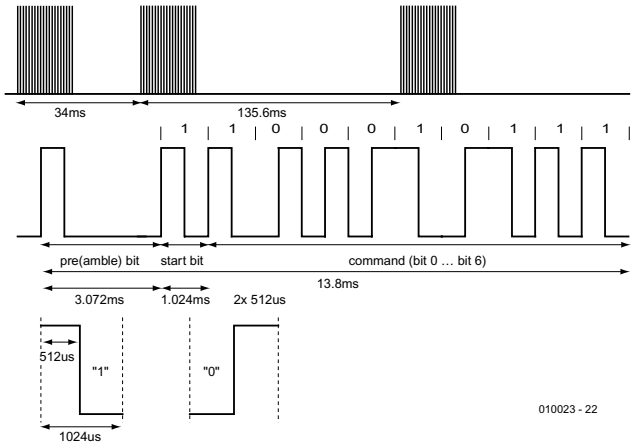


Figure 12. Motorola code message format.

Japanese Code

Similar to the way in which the RC5 code has been standardised in Europe, the Japanese Association for Electric Home Appliances has produced a standard for IR control. This system is called (uninspiringly) the 'recommended standard for infrared remote controls'.

The code is used by a number of manufacturers and has a message length of 48 Bits split into the following fields:

Manufacturers Code (16 bit)
These 16 Bits comprise the unique code for each manufacturer and are registered by the standards organisation. This code is programmed into the IC mask.

Parity Code (4 bit)
These four bits detect data corruption in the message.

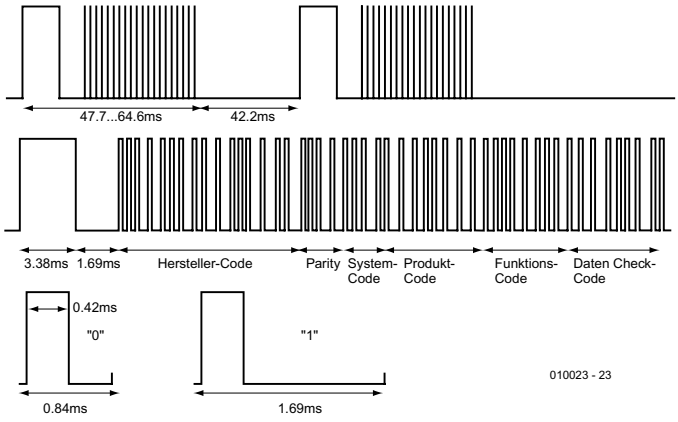


Figure 13. Japanese code message format.

System Code (4 bit)
The four system code bits are pre-programmed into the IC during manufacture.

Product Code (8 bit)
The 8 Bit Product code is made up of two mask programmed bits and 6 user wired bits. The six hardwired bits determine the equipment address.

Functions Code (8 bit)
The 8 bit function code is the value of the key pressed.

Data check code (8 bit)
These eight bits are used to detect data corruption. The system, product and function codes are passed through an algorithm to generate this check code.

A digital zero is represented by a '1' of 0.42 ms followed by a '0' of 1.27 ms. A digital one is represented by a '1' of 0.42 ms followed by a '0' of 0.42 ms.

Sanyo produce the LC7465M this device interfaces to a keypad and sends IR messages using this format.

SAMSUNG Code

The Samsung code consists of a start bit followed by a 12 Bit manufacturers code and an 8 bit command code. The message is always sent a minimum of twice.

A digital zero is represented by a '1' of 0.56 ms followed by a '0' of 0.56 ms. A digital one is represented by a '1' of 0.56 ms followed by a '0' of 1.69 ms. A continuous key press will cause the message to be repeated every 60 mS. The carrier frequency used for this code is 38 kHz.

There are no dedicated transmitter IC's for the Samsung system. A microcontroller is generally used to generate the code e.g. the data sheet of the KS51840 from Samsung gives an application for remote control use.

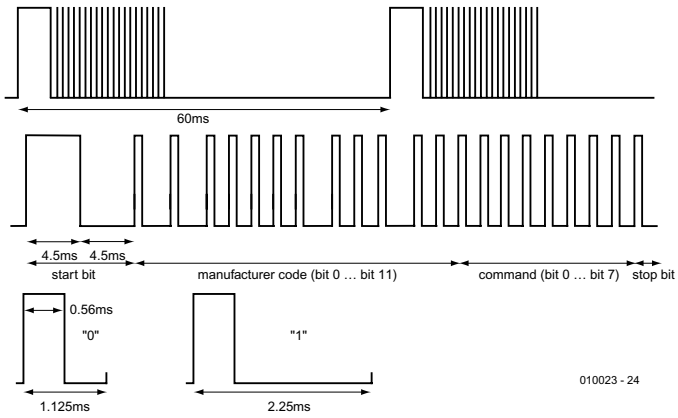


Figure 14. Samsung code message format.

Daewoo Code

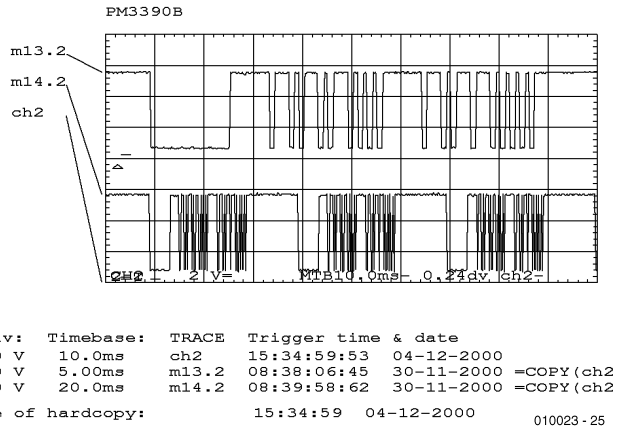


Figure 15. Daewoo code at the output of the TFMS5360 receiver IC

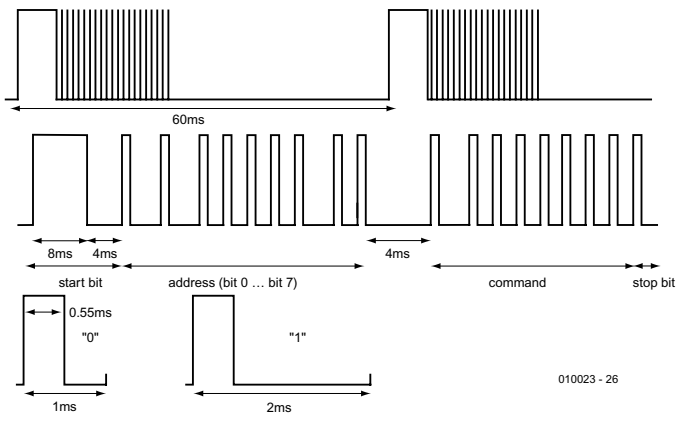
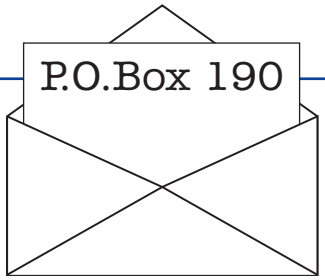


Figure 16. Daewoo code message format.

A digital zero is represented by a '1' of 0.55 ms followed by a '0' of 0.45 ms. A digital one is represented by a '1' of 0.55 ms followed by a '0' of 1.45 ms. This uses a carrier frequency of 38 kHz. A 4 ms space separates the address field from the command field.

(010023-2)

READERS' CORNER



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

Advanced LCR Meter

Dear Editor — I have just abandoned 6 weeks' intensive design work on an LCR meter with a spec similar to the above (published in three instalments, April, May & June 1997, Ed.). The idea was to get a product on the market selling in the £80-90 area, and to get this it was to have been based on a 'PIC' micro with analogue signal conditioning circuitry. I tried my best and came at it from numerous angles of attack but have failed. It is not technically possible to achieve Mr. Bonekamp's spec (there are no women in this industry?) without the use of digital processing ICs.
Mr. Bonekamp's project was a work of art. How about another version of this project using higher bit resolution chips available this past year or so. Or failing that, an article on the programming features of that design?
John Jardine,
Heckmondwike

Many thanks for your complimentary notes — the credit goes to Hans Bonekamp who unfortunately left our company about a year ago in pursuit of other interests. Sorry to disappoint you but there are no plans to update the design which, we agree with you, is to an extremely high standard. The LCR Meter as published in 1997 will certainly cost more than £80-90 to build and the ICs you mention, although technically desirable, will not make it any cheaper.

Camera on Model Train

Dear Editor — In Elektor Electronics November 2000 you describe a model train with

CORRECTIONS & UPDATES

Regenerative SW Receiver
November 2000 — 000112

The circuit diagram incorrectly shows the type TDA7052A IC in position IC1. This should be TDA7052 as stated in the parts list. The 'A' type has a built-in electronic volume control which is driven by a dc voltage at pin 4. If the 'A' type is used in this circuit, a 0.47-µF capacitor has to be inserted between pin 2 and the wiper of potentiometer P2.

Gameboy Digital Sampling Oscilloscope (GBDSO)
October & November 2000, — 990082

The capacitor identified as C1 near input K2 should be labelled C10 (18 pF). Capacitor C5 between the output and the inverting input of IC1a should be labelled C6 (15 pF).
The plugs on the measurement cables you use have to be 3.5-mm stereo types. The measurement signal has to be applied to the centre ring of the plug, while the tip and the remaining ring are connected to ground/screening of the probe. If you use a mono plug, the measurement signal is usually short-circuited.
Nintendo cartridge cases contain two moulded cams that have to be removed because one of these cams may push capacitor C25 off the board when the unit is assembled.
Finally, the author's Gameboy homepage has moved to http://www.semis.demon.co.uk/Gameboy/Gbmain.htm

Valve Preamplifier
June, July, September 2000 — 000063

On the Volume/Balance PCB, the component overlay markings R4 and R5 should be transposed. The same goes for R4' and R5'. The correct value for potentiometer P2 is 22 kΩ.

HEXFET Amplifier Upgrade
September 1995 — 950077

Spurious oscillation may occur if power resistors are used which are different from the low-inductance ones specified in the article. The following component modifications should ensure stable operation of the amplifier:
– at the solder side of the board, fit a 27 nF capacitor in parallel with R31
– connect a 20 kΩ resistor between the collector of T8 and ground
– connect a 20 kΩ resistor between the collector of T9 and ground
– replace R20 by a 1.8 kΩ type
– replace R17 and R18 by 390 Ω types
– replace R3 and R4 by 33 Ω types
On our prototype (which did not suffer from oscillation), these modifications even produced better specifications.

USB Interface
September 2000 — 000079

On disks from an early batch supplied to customers, the filenames have been truncated to a length of 8 characters. This causes a 'driver not found' error to appear during the driver installation. The problem may be corrected by renaming files as follows:
`Cypress~1.INF`, rename to `CypressSemiconductorsCYPRESS.INF`
`Thermo~1.EXE`, rename to `Thermometer.EXE`
`USBele~1.FRM`, rename to `USBelektor.FRM`
`USBele~1.VBP`, rename to `USBelektor.VBP`
Note that renaming of files can not be done on the floppy disk received from our Readers Services, because it is write-protected. You can not use XCOPY to copy our diskettes.

