ELEKTOR ELECTRONICS

FEBRUARY 2003
£3.45

THE ELECTRONICS & COMPUTER MAGAZINE

www.elektor-electronics.co.uk

20/40-MHz Logic Analyser

expandable and with PC control

Autoranging Capacitance Meter

ECC83/12AX7 Microphone Preamplifier

Chess Computer

Active Subwoofer

Soundcard Tuning

Electronic Hourglass

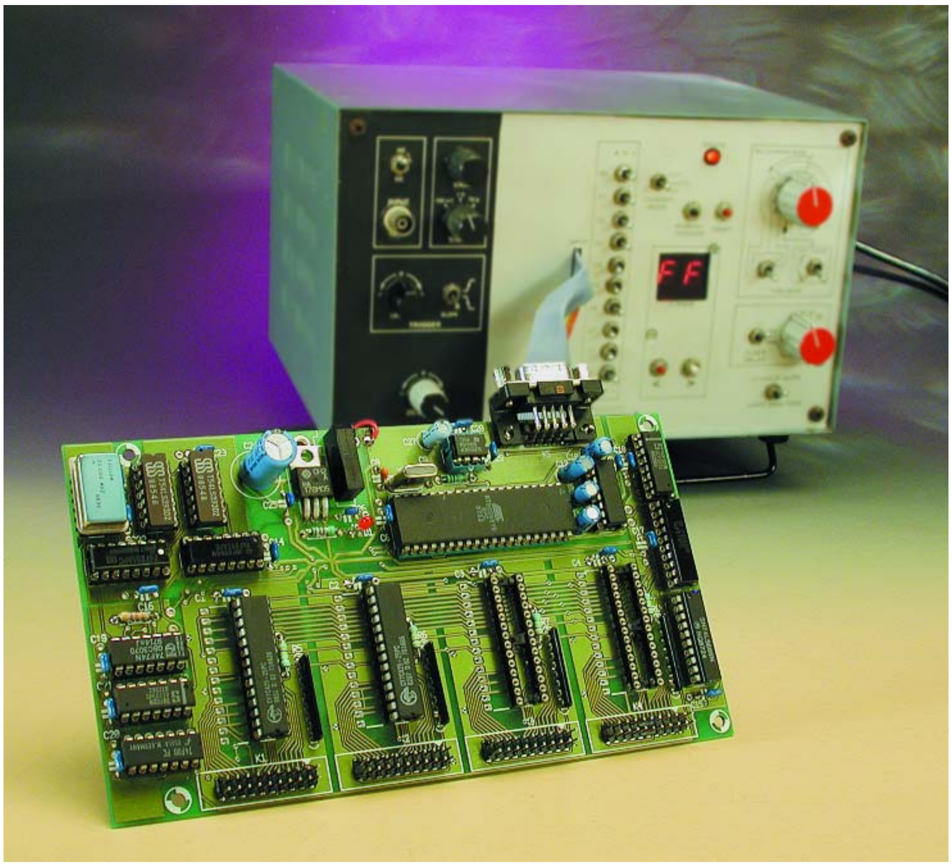
9 770268 451005 02

20/40 MHz Logic Analyser

with expansion capability and PC control

Design by B. Bouchez

With analogue electronics, an oscilloscope is usually sufficient for solving any problems encountered during development or troubleshooting. With digital electronics, things are not so simple. It is often necessary to observe several different logic levels at the same time, and on top of that, high working frequencies are quite common. A logic analyser is clearly the proper tool in such a situation.



A normal oscilloscope is not particularly suitable for working with circuits containing microprocessors, DSPs or other digital components. The bandwidth is usually too small for the extremely fast signals used with such components. Furthermore, it is usually not possible to view more than two signals at once (in the best case, four signals can be viewed). Also, an analogue oscilloscope can only measure repetitive signals, which is not very helpful with digital signal streams.

Although oscilloscopes (including digital storage oscilloscopes) have come into quite general use in recent years, the same cannot be said of logic analysers. They remain expensive items of test equipment, especially for hobbyists.

What does a logic analyser do?

We should first review how a logic analyser works, if only to demonstrate its utility in comparison with

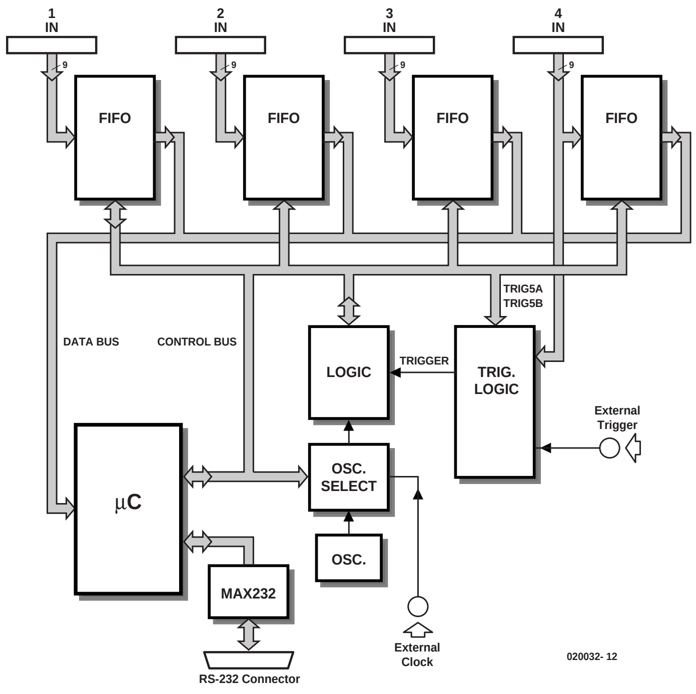


Figure 1. The functional structure of a logic analyser requires a complex logic circuit, which can be considerably simplified by utilising the counters integrated into FIFOs.

that of an oscilloscope.

A logic analyser records digital signals received simultaneously via a certain number of input channels. These signals only have values of '0' or '1', and they are stored in groups of eight (or nine, in the case of this design). In contrast to an oscilloscope, a logic analyser records only a single cycle (comparable to the one-shot triggering mode of an oscilloscope).

The speed at which a logic analyser stores data (which can be compared to the horizontal sweep rate of an oscilloscope) is determined by an internal or external clock. On each rising (or falling) edge of this clock, the states of several input channels are simultaneously sampled and the sampled values are stored.

Another characteristic property of logic analysers is that sampling can be triggered either manually or only after certain (predefined) conditions have been satisfied. The analyser can thus be forced to become active only when the correct binary data are present on specific inputs. With an oscilloscope, trigger control is limited to detecting a certain voltage level on the input with a certain polarity ('trigger level').

Finally, most logic analysers can also be used in the pre-trigger mode. In this mode, data samples are continuously written to the memory of the logic analyser. After the trigger event occurs, the logic analyser continues to store data for a certain number of clock cycles. This makes it possible to determine what happened **before** the trigger event.

Thanks to these characteristics, logic analysers are indispensable tools for designers who what to know what is happening on an address bus, in a circuit containing a microprocessor or in other types of logic circuits.

FIFOs

After reading about the characteristics of logic analysers, you must have already realised that the heart of a logic analyser consists of RAM components that store the data obtained from the logical processes connected to the inputs so that it can be read out as necessary.

Long-time readers of *Elektor Electronics* may remember the logic analyser project described in the June 1981 issue (an example can be seen in the background in the picture at the head of this article). This was probably the first example of a logic analyser that was affordable for hobbyists. It was a monumental circuit, consisting entirely of LSTTL ICs.

The problem that we face when designing a logic analyser is that it must not only generate the control signals for the memory devices, it must also increment the address lines after each sample has been taken. This leads to a complicated circuit design. For this task, 'classic' logic analysers used extremely fast counters driven by an internal or external clock. By interleaving a number of counters and separate memory banks, it is possible to sample logic signals at speeds far greater than the rated speeds (access times) of the memory components.

Here we want to keep the design of the addressing logic simple (no interleaving). Also, it takes a fairly large number of ICs to build counters using discrete devices, since address multiplexers are needed in addition to the counters to allow the data to also be read out.

In order to make our job as simple as possible, here we use a type of memory device that comes with built-in counters: a FIFO. A FIFO (first-in, first-out) memory has a rather complicated structure, as can be seen from **Figure 2**. In contrast to normal memory components, which have a bidirectional data bus, FIFOs have two separate busses, one for writing and the other for reading, with data being input via the one and output via the other.

You will also notice that there is no address bus, at least not external to the IC. The addresses are instead generated internally by two independent counters, one of which is used for writing and the other for reading. When the FIFO is reset, both counters are initialised to a value of '0'.

If we want to store data in a FIFO, we must place the data on the Write bus and then take the $\overline{\text{WR}}$ line Low. The data will then be stored in the RAM location indicated by the Write pointer. If we make the $\overline{\text{WR}}$ line High again, this counter will be incremented. This process is repeated each time a sample is taken.

On the read side, exactly the same process takes place. Each time the $\overline{\text{RD}}$ line is made Low, the data indicated by the Read pointer are placed on the other (Read) data bus. The read counter is in turn incremented when the $\overline{\text{RD}}$ line again goes High.

These write and read operations can occur at completely different and unrelated speeds, which is why this type of device is called an asynchronous FIFO.

To illustrate how such a memory works, **Figure 3** shows what has happened inside a FIFO after four write cycles. The first four memory cells have been filled with data in the order in which they were received (in this example, '$AA' arrived first, then '$55', then '$00' and finally '$FF'). At this moment, the processor connected to the read port pulls the $\overline{RD}$ line Low. Since the read counter has a value of 0, the read bus outputs the value '$AA' (the first value received). After this, the read counter is automatically incremented when the $\overline{RD}$ line goes High again (at the end of the read cycle). The read counter now has a value of '1', so the processor receives the value '$55' in the next read cycle. This was the second value to be stored in the memory. The read counter is repeatedly incremented in this manner.

As can be seen, the data emerge on the read output in the same order in which they arrived at the write input. This explains the name given to this type of memory: first-in, first-out.

Now suppose the FIFO receives a new Write command while data are being read out. The write counter has reached address 4. The data will thus be stored at this address without in any way disturbing the processor that is reading data from the FIFO.

In this example, the read counter will have the same value as the write counter as soon as the processor reading the data has completed five read cycles. There is then nothing more to be read, since the FIFO is empty. This status is automatically signalled by the $\overline{FE}$ (FIFO Empty) line, which goes Low in such a situation.

Naturally, the opposite situation can also occur: the write counter can catch up to the read counter if the memory is emptied more slowly than it is filled. In this case, there will be no more room in the memory, since the FIFO will be full. In this case, the write counter is automatically inhibited until the read counter again starts moving, which indicates that memory locations have become available. The $\overline{FF}$ (FIFO Full) goes Low to signal this status.

It's clear that FIFOs contain a considerable amount of logic, and this considerably simplifies their use. FIFO memories are usually used as buffers between two processors that must exchange data at different rates without mutual synchronisation.

The internal counters roll over automatically when they reach the ends of their ranges (which does not mean that the memory is full, as explained below). Unfortunately, when variable-length data blocks are being exchanged, it is not possible to determine the location of the start or end of a block from the contents of the counters. That is why FIFOs have a 9-bit structure instead of an 8-bit structure. Although it is not always needed, this extra bit can be used to mark the end of block so it can be easily recognised by the processor.

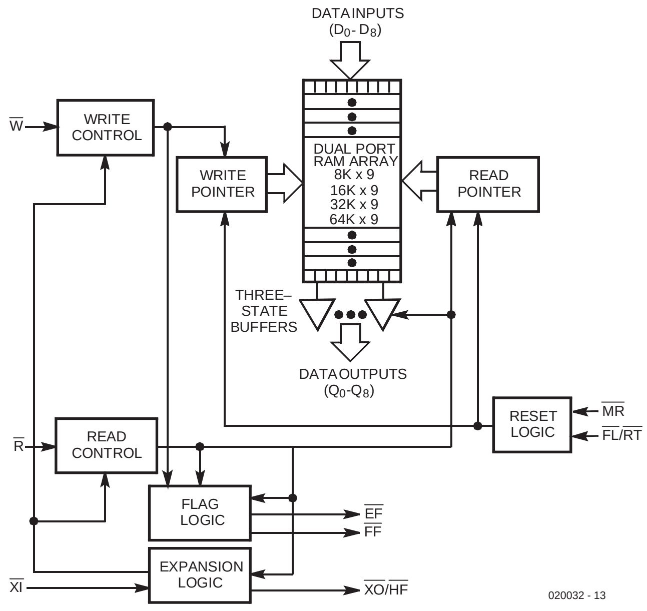


Figure 2. The internal structure of a FIFO memory (source: Cypress).

The circuit

Now that you know now FIFOs work, let's look at the schematic diagram of our logic analyser. It should be a lot easier to understand when you are aware of what FIFOs do.

As can be clearly seen from **Figure 4**, the circuit is built around four such ICs. Incidentally, you do not necessarily have to fit all four of the FIFOs. The circuit will work perfectly with one, two, three or all four FIFOs installed, which is why we say that it has 'expansion capability'.

The memory capacity of different types of FIFOs also varies, from 1 Kword (the 'bytes' here are nine bits

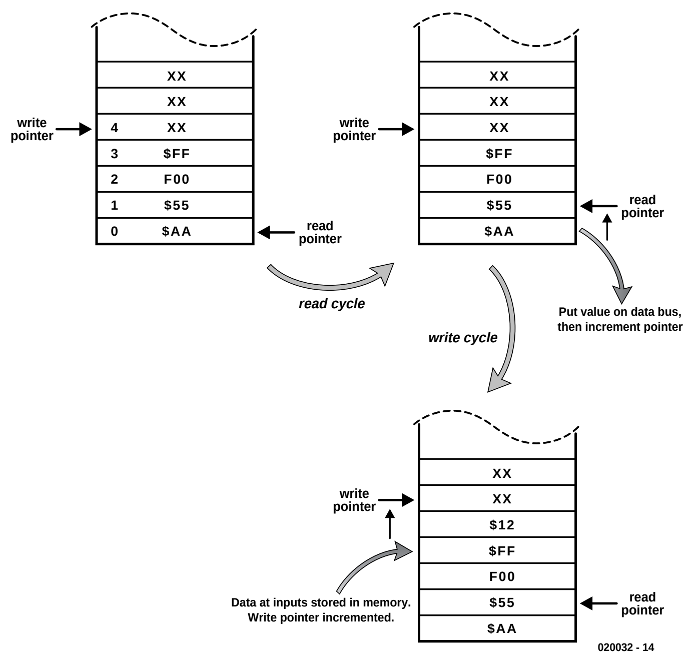

write pointer
4 XX
3 $FF
2 F00
1 $55
0 $AA
XX
XX
read pointer
read cycle
write cycle
Put value on data bus, then increment pointer
$12
Data at inputs stored in memory. Write pointer incremented.
020032 - 14

Figure 3. The working principle of an asynchronous FIFO: the read and write cycles can be executed independently.

long) to 16 Kwords. All of them have the same pinouts (although the same cannot be said of their prices…). With our flexible design, it is thus possible to build a logic analyser with 9 to 36 channels. That's good news for your bank balance!

Finally, the logic analyser can be built in two different speed versions: 20 MHz and 40 MHz. We have more to say about this later on.

As you have already read, the write counters form part of the FIFOs. Although they cannot be seen in the schematic diagram, they do require a clock signal (on the $\overline{\text{FIFO}}$ $\overline{\text{WR}}$ line). This clock signal is generated by a programmable clock generator consisting of a crystal oscillator (IC19) and several binary dividers (IC8 and IC9). The sampling clock is selected using two multiplexers (IC10 and IC12), which are controlled by the microcontroller via the CLKSx lines. One special feature is that the last input of IC10 is not connected to the binary dividers, but instead to an external connector (EXT CLOCK), allowing clock frequencies not available on the circuit board to be used.

The clock signal reaches the FIFO via IC6a. On each falling edge of SYSCLK, the FIFOs store the values of the signals on their input buses. These signals enter via connectors K1–K4. The characteristics of these inputs are described further on.

As already noted, the clock signal passes through IC6a. This gate is in turn controlled by the ACQ ACT line. The ACQ ACT signal is generated by a flip-flop (latch) formed by IC7a and IC7b. This latch can only be set when the output of IC5b goes High. There are two ways in which this can be made to happen: either the output of IC5b can be set High via the $\overline{\text{START}}$ line (the option used for manually starting data acquisition), or the GO SYNC line can be set High. In the latter case, IC5b will only change states when there is activity on the TRIG line. The latter option is used for automatically starting data acquisition.

Flip-flop IC5b can also be reset to '0' by the microcontroller to prevent restarting. This does not affect the circuitry around IC7.

As long as IC7 remains inhibited, a clock signal will be present and the FIFOs will record data samples. However, at a certain point the FIFOs will be full, since the microcontroller does not read the FIFOs while signals are being sampled. As we have seen, the $\overline{\text{FF}}$ line will then become active. This signal is inverted by IC6c, causing IC5a to change state on the next clock edge. Its Q output will go High, releasing the inhibition of IC7 via the ACQ DONE line. This also signals the microcontroller that the sampling cycle is completed. The ACQ ACT line goes Low, preventing the clock signal from reaching the FIFOs, so they stop taking data samples (once a FIFO is full, it will anyhow refuse to store any new data).

The microcontroller can then take as much time as it wants to read the contents of the FIFOs, which contain the sampled data, at its own rate. As we will see later, the microcontroller only has an 8-bit bus for reading data from the FIFOs, while the sampled data can be up to 36 bits wide. The microcontroller simply selects each FIFOs in turn using the $\overline{\text{RAM0RD}}$, $\overline{\text{RAM1RD}}$ and $\overline{\text{RAM2RD}}$ lines. The ninth bit of each FIFO, which is used here as an extra data bit, is read via a separate microcontroller I/O line.

The final subject is triggering. As you already know, there are two triggering options: manual triggering, in which case the analyser is started immediately, and automatic triggering. In the latter case, a rising edge must be present on pin 11 of IC5 to start sampling The microcontroller indicates that this edge is valid by setting pin 12 of IC5 High.

This rising edge can come from several different sources, which are selected using multiplexer IC14. The first source is simply the EXT TRIG connector. This input allows the logic analyser to be triggered by the rising edge of an external signal. The EXT TRIG input is also connected to another input of IC14 via a gate wired as an inverter (IC7d), so it is also possible to trigger the analyser on the falling edge of an external signal.

Another option is internal triggering. In this mode, the analyser must detect a predefined combination of bits on several of its inputs. In professional logic analysers, it is possible to select individual inputs and their associated states. Since we want to keep our logic analyser simple, the number of inputs is limited to eight and masking is not possible.

Internal triggering is achieved by comparing the value of the first eight inputs of K4 to a value stored in IC13 by the microcontroller. If the bit pattern on the inputs matches the bit pattern stored in IC13, the output of IC11 goes High. This output can be selected as the trigger source via IC14.

If the signal on pin 5 of IC14 is used for internal triggering, the analyser will be started as soon as the proper binary pattern (matching the stored pattern) appears on K4. However, if the inverted signal is used (the signal on pin 6 of IC14), the analyser will instead start when the specified pattern (again compared with the value in IC13) is no longer present at K4.

Note that there is a 'work-around' for the absence of a configuration mask for the triggering condition. The inputs on K4 (as well as those on the other connectors) are pulled High by a resistor network. To cause certain inputs to be ignored, simply leave them unconnected and program a '1' for the corresponding bits of the trigger pattern byte.

The last (but certainly not least important) part of the circuit is the local processor (microcontroller). The task of the local processor is to generate the various control signals in accordance with commands received from the attached computer, whose task is to display the signals. For this job, we have selected an Atmel AT90S8515 RISC microcontroller, clocked at 8 MHz. An AT90S4414 could also be used. The latter type has less internal memory and is less expensive, but it is not as readily available.

The AT90S8515 is without doubt very well suited to this application. It has everything on board that we need to control the various parts of the circuit: 8 Kbytes of Flash program memory, 512 bytes of RAM, an asynchronous port, an ultra-fast core and so on. It also contains timers and an EEPROM, but they are not used in this circuit.

The microcontroller is accompanied by a TL78705 reset IC (IC18), which looks after generating the correct initialisation pulses.

IC15 is a traditional MAX232 (or equivalent), which adapts the signal levels between the microcontroller (which works with 5 V) and the RS232 lines (which work with ±12 V.)

Finally we have IC17, a trusty 7805, which provides a stable operating voltage for the circuit. This IC has a fairly demanding job, since the analyser is not particularly economical in its current consumption (especially the 40-MHz version).

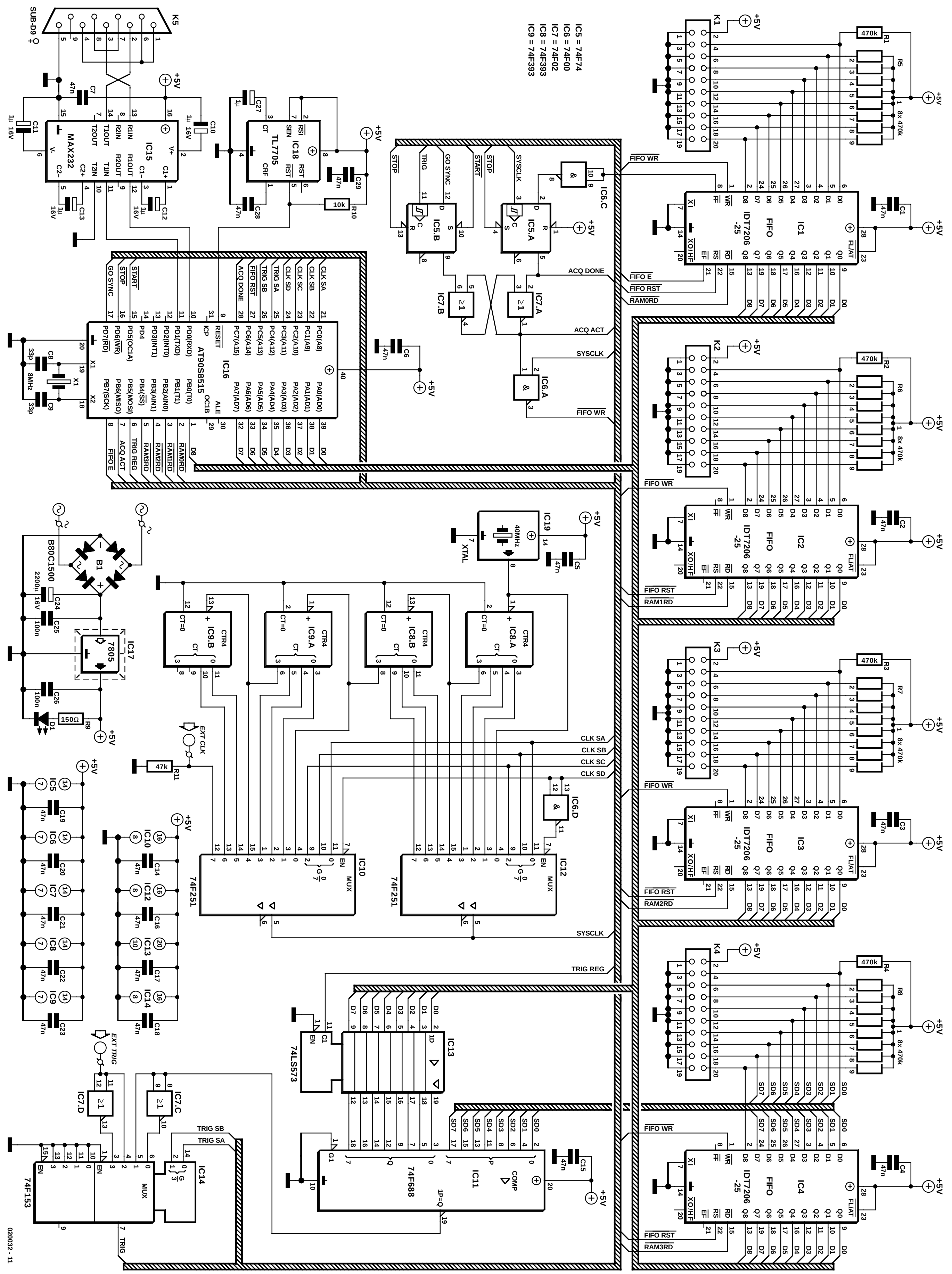


Figure 4. The schematic diagram of the logic analyser.

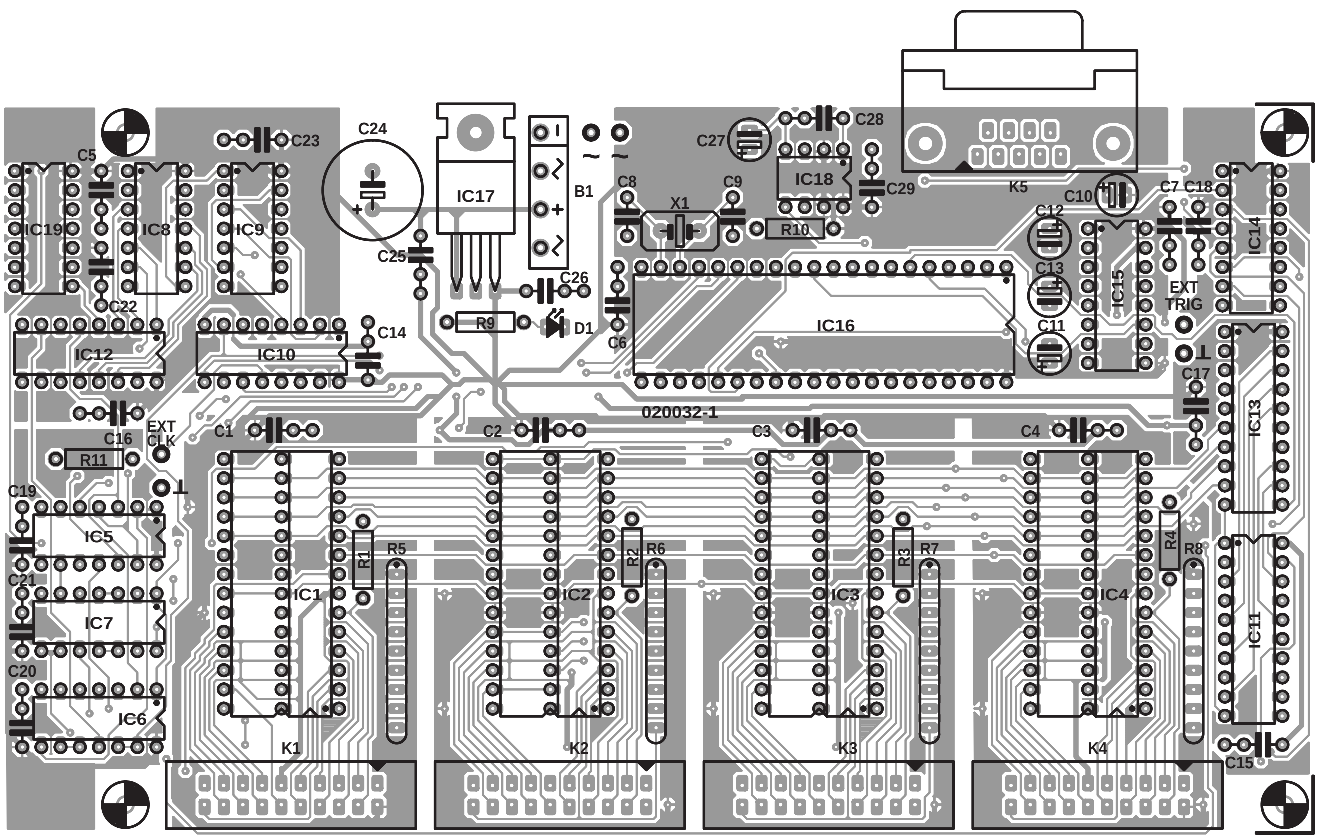


Figure 5a. The component layout of the printed circuit board for the logic analyser.

COMPONENTS LIST

Resistors:
R1-R4 = 470k_
R5-R8 = 8-way 470k_ SIL array
R9 = 1k_2
R10 = 10k_
R11 = 47k_

Capacitors:
C1-C7,C14-C23,C28,C29 = 47nF
C8,C9 = 22pF
C10-C13 = 1µF 16V radial
C24 = 2200µF 16V radial
C25,C26 = 100nF
C27 = 1µF 16V radial

Semiconductors:
B1 = B80C1500 in rectangular case, (80V piv, 1.5A)
D1 = LED, red, low-current
IC1-IC4 = IDT7206 (-15, etc. see text) or CY7C425-15PC
IC5 = 74F74
IC6 = 74F00
IC7 = 74F02
IC8,IC9 = 74F393
IC10,IC12 = 74F251
IC11 = 74F688
IC13 = 74LS573
IC14 = 74F153
IC15 = MAX232
IC16 = AT90S8515-8PC, programmed, order code **020032-41**
IC17 = 7805
IC18 = TL7705-ACP
IC19 = 40MHz or 20MHz oscillator module in DIL14 case (see text)

Miscellaneous:
K1-K4 = 2-way boxheader (angled pins possible)
K5 = 9-way sub-D socket (female), PCB mount, angled pins
X1 = 8MHz quartz crystal
PB, order code **020032-1** (see Readers Services page)
Disk, demo program, order code **020032-11** or Free Download

Before you start...

There's just one more thing before you pick up your soldering iron — you must decide which version of the analyser you want to build. As already mentioned, it can be built to work at two different clock rates: 20 MHz and 40 MHz.

The 40-MHz version can take 40 million measurements (samples) per second, which is quite fast for a circuit of this level of sophistication. Such a sampling rate is quite suitable for testing fast circuits. The following considerations are essential at this sampling rate:

- The FIFOs must be very fast types, with an access time of 15 ns or less (10-ns types are available).
- The ICs connected to the clock line (IC5, IC6, IC7, IC8, IC9, IC10 and IC12) must be 74F types. In no case may these ICs be replaced by 74LS or (even worse) 74HCT types.
- The ICs in the triggering section (IC11, IC13 and IC14) should preferably also be 74F types, but this is less critical than with the previously mentioned group. ICs from the 74LS family can be used, with the understanding this will result in slight delays during triggering, which usually will not be particularly troublesome.

If it is not possible to comply with the above requirements (such as if the ICs are unavailable or too expensive), you can fall back on the 20-MHz version, which can always be upgraded to the 40-MHz version at a later date.

In the 20-MHz version, all of the ICs may come from the 74LS family, but 74HCT devices should be avoided. Also, FIFOs with an access time of 50 ns (or less) can be used; these are easier to obtain and in any case less expensive.

While we're on the subject of FIFOs, besides speed there are two other parameters that must be taken into account: memory capacity and quantity. FIFOs are available in 1K × 9, 2K × 9, 4K × 9, 8K × 9 and 16K × 9 versions. However, the greater the capacity, the more expensive they are. In practice, the 4K version is just sufficient for most applications. However, there is nothing to stop you from using types with greater capacity, or even with less capacity.