Emulator for 27C256 EPROM
January 2001 — 000153

In the circuit diagram, the outputs of IC4 have been shifted. Pin 12 is not connected, and the rest of the address lines has to be moved down one position (pin 13 = A8, pin 14 = A9, etc.). A15 is not used. This error does not apply to the PCB. Input pin 9, not 11, of IC8 has to be connected to ground. This requires a modification to the PCB.

an on-board camera. The article mentions the camera, transmitter and receiver as the main ingredients. Because I am under the impression that these components are supplied by Conrad Electronics, I would like to have their order code numbers. I tried a comparison with the Internet but I was unable to obtain conclusive information. The illustrations shown in your article appear to be different from those in the Conrad Electronics catalogue.

Oscar Hill, by email

You are right, The Conrad catalogue is not clear in this respect. The transmitter has order number 117455 (GigaLink Mini H0 Sender) and costs approx. DM 300 (go to www.conrad.de and enter the order number in the 'produktsuche' box) The receiver is half a Marmitek transmitter/receiver which goes by order number 888664 (Gigavideo 30), price approx. DM 200. Any one of these cameras may be used: 15000124 (b/w, DM 50), 15001424 (b/w, DM 70) or 15002624 (colour, DM 130).

USB Interface Driver for Windows 2000

Dear Editor — the driver you supply for your USB Interface project does not work under Windows 2000. I would like to present you and your readers with a suitable driver.

P. van Lith, by email

Thank you for your kind offer, which we are pleased to take up. The two files, W2KUSB.INF and W2KUSB.SYS have been compressed into a single zip file called W2KUSB.ZIP which may be downloaded from our Free Downloads page (April 2001 items).

PC Dominance?

Dear Editor — I have been a reader of your magazine since 1982 and have seen several changes in this time. The main change is the availability of the home PC. When I started, the Junior Computer was an advanced project — now we take a very high level of computer power for granted. Nowadays we can use the PC to run compilers or assemblers or even simulators for microprocessors, and in real time speed.

The problem with using the PC as a part of a project is the ever-increasing complexity of the operating system (DOS, Windows 3.1, 95, 98, 2000, NT and Linux) and the architecture of the main bus has changed several times with the introduction of new processors. The computer I am using for this letter is considered obsolete (133 MHz Pentium running Windows 98) but I feel it is still very usable for many tasks. These differing tasks mean that one task cannot dominate like in single purpose programs which can hog CPU time and input and output.

The recent articles on the Universal Serial Bus are extremely promising, as the USB device as used is a processor which can be programmed from the host computer. Please keep up this project, as it can be the core for many projects that need communication with a PC but keeping a single purpose for programming. If possible, can the required software and drivers be written with an open licence similar to Linux, to encourage development and use? This seems to be a problem with the I^2C bus as so many interface boards and adapters have been published.

The use of software in projects is now so common. Its descriptions and examples are as important as circuit diagrams and electrical description, so please show more. A software forum might be a good idea.

The Gameboy Digital Sampling Oscilloscope (GBDSO) is a project which looks very likely to be purchased as a kit only, but what a brilliant project, more like this (not too often — funds not unlimited).

There have been several bench power supplies in the past but I think the time might be right for another, Possible specification 0 to 50 V at 2 A. Good at the low voltages for 3.3 V logic and the higher voltages for audio work. If possible using a text LCD module with an MPU for the instrumentation as an option.

Peter Collins, Bostion

It is now almost impossible to be actively involved in electronics and ignore the PC or the Intenet. None the less, we try to keep a balance, see, for instance this month's Millennium Crescendo power amp, which is "100% PC-Free" as far as components, construction and use is concerned (which is not to say that we did not use a PC to design the circuit).

Being publishers only, open-licence software is really out of our league as it involves a lot of co-ordination and feedback to/from individual users. Arguably, that would rob us of the time to prepare other projects for publication!

The free downloads we supply with many (but not all) software-related projects in our magazines and books should be your key to further understanding of programming techniques. Here, the Internet is a powerful tool as it is simply impossible to print complete source file listing in the magazine — a medium-size one for, say, a PIC, would easily fill 10 pages! USB is indeed promising and you may expect further projects in the future.

The Gameboy project was our biggest hit in about 10 years' time and we are still 'enjoying' its impact.

Finally, the cost of ready-made CE approved power supplies of the spec you mention, complete with a digital readout, is now competitive to say the least with any attempt at home brewing. None the less, we should consider the fun in building it yourself, and your suggestion has been forwarded to our design staff.

London Amateur Radio & Computer Show is Moving

Thanks to the continuing efforts of the organisers, the London Amateur Radio & Computer Show retains its popularity to this day; however, with the impending closure of its original venue, RadioSport are taking the event to one of the country's most prestigious exhibition venues — Alexandra Palace.

See the web site http://www.radiosport.co.uk for details of how to get there, lectures, admission prices, etc.

an on-board camera. The article mentions the camera, transmitter and receiver as the main ingredients. Because I am under the impression that these components are supplied by Conrad Electronics, I would like to have their order code numbers. I tried a comparison with the Internet but I was unable to obtain conclusive information. The illustrations shown in your article appear to be different from those in the Conrad Electronics catalogue.

Oscar Hill, by email

You are right, The Conrad catalogue is not clear in this respect. The transmitter has order number 117455 (GigaLink Mini H0 Sender) and costs approx. DM 300 (go to www.conrad.de and enter the order number in the 'produktsuche' box) The receiver is half a Marmitek transmitter/receiver which goes by order number 888664 (Gigavideo 30), price approx. DM 200. Any one of these cameras may be used: 15000124 (b/w, DM 50), 15001424 (b/w, DM 70) or 15002624 (colour, DM 130).

USB Interface Driver for Windows 2000

Dear Editor — the driver you supply for your USB Interface project does not work under Windows 2000. I would like to present you and your readers with a suitable driver.

P. van Lith, by email

Thank you for your kind offer, which we are pleased to take up. The two files, W2KUSB.INF and W2KUSB.SYS have been compressed into a single zip file called W2KUSB.ZIP which may be downloaded from our Free Downloads page (April 2001 items).

PC Dominance?

Dear Editor — I have been a reader of your magazine since 1982 and have seen several changes in this time. The main change is the availability of the home PC. When I started, the Junior Computer was an advanced project — now we take a very high level of computer power for granted. Nowadays we can use the PC to run compilers or assemblers or even simulators for microprocessors, and in real time speed.

The problem with using the PC as a part of a project is the ever-increasing complexity of the operating system (DOS, Windows 3.1, 95, 98, 2000, NT and Linux) and the architecture of the main bus has changed several times with the introduction of new processors. The computer I am using for this letter is considered obsolete (133 MHz Pentium running Windows 98) but I feel it is still very usable for many tasks. These differing tasks mean that one task cannot dominate like in single purpose programs which can hog CPU time and input and output.

The recent articles on the Universal Serial Bus are extremely promising, as the USB device as used is a processor which can be programmed from the host computer. Please keep up this project, as it can be the core for many projects that need communication with a PC but keeping a single purpose for programming. If possible, can the required software and drivers be written with an open licence similar to Linux, to encourage development and use? This seems to be a problem with the I^2C bus as so many interface boards and adapters have been published.

The use of software in projects is now so common. Its descriptions and examples are as important as circuit diagrams and electrical description, so please show more. A software forum might be a good idea.

The Gameboy Digital Sampling Oscilloscope (GBDSO) is a project which looks very likely to be purchased as a kit only, but what a brilliant project, more like this (not too often — funds not unlimited).

There have been several bench power supplies in the past but I think the time might be right for another, Possible specification 0 to 50 V at 2 A. Good at the low voltages for 3.3 V logic and the higher voltages for audio work. If possible using a text LCD module with an MPU for the instrumentation as an option.

Peter Collins, Bostion

It is now almost impossible to be actively involved in electronics and ignore the PC or the Intenet. None the less, we try to keep a balance, see, for instance this month's Millennium Crescendo power amp, which is "100% PC-Free" as far as components, construction and use is concerned (which is not to say that we did not use a PC to design the circuit).

Being publishers only, open-licence software is really out of our league as it involves a lot of co-ordination and feedback to/from individual users. Arguably, that would rob us of the time to prepare other projects for publication!

The free downloads we supply with many (but not all) software-related projects in our magazines and books should be your key to further understanding of programming techniques. Here, the Internet is a powerful tool as it is simply impossible to print complete source file listing in the magazine — a medium-size one for, say, a PIC, would easily fill 10 pages! USB is indeed promising and you may expect further projects in the future.

The Gameboy project was our biggest hit in about 10 years' time and we are still 'enjoying' its impact.

Finally, the cost of ready-made CE approved power supplies of the spec you mention, complete with a digital readout, is now competitive to say the least with any attempt at home brewing. None the less, we should consider the fun in building it yourself, and your suggestion has been forwarded to our design staff.

London Amateur Radio & Computer Show is Moving

Thanks to the continuing efforts of the organisers, the London Amateur Radio & Computer Show retains its popularity to this day; however, with the impending closure of its original venue, RadioSport are taking the event to one of the country's most prestigious exhibition venues — Alexandra Palace.