As regards the number of FIFOs you want to fit, you only have to decide how many channels you want to sample simultaneously (9,

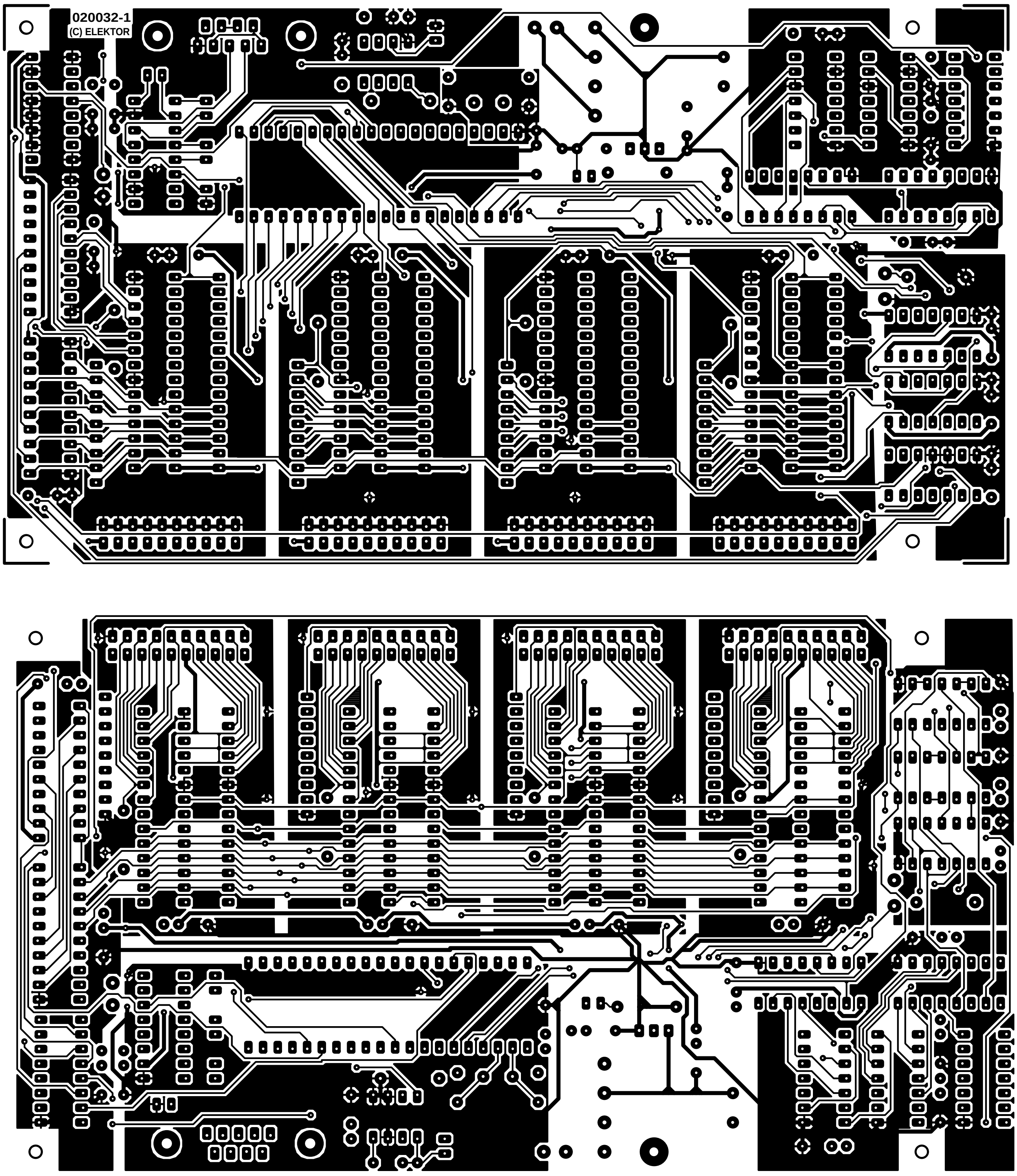


Figure 5b. The track layout of the printed circuit board for the logic analyser.

18, 27 or 36). Naturally, the depth of your (financial) pocket is also a consideration.

Besides the fact that all these options allow you to build the analyser to meet your particular needs, you should be aware that numerous manufacturers make FIFOs that are pin-compatible. **Table 1** shows several types from the most important manufacturers.

Be sure to pay attention to the package type when buying these ICs, since they are available in PLCC32, normal DIL28 and wide DIL28 packages. Our printed circuit board is suitable for both DIL28 versions, but not for the PLCC32 version.

The final consideration relates to the microcontroller (IC16). You can order a pre-programmed example from *Elektor Electronics* Readers Services (order number **020032-41**), or you can program it yourself. Since this microcontroller does not have an ISP port, it must be programmed using an external programmer.

Practical considerations

Once you have collected all the necessary components, you can start assembling the circuit board.

Despite the use of FIFOs, the number of components is rather large, which can also be seen from the dimensions of the printed circuit board (see **Figure 6**).

It is recommended to fit the ICs in sockets, preferably types having turned contacts. No particular problems should be encountered in fitting the components.

With regard to the voltage regulator (IC17), it must be noted that it has to dissipate a fair amount of heat, particularly with the 40-MHz version. The FIFOs draw 200–300 mA, while the 74F logic takes 150–200 mA, for a total of around 500 mA. Thus when fitting this IC, leave room for a heatsink or fit it vertically. A simple solution is to bolt a piece of metal L-section under the voltage regulator, so that a heat sink can be fitted to the upright part above the circuit board.

We also want to emphasise that is essential to use good-quality decoupling capacitors, in order to prevent undesirable surprises when making measurements on fast circuits.

After fitting all the passive components and the power supply portion, check that the 5-V supply voltage is present at the IC sockets before fitting the ICs.

Once all the components have been fitted, you can continue with the next step: testing.

Testing the circuit

Despite the complexity of this circuit, testing its operation is not difficult. All you need is a PC with Windows and the HyperTerminal program, which is normally provided with Windows. Any other type of computer with an RS232 port and a terminal emulator program is also adequate.

Start the terminal emulator program and configure it as follows: 8 bits, no parity, 1 stop bit and 38,400 baud. Connect the analyser to the serial port of the computer and switch on the supply voltage of the analyser.

You should now see a message on your monitor showing the program version and certain other data, as well as a cursor ('>'). If the program was not yet running when the analyser was switched on, type **#I** on the keyboard to reinitialise everything. After this, the start-up message should appear. If it appears correctly, you can assume that the microcontroller is working properly and proceed further. Otherwise, first check the serial link to the PC, since practical experience shows that most problems occur there. Also check the settings of the terminal emulator program. Don't forget that character echoing (returning transmitted characters) is performed by the analyser, so the echo function of the program must be disabled.

If everything appears to be in order but the circuit still refuses to communicate, check the portion of the circuit around IC16, IC15 and IC18.

If during initial testing it appears that some things are not working properly in the analyser, the only solution is to simply test the various portions of the circuit by sending the appropriate commands from the PC. For instance, the clock generator can easily be checked by connecting an oscilloscope to the SYSCLK line and using the **#F** command (see 'Controlling the analyser' below).

Enclosure

Other than perhaps a power switch, the analyser does not have any controls, since everything is controlled via the serial interface. The only visual user indication of its operation is the LED that shows that the supply voltage is present.

A simple enclosure is thus adequate for our analyser. Headers K1–K4 are located at the edge of the circuit board to make them readily accessible, so right-angle connectors can be used here. The external trigger and external clock inputs can also be located on the same side of the enclosure, for example

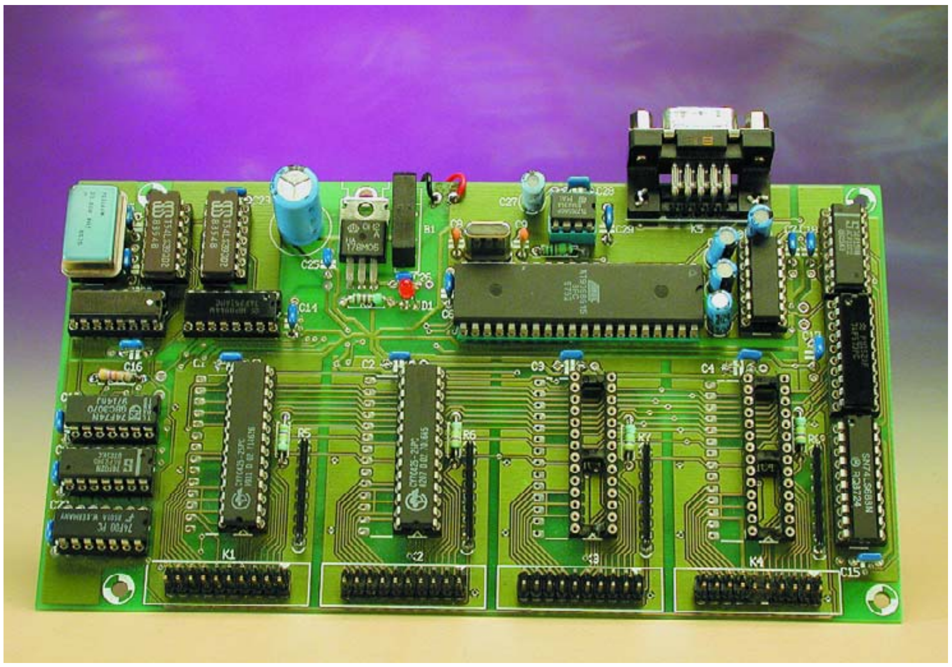


Figure 6. The fully assembled prototype of the logic analyser (two-channel version).

Table 1.

Several types of FIFOs that can be used for this analyser. The operating speed of a FIFO can be determined from its full type number.

Manufacturer	Type	Capacity
AMD	AM7202	1K x 9
	AM7203	2K x 9
	AM7204	4K x 9
	AM7205	8K x 9
IDT	IDT7203	2K x 9
	IDT7204	4K x 9
	IDT7205	8K x 9
	IDT7206	16K x 9
CYPRESS	CY7C425	1K x 9
	CY7C429	2K x 9
	CY7C433	4K x 9
	CY7C460	8K *x 9
	CY7C462	16K x 9

Table 2

Summary of the commands for controlling the analyser.

#I	Initialise the analyser
#G	Start sampling
#C	Stop sampling
#R	Read out data from the FIFOs
#P	Repeat the data last read using the #R command
#Mxx	Set the trigger pattern
#Tx	Select the triggering mode
#Fx	Set the sampling rate
#S	Read the current analyser state
#A	Arm the trigger
#Z	Clear the FIFOs

using banana sockets.

Use a metallic box for the analyser enclosure to provide good screening and prevent the emission of electromagnetic radiation. Remember that the analyser works with in quite high frequencies!

The power supply can consist of a small mains transformer with a secondary voltage of approximately 9 V (at around 8 VA), or a small mains adapter. Be sure the read the Electrical Safety Guidelines page (published occasionally in *Elektor Electronics*) before fitting a mains transformer in the metallic enclosure.

Controlling the analyser

The analyser can be controlled very easily using a PC. This control works according to the master/slave principle, in which the PC sends a command and waits for a response. The analyser never sends anything on its own initiative. Bear in mind that the local processor continues to respond to requests from the PC during the sampling phase (since the FIFOs do their jobs without any involvement of the processor).

The link between the PC (or another type of computer) and the analyser is formed by an RS232 cable with the leads connected 1-to-1 (no crossed connections). Each character received by the analyser is returned as an 'echo' so the PC can verify that no communications errors have occurred.

Each command to be used for the analyser always starts with the symbol '#'. This symbol must be followed by a letter (upper or lower case) representing a command. The letter may be followed by one or more hexadecimal ASCII characters that serve as parameters. For example, the hexadecimal value $5A is represented by the two characters '5' and 'A'.

The analyser recognises the carriage return (CR = ASCII 13) and line feed (LF = ASCII 10) characters, but they are not mandatory.

Once a complete command has been sent to the analyser, it can respond with one of the following characters:

- **>** The command has been correctly executed.
- **?** The command is not recognised (non-existent command or incorrect parameter).
- **!** The command is recognised, but it is not accepted in the current state of the analyser.

The various commands recognised by the analyser are summarised in **Table 2**. Since these commands form the core of our device, we will describe them in more detail.

#I_Initialise the analyser.
Everything is reset (FIFOs cleared, sampling stopped). This command also causes the welcome message to be displayed on the PC monitor, including the version number of the program.

#G Immediately start sampling.
After receiving this command, the analyser starts to store the data present on the inputs. The sampling rate is set by the #F command. The FIFOs are cleared before sampling is started. This command will be refused if the analyser is already busy with sampling.

#C_Immediately stop sampling.
When this command is received, the analyser immediately stops taking samples (even if it has not yet been triggered). Data that have been recorded up to the time the command was received can be read using the #R command.

#R_Read data from the FIFOs.
In response to this command, the analyser sends a 36-bit character string containing data read from the FIFOs, terminated by CR/LF. This string has the structure WWXXYYZZ0V, where the value of 'WW' is determined by the eight bits from the first FIFO, 'XX' by the second FIFO, 'YY' by the third FIFO and 'ZZ' by the fourth FIFO. The 'V' position is for the four extra bits (bit 9 of each of the four FIFOs).

This command will not be executed if the analyser is busy sampling data or FIFOs are empty. If one or more FIFOs are not fitted, the code 'FF' will be sent for each missing FIFO.

#P_Repeat the last data read using the #R command.
If the program finds discrepancies in the data retrieved using a #R command, (e.g., due to transmission errors), it can use the #P command to cause the data block to be re-sent as often as necessary. The #P command cannot be used to read out the FIFOs.

#Mxx_Set the trigger pattern.
The #M command sets the value appearing at the output of the trigger comparison register. The analyser will start taking samples when the data on the eight inputs of K4 match the value transferred via the parameter of the #M command.

#Tx_Select the triggering mode.
The value of the parameter (0–3) determines the trigger source:

0: The analyser starts when the binary value on K4 is equal to the value of the trigger pattern (as specified using #M).
1: The analyser starts when the binary value on K4 is different from the trigger pattern.
2: The analyser starts on a rising edge at the EXT TRIG input.
3: The analyser starts on a falling

edge at the EXT TRIG input.

#Fx_Set the sampling rate.
This command is used to select the output of the clock divider for the FIFO read clock. This command will not be executed if the analyser is busy taking samples. The actual sampling rate depends on oscillator IC19. The following table shows the time between successive samples for the various parameter values at the two oscillator frequencies.

x	20 MHz	40 MHz
8	50 ns	25 ns
9	100 ns	50 ns
A	200 ns	100 ns
B	400 ns	200 ns
C	800 ns	400 ns
D	1.6 µs	800 ns
E	3.2 µs	1.6 µs
F	6.4 µs	3.2 µs
0	12.8 µs	6.4 µs
1	25.6 µs	12.8 µs
2	51.2 µs	25.6 µs
3	102.4 µs	51.2 µs
4	204.8 µs	102.4 µs
5	409.6 µs	204.8 µs
6	819.2 µs	409.6 µs
7	Ext. clock	Ext. clock

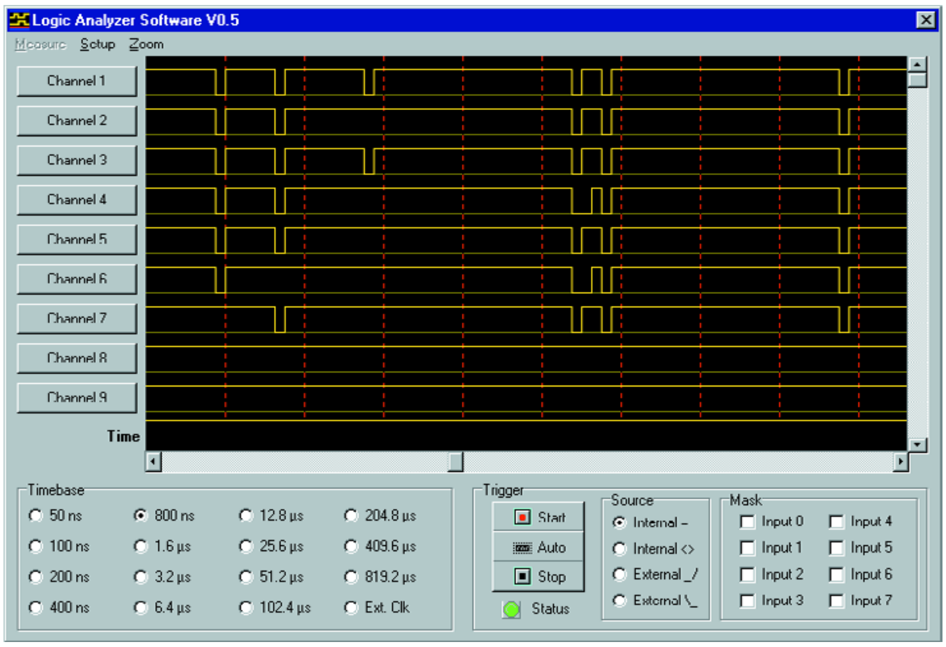
#S_Read the current analyser state.
The analyser responds to this command with the code XX<CR><LF>. The binary value 'XX' in the response is constructed as follows:

bit 0:	1 = analyser started
bit 1:	1 = analyser stopped
bit 2:	0 = FIFOs empty
bit 2:	1 = FIFOs not empty

Immediately after the analyser has been switched on, the response to the #S command will be the value '00'.
#A_Arm the trigger.
After the trigger has been armed, the analyser will start as soon as the trigger condition set using the #T command is satisfied.
#Z_Clear the FIFOs.

Graphic user interface for the analyser

As can be seen from the above, it is very easy to control the analyser. It is relatively easy to write a program to control the analyser (the language and platform are not important) and display the data it sends



Timebase: ( ) 50 ns ( ) 100 ns ( ) 200 ns ( ) 400 ns [✓ 800 ns] ( ) 1.6 µs ( ) 3.2 µs ( ) 6.4 µs ( ) 12.8 µs ( ) 25.6 µs ( ) 51.2 µs ( ) 102.4 µs ( ) 204.8 µs ( ) 409.6 µs ( ) 819.2 µs ( ) Ext. Clk

Source: [✓ Internal –] ( ) Internal <> ( ) External _/ ( ) External _

Mask: ☐ Input 0 ☐ Input 1 ☐ Input 2 ☐ Input 3 ☐ Input 4 ☐ Input 5 ☐ Input 6 ☐ Input 7

in graphic form.

For readers who are not keen to write their own programs, the author has written a program (a screen dump) that is available as a demo version at no charge from the *Elektor Electronics* website. It can also be obtained on a diskette from Readers Services (order number **020032-11**).

The full version of the program is available from the author. You can find more information on a dedicated page at the author's website: http://benoit.bouchez.free.fr/logic_analyzer.

A few final remarks

As should be evident, the logic analyser described here is intended to be used with 5-V TTL signals (including CMOS circuits operating at 5 V). The analyser can also be used with 3.3-V circuits, but in that case resistors R1–R4 and resistor networks R5–R8 must not be fitted, since there is otherwise a risk of damage to the circuit being tested, due to the fact that the 3.3-V line would connected to 5 V via these resistors (although the high values of these resistors limit the chance of damage). It may seem strange that the FIFOs, which operate with a 5-V supply voltage, can nevertheless correctly interpret signals that use the 3.3-V standard. However, the 3.3-V TTL standard actually uses the same switching thresholds as the 5-V TTL standard, with 2.4 V or more being High and 0.4 V or less being Low. This guarantees the compatibility of 3.3-V signals connected to 5-V inputs.

To test logic circuits that work with a supply voltage of 3 V or 2.5 V supply voltages, which are becoming increasingly common, it will be necessary to use input adapters, since the thresholds for High and Low levels are not the same as for 3.3-V and 5-V TTL. Nevertheless, we were able to successfully use the analyser to test a 3-V circuit.

An adapter is also necessary for testing CMOS circuits working at voltages greater than 5 V, since the FIFOs absolutely cannot handle such voltages. Several adapters will be described in a future article.

We would like to again point out the relative vulnerability of the inputs, which are connected directly to FIFOs. If you connect an input to a line carrying a voltage greater than 5 V, there is a risk that the FIFO will quickly perish. Even professional logic analysers cannot escape this risk.

Why then are no protective measures used on the inputs? The reason is very simple: protective measured affect the characteristics of the circuit being tested, and after all, the analyser has been designed to track down faults, not to generate them.

The final important consideration is that you should use **very short** leads to connect the analyser to the circuit being tested, preferably using flatcable — and there is a good reason why the input connectors have ground connections. Excessively long connections can cause significant distortion of the signals and can also disturb the operation of the circuit being tested, especially when fast signals are involved.

(020032-1)

Passive-Optical Person Detector

responds to changes in ambient lighting

Design by B. Kainka

Ordinary proximity detectors use infrared sensors which respond to the warmth of the human body. But there's a simpler way: we can use ambient lighting and detect changes in illumination.

Any move a person makes causes a change in the ambient lighting. This circuit uses an ordinary phototransistor as its 'eyes' to detect these changes. The circuit can detect even the smallest change in illumination: it is not even necessary for a shadow to fall directly on the detector. It is enough if just a part of the sky is occluded. The sensitivity of the sensor can be set using a potentiometer.

The circuit will work over a wide range of ambient lighting conditions. A control loop with a logarithmic characteristic ensures that any given percentage change in light level is equally well detected, whatever the ambient level may be. The circuit will therefore even work in a darkened room. Gradual changes in illumination are ignored, while rapid changes are greeted with an acoustic alarm. However, direct sunlight is not good for the sensitivity of the circuit: it is better to use indirect lighting, reflected from other objects in the environment.

Component selection is not particularly critical. Any ordinary phototransistor can be used for T1. For T2, as well as the type BC547B suggested in the parts list, any similar NPN transistor such as the BC548, BC549 or BC550 will do fine. The transistor's current gain should not be too small, however, and so it's important to check the identifying letter ('suffix') after the part number, which indicates the gain group. Type A transistors are not suitable: types B or C should be used instead.

For the opamp (IC1.A and IC1.B), other dual opamp devices such as the TL072 or TL082 can be used.

The LEDs used should be low current types, able to operate at their rated intensity with a current of 2 mA. In comparison, conventional (generally older) LEDs need considerably more current, being specified to reach their rated intensity at 10 mA or even 20 mA. It's not surpris-

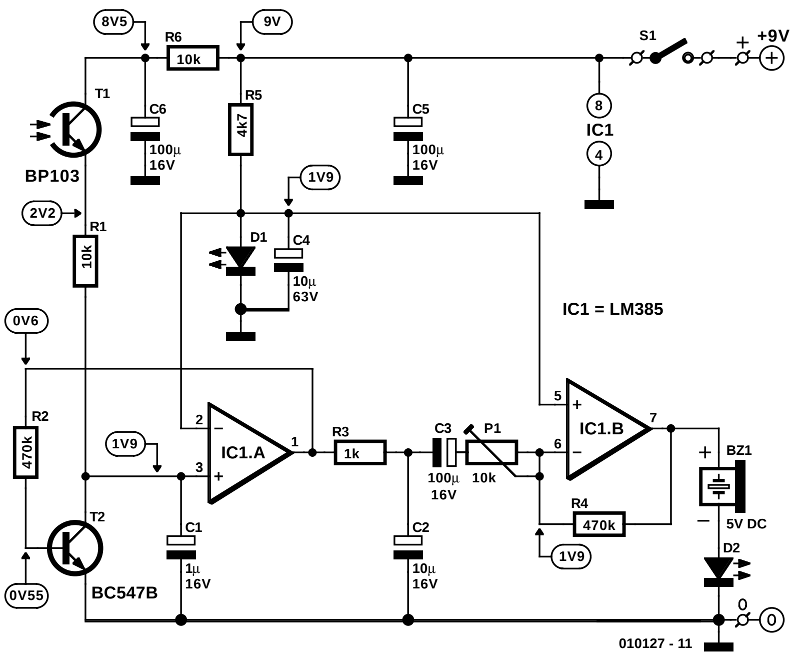


Figure 1. The control circuit around T2 provides automatic compensation for different intensities of ambient illumination.

ing that low current LEDs have been so widely successful.

We should also remark on the buzzer, which is a DC (or 'active') type rated for operation at 5 V. The output of opamp IC1.B will be 1 V or 2 V less that than the battery voltage of 9 V, and we also need to take into account the voltage drop of about 2 V across LED D2, which is in series with the buzzer and which lights when the opamp drives the buzzer.

The battery should last for a reasonably long time. Although LED D1 is continuously illuminated, it only draws about 2.5 mA in the quiescent condition. In the alarm condition, extra current flows through the buzzer

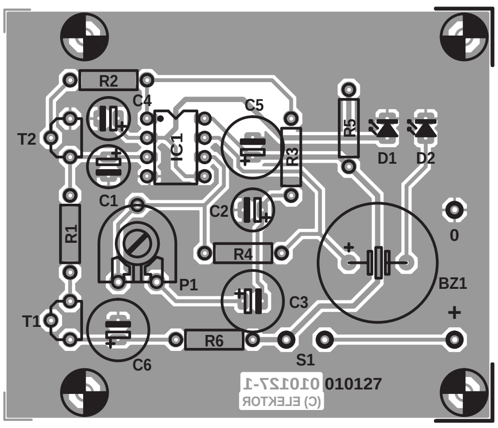
How does it work?

The circuit uses two operational amplifiers, or opamps, contained in a single package. The first stage, around IC1.A, forms a control loop. The further the output voltage rises (above about 0.6 V) the more transistor T2 conducts. T2 thus forms a variable load for phototransistor T1. As the illumination increases, more current flows through the phototransistor. T2 then also turns more fully on, keeping the voltage at the output of the opamp more-or-less constant. In this way the circuit can work over a very wide range of illumination levels.

The transfer characteristic curve of a transistor is very close to exponential over a wide range. Each increase in base voltage of about 20 mV gives rise to a doubling in the collector current. This is used in reverse in this circuit, where a doubling of the ambient illumination intensity will lead to a rise of about 20 mV in the output voltage of IC1.A. If the ambient illumination changes by only 10%, the output voltage will change by just a couple of millivolts, essentially independent of the absolute level at which this relative change occurs.

The second sensor amplifier stage is coupled via a high-pass and a low-pass filter. In the quiescent state, with constant illumination, the output voltage of IC1.B will be at about 1.9 V. Changes in the input voltage will be greatly amplified. Very rapid changes in intensity will be filtered out by the low-pass filter formed by R3 and C2. This reduces the sensitivity of the circuit to the 100 Hz flicker from mains-powered lighting. Very slow variations, on the other hand, are filtered out by the high-pass filter formed by C3 and P1. This ensures that the alarm is not triggered by clouds or by changes in the level of sunlight over the day. However, in the middle frequency range, between about 0.1 Hz and 10 Hz, the circuit is very sensitive, and detects the normal rapid movements of people. The gain, and hence the sensitivity, of the circuit can be set using P1.

IC1.B operates as an inverting amplifier. This means that a sudden fall in light intensity will cause a rise in the output voltage. As soon as this voltage rises above about 2.5 V, and so more than 0.5 V appears across the buzzer, it will start to sound. Thus a shadow crossing the sensor will result in a brief tone. At the same time, current flows through LED D2, which will therefore light briefly. The first LED (D1) not only shows when the circuit is operating, but is also used to generate a stable auxiliary voltage of about 1.9 V.



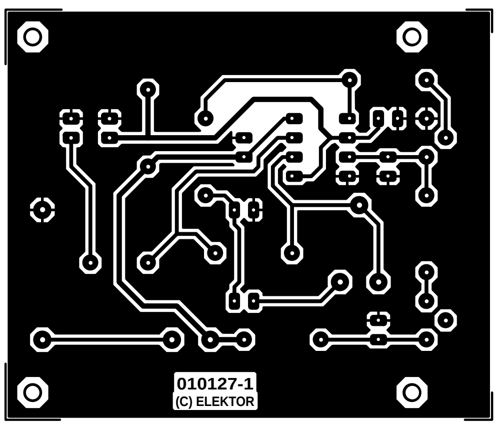


Figure 2. The printed circuit board makes constructing the person detector straightforward (board available via The PCB Shop).

and the other LED, and the total current consumption rises to about 10 mA.

Assuming that the circuit board has been correctly populated (and that the battery is connected with the right polarity!), the buzzer should emit a brief sound and LED D2 should briefly light as soon as the circuit is switched on by closing S1. D1 should light continuously as long as S1 is closed. If D1 does not light, either the diode or the battery is connected the wrong way around. Or the battery is flat — which you can always check with a meter.

The sensitivity of the circuit can be adjusted using trimmer potentiometer P1. The further P1 is turned to the right, the more sensitive the circuit. At high sensitivity even the flickering of fluorescent tubes (including so-called 'energy saving bulbs'), although invisible to the naked eye, will be amplified by the circuit. In electrical terms this is like a 100 Hz AC hum signal and so it is not surprising that it interferes with the circuit's operation, and in some cases can cause the buzzer to sound continuously. The sensitivity control must be backed off until the circuit no longer responds to the fluorescent light, but only to changes in the ambient illumination.

COMPONENTS LIST

Resistors:
R1,R6 = 10kΩ
R2 = 470kΩ
R3 = 1kΩ
R4 = 470kΩ
R5 = 4kΩ7
P1 = 10kΩ preset

Capacitors:
C1 = 1µF 16V radial
C2, C4 = 10µF 16V radial
C3,C5,C6 = 100µF 16 V radial

Semiconductors:
D1 = LED, green, low-current
D2 = LED, red, low-current
T1 = BP103
T2 = BC547B
IC1 = LM358P

Miscellaneous:
S1 = switch, on/off, 1 contact
BZ1 = 5V DC (active) buzzer
9V PP3 (6LR22) battery with clip-on lead
Enclosure with battery compartment, size 101 x 60 x 26 mm

PCB, available from The PCB Shop, see www.elektor-electronics.co.uk

The DC voltage measurements shown in the diagram were made under normal room lighting using a digital multimeter with a 10 MΩ input impedance.

The printed circuit board shown here is unfortunately not available ready-made through the Publishers' Readers Services. However, it may be obtained as a one-off from **The PCB Shop** — for more details see the PCBs & Software page on our website.

(010127-1)

Practical Neural Networks (2)

COURSE SUPPORT WEBSITE
http://www.rgu.ac.uk/eng/compint/page.cfm?pge=6732

Part 2: Back Propagation Neural Nets

By Chris MacLeod and Grant Maxwell

Back Propagation (BP) Networks are the quintessential Neural Nets. Probably eighty percent of nets used today are of this type. Actually though, Back Propagation is the learning or training method, rather than the network structure itself.

The network operates in the same way as the type we've looked at in part 1 — you apply the inputs and calculate an output exactly as described. What the Back Propagation part does, is allow you to change the weights, so that the network learns and gives you the output you want. The weights which the network starts off with are simply set to small random numbers (say between –1 and +1).

What is BP good for?

Back Propagation is excellent for simple pattern recognition and mapping tasks. It learns by example.

To give a typical application, we can train a BP network for character recognition. All you need to do is give it examples of the characters, and what output we would like the network to have, and it will learn from them, see **Figure 1**.

The algorithm works by calculating an error — which is the amount by which the output differs from an ideal value (chosen by you, and called the Target), and then changing the weights to minimise this error. Once the network is trained, it will correctly give the output when a character is applied, even if the character is distorted, imperfect or noisy. In this case, because the Target has two bits, we need two output neurons (one for each bit). Each input and its associated Target is called a Training Pair.

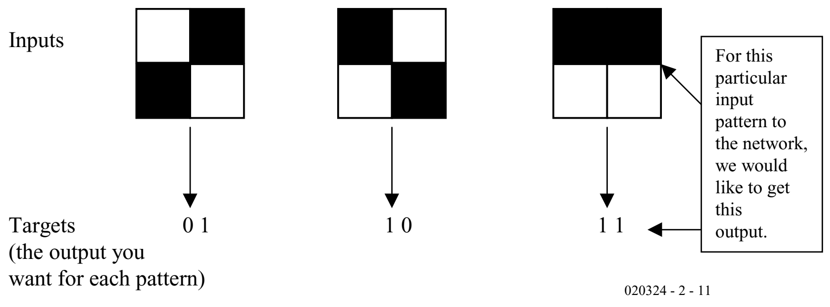

Inputs
For this particular input pattern to the network, we would like to get this output.
Targets
(the output you want for each pattern)
0 1
1 0
1 1
020324 - 2 - 11

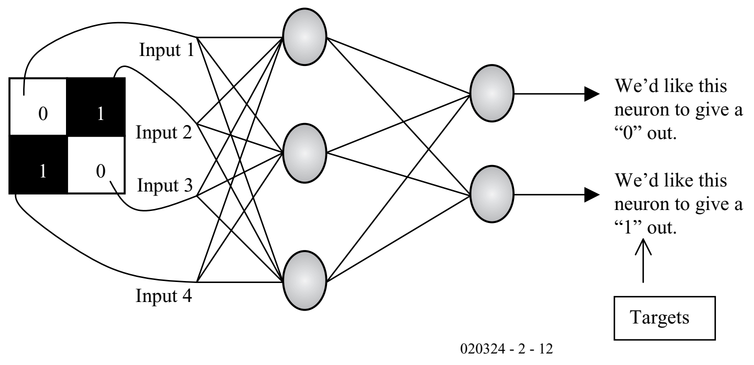
Figure.1. Use of a BP network for image recognition.