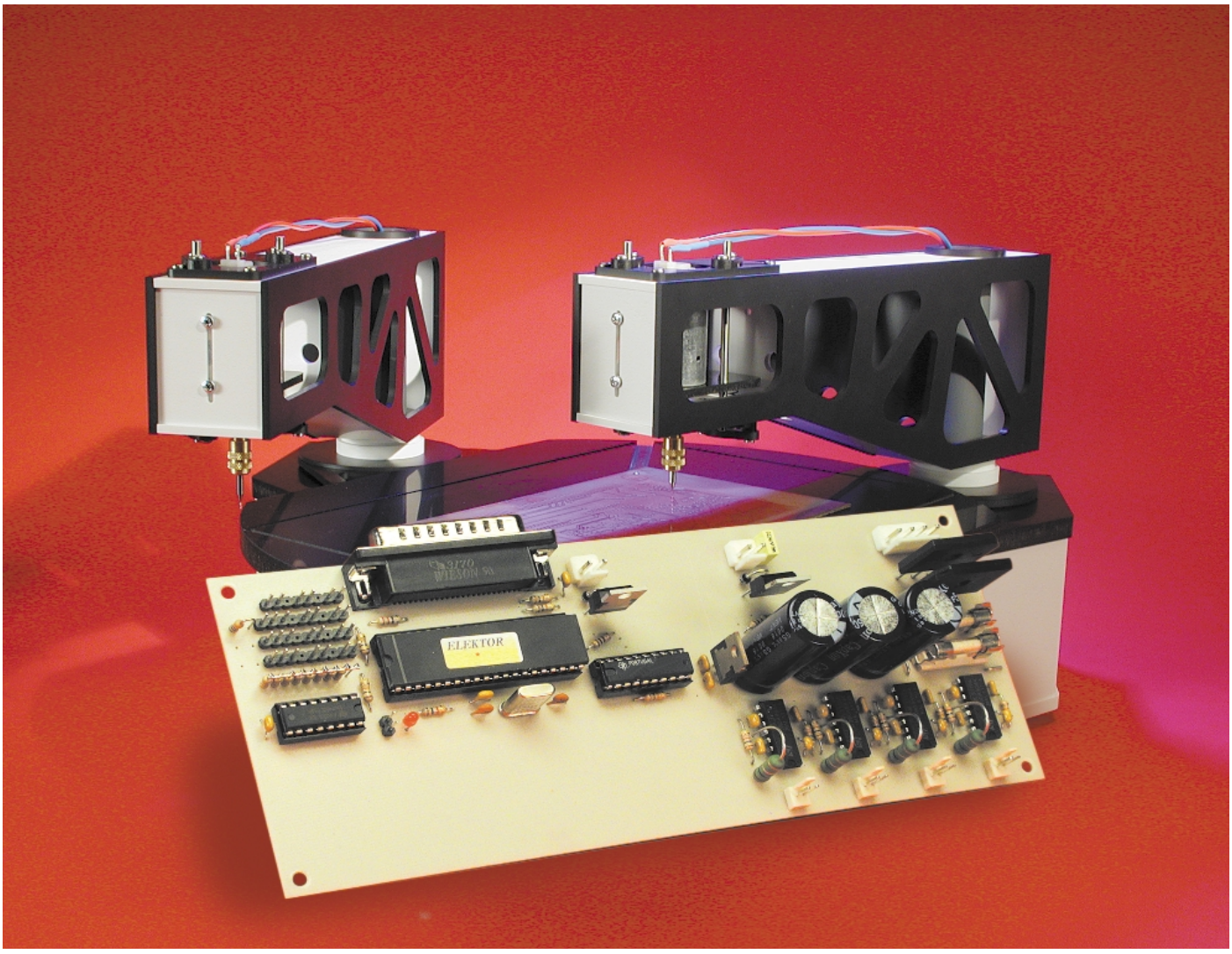
See the web site http://www.radiosport.co.uk for details of how to get there, lectures, admission prices, etc.

PCB Drilling Machine (2)

Part 2: The controller

Design by T. Müller (Radix GmbH) www.radixgmbh.de

Our PCB drilling machine provides a link between a Windows PC and the motors and solenoids of the machine itself: a microcontroller integrated with a high current drive circuit.



A universal controller board

The controller board, with its power output stages and integrated microcontroller, has been designed not just for use with the PCB drilling machine, but also with enough flexibility to be used for many other purposes. The driver stages make it a very handy controller board for other equipment or machines that employ stepper motors and other high current devices. For this reason there are a few places in the layout where we have made additions that are not strictly necessary for the drilling machine application.

To take one example, the output stages are controlled by a GAL. The GAL is simply programmed to combine the PWM signal with a two-bit address to activate the solenoid drive, and (with an enable input) the drill motors. This function could easily have been carried out using a simple address decoder such as a 74138. However, by using a GAL we can completely reconfigure the eight output stages. To this end, you will see that we have routed extra signals from the microcontroller to the GAL, in particular signals that can be assigned special functions in the microcontroller.

Series resistors and capacitors can be fitted at the inputs to the driver stages. The power MOSFETs used in the drilling machine controller do not require these; but if you wish to use bipolar transistors in the drive stages, the spaces we have left on the board will come in handy.

In the previous instalment (March 2001) we described in detail the basic principles of the CNC drilling machine. The controller is just as unconventional: the machine is not driven directly from a PC, but indirectly via a special controller circuit. For trouble-free communication between the Windows PC and the controller over the Centronics interface we need to make use of several tricks that take us beyond the standard Centronics protocol.

Let's start by taking a look at the rather straightforward hardware of the controller board. The basic circuit consists on one side of a microcontroller with a PC interface and on the other the output drive circuits, connected to I/O-port pins. Of course there is rather more to the design than that.

Real-time with Windows?

How can we control a stepper motor from a PC? It goes without saying that we need an output drive circuit capable of supplying the current required by the motor, since none of the PC's interfaces can drive a motor directly. And then, in order to make the motor step smoothly, we need very accurate timing.

At first blush it would be ideal if we could somehow connect the output stages directly to the PC and handle practically all the control signals directly as bits from the PC. The circuit would then be very simple, consisting of just a couple of latches and an address decoder.

Unfortunately, Windows has a rather unpleasant characteristic: its operations are totally asynchronous. It is impossible to say at exactly what time an event will occur. Approximate timing is possible, but as the specified time between events gets smaller and more precise, something eventually will go wrong. Windows on its own was not designed for control applications, and indeed is not suitable for them. It may be easy to use, but from the hardware point of view it is no more suitable than older operating systems such as DOS.

First of all, modern programming languages offer hardly any commands for access to the I/O-ports; and second, multitasking prevents direct hardware access by programs that attempt to bypass the operating system. Special drivers are required, such as `port.dll` found in the *Elektor Electronics* book 'PC Ports under Windows' (to be published soon).

Let us suppose that you have available a programming language that lets you control the port signals at will. We wish to generate a continuous square wave on one of the data pins of the Centronics port at a given frequency. In software this is a trivial matter: we simply need to toggle the bit in question regularly. In order to obtain the desired frequency, the appropriate delay must be used. If the delay is reasonably large, say 0.5 s, we obtain a perfect output signal with a frequency of 1 Hz.

If, however, we reduce the delay to say 0.5 ms and hence raise the frequency to 1 kHz, we hit a problem. The right frequency appears at the output, but only when averaged over a long period. The individual pulse lengths are a little longer or shorter in a random pattern, and some are very different from the specified period. Why is this?

Of course, modern processors are fast enough to toggle a bit 2,000 times per second. No, the reason is that Windows, as a multitasking operating system, has other things to do while generating our output waveform. Updating the mouse position, checking for incoming e-mail, reading audio data from a CD, communicating with peripherals: all these take processor time.

Windows is event-driven, which means that the various parts of the machine can cause the operating system to carry out operations. This can be seen when printing, for example: Windows launches a process to communicate with the printer, and supplies it with new data as and when it is ready to accept it. When that happens and how much data is transferred at a time is totally undefined, but nevertheless consumes processor time. This can interfere with the timing of our square wave and cause variations in its period.

There are two ways to overcome this problem. We could shut down all other processes and disable multitasking while we generate our square wave. Then we might as well use DOS, which is what most manufacturers of this type of control software for small machines do. Under DOS, an application program can use the full power of the processor for itself. Of course, we sacrifice the ease of use of Windows, and, in spite of its power, the computer cannot be used for anything else.

There are various operating system add-ons available for Windows which ameliorate this situation and provide quasi-real-time facilities. These are mostly VXD or TSR pro-

grams that are loaded when Windows starts up and then run in the background. These programs run briefly, but at an exact moment in time; and they run at the highest priority to guarantee having the necessary processor power available to them.

These programs are rather critical and considerably increase the chances of Windows crashing. They also permanently consume memory and processor time, because they are always present and must always be ready to spring into action when required.

These programs are also often written using undocumented features or 'back doors' in the operating system, and so are usually only fully compatible with one particular version of the system.

Offloading the synchronisation problem

We have therefore chosen an alternative approach to driving a stepper motor under Windows. Synchronisation is not left up to Windows software, but rather transferred to a microcontroller. The microcontroller receives a control command (for example for a motor pulse), and rather than executing it immediately, waits for further timing information to indicate when the data are to be processed. In this way, as long as the overall system is fast enough, we can obtain perfectly synchronised results using Windows.

The information passed to the microcontroller to generate a square wave runs as follows: bit high in 500 µs, bit low in 500 µs, Repeat. This information is sufficient for the microcontroller to produce the signal independently, requiring no further intervention from the PC. The PC must deliver the data fast enough, each command arriving within 500 µs. This should not present any difficulty for a reasonably fast PC.

To prevent the PC having to wait until the microcontroller is ready for a new command, the controller is equipped with a small FIFO memory with space for 40 entries. This allows the controller to be well decoupled from the PC. The PC can send off 40 precalculated commands for switching the various output signals to the controller and then has 40×500 µs to prepare new data or carry out other tasks.

Shortly before the FIFO empties, this state is flagged to the PC, which can then refill the FIFO with new data. This does not affect the controller, which continues executing commands one after another with perfect timing.

Using the Centronics interface

Communication with the controller could be carried out over any PC interface. The Centronics and V24 (serial) interfaces are always available, and are particularly suitable candidates.

There are certain problems associated with the V24 interface: getting the wiring right and arranging handshaking can be full of pitfalls. Matters are also more complicated for the microcontroller, which must first convert the serial bit stream into parallel bytes. And, if the microcontroller does not contain a UART, the controller must detect and interpret the start bits itself. This consumes processor time in the controller and ultimately creates difficulties in ensuring timely execution of commands.

For these reasons, we have chosen the Centronics interface. This interface is also superior from the point of view of speed, because eight bits are transferred at a time: this also makes the data easy to process. The data are simply written to a specific I/O address and the job is done. It is also convenient for the microcontroller to receive the data in a simple 8-bit format.

Eight bits may be a convenient number, but it is too few to specify a command with timing information. Multiple bytes must be assembled together to construct a useful command so that the controller can know what to do with the data. For our application, two bytes are enough for each command, but that is too much to transfer in one go over the Centronics interface. But when the 16 bits are divided into two independent bytes, a synchronisation problem can arise.

The strobe-edge trick

How can the controller know which of the bytes is the first and which the second? Counting is one option, but not a very reliable one: a single transmission error and all successive commands will be wrongly interpreted. The error must be detected and the controller reset. Alternatively, the data bytes could include further information to allow the controller to identify which is which.

However, this also has its disadvantages. The marker bits naturally reduce the amount of real information transferred, and so we have fewer bits available for commands.

For this reason we distinguish the bytes externally using a means already provided within the Centronics interface. Take a look at the protocol as shown in **Figure 1**. First consider normal operation, described at the top of the figure. The interface, however, will let us transfer two bytes at a time without modification. How that is achieved is described at the bottom of the figure.

The handshaking process is unchanged: we have simply changed the meaning of the signals. Two bytes are coalesced into a single packet, where the falling edge of the STROBE signal marks the first byte, and the rising edge, the second.

If an error occurs when the first byte is already in the process of being sent, STROBE will eventually rise, if only because of the pull-up resistor fitted to the controller board. The transfer is then terminated, and the pair of bytes remains together. Synchronisation problems are therefore impossible, even though the contents of the two bytes will be in error. If on the other hand we had simply counted bytes, then all subsequent bytes would be interpreted wrongly, as they would all be interchanged. If we were to continue in this manner, the controller would cease operating correctly hence the system would have to be re-initialised.

At start-up the control software sends a RESET command, which causes the controller to clear its FIFOs and bring to a halt all movements in progress in the system.

Controller functions

We have now described the main functions of the controller: accepting the 16-bit wide instructions, storing them in the FIFO, and synchronously executing the commands stored in the FIFO. Of course, the calculation of trajectories is not handled by the microcontroller, since it is not designed for trigonometric calculations and would be too slow. It would also imply programming the dynamic characteristics of the system once and for all into the microcontroller.

Instead, the movements are pre-

processed in the PC and then sent to the controller in the form of primitive instructions such as *motor 1 one step counterclockwise*. All movements are defined in the PC in this way. It is possible to compute the step information for any desired motion and then send it to the controller: and thus the microcontroller never needs to be reprogrammed.

Data format

Now that we have explained how the command data are calculated in the PC and transferred over the Centronics interface to be synchronously processed by the controller, we can take a look at to the internal structure of the command data.

The two bytes are concatenated to form a word. This word contains a data value, or operand, along with an address which determines how the operand is interpreted: in other words, the particular command to be executed. The number of address and data bit is variable, the number of data bits shrinking as more address bits are required. A command with a short address field can contain a longer operand, and vice versa. This kind of addressing scheme can be processed using a sequence of branches, decoding the command using the 'divide and conquer' principle.

This works as follows: the first address bit determines whether we are dealing with one particular type of command (when the bit is a '1') or not (when it is '0'). In the second case we still do not know what type of command we are processing, so we examine the second bit. If this is a '1', then we have decoded a second class of commands, while if it is '0', we proceed to examine the third bit, and so on. Of course, as we continue to examine bits is this way, there remains less and less space for the operand data.

The advantage is that we only need one address bit for the first command type, and so we have a full 15 bits available for the operand data. For commands whose address field is, say, three bits long we have 13 bits available. Note that a three-bit address field allows at most eight different commands to be separately decoded.

1. Quiescent condition:
STROBE from PC to microcontroller is high. BUSY from microcontroller to PC is low.

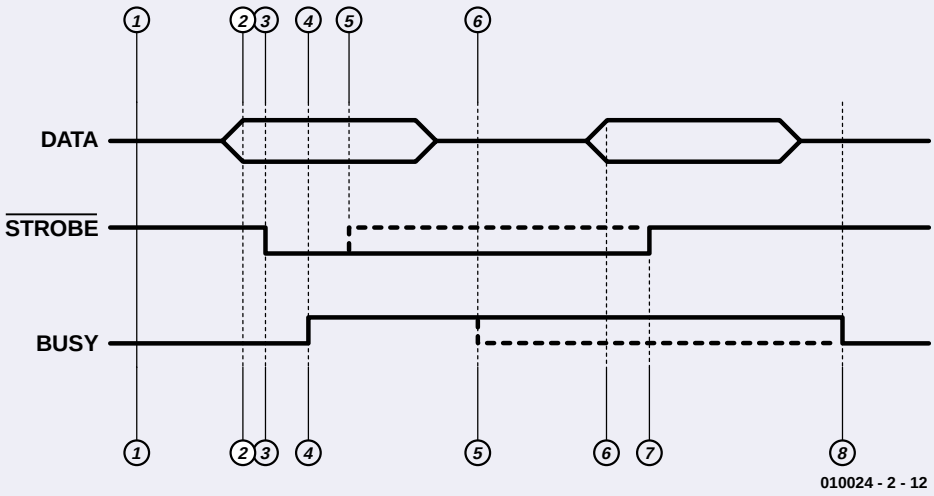
2. The PC wishes to transfer data as usual, and puts the data byte on the eight data lines.

3. The PC takes STROBE low. The connected microcontroller (or peripheral device) sees the low state and knows that the data are ready to be read.

4. The microcontroller takes the data byte and sets BUSY high in acknowledgement.

5. The PC takes the STROBE signal high again.

6. When the microcontroller has processed the byte, it indicates this state by taking the BUSY signal low. This completes the transfer of a byte. The system is ready to transfer another byte, the control signals (BUSY and STROBE) being once more in their original states.



1. Quiescent condition:
STROBE from PC to microcontroller is high. BUSY from microntroller to PC is low.

2. The PC wishes to transfer data using the special mode, and puts the data byte on the eight data lines.

3. The PC takes STROBE low. The microcontroller sees the low state and knows that the data are ready to be read.

4. The microcontroller now takes the data byte and sets BUSY high in acknowledgement.

5. The microcontroller processes the byte immediately and leaves the BUSY signal high.

6. The PC observes that the BUSY signal is high and therefore that the first byte has been accepted. It then puts the second byte on the data lines.

7. The PC indicates that the second byte is present by taking STROBE high again. This causes the microcontroller to accept this byte also.

8. When the microcontroller has finished its processing, it sets BUSY low again.

Figure 1. The original Centronics protocol (above) and as modified for the drilling machine (below).

The variable-length address field also allows a higher priority to be assigned to more common or more urgent commands. For a rarely used command, or one that initiates a lengthy action, we can allow the address decoding to take a little longer.

The commands which are most commonly used, and which sent to the controller at the fastest rate, are without doubt the stepper motor commands; they also require a long operand, which is provided by the highest priority command format. The format of the commands for sending stepping pulses to the motors is set out in **Table 1**.

The second command class is the drilling command. Three address bits can be used for this command without causing any difficulty because its execution time is much longer than the motor step commands, and because it requires a shorter operand. The meaning of the various bits in the drilling command is set out in **Table 2**.

These two commands suffice for basic operation of the PCB drilling machine. In fact there is a range of other commands which are less relevant to our discussion and which are not described here.

Brawn and brains

At first glance the circuit diagram in **Figure 2** may look like an impenetrably complex piece of electronics, but on closer inspection we see that the brains of the circuit are actually very simple. The intelligence is concentrated in a type PIC16C64 microcontroller from Microchip, clocked at 20 MHz. This microcontroller boasts the magnificent total of 33 individually addressable I/O-ports, but only 2 K of EPROM program memory and 128 bytes of internal SRAM, here used for (among other things) the FIFO memory. The PIC16C64 is equipped with peripherals such as a real-time clock, timer/counter and capture/compare inputs. In this application we do not use the 3-wire

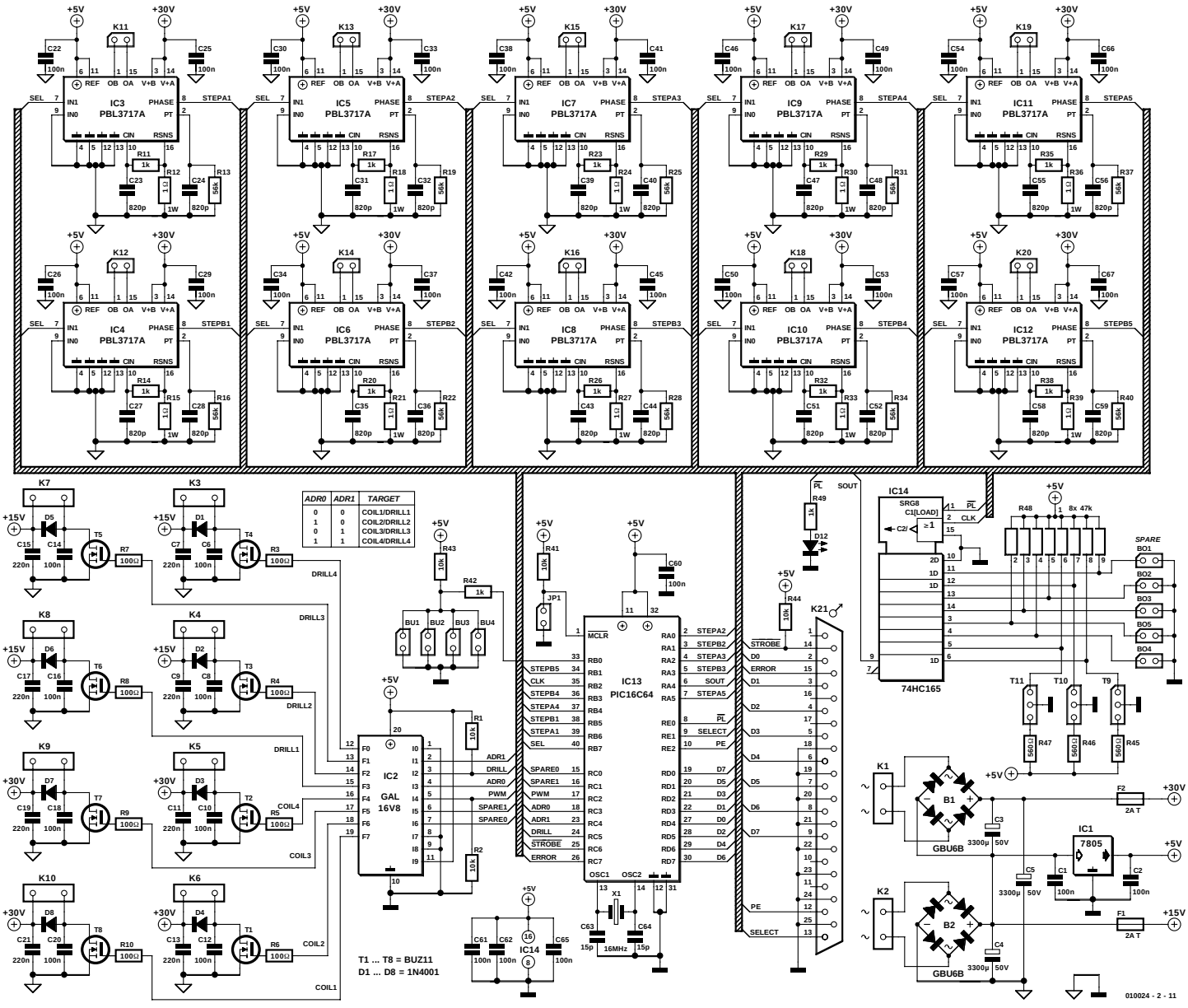


Figure 2. The drilling machine controller board: lots of brawn, and a little bit of brains!

Table 1: Motor pulse command

MSB	Byte1				Byte2				LSB
1	Z	Z	Z	D	T	T	T	Z Z Z Z	Z Z Z Z

Addressing: bit 7, byte 1 = '1'.

Z (11 bits): 'Time'
The time value specifies how long AFTER the execution of the motor pulse command the controller must wait before executing the next instruction. If the value is zero, then the next command (assuming it is also a motor pulse command) is executed immediately and synchronously with the present one.

T (3 bits): 'Target'
These three bits select the stepper motor drive stage affected by the command. The order runs from Target 1 ('000'), the output stage driving connectors K11 and K12, through to Target 5 ('100'), the output stage driving connectors K19 and K20.
Target addresses 6, 7 and 8 ('101', '110' and '111') are reserved.

D (1 bit): 'Direction'
This bit specifies the direction in which the motors are to turn. D=1 specifies clockwise rotation, D=0 counterclockwise.

Table 2: Drilling command

MSB	Byte1							Byte2				LSB
0	T	T	T	1	1	B	B	B B B D	D	D	D	D

Addressing: bit 7, byte 1 = '0'; bits 2 and 3, byte 1 = '11'.