What does a BP network look like?

Figure 2 shows a BP network being used for Pattern Recognition.

A common question is: How big should the network be? We can see from Figure 2 that the number of inputs is fixed by the pattern we are trying to process. In the case of four pixels, there must be four inputs. Likewise, the number of output neurons is fixed by the number of patterns we what to recognise. If we had nine patterns we could either use three output neurons and binary code their outputs, or we could use nine and assign them so that, for example, when pattern 2 appears, output neuron 2 gives a '1' (and the rest are zero).

This only really leaves the number of neurons in the hidden layer to decided on. Fortunately, networks are quite flexible about this parameter and will operate over a wide range of hidden layer neurons; although, the more patterns the network needs to remember, the more neurons you will need. In a network designed to recognise all 26 letters of the alphabet (26 output neurons) on a 5×7 grid (35 inputs), the network will function with anywhere between about 6 and 22 neurons. If you have too few, then the network hasn't got enough weights to store all the information in; if there are too



Input 1
Input 2
Input 3
Input 4
0
1
1
0
We'd like this neuron to give a "0" out.
We'd like this neuron to give a "1" out.
Targets
020324 - 2 - 12

Figure 2. A network wired for recognising patterns.

many, it becomes inefficient and prone to a problem called local minima (discussed later).

The BP algorithm

Now let's have a look at the training algorithm itself. To do this, we'll refer to three neurons labelled A,B and C in **Figure 3**.

The weight that we'll train is that between neuron A and neuron B and is labelled W_{AB} in the diagram. The diagram also shows another weight — W_{AC} — and we'll return to that one in a moment.

The algorithm works like this:

1. First, apply the inputs to the network and calculate its outputs as described last month in Part 1 (this is the *forward pass*).

2. Next, calculate the output error for neuron B. The error is basically: *What you want - What you get*. What you want is your target and what you get is your output. Mathematically:

$$Error_B = Output_B * (1 - Output_B) * (Target_B - Output_B)$$

The term $Output_B * (1 - Output_B)$ is present because of the effect of the sigmoid function — if we were just using a binary threshold, we would omit it.

3. Change the weight. Let W^+_{AB} be the new (trained) weight and W_{AB} be the original (untrained) weight:

$$W^+_{AB} = W_{AB} + (Error_B \times Output_A)$$

Notice that we use the error of the second neuron (B), but the output of the feeding neuron (A).

4. Change all the other weights in the output layer in this manner.

5. To change the weights of the hidden layers you need to calculate an error for the hidden neurons. We do this by *Back Propagating* the errors of the output neurons back. For example, suppose we want to calculate the error for neuron A. We use the errors calculated for all the output neurons attached to it, in this case B and C and propagate them back — hence the name of the algorithm.

$$Error_A = Output_A * (1 - Output_A) * (Error_B * W_{AB} + Error_C * W_{AC})$$

Again, the $Output_A * (1 - Output_A)$ serves the purpose noted in 2.

6. Having obtained the errors for the hidden layer neurons, we now proceed back to stage 3 and change their weights.

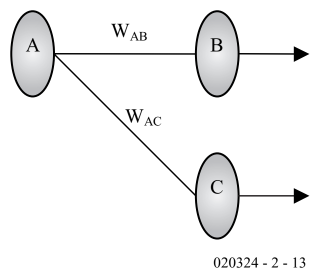


Figure.3. Three neurons which are part of a larger network.

Now this might be a little confusing, so let's show a full example, **Figure 4**.

1. Calculate errors of output neurons

$Error_\alpha = out_\alpha (1 - out_\alpha) (Target_\alpha - out_\alpha)$

$Error_\beta = out_\beta (1 - out_\beta) (Target_\beta - out_\beta)$

2. Change output layer weights

$W^+_{A\alpha} = W_{A\alpha} + \eta Error_\alpha out_A$ $\quad W^+_{A\beta} = W_{A\beta} + \eta Error_\beta out_A$

$W^+_{B\alpha} = W_{B\alpha} + \eta Error_\alpha out_B$ $\quad W^+_{B\beta} = W_{B\beta} + \eta Error_\beta out_B$

$W^+_{C\alpha} = W_{C\alpha} + \eta Error_\alpha out_C$ $\quad W^+_{C\beta} = W_{C\beta} + \eta Error_\beta out_C$

3. Calculate (back-propagate) hidden layer errors

$Error_A = out_A (1 - out_A) (Error_\alpha W_{A\alpha} + Error_\beta W_{A\beta})$

$Error_B = out_B (1 - out_B) (Error_\alpha W_{B\alpha} + Error_\beta W_{B\beta})$

$Error_C = out_C (1 - out_C) (Error_\alpha W_{C\alpha} + Error_\beta W_{C\beta})$

4. Change hidden layer weights

$W^+_{\lambda A} = W_{\lambda A} + \eta Error_A in_\lambda$

$W^+_{\Omega A} = W^+_{\Omega A} + \eta Error_A in_\Omega$

$W^+_{\lambda B} = W_{\lambda B} + \eta Error_B in_\lambda$

$W^+_{\Omega B} = W^+_{\Omega B} + \eta Error_B in_\Omega$

$W^+_{\lambda C} = W_{\lambda C} + \eta Error_C in_\lambda$

$W^+_{\Omega C} = W^+_{\Omega C} + \eta Error_C in_\Omega$

The constant η (called the learning rate, and nominally equal to one) is put in to speed up or slow down the learning if required.

Using BP to train a network

Now that we've seen the algorithm in detail, let's look at how to use it. One of the most common mistakes made when programming a BP network for the first time is the order in which you apply the patterns to the network. Let us take an example. Suppose you wanted to teach the network to recognise the first four letters of the alphabet, placed on a 5?7 grid.

The correct way to train the network is to apply the first letter, and then change all the weights of the network **once** (i.e., do all the calculations in Figure 4, once only). Then apply the second pattern and do the same again, then the third and finally the fourth. Once you've gone through this cycle once start all over again with pattern 1. **Figure 5** shows the idea.

We stop the network when the total error is low enough — that is, when the sum of all the errors (the positive error from every neuron, summed over every pattern) is below a threshold. This threshold is usually set by the user to be some arbitrary low number, like 0.1. In the example above the total error of the network would be:

(Errors of all neurons in pattern 1) + (Pattern 2 errors) + (Pattern 3 errors) + (Pattern 4 errors)

Before doing this, it is necessary to make all the errors positive — we can do this by squaring them.

The learning process is shown in the algorithm below:

1. Apply first pattern, perform forward pass, perform reverse pass.
2. Apply second pattern, perform forward pass, perform reverse pass.
3. Apply third pattern, perform forward pass, perform reverse pass.
4. Apply fourth pattern, perform forward pass, perform reverse pass.
5. Test: is total error small enough? If yes, then go to 6.
6. Go to 1.
7. Stop, network has trained.

A common mistake to make is running the program on pattern one until the error is low, then on pattern two and then on pattern three. If you do this, then the network will only learn the last pattern you've presented it with.

Once the network has learned, you can apply any of the inputs to it (just apply the input and run a forward pass with the trained weights) and it should recognise them. We can then use the network to recognise patterns in a real system.

A more accurate way to train the network is to use a validation set. This is similar to the set of the patterns which you are training the network with — but with noise or other imperfections added. After the training set has been applied, the validation set is run through the network to check its performance (we don't use the validation set to change the network weights). When the net has fully trained both the validation set and the training set will give a low error. If you're training the network too much, then the validation set error will increase as shown in **Figure 6**.

Algorithms in software

In part 1 we discussed various ways of coding the network. One way was to store the weights in a three dimensional array, with indexes denoting the layer number, the neuron number and the connection number. A suitable algorithm for a Back Propagation reverse pass in such a network might be:

1. Initialise all unused weights, targets, errors and outputs to zero
2. Calculate output errors, see **Listing 1**, first part.
3. Change weights, see Listing 1, second part.
4. Calculate error of hidden layers, see Listing 1, third part.

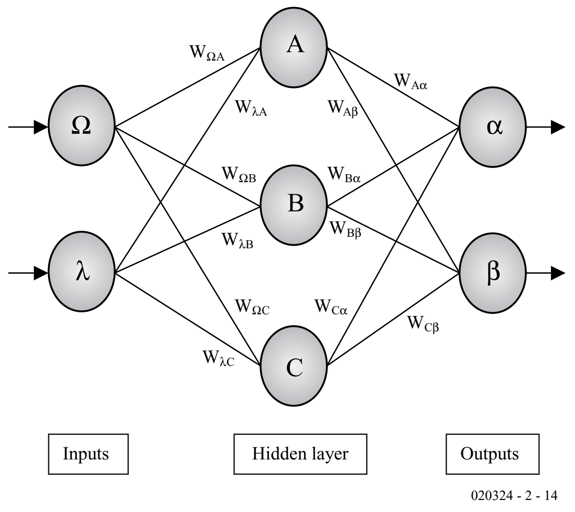


Figure 4. All the calculations for a complete reverse pass in a network.

Where, in addition to the variables explained in part 1 of this course, E(L,n) and T(L,n) are the errors and targets respectively of layer L, neuron n.

Putting it all together

Now that we have algorithms for both the forward and reverse pass of the network, we can put them together into a coherent whole. Given below is a suggestion, showing how this can be done:

1. Set up inputs and targets for network (either in a file, or in arrays).
2. Randomise weights being used.
3. Apply first pattern, calculate network output (forward pass) and error, use error to change weights (reverse pass) — once only. Keep a note of the error.
4. Do the same for second pattern. Add error to the running total from pattern one.

Listing I

```
FOR x = first_output_neuron TO final_output_neuron_number
  E(output_layer, x) =    O(output_layer, x) * (1 -
O(output_layer, x) * (T(output_layer, x) -
O(output_layer, x))
NEXT x

FOR L = number_of_layers TO 1 STEP –1
    FOR n = 1 TO max_number_of_neurons
      FOR c = 1 TO max_number_of_weights
        W(L, n, c) = W(L, n, c) + E(L + 1, n) * O(L, c)
      NEXT c
    NEXT n

    FOR n = 1 TO maximum_number_of_neurons
      FOR c = 1 TO max_number_of_weights
        E(L, n) = E(L, n) + E(L + 1, c) * W(L, c, n)
      NEXT c
      E(L, n) = E(L, n) * O(L, n) * (1 - O(L, n))
    NEXT n

NEXT L
```

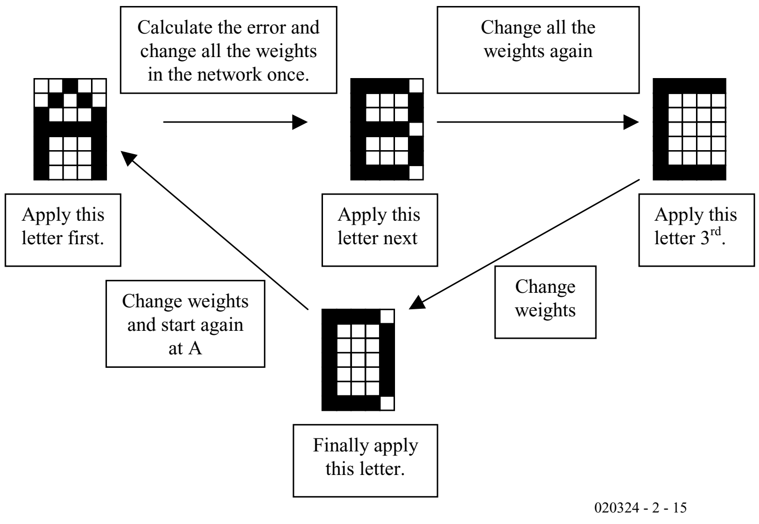

Calculate the error and change all the weights in the network once.
Change all the weights again
Apply this letter first.
Apply this letter next
Apply this letter 3rd.
Change weights and start again at A
Change weights
Finally apply this letter.
020324 - 2 - 15

Figure 5. How a network learns four patterns.

5. Repeat for all subsequent patterns, keep running total of error.
6. If error is too great (network still not fully trained) then zero running total and go to 3, else go to 7.
7. Network is trained and ready to be used, either use directly or store trained weights in a file for future use.

Problems and additions

Although BP is a very useful and simple algorithm, it does have some problems and limitations. Let's start with its limitations.

BP is excellent for the sort of simple pattern recognition and mapping tasks explained above and in the first article. However, it only works well when the image it is to recognise is the correct size and placed in a central position on the grid. It's no good at, say, recognising a face in a crowd — unless you can centre the face or make the network 'scan' the picture until it falls onto the face (and even then you still have to make the face the correct size). In other words, many problems need to be 'pre-processed' before being presented to the network.

So these networks need to operate in a controlled environment, which means that applications such as Optical Character Recognition (OCR) are more suitable. They have problems dealing with the crowded and confusing real world.

Incidentally, the human brain solves this problem by first identifying 'features' in an image, for example, horizontal or vertical lines and then integrating these progressively into a whole image in a layered structure. So if you can identify a horizontal line along the top of an image and a vertical line down the middle, you can integrate these to find the letter T. This approach is more tolerant because these features (the two lines) are always present in T, no matter where its placed in the image or what size it is.

When running your network, you may run into problems with its training. The most common is known as 'local minima'. This occurs because the algorithm always follows the error downwards (it can't cause a change of weights which causes the error to increase). But sometimes, as part of a downwards trend, the error must go up as shown in **Figure 7**. In this case the training gets stuck and the weights can't move out of the local minima.

This problem doesn't really effect small networks, but becomes a problem as the network size increases. One solution is to add 'momentum' to the network. This involves allowing the change of weight to continue for some time in a particular direction as shown below:

New_weight = Old_weight + weight_change + *Weight_change_from_previous_iteration.*

However, a simpler way to overcome this problem (and several others which effect training) is simply to monitor the training progress of the network and if the error gets 'stuck' (does not decrease for some time), reset the initial weights of the network to different random values and start training again.

In next month's instalment of this course, we'll have a look at networks which have recurrent connections including the famous 'Hopfield' network.

(020324-2)

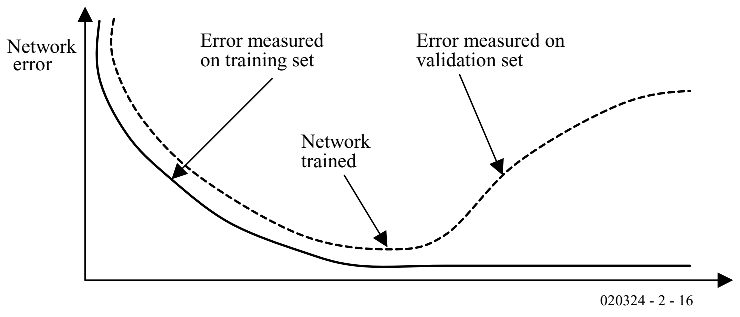


Figure 6. Use of a validation set.

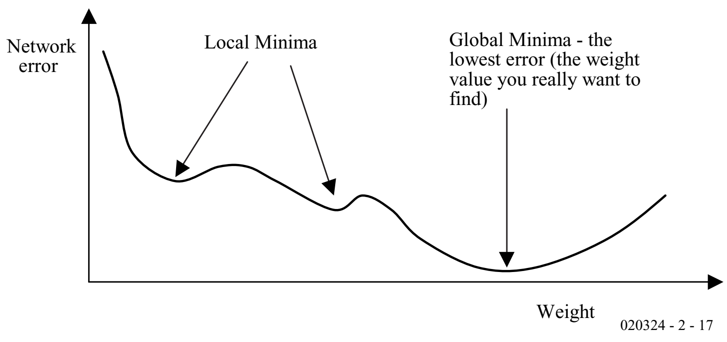


Figure 7. Local minima.

SMDs? Don't panic!

Part 2: practical tips on SMD mounting

By C. Tomanik

In this second and concluding instalment we concentrate on the techniques necessary for soldering and desoldering SMD components.

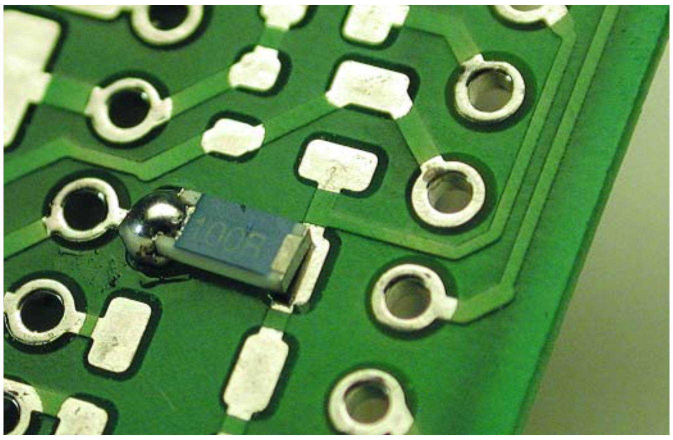
In the last article we looked at some of the different component outlines available for SMDs and some of the basic tools necessary to begin work. This article looks more closely at the techniques necessary for soldering and desoldering SMD components. A standard soldering iron with an interchangeable bit together with some additional solder paste are the only soldering tools necessary for most SMD work. A normal soldering iron would be something similar to the Ersa type Analogue 60 A. This iron is rated at 60 W and has interchangeable bits. Weller and Antex also produce soldering irons with a similar specification. For the ERSA iron it is necessary to fit a fine soldering bit type 832 UD. This bit has a tip diameter of 0.4 mm. The disadvantage of the simple conical soldering bit is that it does not provide an even heat distribution when applied to the joint area and pad but nonetheless, with a little care, it is still possible to produce good quality soldered joints.

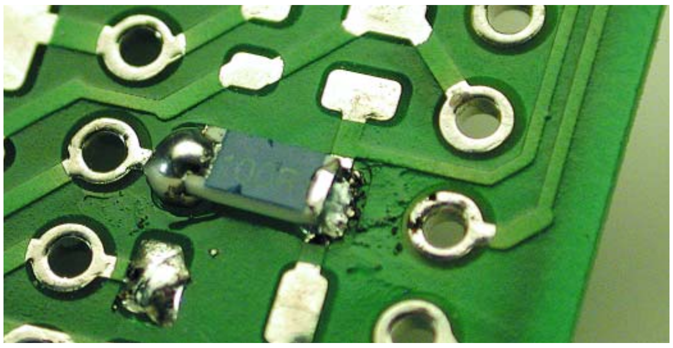
The iron temperature should be set between 350 and 400 °C. This may seem a little hot but a high bit temperature speeds up the soldering process and actually reduces the risk of overheating both component and pad.

Resistors, ceramic and tantalum capacitors

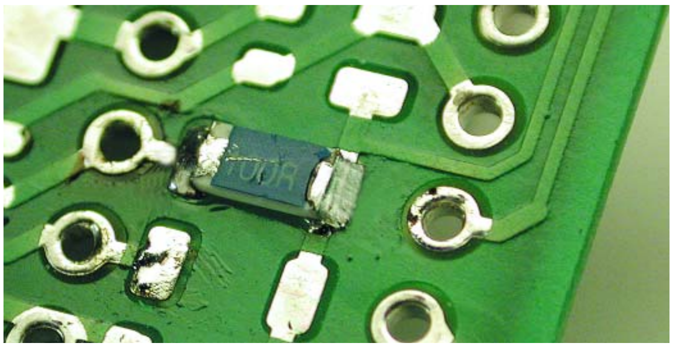
These component packages are broadly similar and can be treated identically for the purposes of soldering.

Place the component down on the PCB pads and line it up accurately using tweezers. Put a small bead of solder on the iron bit and tack down one end of the component with this solder. Don't worry about making a perfect joint just yet because we are just fixing the component in the correct position.





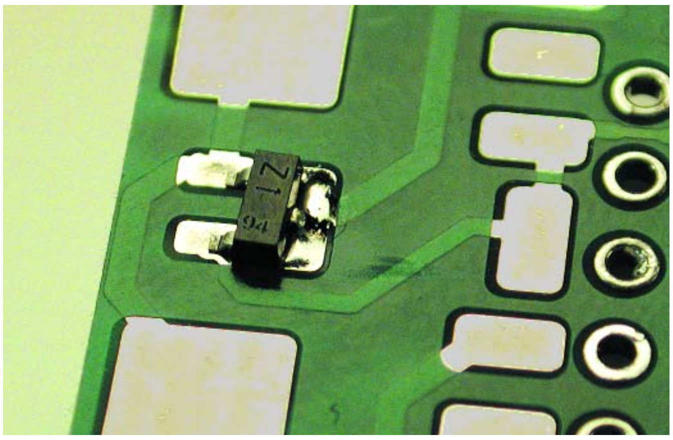
Now go to the other end of the component and place the soldering iron tip in contact with both the component lead and the PCB pad. The iron will be at 90° to component lead axis. Feed solder into the joint between the component and pad. The flux will flow over the joint, drawing solder neatly around the pad and component by capillary action. Don't feed in too much otherwise a ball will start to form. The optimum amount will give a concave shape to the finished joint surface a bit like a miniature version of a trumpet bell-end.



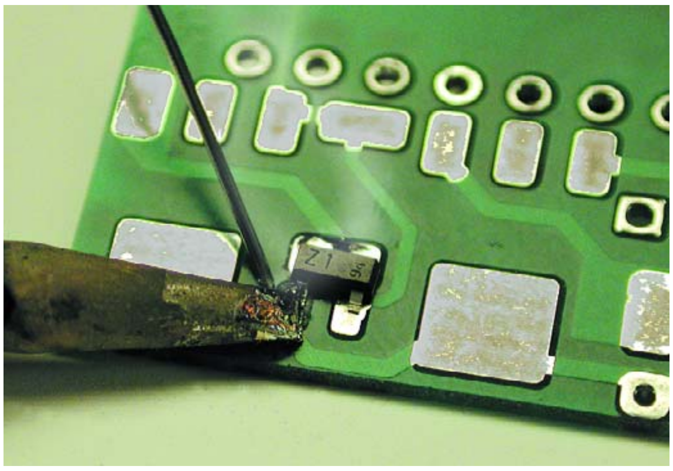
Back to the first joint, we can now re-heat it with the iron and add a touch more solder to make it flow around the pad and lead. Take the iron away as soon as possible to avoid component overheating and don't add too much additional solder.

MELF Diodes
The MELF outline is cylindrical which makes it a little tricky to handle and the lead area in contact with the PCB pad is less than a resistor outline, producing a much smaller capillary effect when soldering. A little additional flux added to the PCB pads will help here. Otherwise follow the procedure for soldering resistors.

SOT23 Transistors and chip electrolytics
These components have relatively widely-spaced short leads for soldering to the PCB pad. We will show the procedure for an SOT 23 type transistor package.



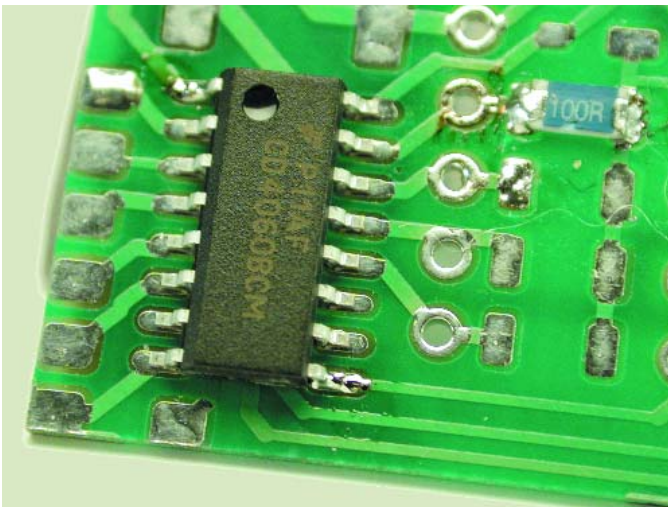
As before, position the component and tack down one of its legs with a small blob of solder.

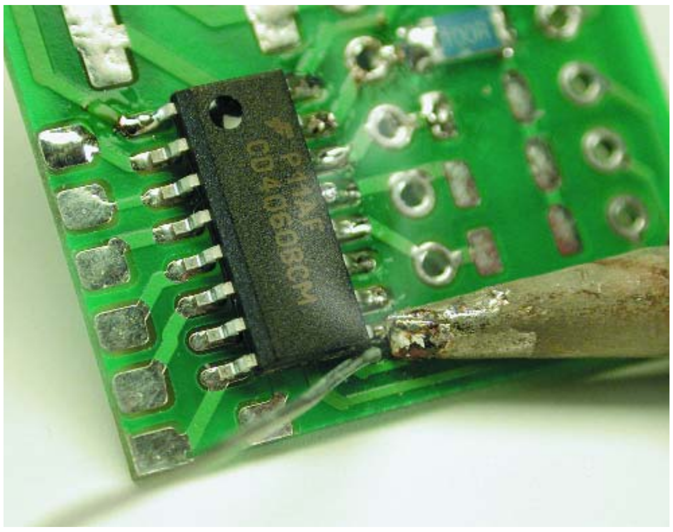


Now to the remaining leg(s), hold the iron bit at 90° to the leg axis using light pressure and feed solder over the leg, from above this time, so that solder coats the leg and flows down over the pad. This will produce a more evenly finished joint. Go back to the first tacked leg and reheat it with a little extra solder as above.

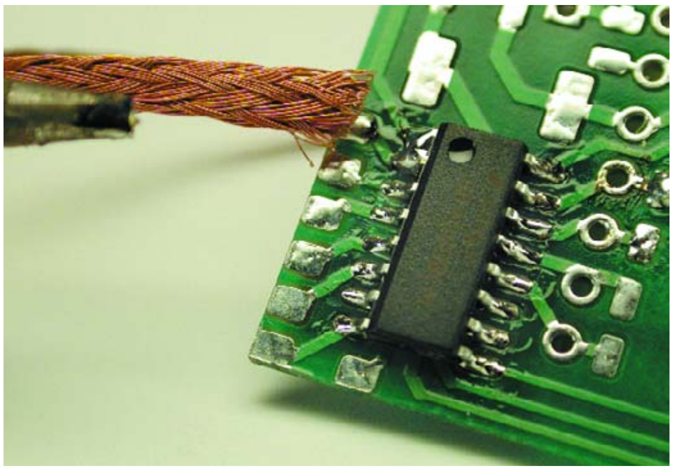
Tackling components with finely pitched leads.
As a example for soldering these types of IC's we will use a QFP type of package outline. The leg spacing is 0.635 mm (not much bigger than the soldering iron bit diameter).

Dab some flux paste over each of the PCB pads.
Position the component accurately over the pads (make sure pin 1 on the IC corresponds to pin 1 on the PCB). Using a soldering iron, tack down two of the component legs (choose diagonally opposing corner legs). This time it's easier if the soldering iron bit is held along the component lead axis rather than at 90° to it.





Now we get to the tricky bit, firstly this is your last chance to check that the component really is accurately centred. Each component leg is now soldered identically: With the bit and component lead in line, quickly heat up the lead and solder pad then feed solder onto the bit.



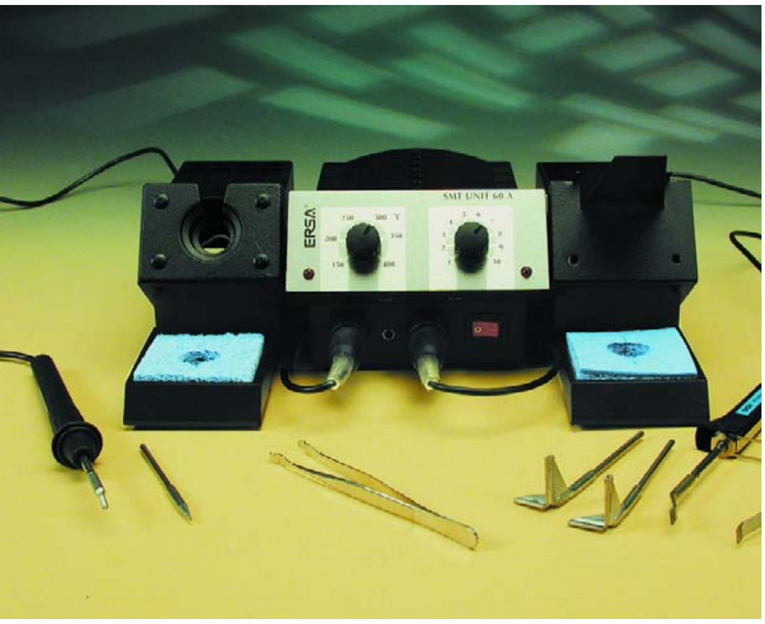
Too much solder will cause a blob to form which bridges over to an adjoining pad. Don't worry because this can be simply removed with the application of solder wick. Position wick over the blob and apply light pressure with the iron bit until solder is drawn into the wick. Finally re-solder with a little additional flux and solder.

Desoldering SMDs

Desoldering can be a bit of a headache with a standard soldering iron because the component is rigid and mounted directly on the PCB surface. First begin by removing as much of the solder in the joints as possible with solder wick, the application of a little flux will maximise the amount of solder removed. Chip components can be heated to 450°C. Place the soldering iron on one side of the component and tug gently with tweezers until the component can be moved. Once removed the component should be discarded.

Components with lots of leads need a certain amount of patience. One technique is to heat up a group of adjacent leads and bend each leg upwards with the help of tweezers to prevent them re-attaching to the pad. With QFP outlines it is best to start at one corner of the chip and desolder individual legs alternately left then right, proceeding away from the corner and around the outline. Alternatively you can snip through the leads with suitable side cutters and desolder each severed leg, removing them with tweezers. As before clean up the pads with solder wick.

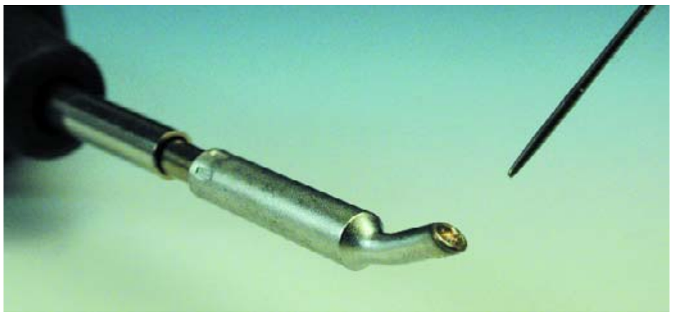
Using a solder station



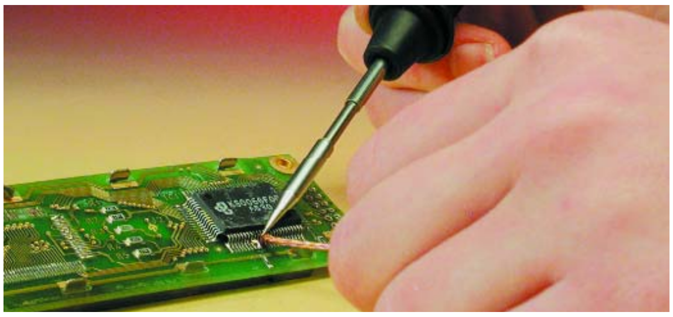
During the preparation of this article a solder station type SMT Unit 60 A (Figure 58) produced by ERSA was used to explore the advantages of working with a dedicated SMD soldering station. This station comprises a soldering iron with interchangeable bits and a pair of heated desoldering tweezers. Antex and Weller are also well known soldering iron manufacturers and produce similar soldering stations. The techniques of soldering and desoldering using these stations are slightly different compared to the standard soldering iron that we have been using up until now.