T (3 bits): 'Target'
These three bits select the output stage for the drilling command. The machine can be fitted with up to four arms, and each of these arms has its own drill and a dedicated driver stage on the controller board. The targets are numbered from left to right, starting with Target 1 ('000') at connector K10, where the solenoid for the first arm is connected. The corresponding drill motor is connected to K8. Target 4 ('011') is driven from connectors K5 (solenoid) and K3 (drill motor). Targets 5 to 8 ('100' to '111') are reserved.

D (5 bits): 'Drilling force'
The force applied to the drill by the solenoid can be controlled by this five-bit data value. The value sets the amuont of energy delivered to the solenoid, and can be set to any number in the range 0 to 31.

B (5 bits): 'Braking control'
When the drilling head is withdrawn the solenoid cannot simply be switched off, since this would cause long-term wear on the mechanism. To prevent undue stress on the motor guide spring, the magnetic field in the solenoid is gradually reduced, so that the drilling head is withdrawn gently.
In principle this value can be set in advance for a particular weight of drilling head. Here, however, we have the possibility of optimising the value. If, for example, you use a more powerful drill motor or a 3-jaw chuck instead of the collet chuck, the weight of the drilling head may be significantly different.

synchronous serial SPI/I²C bus.

The modified Centronics interface uses port D (data signals), RC6 (active-low STROBE), RE1 (SELECT) and RC7 (ERROR). The BUSY signal from the microcontroller is not wired to the standard connection on pin 11, but to the PAPER EMPTY signal (PE) on pin 12. There are several reasons for this: first, it prevents the PCB drilling machine from springing into action when a print job is started under Windows, and second, it allows a printer to operate in parallel with the machine. Further, pulses on the STROBE and BUSY signals appear to affect the internal interrupt processing of Windows: and that is best left well alone.

R44 is the pull-up resistor referred to above that pulls STROBE high in the event of a system crash, thus deactivating the signal.

The microcontroller determines the configuration of the drilling machine via the drilling head limit switches connected to **Bo1-Bo4**. If a drilling arm is fitted and the head is in the 'up' position, the corresponding switch is closed. If an arm is not equipped with a limit switch, the corresponding contacts remain open.

OptSpare is an optical input that, like the switch input Spare, is not used in the present design. **OptBase** is an optical input used to calibrate the reference positions of arm 1, arm 2 and the rotating table. **OptAdd** does the same job for arms 3 and 4. How exactly the calibration is carried out will be described in a later article in this series. The LED can be used to indicate that the machine is operating: it lights during drilling and during the calibration process.

It may seem that 33 I/O-ports are plenty, but unfortunately are not enough for all the signals just described. We therefore use a shift register type 74HC165 (IC14) to combine the bits into a single byte read serially by the microcontroller over a single port bit RA4 (SOUT). A parallel load pulse loads the bits into an intermediate register where they are buffered before being clocked out using the CLK signal (RB2).

The final item of information read by the controller is the 'stop' signal from the solenoid limit switch. These switches are connected to BU1-BU4. Since the current must be switched off quickly, it is not feasible to read these signals out slowly via a shift register. Instead, the stop signal drives the interrupt input RB0 low. Since only one drilling head is actuated at a time, the four limit switches can be connected in parallel.

The rest of the circuit consists of the four identical output drive stages for the solenoids (T1, T2, T7 and T8), the drill motors (T3-T6) and ten identical integrated stepper motor

drivers type PBL3717A from ST Microelectronics. The datasheet for this IC can readily be found on ST's website at www.st.com.

First let us consider the drill motors and solenoids. These are not driven directly, but rather via a programmable logic device type GAL 16V8. You can read the reasons for this in the box *A universal controller board*. Address signals ADR0 (RC3) and ADR1 (RC4) select from the at most four motors and solenoids in the order shown. The level on DRILL (RC5) selects either the motor (when RC5 is low) or the solenoid (when RC5 is high). In this way we make sure that in the case of a failure in the circuit the drill will not turn and the drilling head will be up. When the circuit is switched on, the PIC's ports are in a high impedance state, and resistors R1 and R2 hold the signals at suitable levels to ensure that the motors and solenoids are safely turned off.

The controller activates the selected motor/solenoid pair using port pin RC2. The pin does not just turn on and off, but provide a current control using a PWM signal, different for the solenoid and for the motor.

The solenoids are controlled using a rather complicated current waveform. The timing depends on the instantaneous position of the drilling head (which is determined from the two switches on the head guide), the value specified in the drilling command, and on whether an error situation arises due, for example, to a timeout resulting from a jammed or dirty guide. The important fact is that the PWM control value is calculated at each point in time to actuate the guide solenoid as smoothly as possible.

The solenoid coils are rated for continuous operation at 12 V, and they are driven (although only momentarily) from the same +30 V supply as the stepper motors. This produces an enormous magnetic field to force the drilling head down. As soon as the appropriate Bo switch opens, the PIC can deduce that the solenoid has moved and it then proceeds to regulate the current to the value specified in the drilling command using the PWM signal. Although the solenoid is initially overdriven, in continuous operation it is only driven at effectively 12 V with an 80% duty cycle. This is within the specified limits. Overall, with typical activity, the duty cycle is only a few percent, and in practice the coils only get warm to the touch.

The drill motors are controlled during switch-on and running using a PWM ramp that reaches 100% duty cycle during drilling.

The stepper motors are activated using the STEP signals. Each PBL3717A integrated driver controls one motor winding, and two drivers (STEPAx and STEPBx) are allocated to each motor. The drivers are responsible for producing the drive pulses and for converting the +5 V logic levels to +30 V, the operating voltage of the stepper motors. The microcontroller activates the driver via the level (repsectively from STEPAx and STEPBx) at the PHASE input, and a current then flows from OA to OB. Inputs IN0 and IN1 have a special function: they allow control of the output stage drive current.

This is normally achieved using a retriggerable timing circuit for each output stage. If no more motion is required of the motors, the pulses stop and the current is reduced in the motors. Here, this is realised via the signal from RB7 (pin 40) connected to the IN1 pins of all the PBL3717s. When motion is required, all the motors are driven with full current (even those which are not to move), by setting IN1 = 0. Shortly after the motion stops, IN1 is set to 1 again, and the current drive is reduced to 19%.

Why do we increase the current to the motors not involved in a motion? Partly because in this way the power supply is loaded to the same extent independent of the motion, and partly because, as a result of various resonances in the system, those motors driven at 19% of the maximum current can be jogged out of position. Although this is rather unlikely, it would lead to errors almost impossible to track down.

(010024-2)

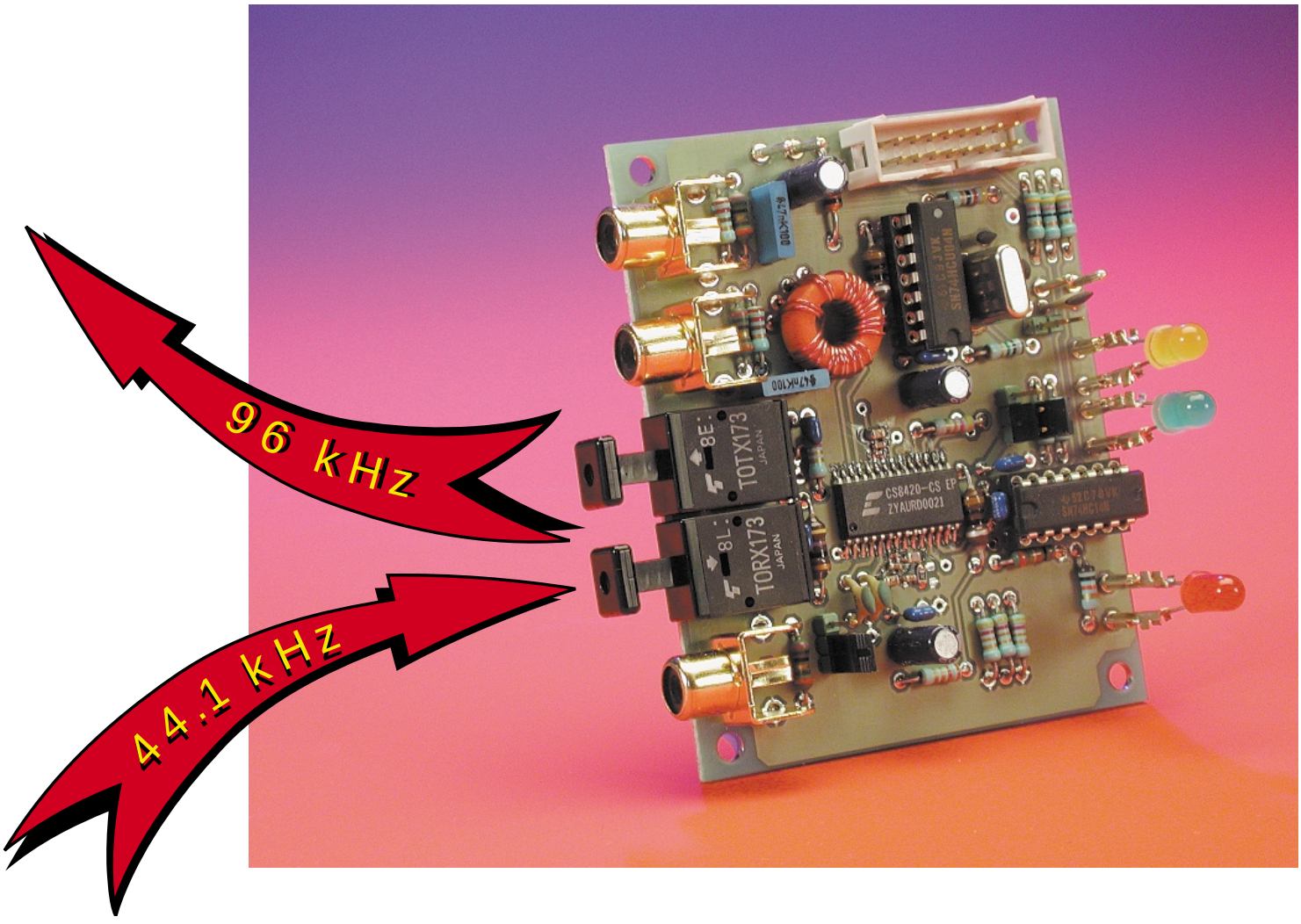
In the next instalment of this series we get down to the construction of the controller board as well as the nuts and bolts of the project. The various parts of the PCB drilling machine will be brought together, with illustrations — and lots of glue and grease!

96 kHz Sampling Rate Converter

for high-end digital audio

Design by T. Giesberts

This converter is specifically intended to upgrade somewhat dated, digital signal sources. It recognises all common sampling rates and converts these to 96 kHz. By design, this converter is ideally suited to be combined with our 'Audio DAC 2000'. However, it may also be used as a stand-alone unit.



Specifications

- stand-alone
- 24 bit I/O
- 32 kHz to 96 kHz input sampling-rate range (8 kHz to 108 kHz using external OMCK)
- 1:3 and 3:1 sampling-rate ratio between in- and output possible
- 128 dB dynamic range
- THD+N > 117 dB
- I^2S-output (alternative receiver for *Audio DAC 2000*)
- coaxial input
- optical input
- two coaxial outputs
- optical output
- choice of Master or Slave mode

Quality conscious audiophiles have, in all likelihood, already replaced the standard D/A-converter in their equipment with an up-to-date 24-bit/96-kHz model, such as the 'Audio DAC 2000', published in the November and December 1999 issues of *Elektor Electronics*. The next logical step is to upgrade the audio installation so that the sampling rate of the (digital) signal sources is converted to 96 kHz before presenting it to the D/A-converter.