Soldering components with finely pitched leads

Once again we will use a QFP packaged device. As before, the pads should first be treated with flux.
The end of this microwell soldering bit is offset at 30° and has a small cup-shaped tip for holding a pool of solder.
Position the component on the PCB and fix it by tacking two diagonal opposed leads to their pads with solder.

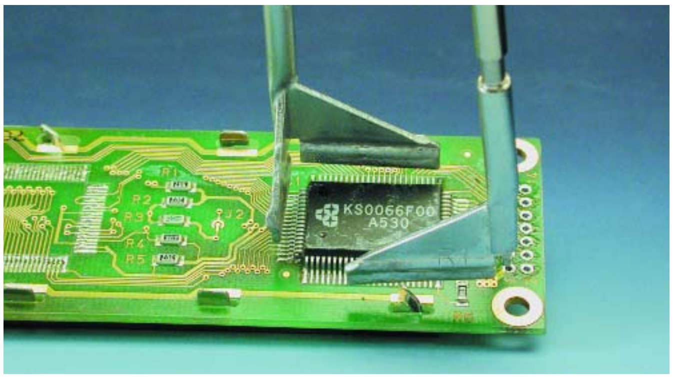


Working with the cup bit it is necessary to flux the joint and then place a small amount of solder into the cup. The cup shape ensures that the solder will remain in place until the tip is placed over the component lead, when the flux will draw the solder out of the cup and into the joint. Judging the right amount of solder in the cup can be tricky, too much will result in a solder bridge (use solder wick to clean up) but with experience this technique is quite effective.

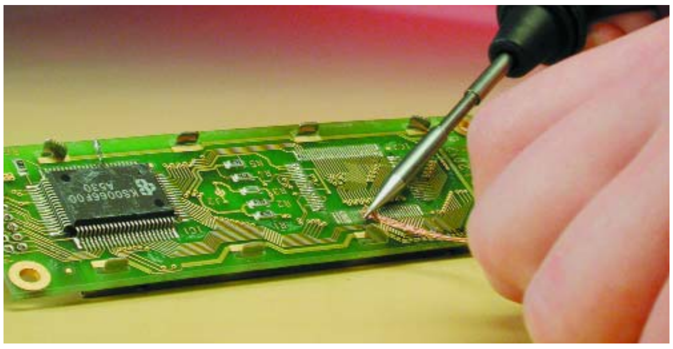


Desoldering QFP outline ICs

Removing these components with the special desoldering tweezers is much quicker and easier than is possible with a standard soldering iron. For each different type of component housing there is a corresponding set of tweezers to desolder



and remove the IC. As an example we will look at the procedure for removing an IC with a QFP type of outline.
Position the pre-heated tweezer jaws over the component



leads and wait for the solder to flow.
A gentle lift of the tweezers will pull the IC clear of the PCB. Finally use some solder wick to clean up the pads.

Conclusions

So working with SMD components need not be so stressful after all, as long as you keep in mind the basic rules: choose a fine soldering bit and work quickly, remember that the joint area is quite small so that it reaches soldering temperature very quickly. If you haven't made the joint after a few seconds, take the iron off and try again later when everything has cooled down. Leaving the bit on the joint for a prolonged period is a sure way to destroy the component and loosen the PCB pad bonding. If you anticipate a lot of SMD work in the future then it will be worthwhile investing in a specialist solder station. These are not cheap (the ERSA SMT unit 60A used in this article retails at around £366 and is stocked by RS Components www.rswww.com). In the long run these speed up SMD work especially if you will be handling large ICs with finely pitched leads.

Providing you have the right tools, a steady hand and are prepared to spend a little time practicing, there really is no need to panic, even if you only have a standard soldering iron it is quite possible to be successful with SMDs.

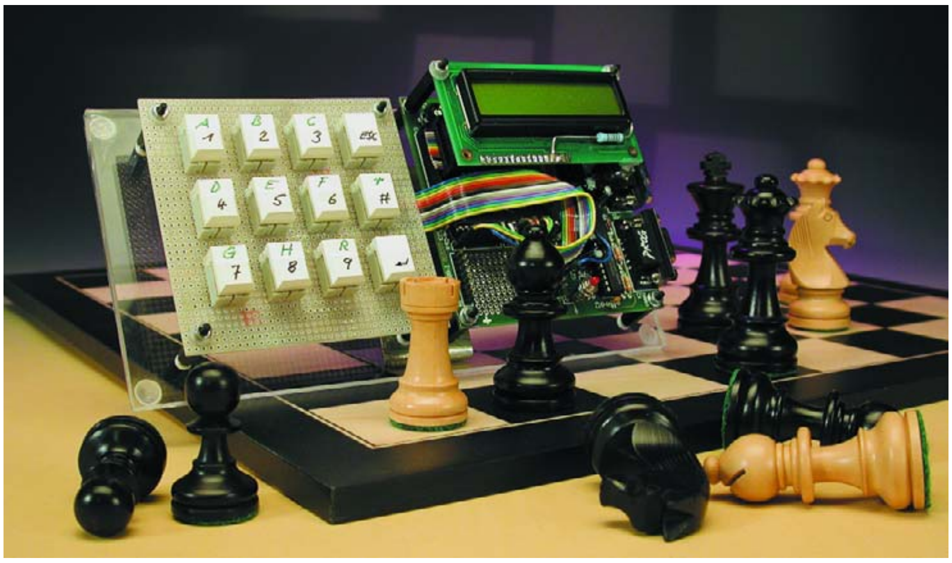
(020305-2)

Chess Computer using the Flash Micro Board

presenting Deep Evelyn, the winner of the Flash Micro Board competition

by P. Azad and T. Gockel

While the subject of chess using large computers can be dealt with in short order (the computer always wins), this project is intended to rekindle interest in user-programmed computer chess by making the task more challenging. The result is a chess computer based on the *Elektor Electronics* Flash Micro Board described in the December 2001 issue.



People have been fascinated by machines and computers that can play chess since the time of Maezel's chess robots. Many people have already attempted to create something that exceeds their own capabilities, occasionally venturing down very interesting paths in the process. For instance, just before the first computer came into existence, the mathematician Alan Turing developed a chess program on index cards — and 'ran' this program using himself as the processor.

Nowadays, you might think that there's not much more to be said about computer chess, since hardly anyone can beat the computer. However, with the Deep Evelyn project we want to rekindle interest in user-programmed computer chess by making the task a bit more challenging: computer chess using an 8-bit Atmel Flash microcontroller with 8 or 12 kB of memory, programmed in C!

Software

Once we had the idea of implementing a chess program using the Atmel microcontroller, we quickly found basic C software for the move generator, position evaluator and recursion (alpha/beta method) on the Internet. However, tests using various C compilers (Rigel and Wickenhüser) failed, since the compilers could not handle the (already stripped-down) syntax, stumbled over the recursion or produced excessively large compiled code. The 'sdcc' open-source compiler, to be described in a future issue of *Elektor Electronics*, ultimately appeared to be the most suitable. What remained was to modify the existing source code to meet our needs. This required the following changes:

– dispensing with standard

Code Segment 1

```
// old...
void foo()
{
 char i;
 for (i = 0; i < 8; i++)
 {
    // common ...
 }

 for (i = 0; i < 4; i++)
 {
    // common ...
    // extra ...
 }
}
```

```
// new...
void help(char n)
{
 char i;
 for (i = 0; i < n; i++)
 {
    // common ...
    if (n == 4) { // extra ... }
 }
}

void foo()
{
 help(8);
 help(4);
}
```

libraries
– reducing the size of the compiled code
– adapting the program to the Atmel memory model
– optimising execution speed

In order to dispense with standard libraries, it was first necessary to remove all `#include` instructions and replace the missing commands with our own code. We removed all I/O instructions and added suitable comments at the affected locations to facilitate inserting new I/O routines.

The first compilation using sdcc produced slightly more than 16 kB of code. Considering the amount of I/O code still to be implemented, this had to be reduced by at least 9 kB. Rather than giving up the spot, we scaled down our objectives. As described below under 'Hardware', the pin-compatible AT89S53 has more Flash memory (12 kB of instead of 8 kB), so in the worst case a 5 kB reduction would be just enough.

To reduce the amount of compiled code, the source code was modified as follows:
– stripping out constants
– factoring out similar code
– replacing the `int` data type with `byte (char)` where possible

Stripping out constants is relatively easy. If a particular value is computed several times within a function, it only has to be computed once and assigned to a variable, which is then used instead of recomputing the value. This not only reduces the amount of compiled code, it also increases efficiency. However, it is easy to introduce errors by removing presumed constants that are actually not constants. All such changes must be carefully considered and checked by running the program.

Factoring out similar code is significantly more complicated. Except from the trivial case of identical blocks of source code, it requires intuition and creativity. If two pieces of code appear to be similar, they can often be combined into a single function, for instance by introducing additional variables and IF queries (see **Code Segment 1**).

Such changes, if made in a reasonable manner, yield a significant reduction in the amount of compiled code, but they increase execution time. Just calling a function twice costs time, due to the implicitly executed push and pop operations for the return address, possible parameters and context saving. Additional code for differentiating the various tasks, such as IF queries, adds to this.

Replacing the `int` data type with *char* (if possible) yields a fundamental improvement in code size, memory usage and execution time. The sdcc compiler interprets `char` as an 8-bit data type and `int` as a 16-bit data type. Since the AT89S53 has an 8-bit ALU, an operation such as multiplying two `int` variables is split into several 8-bit multiplications and additions.

It is generally not possible to simply change `int` to `char`, since this can lead to a considerable loss of information (65,636 vs. 256 values). However, it was possible to replace all instances of `int` by `char` except for the evaluation function. Originally, the `#define` statements for the various chess pieces were used for both differentiation and evaluation. By splitting the IDs and values of the individual pieces into separate `#define` statements, it was possible to reduce the playing-field array to the `char` data type, allowing all routines except the evaluation function to use significantly less costly 8-bit operations.

Adaptation to the Atmel memory model was necessary because the Atmel microcontroller has only 256 bytes of internal RAM. First, all global variables (primarily the arrays for the playing field, moves etc.) were moved to the 32-KB external memory by putting the keyword `xdata` in front of the declarations.

Once we had gotten this far, we could let the chess computer play against itself and follow the moves using the serial interface. The first move was 'a8a9' (!), which led to a tedious debugging session. After we finally cut back the complete chess program to around five lines that simply filled an array using a FOR loop and read it out again, and found that this simple program did not work properly, it was clear that there was something wrong with the sdcc compiler. Totally discouraged, we disabled all compiler optimisation — and were rewarded with success!

After this, our chess computer gave 'a2a3' as the first move, which at least showed a certain similarity to 'e2e3', the move specified by the same program running under Windows…

Although the 256 bytes of internal RAM should have been more than adequate for the remaining local variables and the stack, it turned out that practically all variables not marked with `xdata` are potential sources of errors. At this point, we had enough experience to not worry about why or try to systematically nail down the cause of the problem. Instead, we adopted the drastic solution of entirely avoiding the use of internal memory. In particular, this meant:
– replacing function parameters with global variables
– replacing local variables with global variables
– implementing our own runtime stack

The alpha/beta function is the only recursive function in the chess program. This means that all of its parameters, local variables and return address are implicitly placed on the runtime stack each time it is called, so they

Code Segment 2

```
// old...

int value;
int foo(char a)
{
  char local;
  if (a == 5)
      return a;
  // code1 ...
  value = foo(a + 1);
  // code2 ...
  return value;
}

// new

xdata int value;
// for current stack content
xdata char a;
xdata int local;
xdata char ret;
// corresponds to return value register
xdata char retval;
int foo()
{
begin:
  // fetch parameters and return address
  a = stack[stackp].a;
  ret = stack[stackp].ret;
  if (a == 5)
  {
    retval = a;
    if (ret)
      goto ret;
    return;
  }
  // code1 ...
  // update stack contents (save context)
  stack[stackp].a = a;
  stack[stackp].local = local;
  // push (parameters & return address)
  stack[++stackp].a = a + 1;
  stack[stackp].ret = 1;
  goto begin;
ret:
  // pop (restore context)
  a = stack[--stackp].a;
  local = stack[stackp].local;
  ret = stack[stackp].ret;
  // fetch return value
  value = retval;
  // code2 ...
  retval = value;
  if (ret)
    goto ret;
  return retval;
}
```

can be read out and removed from the stack each time it executes a return. All of this, which is normally handled by the programming language, now had to be explicitly programmed.

A `sruct` array encompassing all of the above-mentioned variables is used for the stack. Since labels are not types in C and thus cannot be assigned to variables, return addresses must be implemented indirectly using IDs and IF queries. It would have been conceivable, and certainly somewhat more efficient, to use an inline assembly-language solution. **Code Segment 2** shows an example of the modified code.

With regard to optimising execution speed, the colours of the pieces are distinguished in the chess program by the sign of the ID: a positive ID describes a white piece, while a negative ID describes a black piece. The efficiency of the evaluation was improved by a factor of around 2 by using two separate branches in the evaluation algorithm to differentiate between the white and black pieces, instead of handling the difference by repeatedly multiplying by a variable (1 or –1) in a single block of code. This allowed so many multiplications to be eliminated that the compiled code was smaller, even though the number of lines of source code was nearly doubled.

It also proved to be very practical to use `#define` and dummy `#define` statements (`#ifdef WIN32 ... #enddef`) to allow the source code to be compiled as a console program running under Windows, so basic program operation could be checked and/or debugging could be conveniently performed in a Windows environment.

In addition, output via the serial interface was a great help in the development phase. The necessary routines can easily be taken from a small demo program (`hi.c`) provided with the sdcc compiler. An even more convenient solution would be to use the freeware program PaulMon2, but we were not able to test this in the short amount of time available (ten rather long evenings).

Operating Instructions

Enter a move when prompted to do so.
For example, to enter **e2e4**, press the **E**/, /**2**, **E**/ and /**4** buttons in sequence.
To correct a false entry or delete a complete line, press **ESC**.
To confirm or transmit your entry, press **RETURN** or <-, respectively.
Short castling: press the **#** button (now labelled /**r**).
Long castling: press the /**l** button (now labelled /**R**).
To select the computer's move (microcontroller computes the next move): press **RETURN** or <- without entering a move.
New game: press the Reset button on the Flash Board.

Hardware

For this project, we concentrated on the software and intentionally avoided fancy inputs and outputs. Nevertheless, we made a few small changes to the hardware of the Atmel AT89S8252 Flash Micro Board as described in the December 2001 Issue of *Elektor Electronics*.

The voltage regulator was replaced by an equivalent low-drop type (such as an LT1086CT-5, which however has a different pin arrangement (gnd/in/out instead of in/gnd/out), or an LM2940 or equivalent) and the input voltage was reduced, since the original 7805 became much too hot with the LCD background illumination switched on (with a 10Ω series resistor), due to its high drop-out voltage (around 3 V).

User input is via a keypad composed of 12 digital pushbutton switches fitted in the prototyping area. Since each switch is wired to ground, 12 port pins are required. There was no need to economise on port pins by using a matrix connection. The assignment of port pins to switches can be easily seen in function `AskPlayerMove()` in the source code (`userinit.c`). The switches are debounced by a simple delay (50 ms, which can be changed as desired). The button arrangement matches a telephone keypad, and the button functions and operation are practically self-explanatory (see the Operating Instructions box). Note that the program does not check entered moves for correctness, so if you want to cheat (yourself) you are free to do so.

The 16 × 2 LCD is connected to the address/data bus. It can be addressed in the syntax of the selected C compiler (sdcc) using the `xdata` (external RAM access) command. An example of such an access is:

```
volatile xdata unsigned char
at 0x8000 cmd_write;
```

The board presently holds an Atmel Atmel AT89S53-24PC, which is pin-compatible with the AT89S252 but has 12 kB of Flash program memory instead of 8 kB (but no EEPROM). However, with the stripped-down user interface, the program would probably also run in the AT89S252. The actual amount of program memory required by the compiled code can be seen under *CSEG* in the `.map` file generated by the compiler.

Results and prospects

The current version of Evelyn is a chess computer that cannot compete with professional devices or PC programs, although it can beat an amateur. However, the purpose of this project was not to create a high-performance chess computer, but rather to show how a relatively complex C program can be transferred from a PC environment to a much less powerful microcontroller environment. The references provide further information that you might find useful in your own projects.

One last thing: the name of the chess computer was inspired by the following highly informative sdcc compiler message:

```
Warning: conditional flow
changed by optimiser
'chess.c(386)': so said
EVELYN the modified DOG.
```

(031001-1)

For further information:

www.artilect.co.uk/lego/default.asp?page=Chess
'The Chess Robot Project': chess on Lego Mindstorm (download)

http://sdcc.sourceforge.net
Home page of sdcc (small-device C compiler): documentation and download

www.turbobit.com/mem51.html
Information about the memory mapping of 8051 derivatives with reference to the sdcc .map file

www.geocrawler.com/archives/3/3278/2000
sdcc user forum

www.estpak.ee/~nq002a/liter/picapps.pdf
'Interfacing a Matrix Keypad' and other PIC applications

www.pedram-azad.de
Download URL for the latest version of the chess computer program and related documentation

Active Loudspeaker System (2)

A subwoofer for use with multimedia loudspeaker systems

Design by T. Giesberts

As with most small enclosures, the bass response of the active 2-way loudspeakers described last month has its limits. The compact subwoofer described here will therefore be a welcome addition for most enthusiasts. With its adjustable filter this subwoofer will also work very well with other small (satellite) speakers.

As a rough guide the bass response of loudspeakers tends to improve in proportion to their dimensions. This is not completely true, since there are a few tricks that can be used to get a reasonable bass response from a relatively small enclosure. But these come at a cost of a reduced power handling capacity, efficiency and sound pressure level that the loudspeaker can produce.

Of course, what everybody wants is a loudspeaker with a volume of a few litres, using a small woofer, which is still capable of reproducing perfect sounding bass guitar or pipe organ sounds down to 30 or 40 Hz. But unfortunately this is just a physical impossibility. Such a bass response requires a larger air displacement and this needs larger speaker diaphragms and enclosures.

The active 2-way system published last month had a 13 cm woofer in a closed box with a volume of about 4 litres. This really is a very small sized enclosure and the response graph made clear that not much should be expected of the response below 100 Hz. We decided not to use artificial (electronic) means of boosting the low frequency response, because we already had our eye on a much better solution: a separate subwoofer that is placed underneath the table (or desk), inconspicuously adding another octave to the response of the two miniature loudspeakers.

Dual bass reflex

So how did the design for the subwoofer come about? We tried to get the subwoofer to complement the previously published 2-way loudspeakers as well as possible. This meant we had to have a good quality design, without it becoming too large or costly. Since the intention was that the subwoofer could be easily placed underneath a desk we based the dimensions on those of a medium sized PC tower case. In this way we arrived at a volume of roughly 50 litres.

Next we started looking for some speakers that would fit in such an enclosure and would also give an acceptable bass response. For the speakers to fit in a PC tower sized enclosure they couldn't be very large. Again they should preferably have magnetic shielding, so they won't cause problems when placed close to a monitor or TV.

After searching for a while we found the W170SC made by Visaton: a robust 17 cm woofer with a load handling capacity of 50 W, a resonant frequency of 36 Hz and a noticeably large peak diaphragm displacement of 20 mm. The W170SC is most at home in a bass reflex enclosure with a volume of about 25 litres and is then capable of reproducing frequencies down to about 45 Hz. Since a single 17 cm speaker only produces a limited amount of air displacement, we have used two of them in a 'double' subwoofer using an enclosure with a volume of 50 litres.

Just like the 2-way loudspeakers, the subwoofer is also an 'active' design, with an electronic filter and its own power amplifier. To give this design a wider usefulness, the cut-off frequency has been made adjustable between about 75 Hz and 145 Hz. The subwoofer can therefore be combined with virtually any compact (satellite) speakers. The power amplifier used here is the same integrated dual bridge amplifier that was used in the 2-way loudspeakers. Since the WS170SC has an impedance

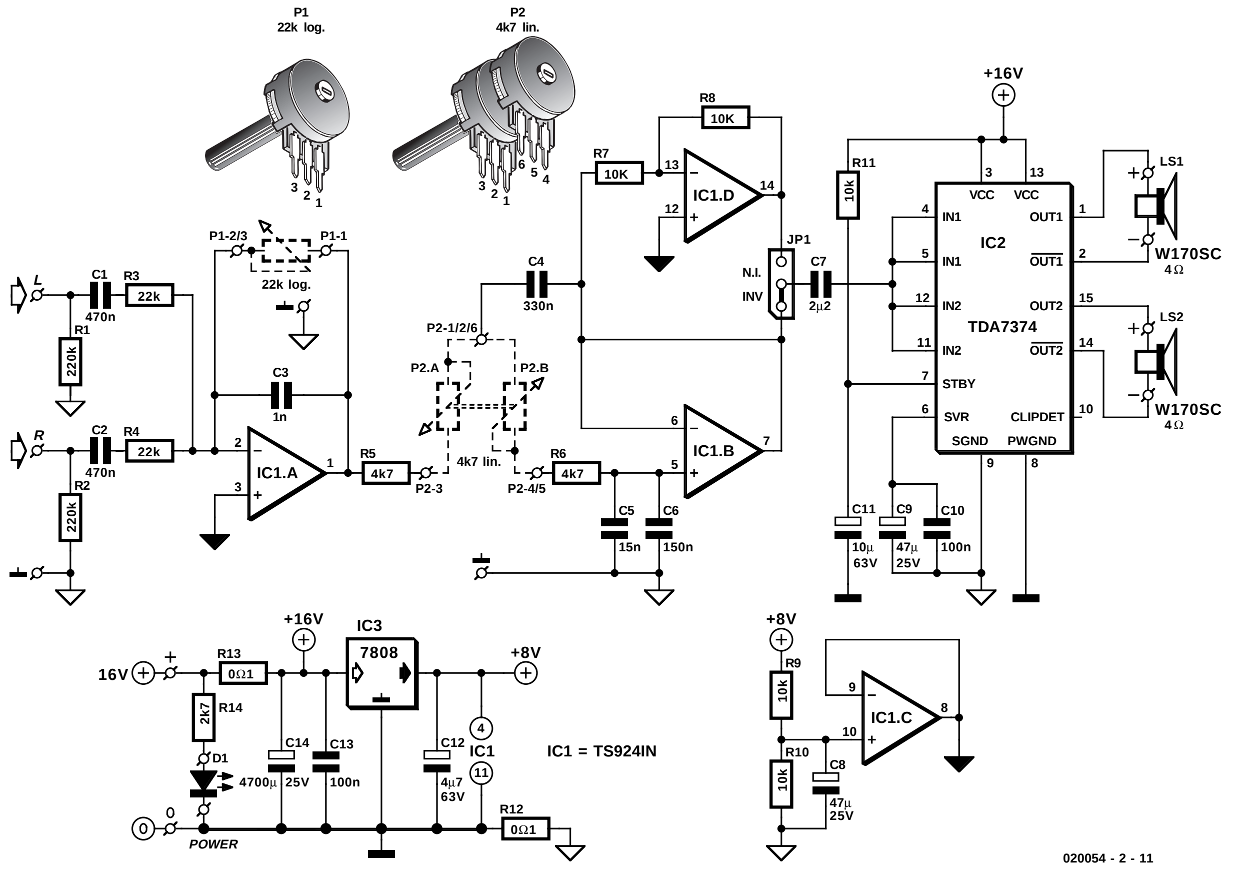


Figure 1. The circuit is very similar to the active 2-way system.

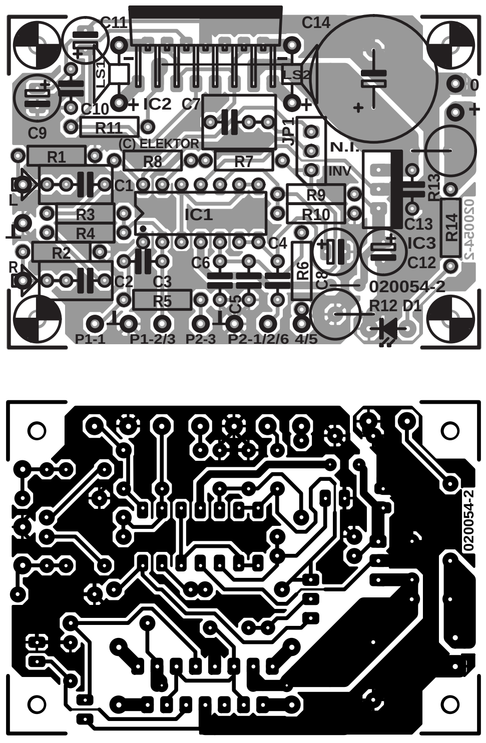


Figure 2. All available space is utilised on this PCB. Note the two wire links!

COMPONENTS LIST

Resistors:
R1,R2 = 220kΩ
R3,R4 = 22kΩ
R5,R6 = 4kΩ7
R7-R11 = 10kΩ
R12,R13 = 0.1Ω 5W
R14 = 2kΩ7
P1 = 22kΩ logarithmic. mono
P2 = 4kΩ7 linear. stereo

Capacitors:
C1,C2 = 470nF
C3 = 1nF, lead pitch 5mm
C4 = 330nF, lead pitch 5mm
C5 = 15nF, lead pitch 5mm
C6 = 150nF, lead pitch 5mm
C7 = 2µF2 MKT,
lead pitch 5 or 7.5 mm
C8,C9 = 47µF 25V radial
C10,C13 = 100nF, lead pitch 5mm
C11 = 10µF 63V radial
C12 = 4µF7 63V radial
C14 = 4700µF 25V radial, lead pitch 7.5mm, max. diameter 17mm

Semiconductors:
D1 = LED, red, high-efficiency, 5mm
IC1 = TS924IN ST (Farnell)
IC2 = TDA7374B ST (RS Components)
IC3 = 7808

Miscellaneous:
JP1 = 3-way pinheader with jumper
LS1,LS2 = Visaton W170SC 4 Ω Conrad Electronics)
2 x bass reflex tube, diameter 72mm, length 145mm (Conrad Electronics # 34 26 10-60)
Heatsink: e.g., SK100 (Fischer), height approx. 50mm *
Sealant tape: MDM-5 (Monacor/Monarch)
BAF wadding
PCB, order code **020054-2** (see Readers Services page)

* see text

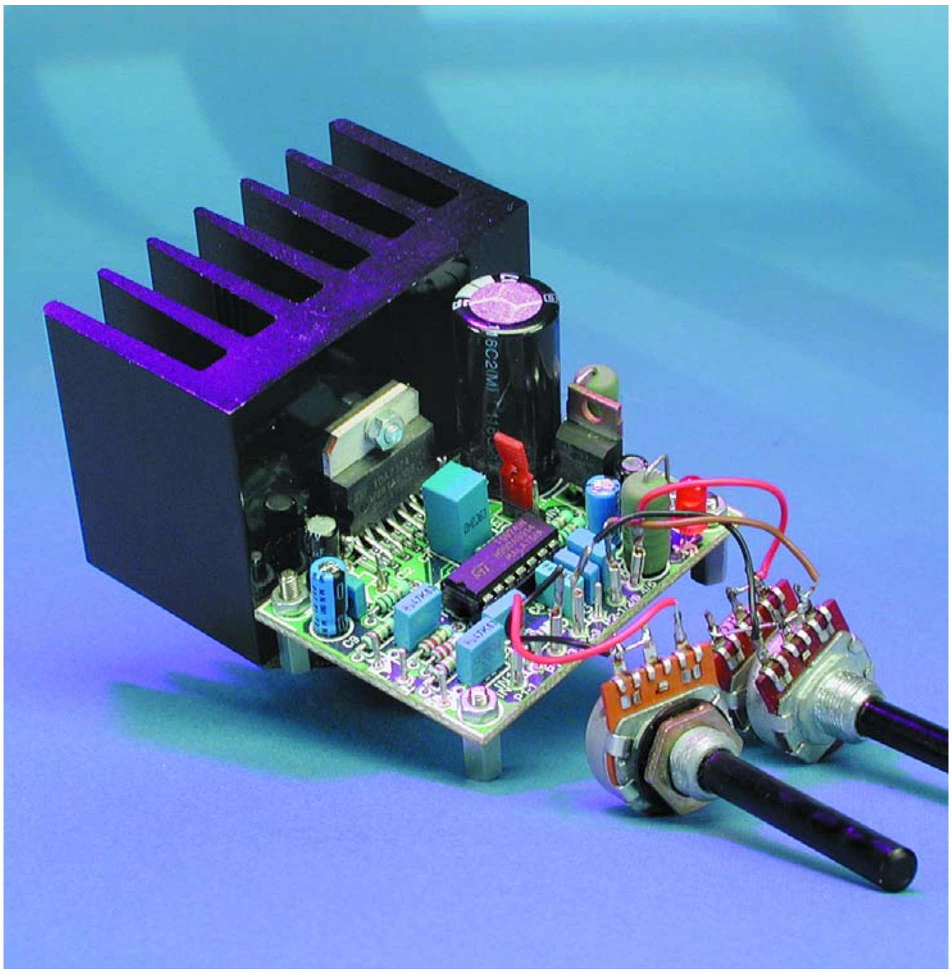


Figure 3. The completed PCB for the subwoofer, including its heatsink.

of 4 W, the amplifier has a power output of around 20 W per woofer.

In practice we found that the frequency response and sound pressure level produced by the subwoofer perfectly complemented the response of the small 2-way loudspeakers.

Mixer and low-pass filter

When you look at the circuit diagram shown in **Figure 1,** the similarities to the design for the active 2-way loudspeaker should be immediately obvious since the same ICs have been used. Again we've used a TS924IN quad rail-to-rail opamp (IC1) for the input stage and filter. Voltage regulator IC3 supplies a stabilised 8 V to this opamp. R12 has been added to separate the signal ground and the supply ground when a single power supply is used for several channels. The fourth opamp in the TS924IN (IC1c) is once more used to create a stable virtual ground that is at exactly half the supply voltage.

The stereo output from the preamplifier is fed to the L and R inputs. Since our subwoofer is monophonic, like most are, (stereo information below 100 Hz is virtually non-existent), the left and right channels are first combined using a simple mixer (IC1a). This has been implemented in its inverting form so that the gain can be turned right down to zero with P1. The mixer can therefore accept signals at practically any level and could even be connected to the output of another power amplifier. C3 suppresses high frequency interference and restricts the bandwidth to about 7 kHz.

The crossover filter consists of a 2nd order Butterworth low-pass filter. This filter is built round IC1b and has its cut-off frequency adjusted using stereo potentiometer P2. When both P2a and P2b are set to 0 W (effectively a short-circuit), the cut-off frequency is about 145 Hz. When the resistance is turned to its maximum, the cut-off frequency moves to about 75 Hz. For use in conjunction with the active 2-way loudspeakers a cut-off frequency of 100 Hz is ideal, but P2 covers a wider range so the subwoofer can also be used with other (satellite) speakers. We found that the subwoofer could be adapted for use with various satellite speakers by adjusting the volume and frequency controls to their best settings through trial and error.

As a bonus we've added an inverter (IC1d) to the output of the filter, which is selected when jumper JP1 is in the upper position. Depending on the relative positioning of the subwoofer and satellites, a (subtle) improvement can sometimes be heard when the phase of the subwoofer is inverted. We should point out that the output of IC1d is actually in phase with the input signal, since the filter round IC1a is also inverting. If you wish to use the phase switching facility after encasing the electronics we recommend that you replace JP1 with a changeover switch.

Power amplifier

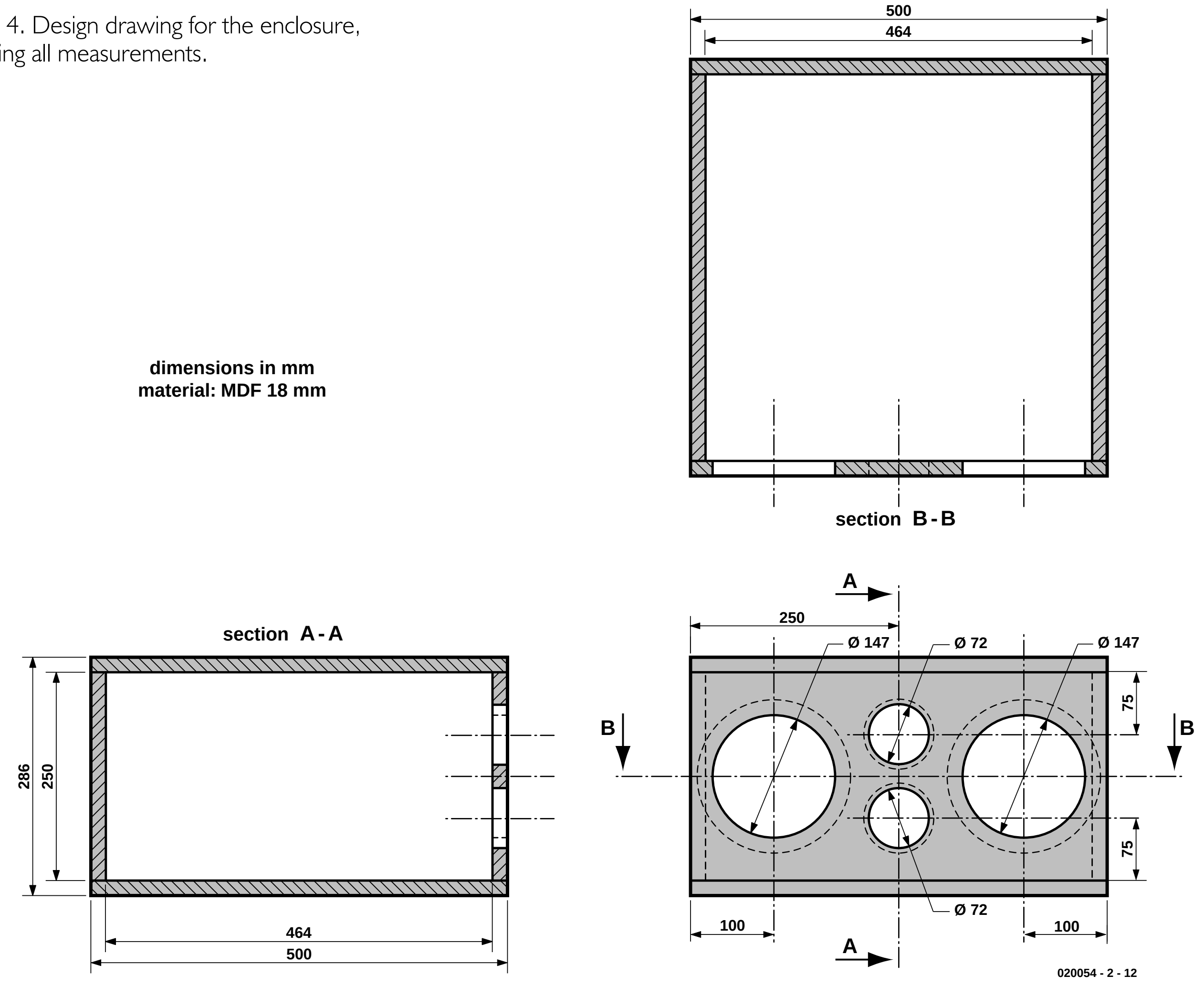
Because our experience with the active 2-way system was entirely positive, we decided to use the same TDA7374B integrated power amplifier. The small number of external components required and the good internal protection circuits are a big advantage of this IC. Another excellent feature of this IC is that it contains two amplifiers, so that each woofer can have a separate power amplifier.

Since the impedance of the woofers is 4 W, the TDA7374B delivers a bit more power than with the two-way loudspeakers; with a supply voltage of 17 V the power output is 2 × 20 W. The heatsink used should of course be capable of handling this. When a continuous signal is played at full power, you should in theory use a heatsink with a thermal resistance of 1.5 K/W. But with normal music a heatsink rated at 2.5 K/W is sufficient in practice. An example of this is the SK100 made by Fischer, which has been used in the prototype shown in **Figure 3**.

Printed circuit board

All of the electronics for the subwoofer fits on a small PCB measuring only 44 by 63 mm. That is quite a bit smaller than the PCB for the 2-way loudspeaker! **Figure 2** shows the layout and

Figure 4. Design drawing for the enclosure, including all measurements.



parts numbers.

Again we find the input pins on the left-hand side of the PCB, the supply pins top-right and the speaker pins on either side of the power amplifier IC. The connections for the potentiometers are at the front of the PCB. Take great care when connecting stereo potentiometer P2; the numbers next to the pins correspond to those in the circuit diagram. As long as the connections between the PCB and potentiometers are less than about 4 cm you can use standard connecting wire. If you have a longer connection then you must use shielded audio cable. With that in mind there are earth pins included near P1 and P2 for connection to the shield.

It is best to start with the wire links when populating the PCB; that way you won't forget them later. There are two of these: one near R12 and the other underneath the pins of IC2. The last link could also be soldered on the underside of the PCB, but in either case you should use isolated wire. The power amplifier IC has purposely been placed at the edge of the PCB, making it easier to mount the heatsink. The use of isolating material between the heatsink and the tab of the IC is recommended.

The rest of the construction shouldn't present any difficulties and as long as you keep to the parts list and the layout in **Figure 2** there is no reason why the circuit shouldn't work first time. In **Figure 3** you can see what a

Amplifier Specifications (with a 17 V supply voltage)

Input impedance		2 x 20 kΩ
Sensitivity (20 W/4 Ω, P1 max, P2 = 0)		250 mV (L=R)
Distortion + noise (65 Hz, B = 22 kHz, P1 max., P2 = 0)		0.01 % (1 W/8 Ω)
		0.032 % (1 W/4 Ω)
Bandwidth	P2 = 0	18.5 Hz - 160 Hz (relative to 65 Hz)
	P2 = max.	16.5 Hz - 86 Hz (relative to 44 Hz)
Output power		2 x 20 W (4 Ω)
Quiescent current	0.16 A	

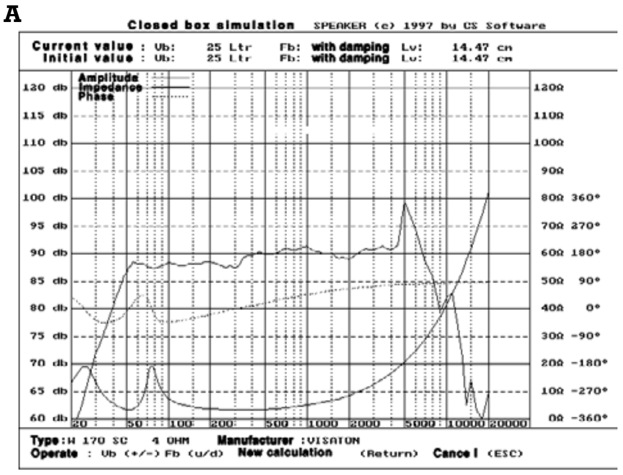
In addition to the measurements taken of the electronics we also have three graphs that illustrate the performance of the active subwoofer as a whole.

Graph A shows the computer simulation of a W170SC in a 25 litre bass reflex enclosure. With the bass reflex port tuned to 46 Hz the graph is almost flat up to 300 Hz. There are some fluctuations above this, but those frequencies are sufficiently suppressed by the active filter and will therefore not affect the response.

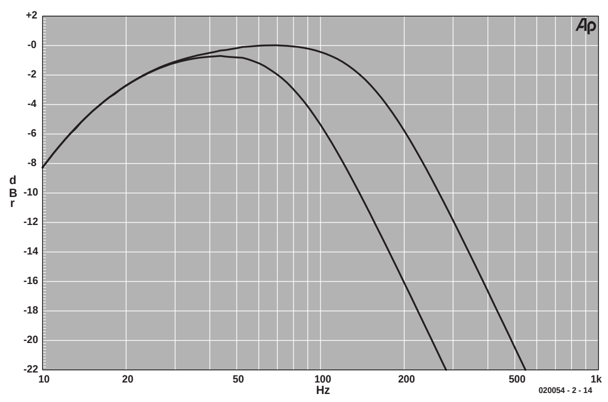
Graph B shows the frequency response of the power amplifier with P2 in its two extreme positions. The exact cut-off frequency as well as the response depends on the tolerance of the capacitors and potentiometer P2. When used in conjunction with the active 2-way system the cut-off frequency should be around 100 Hz; the range of P2 adequately covers this.

Graph C shows the measured frequency response of the complete system. This too has been measured with P2 in its extreme positions. The advantage of the smaller bandwidth is that the lower cut-off frequency is on average a few Hertz lower.

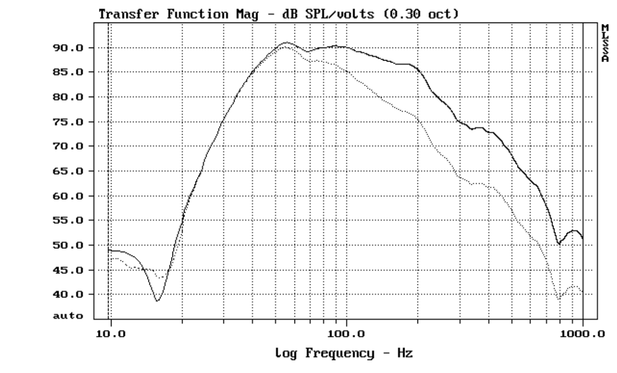
A



B



C



CURSOR: dy = -11.2637 x = 1000.0000 (2048)

FREQUENCY DOMAIN MENU: Go View Reference Acquisition Setup Transfer Macro QC
Overlay Calculate Printer DOS Units Library Info Exit
F1 for Help MLSSA: Frequency Domain

correctly populated PCB looks like, including the SK100 heatsink.

The electronics can either be mounted in a separate case or inside the loudspeaker enclosure. You can choose whichever suits you best. For the power supply a standard transformer/bridge/ electrolytic circuit can be used. A transformer rated at 12 V/50 VA, a 10 A bridge rectifier and a smoothing capacitor of 10,000 μF/25 V should be sufficient. As we mentioned in the article for the 2-way loudspeakers, we will shortly publish a 17 V power supply for this project. This is based on a switched mode design, keeping losses in the voltage regulation to a minimum.

Figure 5. The woofers have to be screwed down tight to avoid unwanted resonance.

Sturdy enclosure

We've already mentioned that the ideal enclosure for two W170SC woofers is a bass reflex enclosure with a volume of 50 litres. According to the calculations the enclosure should be tuned to a bit below 50 Hz. As chance would have it, the required bass reflect port could be implemented perfectly, using two standard reflex tubes made by Conrad, with a diameter of 72 mm and a length of 145 mm. These tubes therefore don't need to be cut to size, but can be mounted straight into the enclosure. In theory this provides tuning at exactly 46 Hz.

Figure 4 shows the complete design drawing for the subwoofer enclosure. It can be seen that this is a straightforward rectangular enclosure, which even the less experienced carpenter should be able to assemble. This will certainly be the case if you can find a timber merchant who will cut the panels to size.

Because this enclosure is substantially larger than that of the satellites and needs to support much heavier speakers, we used 18 mm thick MDF board for its construction. This also prevents unwanted resonances in the panels. In our prototype we glued two small 250 mm support beams between the two largest panels, although this isn't a necessity.

The openings for the speakers and the bass reflex tubes should be cut out carefully. A special tape (speaker foam insulation) should be used to obtain an airtight seal between the speakers and enclosure. The speakers should be screwed down tightly, either with self-tapping screws or (better still) using nuts and bolts. The bass reflex tubes should be fixed with wood glue.

Only a limited amount of acoustic damping is required in the enclosure. It is sufficient to cover all sides with a layer of polyester wadding and there is certainly no need to fill it up completely. The input connectors can be placed anywhere on the enclosure; their position will mostly depend on the orientation of the enclosure. Make sure that the polarity is clearly marked next to the connectors (or use loudspeaker terminals).

Positioning and sound

The positioning of a loudspeaker system consisting of a subwoofer and two satellites is fairly flexible. For an optimal stereo reproduction, the loudspeakers and your usual listening position should form an equilateral triangle. This is just a standard rule for stereo reproduction. The best position for the subwoofer should strictly be in the centre between the satellites. However, since these low frequencies have virtually no directional information, you'll find that it doesn't matter very much in practice. Anywhere in the vicinity of the satellites seems to give a good result.

It was a pleasant surprise when we listened to the two-way loudspeakers in combination with the subwoofer. You don't have to take our word for it, but once you've heard the small loudspeakers in combination with the subwoofer you won't want to part with it again. Even with the volume control at a relatively low setting you will get a much richer sound. The inherently puny response of the small loudspeakers disappears and the sound gets a lot more 'body'. The subwoofer produces a well-defined bass and in practice had power to spare.

Whether it is worth experimenting with phase switch JP1 depends very much on the positioning of the loudspeakers. When the subwoofer and satellites are all in one line, the sound is clearly better in phase rather than out of phase. If the woofer is nearer to or further from this line then experiment with JP1 to obtain the best result. You shouldn't expect a large difference though, since the change really is subtle.

Half a subwoofer?

A nice feature of this subwoofer design is that it can be made as a smaller 'trial' version. When you want to limit the size and/or cost and are happy with a slightly less punchy bass response, you can simply use half the design. So you'll only need **one** W170SC, **one** bass reflex tube and an enclosure of 25 litres instead of 50 litres. Should you change your mind at a later date, you could always build a second identical enclosure, making the end result exactly the same as the dual version just described.

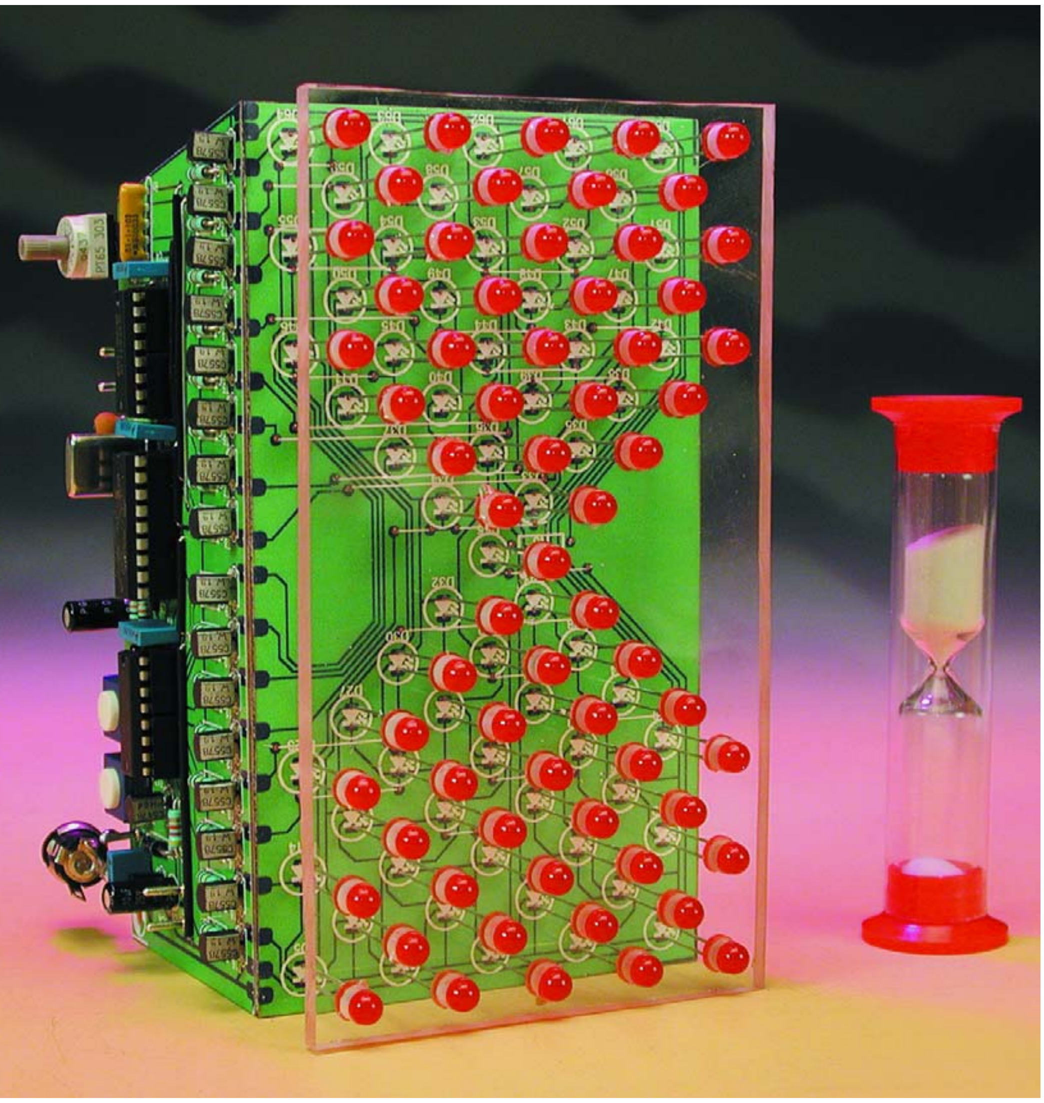
(020054-2)

Electronic Hourglass

an unusual PIC project

Design by S. Buchgeher

In a real hourglass, silicon dioxide (SiO_2) in the form of extremely fine quartz sand trickles from one bulb into the other. In the electronic version, the only silicon to be found is in a small, oscillating wafer of quartz and a few chips in the PIC microcontroller and transistors. And instead of grains of sand, points of light trickle downwards…



Of course, the points of light in the electronic hourglass do not have any direct relationship to the silicon in a conventional hourglass, since there is no silicon among the various elements (primarily gallium and arsenic) in the chips of the LEDs used here. Aside from that, you might wonder why anyone would want to build an electronic hourglass.

One reason is that this project provides an excellent example of how to program and use a PIC microcontroller. By studying the listings (which can be downloaded from the *Elektor Electronics* website), you can see how the program is constructed, and if you wish you can adapt the parameters configured in the software to suit your own needs.

Another reason is that the electronic hourglass also serves a practical purpose, since it can be used as a kitchen timer that has at least three advantages over a traditional hourglass. First, the time required for the 'sand' to flow from one bulb to the other can be adjusted; second, there is an acoustic signal after the time is expired; and third, the whole thing looks great and fits better in the household of a modern electronic hobbyist. Of course, the fact that it's somewhat unique (since you can't just buy it ready-made) is also a significant consideration.

The hardware

The most striking part of the schematic diagram (**Figure 1**) is the 8 × 8 matrix of LEDs, which represents one of the key elements of this design. But what happened to the series resistors for the LEDs? Here the LEDs are driven using the multiplexing technique, so each LED is on for only a fraction of a second. This means that we can do without series resistors. Each of the rows and columns of the 8 × 8 matrix is driven via a transistor stage. Two 3-to-8 decoders (IC2 and IC3) ensure that only one row or column at a time is enabled. The LEDs that should be illuminated at any given time are determined by the microcontroller (IC1). This brings us to the second major part of the circuit, the PIC16F84 microcontroller. This device is readily available and has exactly the number of I/O pins needed for our application. The main advantage of this

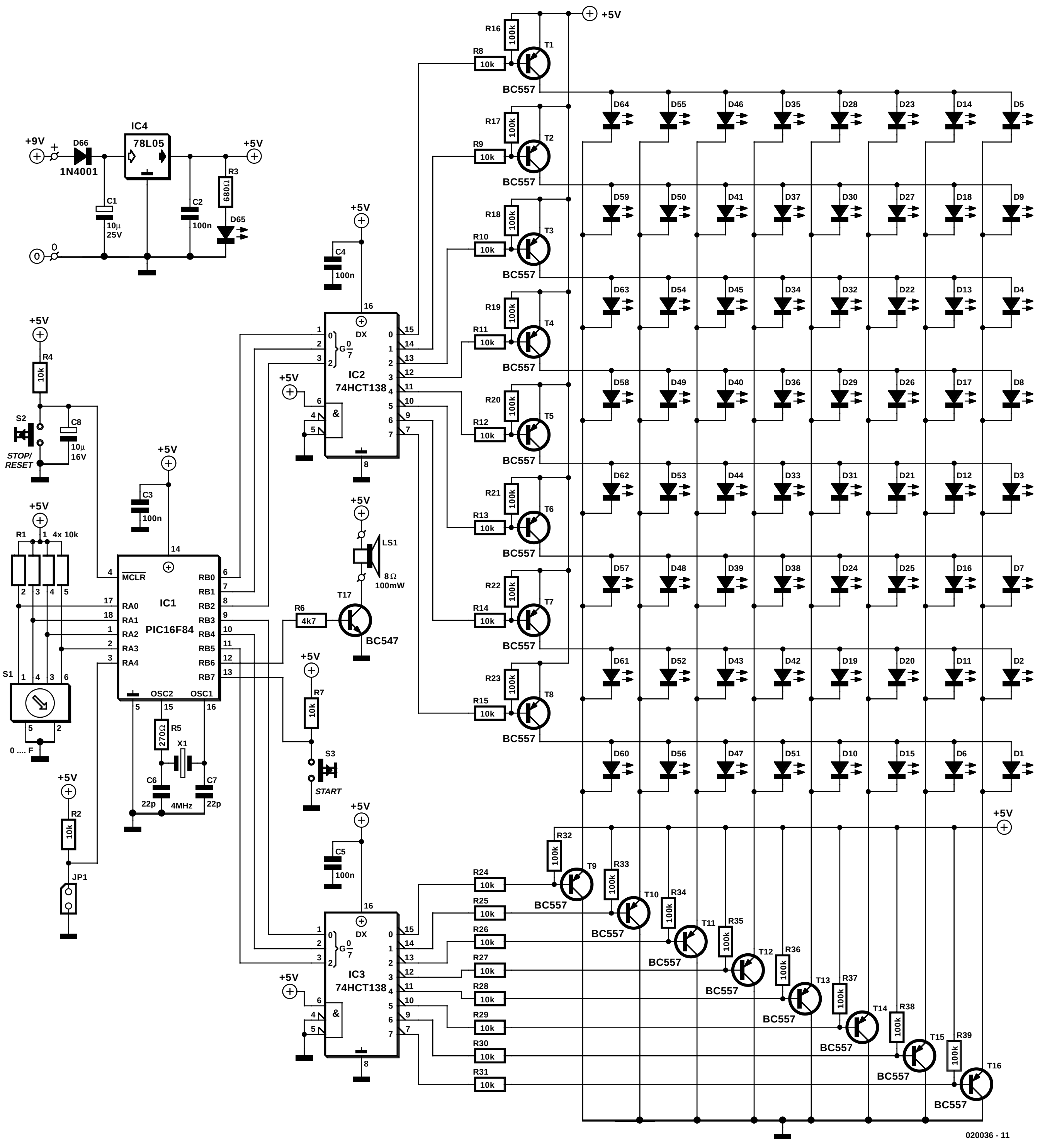


Figure 1. The circuit of the electronic hourglass essentially consists of PIC microcontroller driving a display formed by 64 LEDs

microcontroller is that it has a Flash memory, so it can be reprogrammed practically as often as desired. There is also an EEPROM in the PIC16F84, but it is not needed for this project.

The job of the microcontroller is to control the entire process, from starting the timer to emitting an acoustic signal at the end of the set interval. This is realised using the program described later on.

A standard circuit connected to pins 15 and 16 of the microcontroller is used to generate the clock signal. In principle, a 4-MHz ceramic resonator could be used in place of the quartz crystal (X1), but the times would then be less exact. If a quartz crystal is used, the configured values (timing constants) are sufficiently close to the to the calculated values.

A proven standard solution using a resistor and capacitor (R4 and C8) is used to generate the reset signal. The Stop switch (S2), which is wired in parallel with C8, also generates a reset signal to put the microcontroller back into its initial state.

The row decoder (IC2) is connected to I/O pins RB0–RB2, while the column decoder is connected to pins RB3–RB5. These six I/O pins are thus responsible for driving the LED matrix. The LEDs are 5-mm, red low-current types (2 mA typical). Of course, the circuit will also work with green or yellow LEDs, provided low-current types are used with sufficient brightness.

The loudspeaker (LS1) is driven by pin RB6 of the microcontroller via a transistor stage (R6/T7), using a highly audible square-wave signal.

The desired time is set using a hexadecimal-coded switch (S1), allowing 16 different times to be set. This switch requires four lines, each with its own 10-kΩ pull-up resistor array (R1). Microcontroller pins RQ0–RA3 are used for this purpose.

A tone can optionally be generated each time the state of the LEDs changes by fitting jumper JP1 (connected to pin RA4 of the microcontroller).

The LED hourglass is started using the Start pushbutton switch S3, which is connected to pin RB7 of the microcontroller via a pull-up resistor (R6).

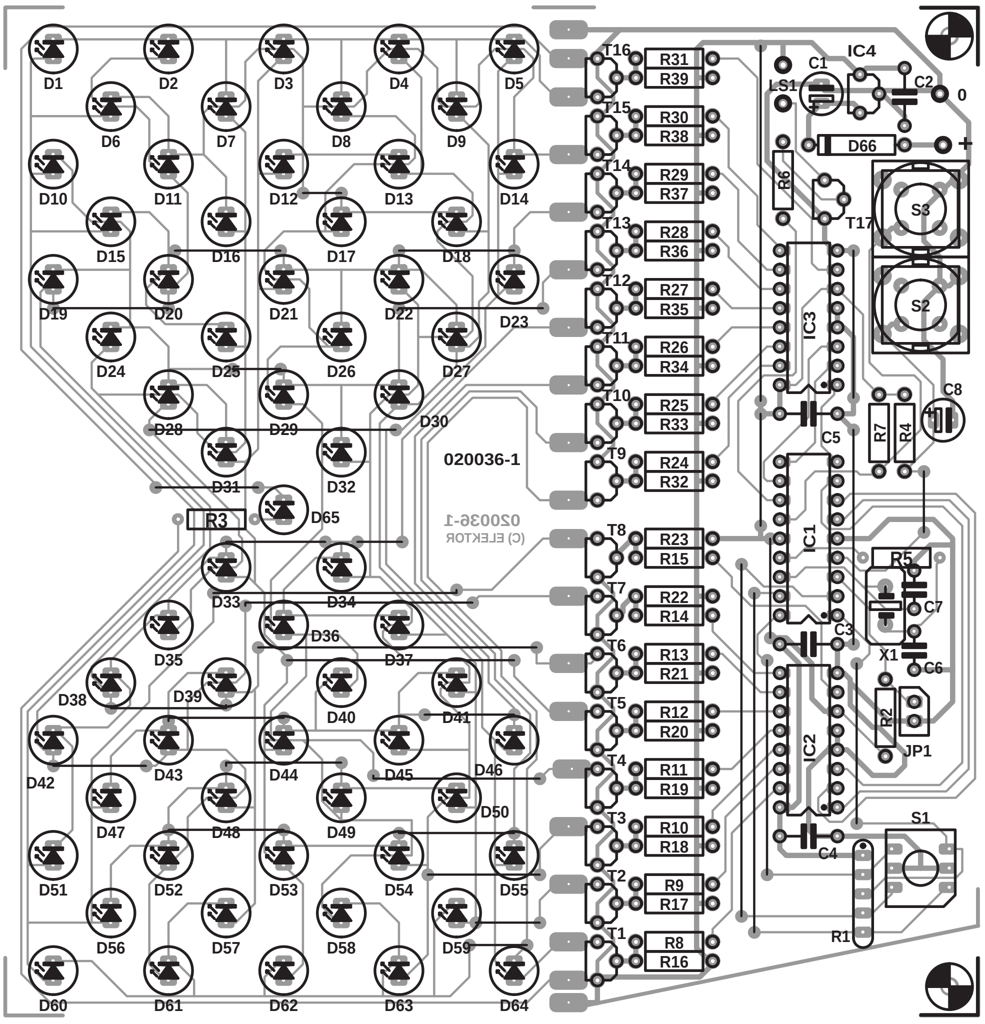


Figure 2. The layout of the printed circuit board betrays its intended use: the LEDs are

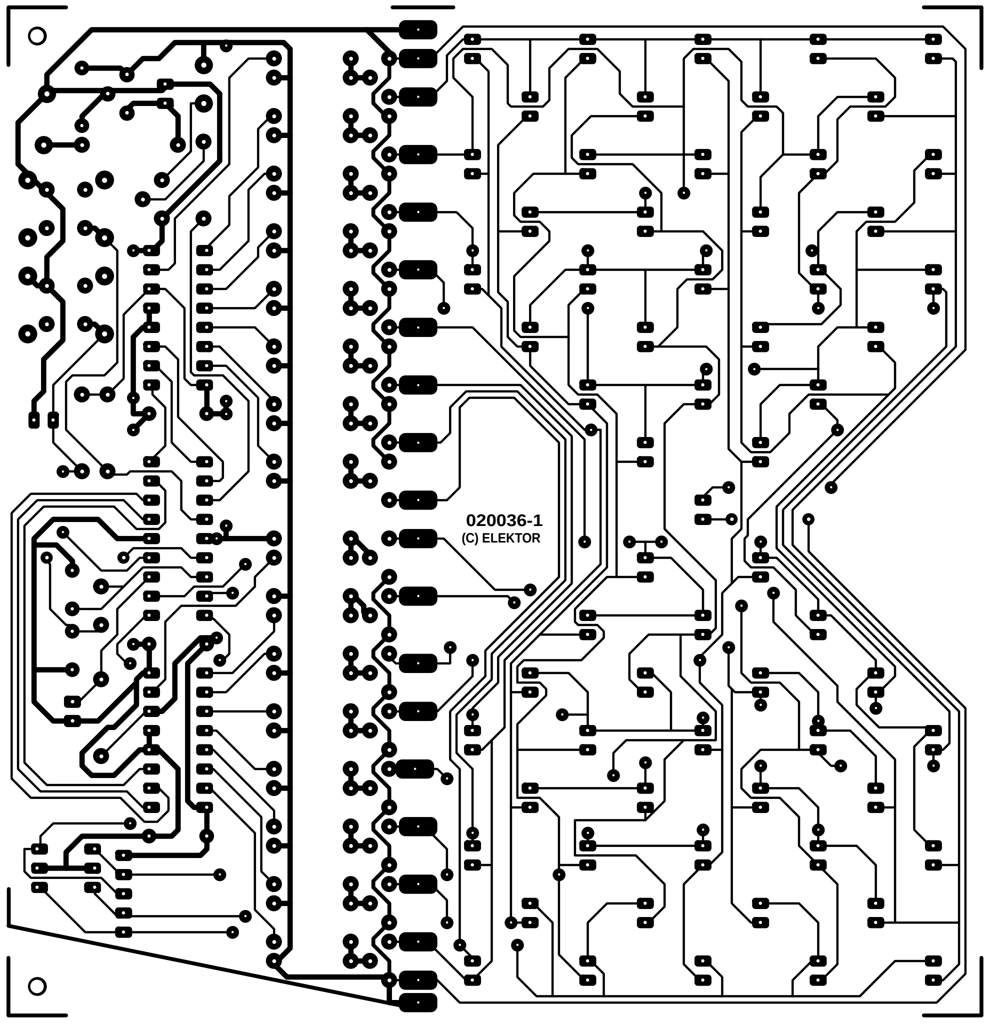
COMPONENTS LIST

Resistors:
R1 = 4-way 10kΩ SIL array
R2, R4, R7-R15, R24-R31 = 10kΩ
R3 = 680Ω
R5 = 270Ω
R6 = 4kΩ7
R16-R23, R32-R39 = 100kΩ

Capacitors:
C1,C8 = 10µF 25V radial
C2-C5 = 100nF
C6,C7 = 22pF

Semiconductors:
D1-D65 = LED, 5mm, red, low-current
D66 = 1N4001
T1-T16 = BC557
T17 = BC547
IC1 = PIC16F84-04/P, programmed, order code **020036-41**
IC2, IC3 = 74HCT138
IC4 = 78L05

Miscellaneous:
X1 = 4MHz quartz crystal
S1 = miniature hex encoder switch, 16 positions (Hartmann type PT65 303, from Conrad Electronics, # 705497)
S2,S3 = single contact switch, make contact, e.g., type D6
Ls1 = miniature loudspeaker, 8Ω 0.1W
JP1 = jumper
PCB, order code **020036-1**
Disk, project software, order code **020036-11** or Free Download



arranged in the form of an hourglass (board available ready-made).