The up-sampling converter described here makes this possible. It accepts all imaginable sampling frequencies and has been designed in such a way to allow it to be used in three different ways. If there is sufficient space inside the enclosure it may, for example, be built into an existing CD player. Alternatively, it may be fitted into its own enclosure and used as a stand-alone device. Alternatively, it can be combined with the 'Audio-DAC 2000' by simply substituting it for the receiver section. The latter has the added advantage of not requiring an additional power supply since this may be obtained automatically from the Audio-DAC.

Sense or nonsense?

The fact that the combination of up-sampling-converter and Audio DAC results in a very universal D/A-converter that can deal with all common sampling rates will be appreciated by most readers as a significant feature. This requires no further debate.

However, a question that will be asked by some is: what is the sense in preceding a D/A converter by a 96-kHz-up-sampling-converter when the actual signal sources under consideration do not use a higher sampling rate than 44.1 kHz (CD) or 48 kHz (DAT)?

In the first instance this appears to be a valid argument indeed, but up-sampling has, in this case, real advantages. Of course, by converting from 44.1 to 96 kHz no additional information is gained from the CD. However, the increased number of steps creates an 'intelligent' rounding of the original steps. Consequently, the digital signal has increased resolution with the result that the D/A-converter can better process this signal without the need for very steep filters.

The result is a measurable 'cleaner' signal that also, according to the true audio fanatic, clearly sounds better. Whether everyone will hear the difference is questionable but it is obvious that up-sampling deserves the benefit of the doubt.

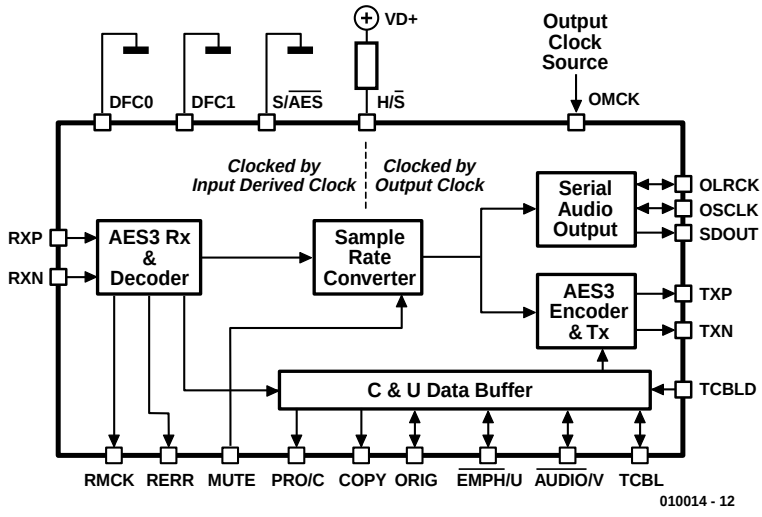


Figure 1. Block diagram of the CS8420 in hardware mode 1a.

The heart

The converter presented here consists essentially of a single IC: the integrated sample rate converter plus AES3 tranceiver type CS8420 from Crystal. This IC is intended for 16-, 20- and 24-bit applications where the input sample rate is unknown or clearly asynchronous with the desired sampling rate. The overall functioning of the IC relies on the conversion of the input sample rate to a very high frequency, which is subsequently divided to obtain the required value. This value is defined by the frequency of the master clock input.

At the input of the CS8420 there is the choice of AES3 or three wire serial format. At the output, both formats are available. The control of the data stream between input and output may be controlled with an external microcontroller. However, it is also possible to do without such a controller by using one of the six hardware modes built into the IC. In our case the CS8420 is used in hardware mode 1a. **Figure 1** shows the internal block diagram when the IC is in this mode. As can be observed, the AES3-receiver (RXP/RXN) has been utilised here and the converted signal is fed to both an AES3-encoder and a serial audio port.

The various pins of the CS8420 have different functions and names depending on the hardware mode. To avoid confusion, only those pins that have a function and designation related to hardware mode 1a are shown on the complete schematic for the up-sam-

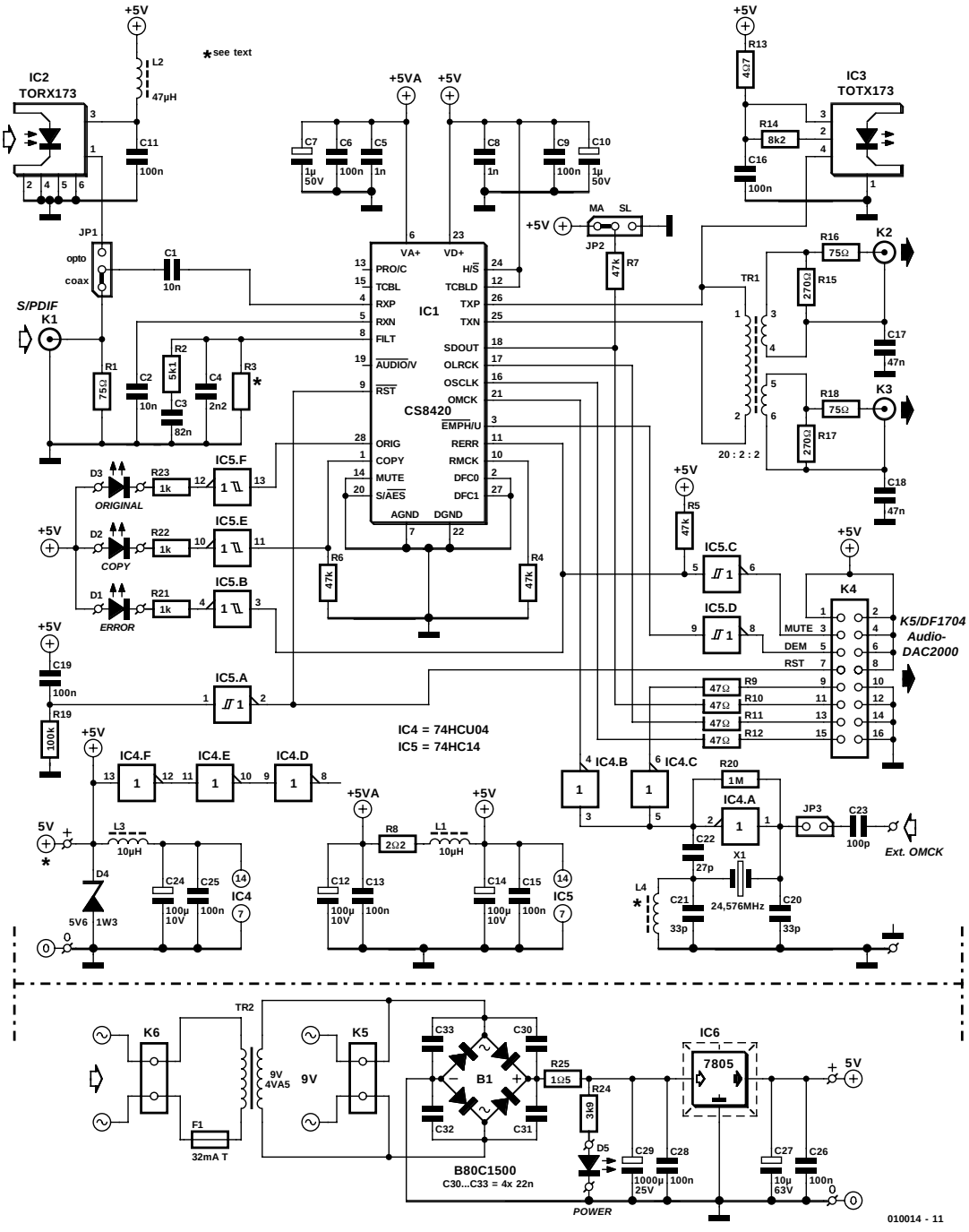


Figure 2. With the exception of the CS8420 the converter hardly consists of any active components.

pling converter. Those of you who would like to know more about the IC, or are curious about the other modes, are referred to the datasheet for the IC and application note AN159 available from the manufacturer's web site at http://www.cirrus.com.

The complete schematic

The practical schematic of the up-sampling converter is shown in **Figure 2**. It is quite clearly laid out and makes it immediately obvious that the amount of hardware required is within reason.

Inputs and outputs

This circuit permits the input signal to take the form of either a coaxial signal (K1) or an optical signal (IC2). The latter is a standard optical Toslink receiver module that we have used on previous occasions. The selection is made by the appropriate location of jumper JP1. The idea of using a real switch was rejected because this selection is normally made only once.

The PLL on the input of the CS8420 has an external filter network (C3/C4/R2/R3) that may be modified for other frequencies, if desired (refer to the application note for details). The values shown in the schematic give the network a frequency range from 32 kHz to 96 kHz (R3 is not fitted).

The AES3-outputs are routed via a wind-your-own transformer to two (electrically isolated) coaxial outputs. The output grounds are decoupled with C17 and C18 to suppress RF interference.

In addition, an optical output is provided by IC3. Its input signal comes from the symmetric output (TXP).

The serial audio output port (SDOUT/OLRCK/OSCLK/OMCK) is used, via K4, to route the correct signals to the 'Audio-DAC 2000'. For this purpose, K4 is connected with a 16-way ribbon cable to K5 on the DAC PCB. The double bandwidth (DBW) feature is than always active and the Bessel filter in the 'Audio-DAC 2000' is activated.

The mute function is derived, via inverter IC5c, from the receiver-error-indicator (RERR). Several channel-status-bits are also connected to the output pins. One of these is the emphasis bit that signals the Audio-DAC via IC5d whether pre-emphasis (50/15 µs) has been applied or not.

The reset signal is the same as the one for the CS8420, which is generated by the circuit consisting of R19-C19-IC5a. The format of the digital audio output port is I^2S-compatible. Therefore, digital filter DF1704 has to be configured for this format.

Master-clock

The frequency of the OMCK signal defines the output sample rate with the condition that the sample rate is smaller by a ratio of 256 compared to OMCK. To obtain 96 kHz a master-clock of 24.576 MHz is necessary. The required clock generator is constructed around IC4a and X1. Pay careful attention to the C_{load} of the crystal used, because this has to be equal to C20/2 (where C20 = C21). C22 compensates the propagation delay of buffer IC4a and has to approximately equal to C_{load}.

The crystal has to be cut for fundamental frequency and parallel resonance. Commonly available crystals are intended for series resonance. If a series resonance crystal is used the oscillator may run at the wrong frequency, or not work at all. There is also the possibility that the crystal is designed for parallel resonance but has been cut for 3rd harmonic resonance (a.k.a. third overtone). To adapt the oscillator to a crystal of this type it is necessary to fit an additional inductor (L4), which, together with C22, is tuned to a frequency just below the harmonic in order to suppress the fundamental. L4 will typically have a value of 1.5 or 1.8 µH, but it may be necessary to experiment a little with this.