LED D65 is connected directly to the supply voltage via a series resistor (R3), so it is always on when the supply voltage is present. It is placed in the middle of the 'LED field' to act as a pilot light indicating that the hourglass is switched on.

There's not much that needs to be said about the power supply. A 78L05 fixed voltage regulator, in combination with capacitors C1 and C2, provides a stabilised 5 V operating voltage. Diode D1 protects against reverse polarity, and an on/off switch on the 9-V input (not shown on the drawing) allows the hourglass to be switched on and off. The 9-V source can be a 9-V mains adapter, a 9-V battery pack (six 1.5-V cells) or a 9-V storage battery, as desired. The current consumption of the circuit is approximately 45 mA.

Construction and operation

All of the components shown in the schematic diagram can be seen in the component layout of the printed circuit board (**Figure 2**). The 65 LEDs are arranged to form a display in the shape of a traditional hourglass. The board can be split into an LED board and a control board (containing the rest of the circuitry) if you do not want to have the control portion immediately adjacent to the LED display. The separation line runs through the middle of the row of solder pads between the two parts of the circuit board. The two halves of the solder pads remaining on each board after they have been separated are used to attach the interconnecting wiring. For the prototype built in the *Elektor Electronics* lab, the two parts of the board were arranged at right angles to each other (see photos). This allows the connections to be directly soldered together (using short lengths of wire if desired).

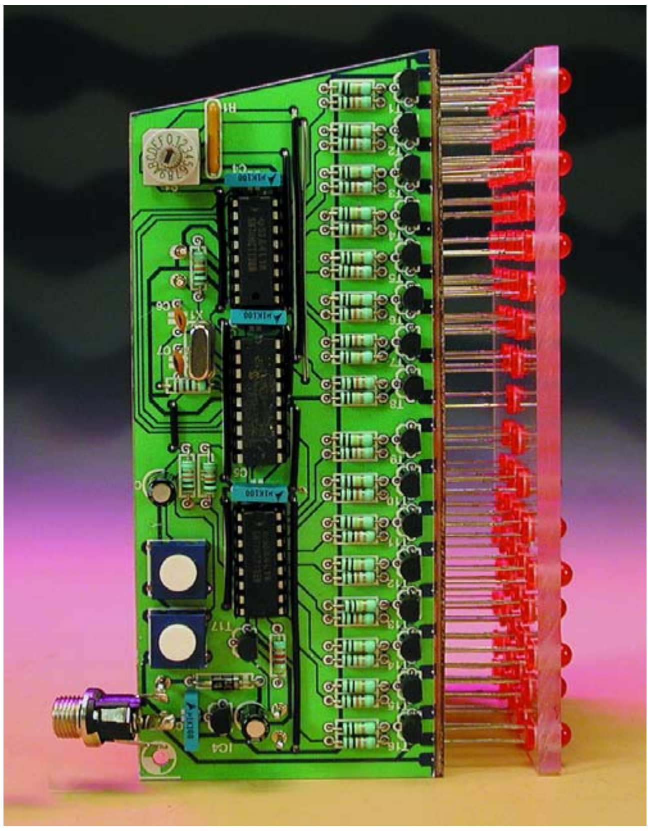
A few suggestions, both general and specific, are in order with regard to fitting the components to the circuit board. Start by fitting the wire bridges, to ensure that you do not forget any of them. Pay attention to the polarity of polarised components, such as electrolytic capacitors, diodes and LEDs. Use sockets for IC1–IC3, and be sure to insert the ICs the right way around (note the markings). Before connecting the operating voltage, thoroughly inspect the fitted components and solder joints of the assembled board. In the Components List, no current gain class is given for the transistors, so A, B or C types can be used.

There is no on/off switch on the circuit board. Such a switch can be fitted to the rear of the enclosure, with the controls (the Start and Stop pushbuttons and the 16-position code switch) at the front, fitted directly to the circuit board or on a front panel.

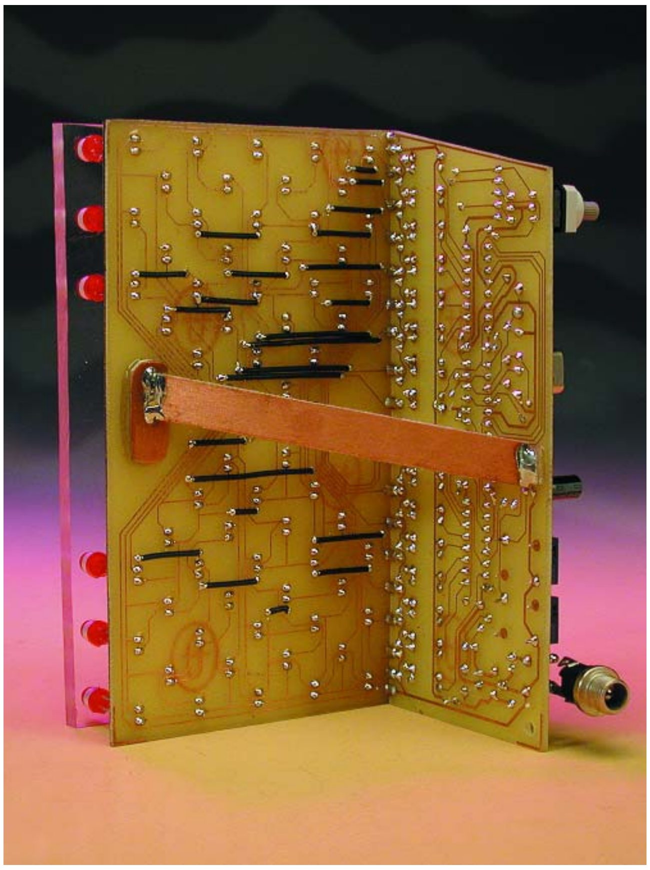
After you apply the operating voltage and switch on the circuit using the on/off switch (if present), the lower group of LEDs will light up. Set the desired time interval using the selector switch. You can choose from 16 intervals, ranging from 15 seconds to 1 hour (see **Table 1**). The intervals are defined in the software using two constants, so they can be modified if necessary.

As soon as the Start button is pressed, the LEDs in the upper half of the display will light up and the set time will start to trickle away. The LEDs in the upper part of the display will be extinguished one by one, while the LEDs in the lower part of the display will be correspondingly illuminated one by one. The timer interval can be terminated at any time by pressing the Stop button.

All of the LEDs in the lower half of the display will be illuminated when the set interval has expired. When the last LED lights up, the loudspeaker (fitted at the rear) starts to emit a tone. This signal can be stopped by pressing the Stop button. If you press the Start button instead of the Stop button, the loudspeaker



will also be muted, but at the same time the hourglass will be started again to time out the interval set by the code switch.



If the previously mentioned jumper is fitted to the board, a brief acoustic signal will be emitted each time the state of the LEDs changes.

The software

The tasks performed by the microcontroller can essentially be divided into the following subtasks:

– Driving the 8 × 8 LED matrix. This should take place in the background, so it is implemented using a timer interrupt.
– Loading the addresses of the LEDs to be illuminated into memory locations POS1–POS32.
– Polling the Start pushbutton and then the hex-encoded switch. Separate polling of the Stop switch is not necessary, since it is directly connected to the Reset input. A reset is triggered when the Stop button is pressed, causing the microcontroller to be re-initialised.

– Activating the loudspeaker on completion of the timing interval.

The special feature of the software for this project is that the addresses of the 32 LEDs that are illuminated at any given time are stored in a set of RAM locations labelled POS1 through POS32. In the initial state, with the 32 bottom LEDs illuminated, RAM locations POS1 through POS32 contain the addresses of LEDs D33 through D64. POS1 thus contains the binary value ‘10011100’ (the address of LED D33), POS2 contains the binary value ‘10010100’ (the address of LED D34) and so on.

Table 1.

Time settings selectable using the code switch S1.

Switch setting	Time	
0	15	seconds
1	30	seconds
0	60	seconds
3	90	seconds
4	2	minutes
5	3	minutes
6	4	minutes
7	5	minutes
8	8	minutes
9	10	minutes
A	12	minutes
B	15	minutes
C	20	minutes
D	30	minutes
E	45	minutes
F	60	minutes

The ISR is given the additional task of decrementing counter register AKTXBITx by 1 each time it is called. This register is a 16-bit counter register, so it must be divided into two 8-bit registers labelled AKTZEITL and AKTZEITH. When the value of this register reaches zero, a flag bit is set (NaechstZust-Flag) to inform the main routine that the state of the LEDs must be changed. Two subroutines, labelled ZUSTAND1 and ZUSTAND0, are provided for the start and exit states (the start state is with all 32 of the upper LEDs illuminated, while the exit state is with all 32 of the lower LEDs illuminated).

Although it would have been possible to write a subroutine for each of the intermediate states, the program memory of the PIC16F84 is probably not large enough to hold the amount of code this would generate. However, other solutions are possible — and if you are interested in the details, you can analyse the program by studying the listings.

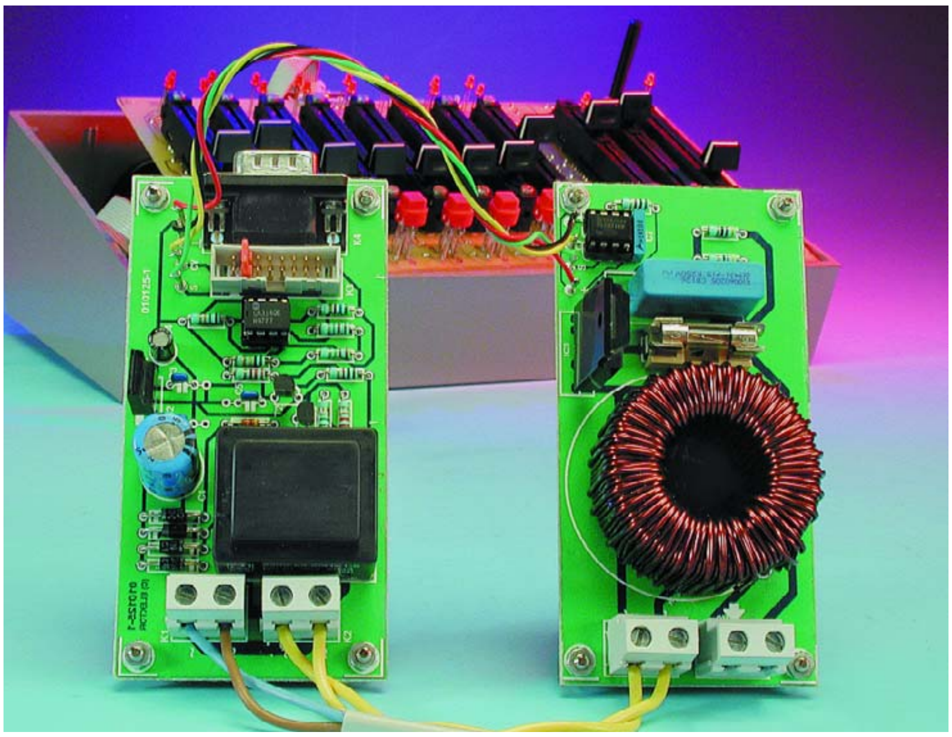
(020036-1)

Dimmer with 0-10 V Control Input

for our Light Mixer Panel

Design by L. Lemmens

At the end of the Light Mixer Panel article published in last year's February issue we promised to complete the project with a compatible dimmer. Sure, the DMX Dimmer published in May 2002 could be used for the purpose, but the design is rather too extensive and complex to act as a simple dimmer. The circuit presented in this article is much simpler and cheaper!



The 8-channel Light Mixer Panel published in February 2002 was well received by many readers interested in sound & light equipment, mainly because it was affordable and fairly easy to integrate in existing lighting systems. However, besides praise for the design we also received many requests for an associated dimmer circuit capable of accepting the standardised 0-10 V control signals supplied by the Mixer panel outputs. Although suitable dimmers with a 0-10 V control input are available commercially, the true DIY enthusiast will, of course, insist on building one himself, in line with his efforts a building the Light Mixer Panel.

Although the DMX Dimmer published in May 2002 partly fulfilled the wishes of lights technicians, its design was obviously geared to semi-professional realms and control from a digital DMX system. Few of you who require a couple of simple dimmers will be prepared to invest in a hefty processor controlled circuit, even if it is designed and built to perfection.

To cut a long story short, there was a continuing need for a simple, straightforward dimmer suitable for lamp powers up to about 1,000 watts and with an input that's compatible with the industry standard 0-10 V control voltage. Well , that's exactly what we present in this article.

Two separate circuits

The basic design of the dimmer circuit follows the simplest and most rudimentary approach we could think of. The actual dimming is done using an opto-triac and phase angle control. The triac is driven by a comparator that compares the 0-10 V control voltage supplied by the Light Mixer Panel (or another compatible source) with a triangular voltage of about 10 V. In fact, we are looking at a pulsewidth control system here — the higher the control voltage, the longer the 'High' time of the pulse at the comparator output. Consequently, the triac will conduct longer and the lamp will light brighter. So, the only thing we need besides the comparator and the opto-triac is a sawtooth voltage. Generating such a waveform is not problematic as we'll see a bit further on. Because the sawtooth voltage may be used for several parallel-connected comparators, the circuit has been designed so that it can be split in two parts: controller and dimmer. The first circuit contains the sawtooth generator, a simple stabilised power supply for the opamps and a connector for the control voltages received from the Light Mixer Panel. The dimmer circuitry comprises little more than the comparator, the opto-triac and a suppressor coil.

Each functional section is mounted on a separate board, allowing several dimmer circuits to be connected to a common control circuit. This is particularly useful if several spotlights are clustered in a rack or on a tripod. Each spotlight then has its own dimmer module (and may be controlled separately), and each cluster has its own control board.

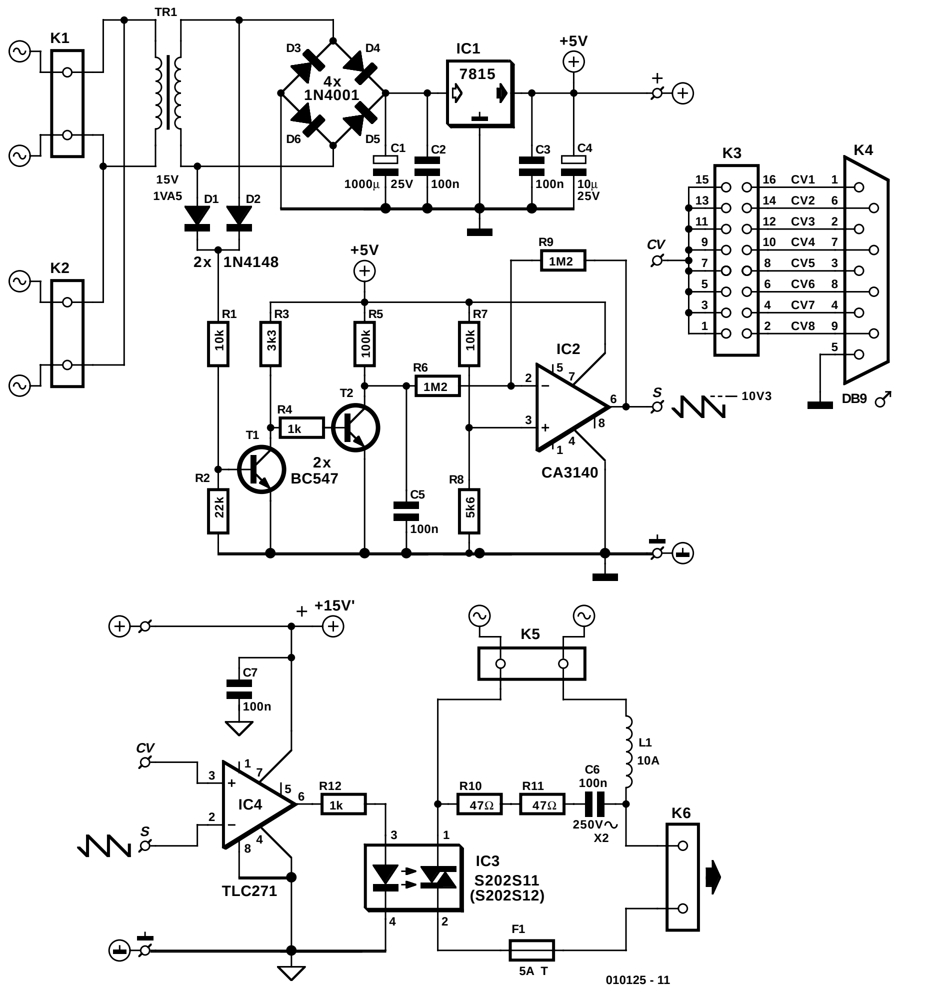


Figure 1. The circuit diagram may be divided into a control and a dimmer section. The control voltages supplied by the Light Mixer panel arrive on connector K4.

Control Module

The complete circuit of the dimmer is given in **Figure 1**. The upper part of the drawing describes the control module.

Let's first have a look at the connectors. The mains voltage is connected to one of the parallel-connected terminal blocks K1 or K2. The other terminal block may be used to feed the mains voltage through to K5 on the dimmer module(s). Note, however, that the maximum current of 15 A for these terminal blocks is not exceeded. In case of doubt it is better to provide each dimmer module with its own mains inlet.

The 9-way sub-D connector labelled K4 is intended for connecting the eight control voltages arriving from K2 on the Mixer Panel. The connection is made using a piece of 9-way flatcable terminated with a female connector (socket) at one end and a male connector (plug) at the other. The socket side goes to K4. Double-row pinheader K3 has been added to set the desired channel number with the aid of a jumper. Starting from 'CV' (control voltage) the control voltage is looped through to the identically labelled terminals on the dimmer module(s). Incidentally, on K3 at the side of K4 it is also possible to take control voltages from other channels.

The next sub-circuit to discuss is the sawtooth generator. The circuit around D1, D2 and T1 generates pulses around the zero-crossing of the mains voltage. Via T2, these pulses quickly discharge capacitor C5. During the Off time of T2, C5 is loaded again via R5. This process causes a sawtooth-shaped voltage on C5. This waveform is inverted by IC2 and amplified to a ramp voltage with a maximum swing of about 10 V. From point 'S', the auxiliary voltage is 'copied' to the corresponding terminal on the dimmer module(s).

The remaining sub-circuit in the control module is a simple stabilised 15-volt power supply unit for the opamps. The PSU consists of transformer Tr1, bridge rectifier D3-D6, reservoir capacitor C1 and voltage regulator IC1.

Dimmer module

First, the connectors again: the mains voltage is connected to K5 and the spotlight to be dimmed, to K6.

IC4 compares the sawtooth voltage (terminal S) with the control voltage from the Light Mixer Panel point (CV). The higher the control voltage, the longer the output of IC4 will remain High — the LED opto-triac IC3 is then driven longer, causing the lamp connected to K6 to light brighter.

Starting from the control board, the ±15-V supply voltage is connected through via the pins marked with the ‘ground’ and ‘plus’ labels.

When several dimmer modules are connected to a single control, module, the necessary control voltages may be taken directly from K3 (pin 2 = channel 1; pin 4 = channel 4; etc.)

Soldering

The artwork for the printed circuit boards of

COMPONENTS LIST

Resistors:
R1,R7 = 10kΩ
R2 = 22kΩ
R3 = 3kΩ3
R4,R12 = 1kΩ
R5 = 100kΩ
R6,R9 = 1MΩ2
R8 = 5kΩ6
R10,R11 = 47Ω

Capacitors:
C1 = 1000μF 25V radial
C2,C3,C5,C7 = 100nF
C4 = 10μF 25V radial
C6 = 100nF 250VAC Class X2

Inductors:
L1 = 10A triac suppressor coil, 50-100μH

Semiconductors:
D1,D2 = 1N4148
D3-D6 = 1N4001
T1,T2 = BC547
IC1 = 7815
IC2 = CA3140E
IC3 = S201S01 or S202S11
IC4 = TLC271CP

Miscellaneous:
K1,K2,K5,K6 = 2-way PCB terminal block, lead pitch 7.5mm
K3 = 16-way pinheader
K4 = 9-way sub-D connector, male, PCB mount, angled pins
TR1 = 15V 1VA5 e.g., Hahn type BV EI 303 2033
F1 = 1.25 A, slow, with PCB mount holder
PCBs, order code **010125-1** (control module), **010125-2** (dimmer module), see Readers Services page.

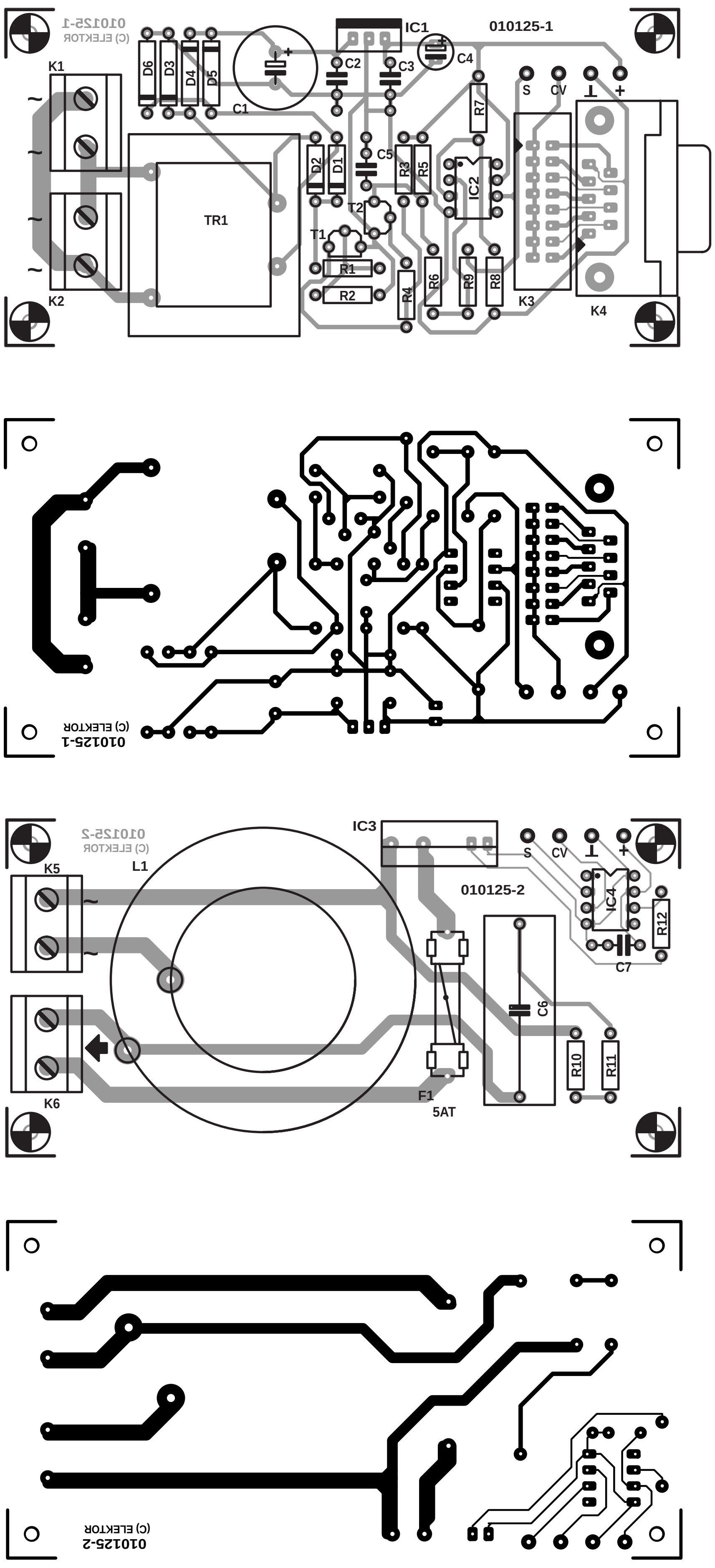


Figure 2. The control and dimmer circuits have their own printed circuit board.
That allows multiple dimmer modules to be hooked up to a common control board (boards available ready-made).

the two modules is shown in **Figure 2**. Although the dimmer board holds fewer components than the control board, the two are of equal dimensions mainly because of the bulky size of the 10-A suppressor coil (L1).

Stuffing these boards is not something to baulk at. The number of components is within reason and there are no pitfalls or problematic details. We would, however, advise to spend just a little more time and attention than usual to the solder work, so as to avoid any risk of short-circuits on the board caused by solder blobs, splashes and other unwanted links. After all, both boards are connected to the mains so you simply cannot afford to be less than careful here.

Now that we are on the subject, please read the *Electrical Safety* page which is occasionally printed in this magazine (if you do not have a copy, we will gladly send you one). **Never work on the circuit when it is connected to the mains. The mains voltage is potentially lethal.**

Enclosure

For safety's sake the circuit may only be built into an ABS (solid plastic) case. Make sure you install an approved mains appliance inlet as well as safe and sound connections for the spotlights. The connections between the various mains inlets and K1, K2, K5 and K6 should, of course, be made in properly insulated wire with a cross-sectional area of at least 1.5 mm^2. The links between points S, CV 'ground' and '+' may be made using light-duty insulated wire.

If a combination of one control board and one dimmer board is applied locally, then a channel is set on K3 using a jumper or wire link. Next, the points CV on the two boards may be linked. In a configuration with several dimmer boards and one control board, each CV terminal on the dimmer boards is separately connected to the desired channel pin on connector K3. That is easily done with the aid of, say, a header and a length of flatcable.

According to electrical safety regulations, the case should have a label attached on it stating salient information like operating voltage and fuse rating. A suggested format is included in the circuit diagram in Figure 1.

Up to 1,000 watts

Although the opto-triac used here is rated for a continuous current of 8 A we deemed it safer to remain well under this specification. Fuse F1 is therefore dimensioned for a maximum lamp power of 1,000 watts (assuming a mains voltage of 230 volts). In most cases, that will be sufficient while minimizing the risk of serious damage when something goes wrong at such a relatively low power level.

For lamp powers up to 500 watts no cooling device is required for IC3. Above 500 watts, we recommend fitting a small heatsink and drilling a few ventilation holes in the case so the heat does not build up.

Short current surges exceeding the maximum value of 8 A, for instance, when cold lamps come on, need not worry us because the opto-triacs type S202S11 and S202S12 mentioned in the parts list can withstand peak currents up to 80 A (!)

(010125-1)

Optimise your PC Soundcard

using a correction filter

Design by R. Badenhausen

Digitising analogue sound media like vinyl records and magnetic tapes is a simple matter if you have a reasonably fast multimedia PC. After all, you have available all the necessary ingredients and tools including a soundcard with an analogue inputs, software and a CD-ROM burner. The circuit described in this article enables the analogue input of your soundcard to be linearised for optimum sonic reproduction.

Please note: Readers' Circuits have not been tested or post-engineered by the Elektor Electronics design laboratory.

The author and designer of this circuit performed a series of tests on several 'standard' soundcards used in PCs. Although most of these were found to have good anti-aliasing filters, in quite a few cases the frequency linearity of the analogue inputs left much to be desired. In most cases, this deficiency is caused by a rather too small input capacitor, a digital input filter dimensioned for a too low frequency, or simply by the user himself unwittingly connecting a cheap input cable of about 3 metres length to the input, which then sees a far too large capacitance.

Correction circuit

The circuit shown in **Figure 1** allows any negative deviations at both ends of the frequency spectrum to be 'pulled straight'. In fact, it causes the frequency response to be lifted by a controlled amount at the two far ends. An example of this is shown n **Figure 2**.

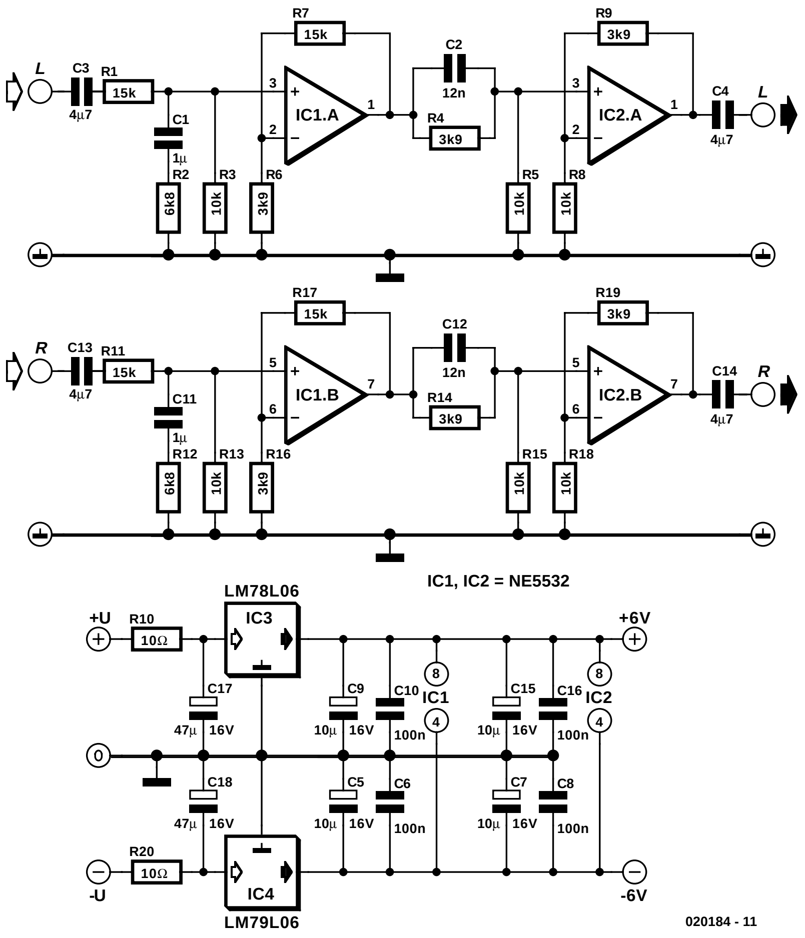


Figure 1. The correction circuit consists of two filter sections with buffer/amplifier stages.

The circuit consists of two passive filters (each having two roll-off points) — one for amplifying the low frequencies and one for the high frequencies. The first filter consists of components R1/C1/R2/R3, the second, of R4/C2/R5. Behind each filter sits a buffer/amplifier (IC1a and IC2a respectively) which serves to compensate the attenuation introduced by the filter. The Right (R) channel is of identical design with the component reference numbers increased by 10.

The values of the resistors in the buffer stages follow these easy equations: R6 =R2/R3; R7 = R1; R8 = R5; R9 = R4. The same goes for the Right channel: R16 = R12/R13, etc.

Two voltage regulators, a 78L06 and a 79L06 together with a number of reservoir and decoupling capacitors, ensure a stable supply voltage for the circuit.

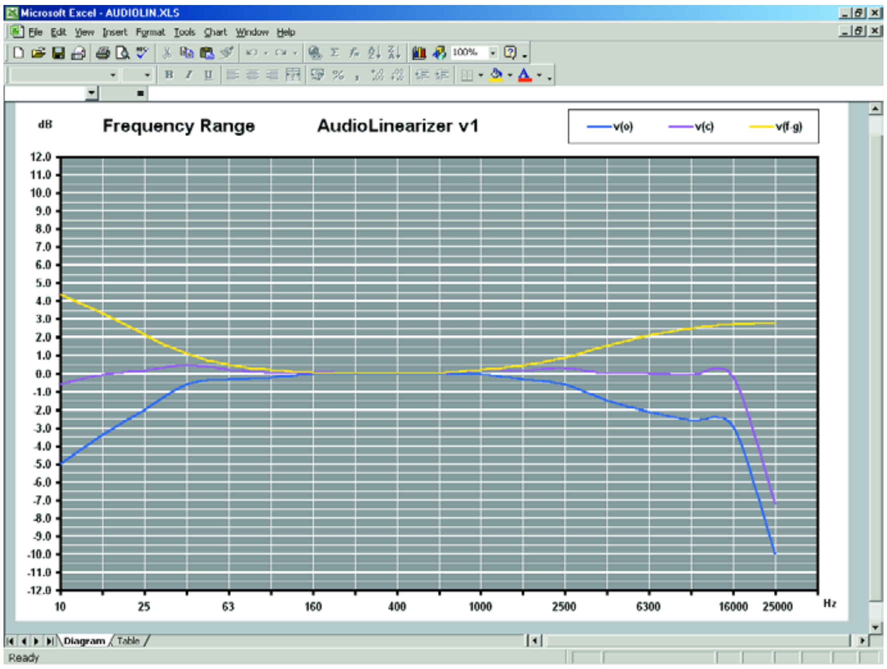


Figure 2. This graph shows the measured frequency response, the correction curve and the resulting curve.

Measurement procedure

Before we can have a look at the calculation of the filter component values , we need to measure the error (non-linear frequency response) introduced by the analogue input.

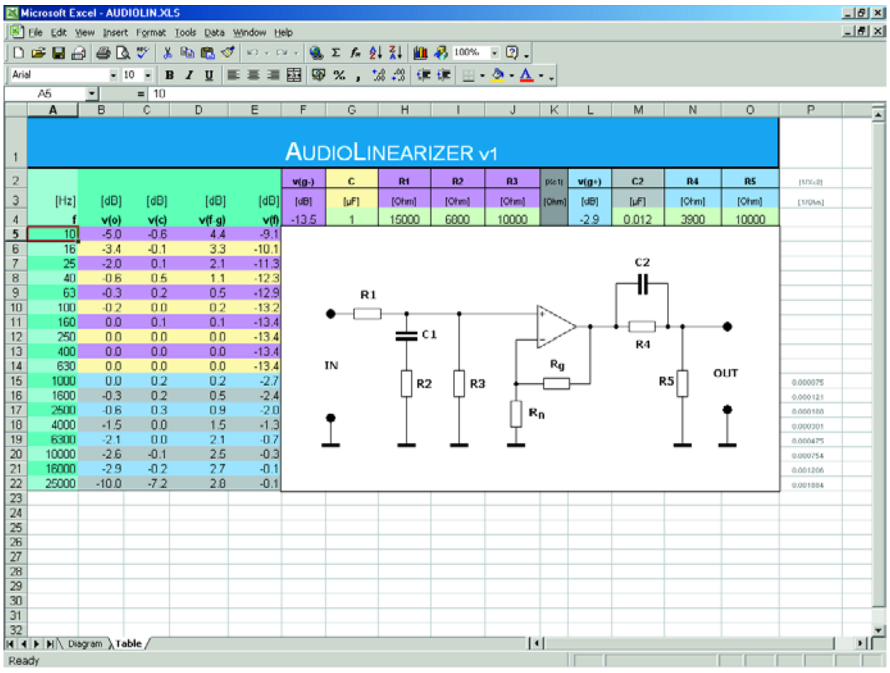
It is unusual for a manufacturer of a soundcard to supply exact measurement data covering the inputs and outputs of the product. So, we have to do that ourselves. The job requires an accurate sinewave generator with a frequency range of 10 Hz to 25 kHz and a sound editor program like Cool Edit. If necessary, an oscilloscope may be connected to measure the output level at different frequencies.

Connect the generator to the inut of the soundcard under test, then adjust it for the highest output level at which the input is not overdriven (for example, 200 mV). Next, use the software to measure the output level at different frequencies. Using Cool Edit, that is done by selecting 'Record' and then checking the signal level indicated by the VU meters. The range of the meters may be adapted by right-clicking on them.

Dimensioning

The author employs an Excel spreadsheet to calculate the values of the components in the correction filters. The spreadsheet allows the measured values to be entered in column B (v(o), see **Figure 3**). The level at 1000 Hz is taken as the reference level (0 dB). Column A shows the frequencies at which the levels need to be measured. Column D shows the response of the correction network of Figure 1, and column C, the resulting frequency response. Finally, in column E we see the attenuation v(f) per filter section — the respective values at 630 Hz and 1000 Hz indicate the attenuation which is to be compensated by the associated buffer/amplifier.

At the right-hand side of the Excel spread-


Microsoft Excel - AUDIOLIN.XLS
File Edit View Insert Format Tools Data Window Help
Arial
10
A5
= 10
R1
C1
C2
R4
IN
Rg
R2
R3
R5
OUT
Rn
Diagram
Table
Ready

AUDIOLINEARIZER v1

[Hz]	[dB]	[dB]	[dB]	[dB]
f	v(o)	v(c)	v(f-g)	v(f)
10	-5.0	-0.6	4.4	-9.1
16	-3.4	-0.1	3.3	-10.1
25	-2.0	0.1	2.1	-11.3
40	-0.6	0.5	1.1	-12.3
63	-0.3	0.2	0.5	-12.9
100	-0.2	0.0	0.2	-13.2
160	0.0	0.1	0.1	-13.4
250	0.0	0.0	0.0	-13.4
400	0.0	0.0	0.0	-13.4
630	0.0	0.0	0.0	-13.4
1000	0.0	0.2	0.2	-2.7
1600	-0.3	0.2	0.5	-2.4
2500	-0.6	0.3	0.9	-2.0
4000	-1.5	0.0	1.5	-1.3
6300	-2.1	0.0	2.1	-0.7
10000	-2.6	-0.1	2.5	-0.3
16000	-2.9	-0.2	2.7	-0.1
25000	-10.0	-7.2	2.8	-0.1

v(g-)	C	R1	R2	R3	p(c1)	v(g+)	C2	R4	R5
[dB]	[µF]	[Ohm]	[Ohm]	[Ohm]	[Ohm]	[dB]	[µF]	[Ohm]	[Ohm]
-13.5	1	15000	6800	10000		-2.9	0.012	3900	10000

[1/(X·2)]
[1/Ohm]
0.000075
0.000121
0.000188
0.000301
0.000475
0.000754
0.001206
0.001884

Figure 3. This Excel spreadsheet allows measured valued to be entered and filter component values to be adapted to requirements. The associated curves (Figure 2) show the results instantly.

sheet the frequency determining components are shown, with an opamp thrown in for the necessary gain. The values of the filter components may be entered above this section, whereupon the spreadsheet will instantly compute the associated correction curve.

If you click on the 'Diagram' tab at the bottom, the three frequency response curves will appear: (1) the characteristic of the filter with the required values; (2) the measured response of the input and (3) the result of combining these two. Now, you are in a position to experiment with the filter components until the resultant frequency response is as flat as possible.

When the spreadsheet is started, you'll find that a number of component values and an example curve are already present. These may be edited as required.

Some rules of thumb are in order at this point to help you get the dimensioning right. A minimum value of 15 kΩ should be observed for R1, to prevent the output of the sound source affecting the filter. The ratio of R1 to R2//R3 determines the amplification of the low frequencies. A higher value of R1 causes a bass boost. The value of C1 determines the frequency at which the correction starts to act. Lowering the value of C1 causes the roll-off frequency to be increased. In the High section the ration of R4 to R5 determines the amplification of the higher frequencies, while C2 provides the onset of the high roll-off point. Here, a larger value results in a lower roll-off point. Of course, all components in the filter sections will interact to some extent. Fortunately, you'll soon see the effects and results when actually entering values into the spreadsheet.

If only one correction is required, (i.e., for high or low only) the other section and the associated buffer/amplifier may be omitted.

The Excel spreadsheet is available from this month's Free Downloads page on our website, look for file number **020184-11**. Readers without access to the Internet may order it on floppy disk using the same number.

(020184-1)

Autoranging Capacitance Meter

wide-range, PIC-controlled

Design by Flemming Jensen

Most commercial digital Capacitance meters can measure capacitors from a few pFs (picofarads) up to 2,000 µF (microfarads). A few are able to measure up to 20 mF. Circuits like audio power amplifiers, switched-mode power supplies, printers and photocopiers often incorporate really large capacitors in the hundreds of mF range (1 mF = 1,000 µF). With this cheap circuit, you'll be able to measure any capacitors of almost any 'size', from picos to Farads.

Main Features

– Low cost, direct reading,
– Wide range, (from a few pFs up to several Farads)
– Auto-ranging, (no range switches)
– Auto-zeroing, (nulls stray capacitance of connected test leads or test jig on power up)
– Low-Battery indication
– Simple setup

Although the specifications of the Autoranging Capacitance Meter are pretty impressive, the circuit diagram is uncluttered and free from surprises — see **Figure 1**. Using connector K1, the capacitor to be tested, C_x, is connected into a circuit based on the CMOS version of the good old 555 timer IC. Here, the TLC555 is configured as a monostable multivibrator (MMV). The real workhorse in the circuit is a PIC16F84 microcontroller running at 20 MHz. Using two output port lines, the PIC controls the 555's $\overline{R}$

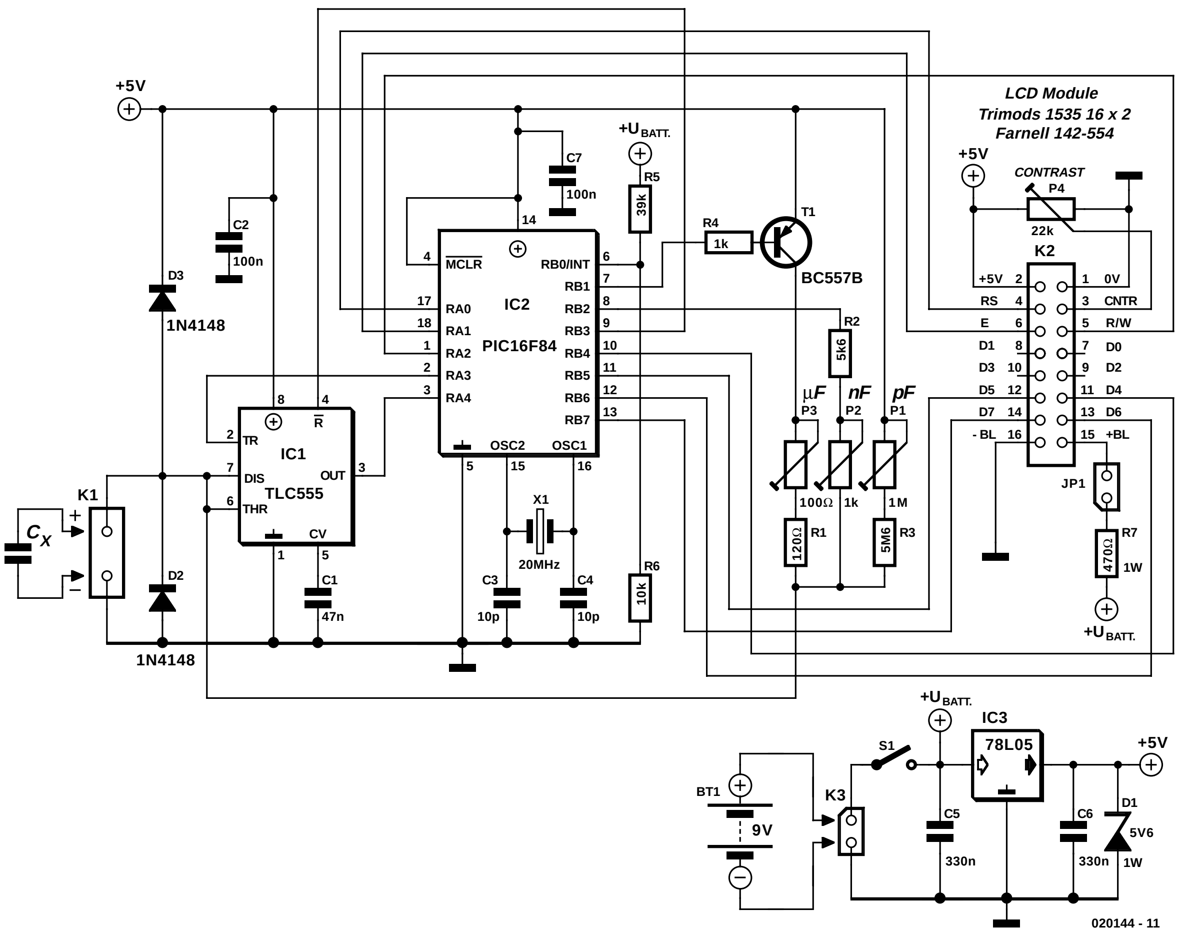


Figure 1. Circuit diagram of the Autoranging Capacitance Meter.

(reset) and TR (trigger) inputs and at the same time senses the logic level at the 555's OUT (output) pin. The larger the capacitance of the capacitor under test, the longer the High time of the pulse at the 555's output. As long as this output is high, the PIC increments a software counter (i.e., a variable). When the 555's output drops Low again the result is processed and sent to the readout. The PIC handles the necessary range switching and controls the LCD module as well.

The meter has three capacitance ranges:

pF	used for capacitors from 1 pF to 9,999 pF
nF	which covers 10 nF to 9999 nF
μF/mF	which covers 10 μF and up.

For your convenience, the read-out is modified so the meter will show 1 pF to 999 pF, and 1.00 nF to 9.99 nF. The same goes for the μF range.

The MMV oscillator frequency and hence the calibration for each range is determined by capacitor C_x and the resistance between the parallel-connected DIS (discharge) and THR (threshold) pins of the TLC555 chip. The resistance is range-dependent and is derived from resistor/preset combinations which can be switched into the circuit by the PIC using its RB1 and RB2 port lines.

The LCD, a 2-line, 16-character type is controlled in 4-bit mode. Its backlight may be switched on optionally using jumper JP1.

The power supply is totally standard and based on the 78L05 three-pin fixed voltage regulator. An extra zener diode, D1, is included to prevent any risk of damage to the circuit when the input is overloaded with a direct voltage in excess of the supply voltage (5 V).

The circuit is powered by a 9-volt PP3 battery. Current consumption is in the region of 7 mA with the LCD backlight left off.

Auto-zeroing and auto-ranging

At power-up, the PIC runs a routine that checks the stray capacitance at the input caused by test leads. and puts the result in a variable. This result is later subtracted from the result obtained from the capacitor measurement. Note, however, that this is only true in the pF range. It is therefore important that no capacitor is connected when the instrument is switched on, unless, of course, you intend to cancel a certain amount of stray capacitance. In all other ranges than the pF range, the capacitor may be connected at power-up.

After the zeroing routine, the meter enters the pF range. At this point, any measured capacitance will be recorded and placed in a variable. The auto-ranging functions like this: if the capacitor is too large for the pF range, the count will overflow, and the PIC will select the nF range, i.e., select a lower value charging resistor, and then continue by performing a new measurement. If the capacitance value is still too large, the μF range is selected, and so on. The result is displayed on the two line alphanumeric LCD module.

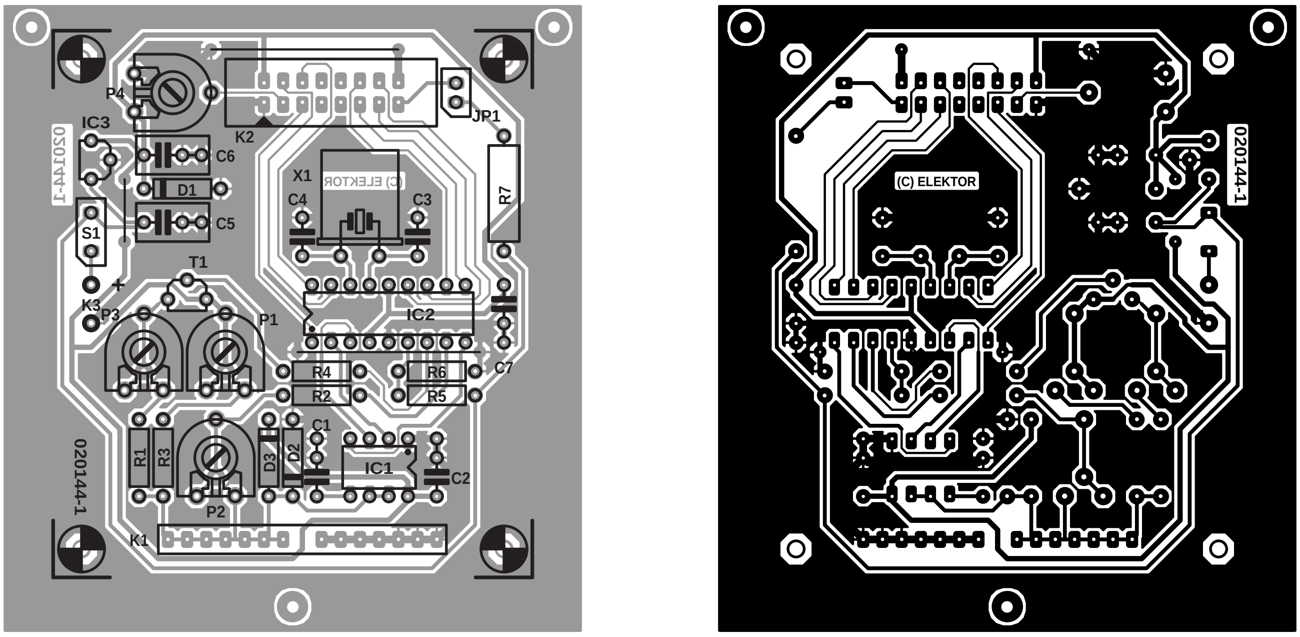


Figure 2. PCB design for the project (board available ready-made).

Hum and noise cancellation

In the pF range, the input represents a very high impedance. In this range the capacitor is charged via a 5 to 6 Megohm resistor and therefore the meter is sensitive to hum and noise picked up via the capacitor leads and the test leads, if used. When measuring capacitors near the low end of the pF range it is essential for the meter to be kept well removed from transformers etc., otherwise a unsteady readout may be the result. To reduce hum and noise pickup even further, the measurement in the pF range is performed twice, with a 10-ms interval. The mean value of the two results is calculated and the result is sent to the readout. This method is sure to give a steadier reading. In the nF and μF ranges, the resistor values in the MMV are relatively low and no special precautions are called for, allowing every single measurement to be read out.

Mode change when measuring large capacitors

Capacitors below 10 mF are measured continuously, i.e., the meter faithfully starts in the pF range. If the count overflows, the nF routine is run. If the count still overflows then the μF routine is run, and so on. Once the proper range is reached, the result is sent to the readout. Next, the whole thing starts all over again: pF routine $\rightarrow$ overflow $\rightarrow$ nF routine, $\rightarrow$ and so on.

Larger capacitors, in excess of 10 mF (10,000 μF = 10 × 10^{-3} F), are not measured continuously but four times. A message ‘wait’ is displayed on the LCD until all four measurements have completed. The result of the fourth measurement is then read out, a message ‘ready’ is displayed and the result is valid. This method ensures that the capacitor has been fully discharged and charged, gives a reliable reading and guarantees a low current consumption.

To prepare for a new measurement the on/off switch has to be operated.

COMPONENTS LIST

Resistors:

R1 = 120Ω
R2 = 5kΩ6
R3 = 5MΩ6
R4 = 1kΩ
R5 = 39kΩ
R6 = 10kΩ
R7 = 470Ω 1W (see text)
P1 = 1MΩ 10-turn multiturn preset, vertical mounting
P2 = 1kΩ 10-turn multiturn preset, vertical mounting
P3 = 100Ω 10-turn multiturn preset, vertical mounting
P4 = 22kΩ

Capacitors:

C1 = 47nF
C2,C7 = 100nF
C3,C4 = 4pF7
C5,C6 = 330nF

Semiconductors:

D1 = zener diode 5V6 1W
D2,D3 = 1N4148
IC1 = TLC555 (Texas Instruments), 3V555 or TS555 (Thomson)
IC2 = PIC16F84A-20/P, programmed, order code **020144-41**
IC3 = 78L05
T1 = BC559B

Miscellaneous:

BT1 = 9V PP3 (6F22) battery with clip-on connector
JP1 = 2-way pinheader with jumper
K1 = two 7-way socket strips, lead pitch 0.1 inch
K2 = 16-way boxheader
K3 = PCB solder terminals
S1 = on/off switch
X1 = 20MHz quartz crystal
LCD display, 2x16 characters, e.g., Trimods 1536, Farnell # 142-554
Length of 16-way flat cable with IDC socket
Enclosure: e.g., Pactec HPS and HPLS series, type HPS-9VB (28x53x91x146 mm), Farnell # 736-351, see also www.pactecenclosures.com
PCB, order code **020144-1** (see Readers Services page)
Disk, source code and hex files, order code **020144-11** or Free Download

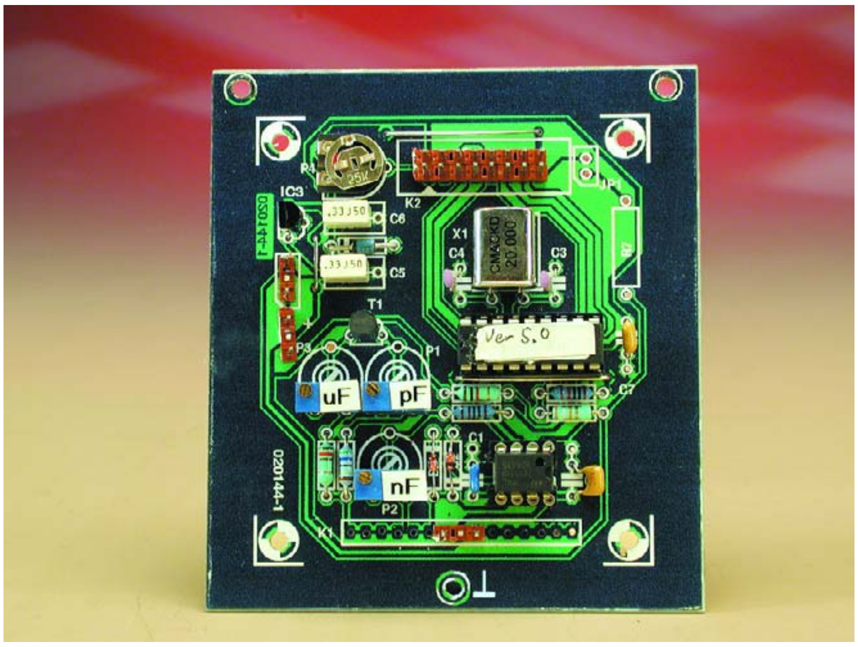


Figure 3. Close-up view of our prototype board.

Construction and Troubleshooting

Populating the printed circuit board shown in **Figure 2** should not present problems if you follow the component overlay printed on the ready-made board (or else gleaned from Figure 2) and the information in the parts list. Connector K1 consists of two socket strips that allow capacitors with different pin spacings to be easily connected without introducing too much stray capacitance. Wander (banana) sockets in the traditional colours red and black are used for the test leads.

Depending on the exact LCD type you intend to use you may need to establish a suitable value for resistor R7 in accordance with the current drawn by the backlight lamp inside the LCD. Here, a value of 470 Ω, 1 watt, is given for initial guidance.

The three wire links on the board should be installed first so they are not forgotten later.

Carefully inspect the board for solder blobs, short circuits, dry joints and the orientation of all polarised components. Our working prototype board is shown in **Figure 3**.

Before fitting IC1 and IC2 on the board, check the presence of the +5-volt supply volt-

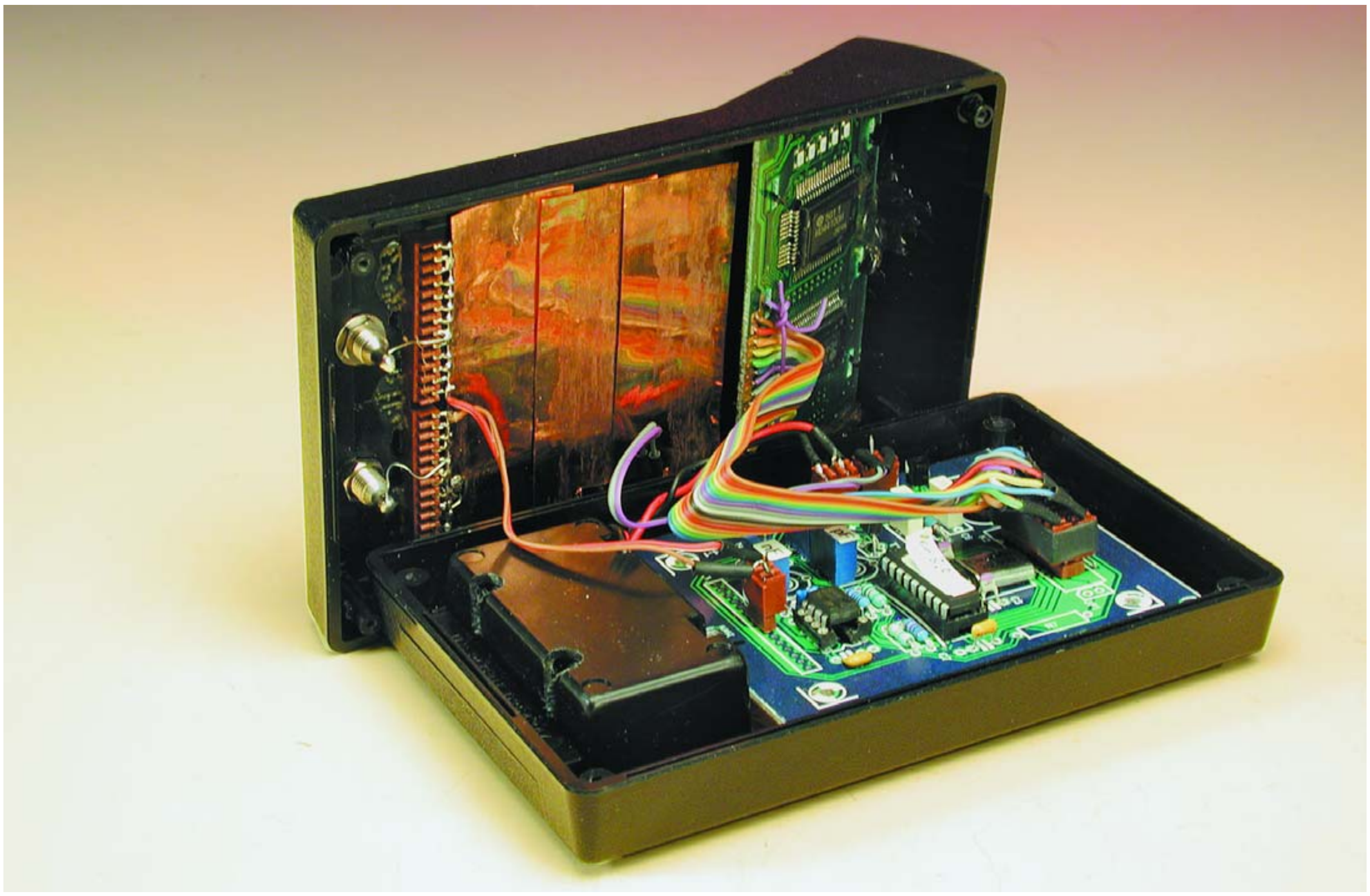


Figure 4. A look inside the case. The copper foil shielding is optional, and only necessary if interference from external sources is a problem.

Things to Remember

Discharge capacitors before testing.

In the pF range: first power up the meter, then connect the capacitor.

Capacitors larger than 10 mF: 4 measurements are performed. When meter signals 'ready' the result is valid. Turn the meter off and back on for a new measurement.

Large capacitors: don't get impatient. A 370-mF cap will take about 10 minutes to test.

age at a few relevant points in the circuit (use the circuit diagram for reference). If everything checks out okay, switch off and install the ICs.

The LCD is connected to the circuit via a boxheader and a short length of flatcable. A suggested mounting method in the Pactec case may be gleaned from the photograph in **Figure 4**.

The board fits exactly on the three moulded bosses in the Pactec case, and is secured using small self-tapping screws. The LCD will have to be secured in the sloping part of the enclosure using glue or a similar substance, but only if its connections to the circuit are known to be correct.

If you don't see anything on the display, turn P4 fully anticlockwise. If this result in a dark upper line, but no text, you probably have a short-circuit between some of the tracks going to the display connector, or you may have some of the display wires mixed up. Check and repair before proceeding.

Figure 4 shows pieces of copper laminate or foil fixed to the inside of the case. A few pieces are also present under the board. This method of screening the sensitive circuit helps to avoid interference. The laminate should be connected to the PCB ground plane.

The suggested Pactec HPS-9VB enclosure has an internal battery compartment and requires a minimum amount of mechanical work. However, you'll have to carefully file away about 1 mm of each side of the LCD module board. This shouldn't be too difficult. There's no need to make any clearance for the LCD module, because the enclosure comes with a loose display panel.

Setting up the instrument using a DMM

Turn the contrast preset P4 fully anticlockwise and then slightly clockwise until a usable contrast is reached. The production tolerance on a number of makes of the CMOS 555 IC has been examined. In conclusion. we prefer the genuine Texas Instruments TLC55, Thomson TS555 or 3V555IN. If you decide to use other brands it may be necessary to alter the range resistors slightly.

Using close tolerance capacitors as reference devices, and the Thomson 555 ICs it proved possible to set up the next Capacitance Meter and achieve repeatable results by using nothing more than an ordinary multimeter. The only requirement is that the DMM is capable of reliably measuring resistance values in excess of 6 MΩ. This way it should be possible for *Elektor Electronics* readers to avoid special reference capacitors when setting up the instrument.

Remove IC1 and IC2 from their sockets. For the μF range: measure the resistance between pin6/7 of IC1 and the collector of T1, and adjust P3 for a 190 Ohm reading. For the nF range: measure the resistance between pin 6/7 of IC1 and pin 8 of IC2, then adjust P2 for a 5.94 kΩ reading. Finally, for pF range: unsolder one end of R3 (the end nearest the K1 connector), then measure the resistance between this end and pin 8 of IC1. Adjust preset P1 for a 6.0 MΩ reading.

Setting up the instrument using reference caps

Turn the contrast preset P4 fully anticlockwise and then slightly clockwise until a usable contrast is reached. To adjust the Capacitance Meter you should get hold of one precision capacitor whose value falls in the pF range, e.g., 470 pF and one that falls in the nF range e.g., 220 nF. These values can be obtained as 1% tolerance from Farnell, among others. A value like 1,000pF is not advisable because it will cause the meter to switch back and forth between a 999 pF and a 1.00 nF reading during adjustment.