As an additional feature, JP3 provides for the possibility to supply OMCK from an external oscillator (input 'Ext.OMCK'). This makes any frequency between 8 kHz and 108 kHz possible, provided, the ratio of the input sample rate and the output sample rate is between 1:3 and 3:1. C20…C22, X1 and L4 have to be removed in this case. And don't forget to fit a jumper on the pins of JP3, of course!

Adjustments

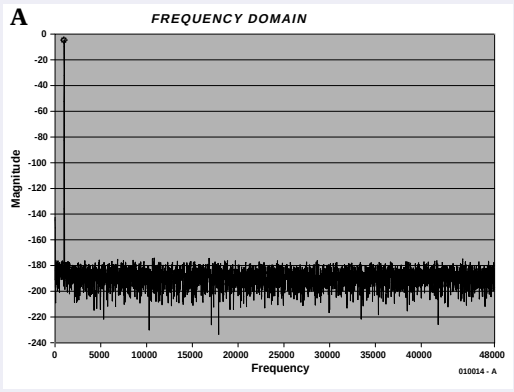
Four pins define the operating mode of the IC: hardware mode is selected by making H/S 'high'. The state of DFC0, DFC1 and S/AES further define which one of six hardware modes is chosen.

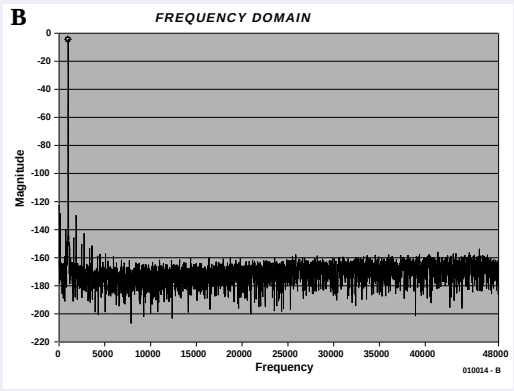
Some Test Results

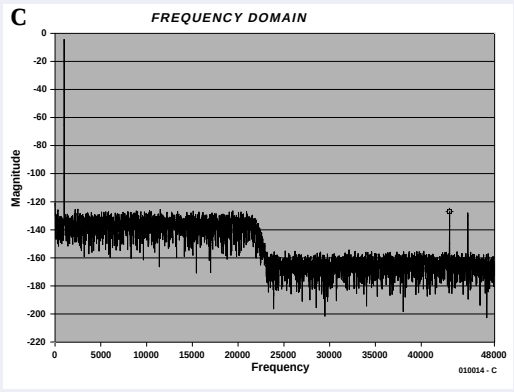
A few measurements have been performed in the digital domain with the help of special measuring equipment ('CDBCapture + board') that clearly show what the up-sampling converter does.

Figure A shows the output of a DVD player with a 24 bit, 1 kHz test signal at a sampling rate of 96 kHz. This signal is passed through the converter and the result is shown in **Figure B**. The only visible irregularities are a few (non-harmonic) frequency components which are probably the result of the up-down-sampling process and added dither. The latter explains also why the noise floor is a little higher. The signal to noise ratio of this signal (measured digitally!) amounts to 120 dB. This is much lower than is possible with a 24-bit-D/A-converter (physical limit, noise, etc.)

Figure C shows the result of a typical application. The 1 kHz signal from the test CD with a sampling frequency of 44.1kHz is now increased to a sampling frequency of 96 kHz. It is clearly visible that the noise floor is considerably higher with a 16-bit signal and that the bandwidth is limited to 22.05 kHz (by the converter). The mixing products of the original sampling frequency have now become visible because of 24-bit resolution. These will disappear completely in the noise at the D/A-conversion stage. The signal to noise ratio is a little more that 98 dB, the exact same value as measured at the digital output of a CD-player.







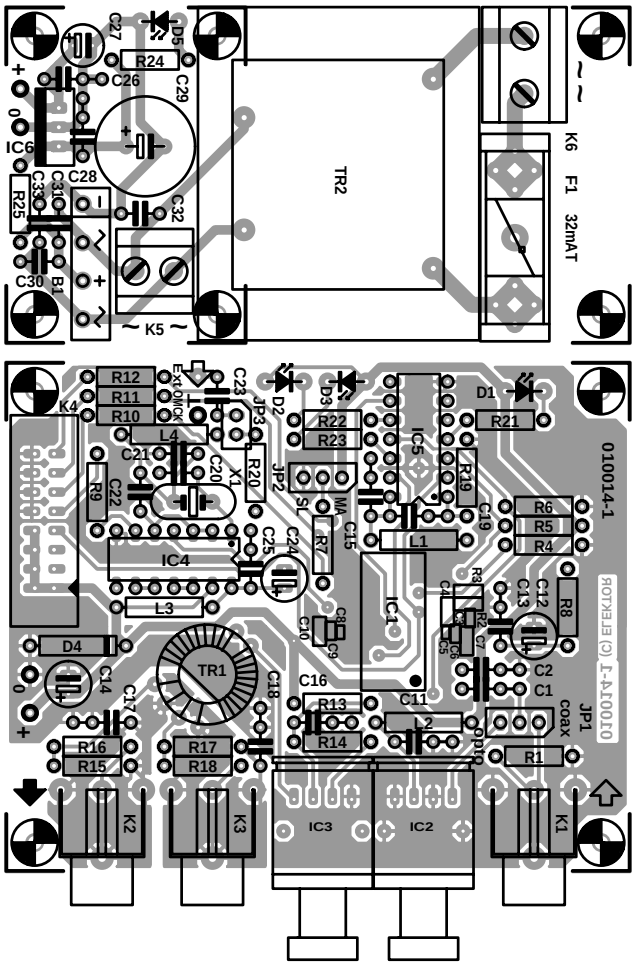
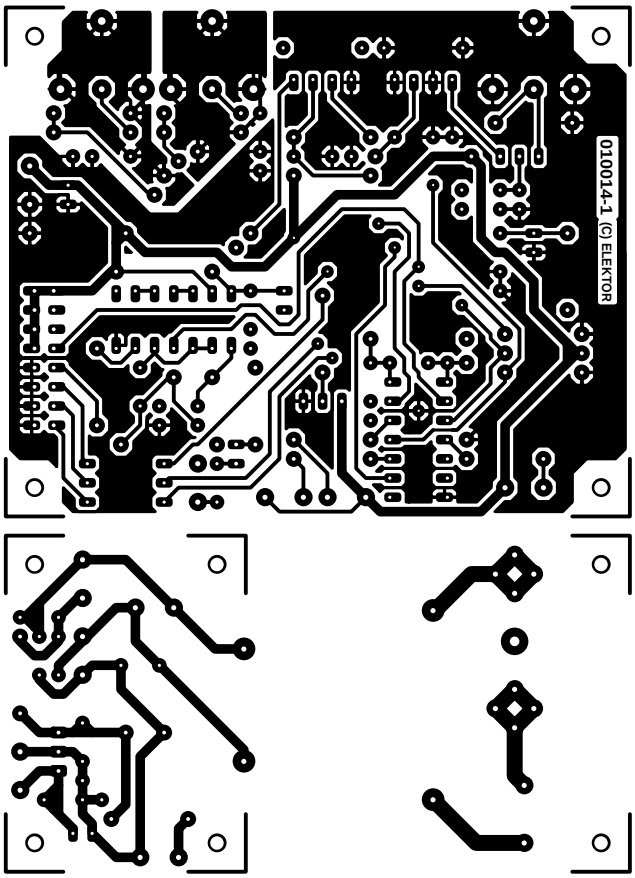
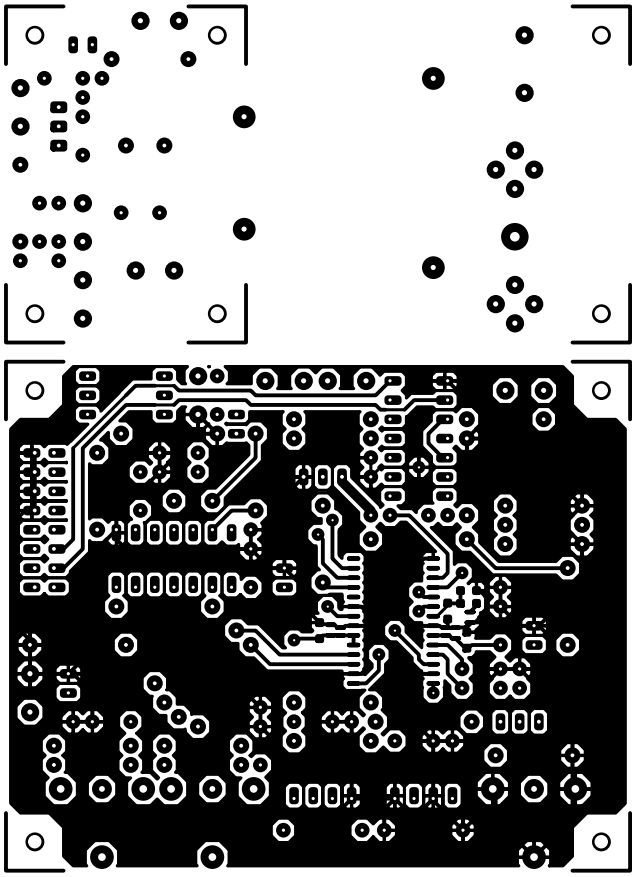


Figure 3. The PCB for the converter is double sided.
The (single sided) power supply may be separated with the aid of a saw.





Pull-up- and pull-down resistors on various pins (start-up option pins) adjust several options more. Of relevance here are Copy (Copy Channel Status Bit Output), RMCK (Input Section Recovered Master Clock Output), RERR and SDOUT (Serial Audio Output Port Data).
The pull-down on Copy causes the CS8420 to operate in the aforementioned mode 1a (refer **Figure 1**), and EMPH, PRO, AUDIO, COPY and ORIG are defined as outputs.

A pull-up on RERR and a pull-down on RMCK select the serial output port output format OF2: I^2S, 24-bit data. The logic level on SDOUT defines whether the serial output port is 'master' or 'slave' (pull-up for master). Jumper JP2 selects between master and slave. This allows multiple circuits to operate synchronously. This is mostly intended for digital mixers. (Further information may be found in the data sheet for the CS8420 as to which connections to K4 are required for this.)

MUTE is not used. IC5b, IC5e and IC5f drive LEDs D1…D3 which respectively indicate error, copy and original bit of the input signal. In this mode the transmitted channel status bits are a copy of the input channel status bits. The U- and V-bits are 0. TCBLD is connected to +5 V, which makes TCBL (Transmit Channel Status Block Start) an output, but this pin is not actually used.

COMPONENTS LIST

Resistors:
R1,R16,R18 = 75Ω
R2 = 5kΩ1, SMD (Farnell # 771-429, case size 0805)
R3 = see text
R4-R7 = 47kΩ
R8 = 2Ω2
R9-R12 = 47Ω
R13 = 4Ω7
R14 = 8kΩ2
R15,R17 = 270Ω
R19 = 100kΩ
R20 = 1MΩ
R21,R22,R23 = 1kΩ
R24 = 3kΩ9
R25 = 1Ω5

Capacitors:
C1,C2 = 10nF ceramic
C3 = 82nF, SMD, (Farnell # 301-9550, dielectric X7R, case size 0603)
C4 = 2nF2, SMD (Farnell # 578-290, dielectric NPO, case size 0805)
C5,C8 = 1nF, SMD (Farnell # 301-9380, dielectric X7R, case size 0402)
C6,C9 = 100nF, SMD (Farnell # 432-210, dielectric X7R, case size 0603)
C7,C10 = 1µF, SMD (Farnell # 317-627, dielectric NP0, case size 0805)
C11,C13,C15,C16,C19,C25,C26,C28 = 100nF ceramic
C12,C14,C24 = 100µF 10V radial
C17,C18 = 47nF, MKT (Siemens, Electrovalue)
C20,C21 = 33pF
C22 = 27pF
C23 = 100pF
C27 = 10µF 63V radial
C29 = 1000µF 25V radial
C30-C33 = 22nF, ceramic

Inductors:
L1,L3 = 10 µH
L2 = 47µH
L4 = 1.5µH or 1.8 µH (see text)

Semiconductors:
B1 = B80C1500 (rectangular case, 80V piv, 1.5A peak)
D1,D5 = high-efficiency LED, red
D2 = high-efficiency LED, yellow/amber
D3 = high-efficiency LED, green
D4 = zener diode 5.6V, 1.3W
IC1 = CS8420-CS (Crystal, 28-pin SOIC; Atlantik Elektronik, Germany)
IC2 = TORX173 Toshiba (Conrad Electronics)
IC3 = TOTX173 Toshiba (Conrad Electronics)
IC4 = 74HCU04
IC5 = 74HC14
IC6 = 7805

Miscellaneous:
JP1,JP2 = 3-way pinheader + jumper
JP3 = 2-way pinheader + jumper
K1,K2,K3 = RCS (line) socket, PCB mount, (e.g. Monacor/Monarch type T-709G)
K4 = 16-way straight boxheader
K5,K6 = 2-way PCB terminal block, lead pitch 7.5 mm
X1 = 24.576 MHz quartz crystal (C_{load} = 20 pF, parallel resonance)
F1 = fuse, 32 mAT (time lag) + PCB mount fuse holder
Tr1 = ferrite core, Philips type TN/7,5/5-3E25, primary 20 turns, secondary 2x2 turns ECW 0.5 mm dia. (26 SWG)
Tr2 = mains transformer, secondary 1 x 9 V/4.5 VA (e.g. Monacor/Monarch type VTR 4109)

Power Supply

If you combine the converter with the 'Audio-DAC 2000' then a power supply is not necessary. The circuit is then automatically powered from the connection between K4 and the Audio-DAC connector K5.

A simple regulated power supply is provided for stand-alone use, consisting of a standard 7805 voltage regulator (IC6). The outputs 0 and + of this power supply are connected to the pins of the same name on the converter PCB. The mains is connected to connector K6.

An unusual feature of this power supply is the possibility to connect to K5 a mains power adapter or other 9 V power source. Tr2, F1 and K6 are omitted in this case.

LED D5 functions as the essential on/off-indicator.

Construction

As you will have come to expect from us, we have also designed a nice (double-sided) printed circuit board for this up-sampling converter. **Figure 3** shows that it has been designed in such a way that it effectively consists of two parts. It is possible to separate the power supply by simply cutting the PCB. If you don't make use of the power supply it may well come in handy on some future occasion when another application requires a regulated 5 V supply.

The components on this circuit board have deliberately not been placed too close to each other. This makes the construction relatively straightforward, with the exception of IC1 and surrounding components. The CS8420 is an SMD component, the handling of which requires more than average skill. A steady hand and a soldering iron with a small tip are indispensable. It gets worse, to some extend, with regard to the capacitors around the IC. With excellent decoupling and small physical dimensions of the PLL network (required to avoid jitter) in mind, small to very small capacitors are used. The detailed picture of **Figure 4** makes it clear that if you are inexperienced with a soldering iron it is better not to attempt to solder these.

Another issue that requires explanation is output transformer Tr1. You have to wind this transformer yourself, but this is not a big

deal. The core is a small toroid with a primary winding of 20 turns and two secondary windings of 2 turns each. The windings are made with 0.5 mm diameter enamelled copper wire (approx. 26 SWG). The 20 primary turns are wound first and are evenly distributed around the circumference of the core, allowing for space in the centre for the four secondary turns.

Figure 5 shows a clear view of the fully built-up circuit board. There is no harm in carefully comparing this with your circuit board before using the converter.

Fitting the converter in an enclosure is a reasonably straightforward job since this has been taken into account when laying out the PCB. It is easy to mount the PCB so that the input and output connectors protrude through the rear panel. Short wires are attached to the status LEDs D1, D2 and D3 and mounted to the front of the enclosure. Otherwise, the essential wiring is limited to connecting the power supply to the converter and connecting K6 to an appropriate mains entry device with power switch. Be careful and observe all electrical safety practices when wiring this. Finally, stick an identification label with the number of the PCB, the power supply voltage and the rating of the fuse on the enclosure.

The job is really simple if the converter is combined with the 'Audio-DAC 2000'. In this case, it suffices to swap the 'old' receiver PCB with the converter PCB and to connect the ribbon cable to K4.

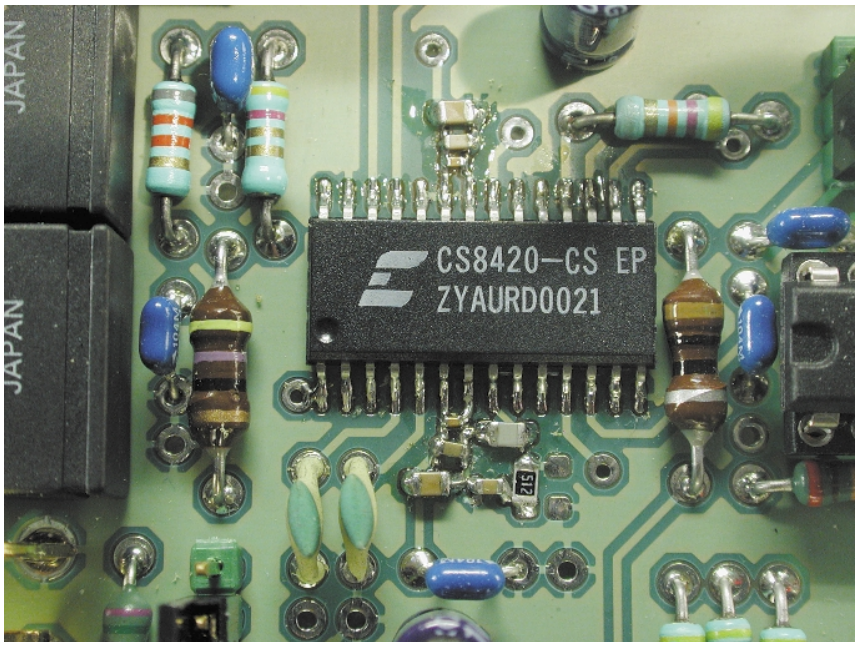


Figure 4. The mounting of IC1 and surrounding SMD capacitors requires considerable soldering skills.

Oops...

Unfortunately, the CS8420 exhibits a small cosmetic flaw. If the input signal is briefly interrupted during use it is possible for the IC to enter the wrong operating mode. The clear audible consequence of this is that the output signal is significantly mutilated. The only remedy is to briefly switch the power supply off and on. It is not likely to happen much in practice, but it is still a little untidy on the part of the manufacturer...

(010014-1)

Note
We extend our gratitude to Unique Memec GmbH, Germany, for making available specialized test equipment from Crystal.

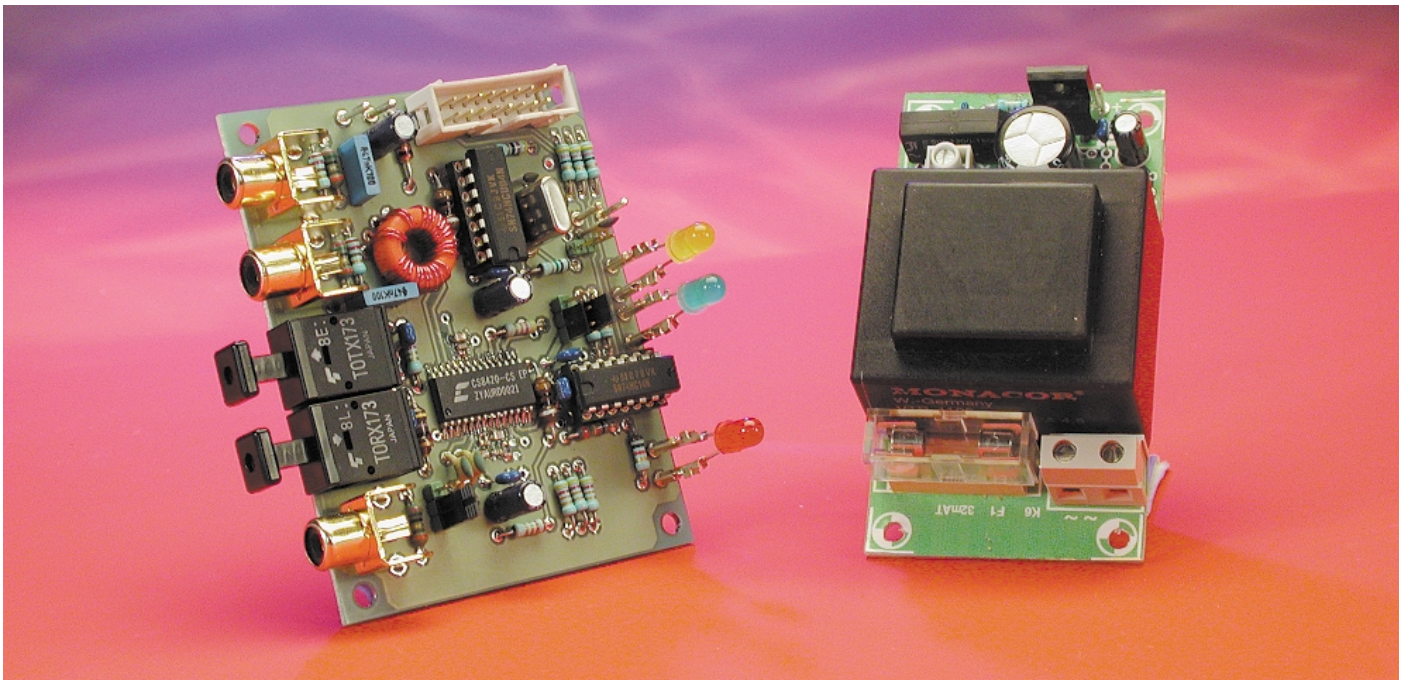


Figure 5. This photograph gives a good impression of the finished boards.