Keep the instrument well away from any mains transformers. Be aware that your soldering iron or halogen light source may generate a strong magnetic 50 Hz field. Switch on the instrument and connect the 470-pF precision cap. Adjust P1 for a 470-pF reading. Remove the capacitor and connect the 220-nF cap, then adjust P2 for a 220-nF reading. For the μF range it will probably not be possible to get hold of a low-tolerance capacitor and if you do not have access to a commercial capacitance meter you may use an ohmmeter to adjust the series combination R1-P3 for a total resistance of 190 Ω.

SMD and trimming capacitors

If you construct a simple test jig (fixture) to hold SMD capacitors, the auto-zero function will cancel out the capacitance of the jig and make it very easy to test pF-range SMD caps. The same goes for trimmer and tuning capacitors (ceramic, PTFE or air types). Make a simple, mechanically stable jig that allows the capacitor to be soldered to (or into).

Switch on the instrument with the test jig connected up. The stray capacitance of the jig will be cancelled out. Next, solder the trimmer to the jig and measure. Adjust the trimmer and observe the varying capacitance. Make a note of its lowest and highest capacitance.

Burn your own

Assembly-code lovers, as well as those with access to a PIC programmer and blank 16F84 chips will be pleased to know that the author's source and hex code files for this project are available free of charge from the Free Downloads page at www.elektor-electronics.co.uk. The file number is **020144-11** under month of publication.

Final notes

Be sure to discharge any capacitor before testing it. Sure, the Capacitance Meter is equipped with a simple diode input protection, but you would not like to stress it into actual operation, would you? Various other, more advanced, protective circuits have been tested but they all seemed to degrade either the wide range or the accuracy of the meter. Still, the Capacitance Meter is rather robust if you just keep in mind that all capacitors should be fully discharged before testing.

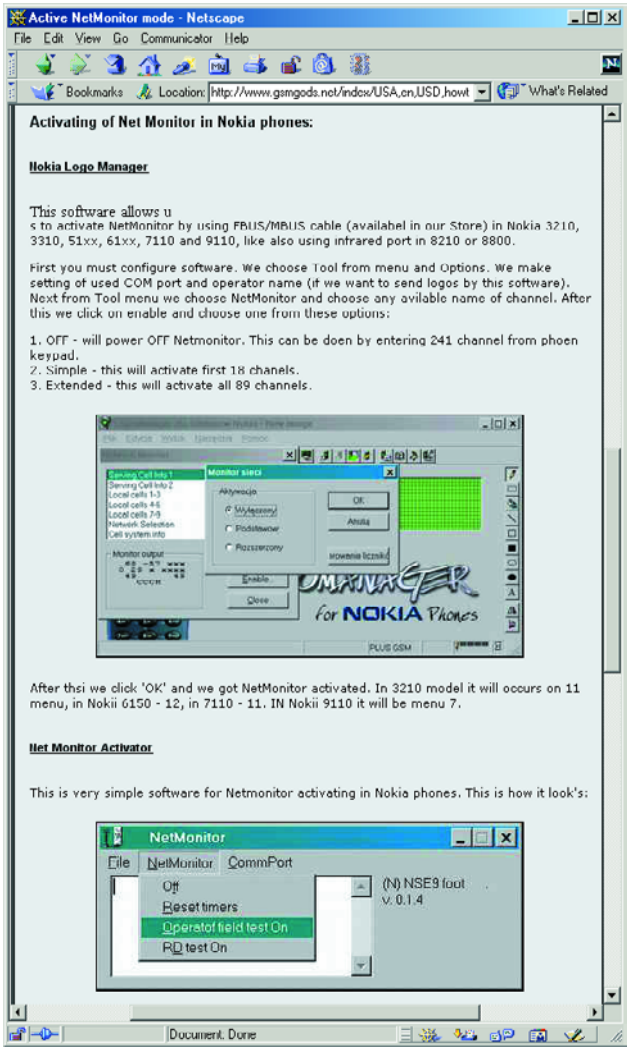
(020144-1)

Network Monitor for Nokia GSM Phones

Measure cellular radio fieldstrength

By Harry Baggen, with a contribution by P. Busch

Today's GSM phones have some many bells and whistles that many of us are confused (to say the least) by the plethora of menus and submenus. To add to the confusion, there exists a vast number of auxiliary programs ('tools') for even more options and functions. Among these programs we discovered a few that are of interest to electronics enthusiasts.


Active NetMonitor mode - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://www.gsmgods.net/index/USA,en,USD,howt
What's Related

Activating of Net Monitor in Nokia phones:

Nokia Logo Manager

This software allows u
s to activate NetMonitor by using FBUS/MBUS cable (availabel in our Store) in Nokia 3210, 3310, 51xx, 61xx, 7110 and 9110, like also using infrared port in 8210 or 8800.

First you must configure software. We choose Tool from menu and Options. We make setting of used COM port and operator name (if we want to send logos by this software). Next from Tool menu we choose NetMonitor and choose any avilable name of channel. After this we click on enable and choose one from these options:

1. OFF - will power OFF Netmonitor. This can be doen by entering 241 channel from phoen keypad.
2. Simple - this will activate first 18 chanels.
3. Extended - this will activate all 89 channels.

After thsi we click 'OK' and we got NetMonitor activated. In 3210 model it will occurs on 11 menu, in Nokii 6150 - 12, in 7110 - 11. IN Nokii 9110 it will be menu 7.

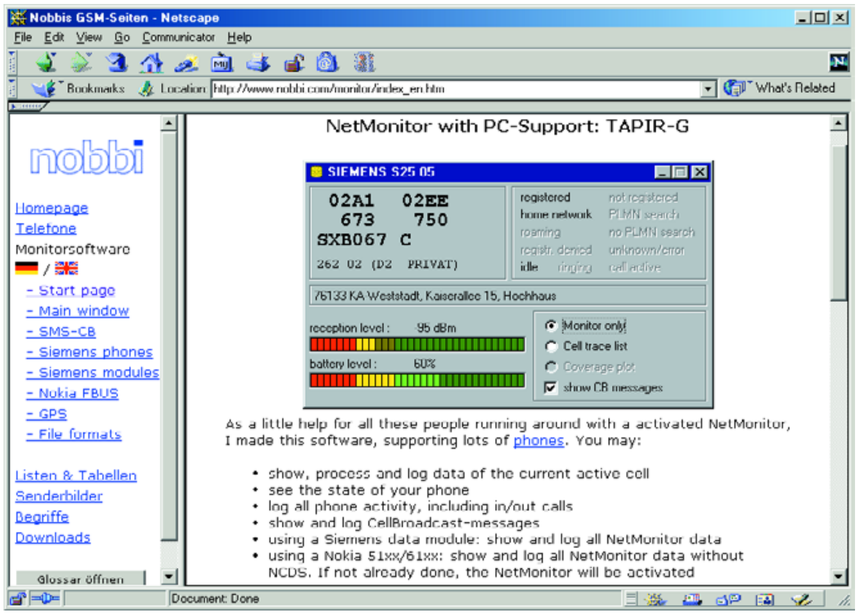
Net Monitor Activator

This is very simple software for Netmonitor activating in Nokia phones. This is how it look's:

Although there's no shortage of tips and tricks for mobile phones, it must be said that most of these are only useful for fanatic GSM users. And yet, there are tools worth experimenting with or having a look at if you're into electronics. For example, our reader Mr. Busch kindly informed us of the possibility to use a Nokia mobile phone to measure the local fieldstrength of GSM transmitter(s) within receiving range of the phone.

Fieldstrength measurement is one of the functions of the program Netmonitor which is available in Nokia 51xx and 61xx GSM phones. The program 'lies dormant' in the phone and needs to be activated. This requires a serial cable between the phone and the PC, plus a PC program like PC Locals or Logo Manager which allows Netmonitor to be activated. Once that has been done, the menu will show one additional option named Netmonitor. Selecting this menu option and then pressing '01' followed by Enter causes the received fieldstrength in dbμV to be indicated on the display, along with the associated channel number. Subsequent browsing then shows all channels being received, complete with the relevant fieldstrength. All very nice, but Netmonitor can do much more…

On the Internet, Netmonitor is a popular subject among technically inclined GSM phone users. This diagnostic mode, actually provided for service engineers, appears to be present on far more Nokia phones and the number of options is reported to vary between 19 and 133! A similar mode is also said to be available on other GSM phone brands, including Siemens, Motorola



NetMonitor with PC-Support: TAPIR-G

```
02A1   02EE
 673    750
SXB067  C
262 02 (D2  PRIVAT)
```

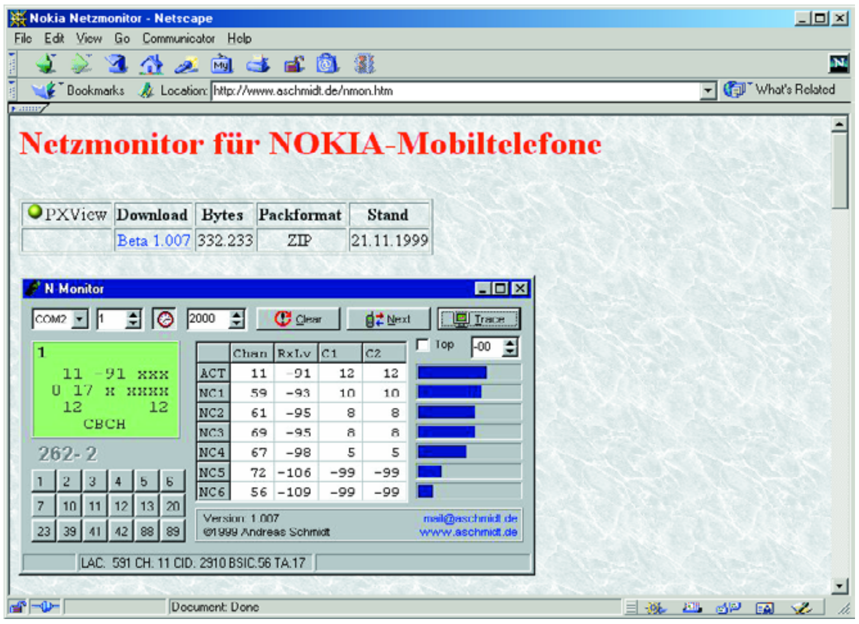
76133 KA Weststadt, Kaiserallee 15, Hochhaus

reception level : 95 dBm

battery level : 60%

As a little help for all these people running around with a activated NetMonitor, I made this software, supporting lots of phones. You may:

- show, process and log data of the current active cell
- see the state of your phone
- log all phone activity, including in/out calls
- show and log CellBroadcast-messages
- using a Siemens data module: show and log all NetMonitor data
- using a Nokia 51xx/61xx: show and log all NetMonitor data without NCDS. If not already done, the NetMonitor will be activated


Nokia Netzmonitor - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://www.aschmidt.de/nmon.htm What's Related
N Monitor
COM2
Clear
Next
Trace
Top
Document: Done

Netzmonitor für NOKIA-Mobiltelefone

PXView	Download	Bytes	Packformat	Stand
	Beta 1.007	332.233	ZIP	21.11.1999

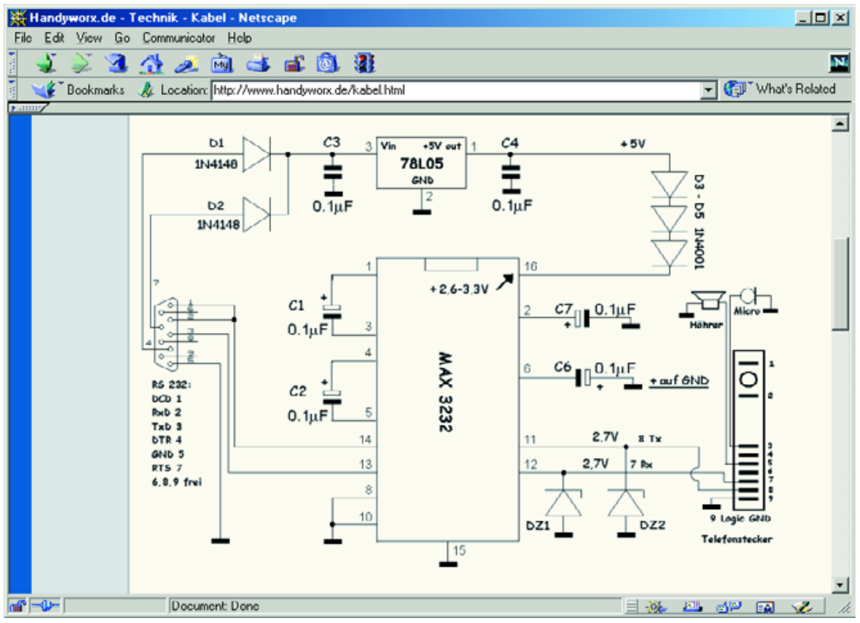
```
1
  11 -91 xxx
 0 17 x xxxxx
  12      12
    CBCH
262- 2
```

	Chan	RxLv	C1	C2
ACT	11	-91	12	12
NC1	59	-93	10	10
NC2	61	-95	8	8
NC3	69	-95	8	8
NC4	67	-98	5	5
NC5	72	-106	-99	-99
NC6	56	-109	-99	-99

1	2	3	4	5	6
7	10	11	12	13	20
23	39	41	42	88	89

Version: 1.007
©1999 Andreas Schmidt
mail@aschmidt.de
www.aschmidt.de

LAC: 591 CH: 11 CID: 2910 BSIC:56 TA:17



and Ericsson.

If you want to know more about Nokia Netmonitor and how to activate it, it is best to start at the **GSMGods.net** website [1]. These pages explain how to activate the hidden program using the Nokia Logo Manager utility. They also mention the program Net Monitor Activator which has been specifically written for the purpose.

Fortunately, extensive manuals are also available for Netmonitor. After all, some documentation will be required particularly in the mode covering 89 test pages! A condensed version is available on **GSMworld** [2], which also shows the various monitor windows that pop up. A slightly more extensive manual may be found at, among others, the website published by **Marcin Wiacek** [3] as well as at **Nobbis GSM Seiten** [4].

The latter website has more interesting things to offer. For example, it contains **Monitorsoftware** [5] that allows several data (including fieldstrength) of the active cell transmitter to be shown on the PC screen.

Another useful program that works in combination with Netmonitor is called N-Monitor by **Andreas Schmidt** [6]. All data supplied by Netmonitor is neatly arranged and displayed on the PC screen.

As an aside, if you do not yet have a PC link cable for your GSM phone, have a look at **Handyworks.de** [7]. This website not only provides details on Netmonitor and various other software, but also contains a guide to home construction of a data cable for Nokia GSM phones.

(035012-1)

Internet Addresses

[1] GSMGods:
www.gsmgods.net/index/USA,en,USD,howto,subpage_id,active_netmonitor.html

[2] GSMworld.net:
www.gsmworld.net/nokia/netmonitor/net_main.shtml

[3] Marcin Wiacek:
www.mwiacek.com/english/gsm/netmon/faq_net0.htm

[4] Nobbis GSM Seiten:
www.nobbi.com

[5] Nobbis Monitorsoftware:
www.nobbi.com/moni1_nok.htm

[6] Andreas Schmidt:
www.aschmidt.de/nmon.htm

[7] Handyworks.de:
www.handyworx.de/netmonitor.html

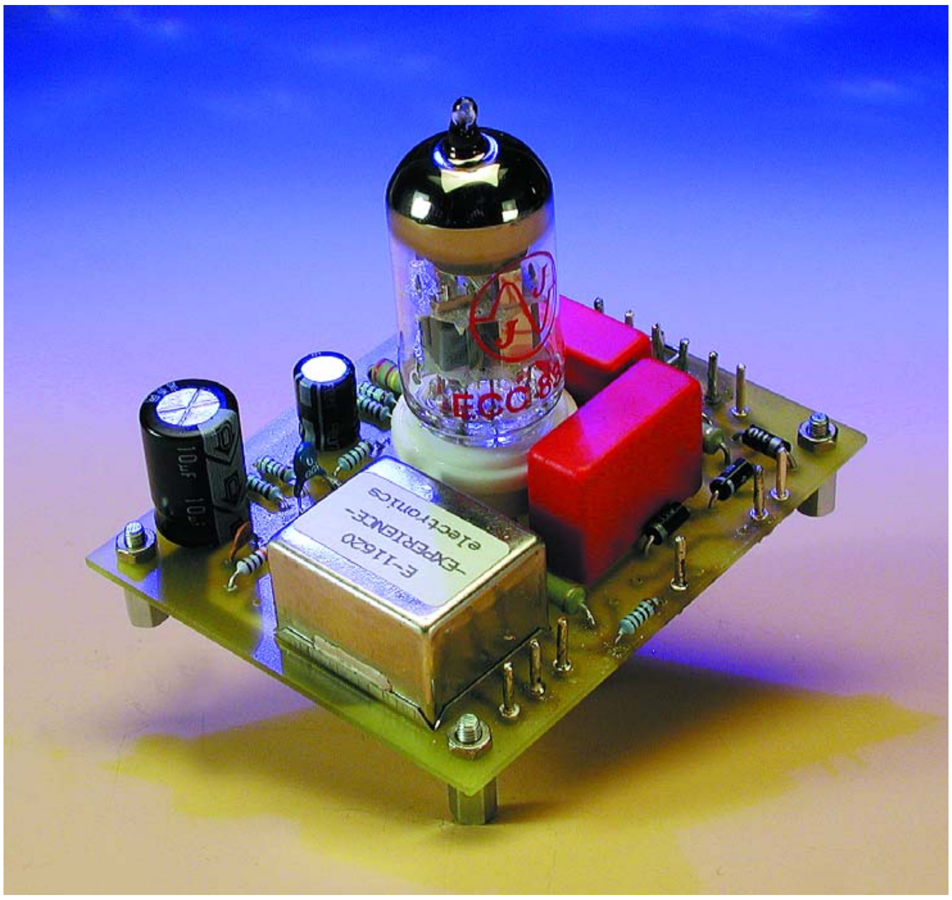
ECC83 (12AX7) Microphone Preamplifier

studio quality with valves

Design by G. Haas

experience.electronics@t-online.de

In this semiconductor age, we find valves being used increasingly often for hi-fi and guitar amplifiers, top-end condenser microphones and studio equipment. This article presents an excellent microphone amplifier with a uniquely attractive sound.



Microphone amplifiers must amplify extremely small signals to much higher levels while introducing the least possible amount of additional noise. In principle, it does not matter whether a transistor, operational amplifier or valve us used as the gain element.

A signal can be amplified by any desired amount, but the limit is set by the signal-to-noise ratio. If the magnitude of the noise signal is equal to or greater than that of the desired signal, any amplification is pointless. Consequently, microphone amplifiers must be designed to have the lowest possible levels of hum, noise and distortion, since every corruption of the signal originating in the microphone amplifier will be magnified by the following amplifier. Particular attention must therefore be given to the design of the input stage.

A low-noise transistor or low-noise valve will not by itself automatically yield a low-noise amplifier.

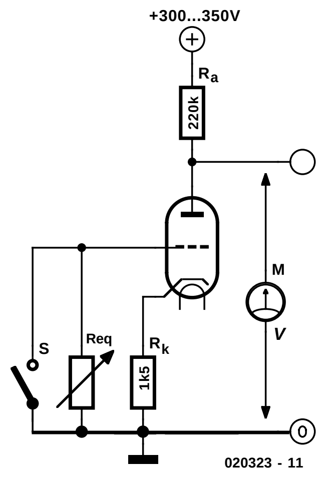


Figure 1. Basic noise measurement circuit

$U_{ntot} = \sqrt{(U_V{}^2 + U_{Req}{}^2)}$

$U_{V2} = U_{Req}{}^2$

$U_{ntot} = UV \cdot \sqrt{2}$

U_{ntot}	= total noise voltage
U_V	= valve noise voltage
U_{Req}	= noise voltage of resistor
R_{eq}	= equivalent noise resistance

Specifications

Supply voltages		350 V at approx. 4 mA
	for ECC83S	12.6 V/0.15 A
	for ECC808	6.3 V/0.34 A
Frequency response	a_u = 40 dB	28 Hz - 24 kHz (–1 dB)
Input impedance	1 kHz	approx. 900 Ω
Unweighted noise voltage	20 Hz - 20 kHz	–72.5 dBm
Noise voltage		–81.0 dBm(A)
	CCIR-468	–67.8 dBm
Input referenced noise voltage	CCIR-468, a_u = 50 dB	–117.8 dBm

Harmonic distortion	dtot	d2	d3	d4	d5	
–40 dBm, a_u = 30 dB	0.342%	0.020%	0.287%	0.018%	0.041%	at 80 Hz
	0.023%	0%	0.001%	0%	0%	at 1 kHz
–40 dBm, a_u = 40 dB	0.353%	0.030%	0.294%	0.018%	0.040%	at 80 Hz
	0.025%	0.006%	0.001%	0%	0%	at 1 kHz
–40 dBm, a_u = 50 dB	0.350%	0.023%	0.293%	0.018%	0.040%	at 80 Hz
	0.046%	0.036%	0.003%	0%	0%	at 1 kHz

Noise arises from the motion of electrons in any type of electrical conductor. The fundamental noise level of a given component is set by its construction and the materials used. The noise generated by an input stage is determined by the valve noise (or semiconductor noise) and the internal resistance of the signal source (resistance noise).

Noise measurements

Figure 1 shows a measurement circuit that can be used to determine the equivalent noise resistance (R_{eq}) of the valve used here (ECC83). The values of Ra and Rk are typical for this type of valve, but they anyhow do not have any effect on the measurement. First, the noise voltage of the valve (U_V) is measured at the anode with switch S closed, using a millivolt meter. The switch is then opened, and the value of R_{eq} is adjusted until the measured value is a factor of √2 greater. The value of R_{eq} is then recorded; this is the equivalent noise resistance of the valve. From the formulas, it can be concluded that if R_{eq} is smaller than R_V, valve noise predominates, while if R_{eq} is greater than R_V, resistance noise predominates.

If a pentode is used instead of a triode, there is an additional noise source in the form of partition noise. In a pentode, the number of electrons leaving the cathode is larger than the number arriving at the anode. As more electrons leave via the screen grid, the noise level increases. This is why we often see an EF86 pentode, which has low noise and microphonics, wired as a triode. The larger gain that can be achieved with the pentode configuration has been foregone in favour of better noise performance. A pentode in the triode configuration, or just a triode, is often used in such cases. Triodes also have a structural advantage over pentodes, in that they tend to produce second-harmonic distortion. This is

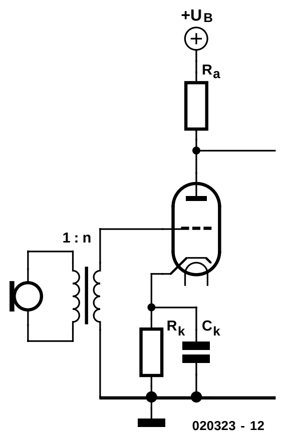


Figure 2. Microphone impedance matching using an input transformer.

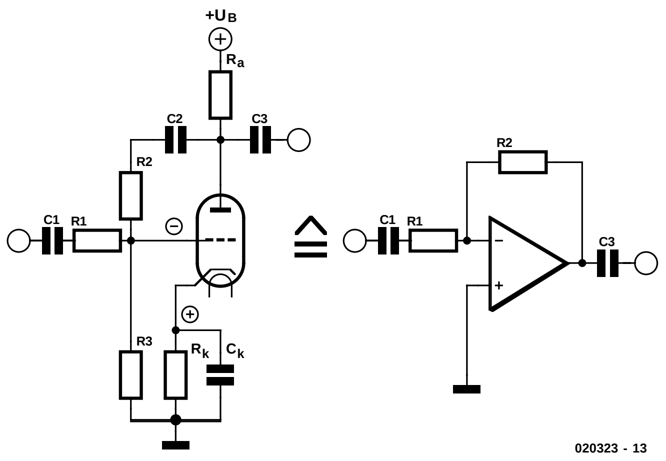


Figure 3. An inverting operational amplifier using a valve.

more pleasant to the ear than the 'scratchy' third-harmonic distortion produced by pentodes due to variations in the division of the cathode current between two electrodes, the anode and the screen grid, which depends on the drive level.

Transformer matching

In traditional circuits, such as that shown in **Figure 2**, an input transformer is used to match the microphone impedance to that of the valve. This transformer typically has a turns ratio of 1:10 to 1:30. With an input transformer, it is possible to boost the input signal level with practically no noise. However, stray circuit capacitances in combination with transformer capacitances limit the upper corner frequency and linearity of this arrangement, especially at large turns ratios. This problem can only be mastered using an elaborate transformer construction and sophisticated circuit design. The valve in Figure 2 works without feedback, so the amplification factor depends only on the turns ratio of the input transformer and the transconductance (g_m) of the valve. If the valve is replaced, the gain may also change.

Opamp circuits

A valve can also be wired as an operational amplifier, as shown in **Figure 3**. The plus and minus signs next to the valve electrodes identify the corresponding inputs of the valve opamp. Capacitors C1–C3 serve only to separate dc and ac voltages; in principle, they have no further effect. The grid-leak resistor R3 is needed by the valve, but its resistance is so large that it has no significant effect on the overall circuit. The cathode of the valve corresponds to the non-inverting input of the opamp. Since Rk is needed to set the dc operating point of the valve, it must be bypassed for ac signals by Ck to connect this input to signal ground. Now we have an inverting opamp whose gain is set by the resistance ratio R2:R1, independent of the amplifying component. Of course, the open-loop gain of this component must be significantly greater than the value of R2:R1. The input resistance of the circuit is equal to that of R1. As the value of R2 cannot be made arbitrarily large, since the value of grid-leak resistor R3 also cannot be made arbitrarily large, the value of R1 will be relatively small for large amplification factors. This imposes a significant load on the signal source. The internal resistance of the signal source forms a voltage divider in combination with R1. The control grid, just like the inverting input of an opamp, represents a virtual ground.

If the opamp circuit is modified as shown in **Figure 4**, the gain is essentially determined by the ratio R_B:R_A. This gives us considerably more freedom in selecting the values of R1 and R2. If R1 and R2 are now replaced by an impedance-matching transformer, R1 becomes the source impedance of the signal source and R2 becomes R1 × n^2. An equivalent circuit using a triode guarantees high gain with low noise. However, this arrangement has the disadvantage of having a limited amount of fundamental gain.

This situation can be improved using the circuit shown in **Figure 5**, which includes an additional valve. V2 acts as an impedance converter, since the feedback signal is taken from the cathode resistor. This yields the same considerations for R_A and R_B as in Figure 4, but since the cathode resistor of V1 is not bypassed, the fundamental gain is less. This has a beneficial effect on the distortion characteristic and long-term stability of the circuit, due to the use of negative feedback. The emissivity of valve cathodes decreases with age. If a lower level of system gain is used from the start, the useful life of the valves is extended. Valve V2 makes up for the missing gain. Here again, the cathode resistor is not bypassed with a capacitor, since the ac voltage on the cathode is needed for the negative feedback. Overall negative feedback is also provided via R_{FB}, in order to constrain the characteristics of the overall system without requiring selected valves to be used.

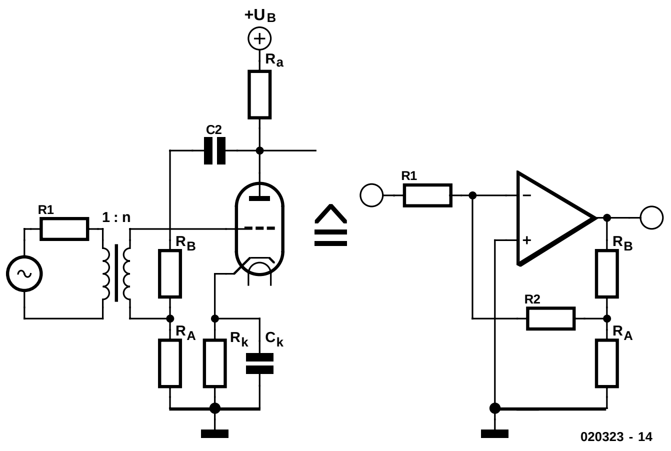


Figure 4. A non-inverting operational amplifier using a valve.

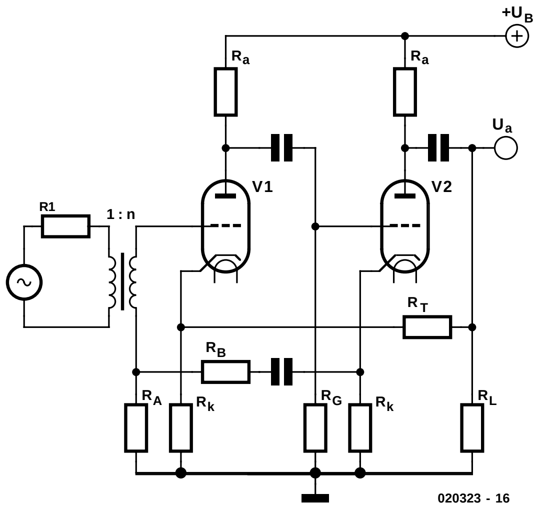


Figure 5. Amplifier with impedance converter.

Microphone preamplifier

Figure 6 shows the complete schematic diagram of the preamplifier, with all component values. The input transformer (type E-11620), which is one of the most important components for this application, is wound with a turns ratio of 1:8+8. Here it is wired for a 1:16 ratio. This provides a good compromise between signal level boosting and the noise performance of the circuit. Furthermore, this transformer can also be used for other purposes, so its price can be kept within reasonable limits by virtue of a relatively large production volume.

The input transformer can be used with an input level of around 800 mV$_{eff}$ at 40 Hz, but that does not mean that the amplifier circuit should be fed such a strong input signal. The maximum input level depends on the maximum output level of the complete installation. The transformer is fully encased in mu-metal, since otherwise even minute amounts of coupled-in noise would be amplified to high levels by subsequent amplifier stages.

The component values have been chosen to allow a gain of around 25 to 60 dB to be used with high sound quality. The gain is essentially determined by the values of R6 and R15. A gain of 25 dB is provided by the signal level boost of the input transformer alone. A fixed minimum gain can be thus set using R6. R15 can also be replaced by a wire bridge, a selector switch with fixed dB settings, or a trimpot. Of course, only premium-quality components should be used for this purpose. The selector switch must have gold-plated contacts and make-before-break switching, since otherwise it will produce crackling noises and switching clicks.

Coupling capacitors C4 and C5 are specially marked in the schematic diagram. The marking indicates the lead connected to the outer foil of the capacitor, which should be connected to the non-critical side of the circuit. Many types of film capacitors are correspondingly marked. The result is that the capacitor screens itself, thereby reducing the susceptibility of the circuit to interference.

The printed circuit board, whose layout is shown in **Figure 7**, allows the input transformer to be used at a ratio of either 1:16 or 1:8 by means of wire bridges. This allows other types of valves with the same basing to be used, such as the ECC81, ECC82 or similar dual triodes. However, if a different type of valve is used, the component values cannot simply be used as is. It is essential to modify them as necessary to match the dc operating point of type of valve used.

Components R3, C1 and C9 attenuate the resonance peak formed by the input transformer in combination with the amplifier circuit, in order to make the frequency response of the amplifier as flat as possible. The indicated component values can be adjusted as necessary according to circumstances. With the indicated values, the overall arrangement has a slight rise in the frequency response (around 0.8 dB) at 17.7 Hz. This could be suppressed even more, but only at the expense of a lower corner frequency at the high-frequency end.

Resistor R1 provides a finite load for the input transformer. The grid of the valve has such a high impedance that the transformer would otherwise operate with practically no load on the secondary. Since this can also result in a non-linear frequency response, a finite load impedance provides a definite benefit.

High-quality power supply

Both the enclosure for the circuit and the power supply must meet demanding require-

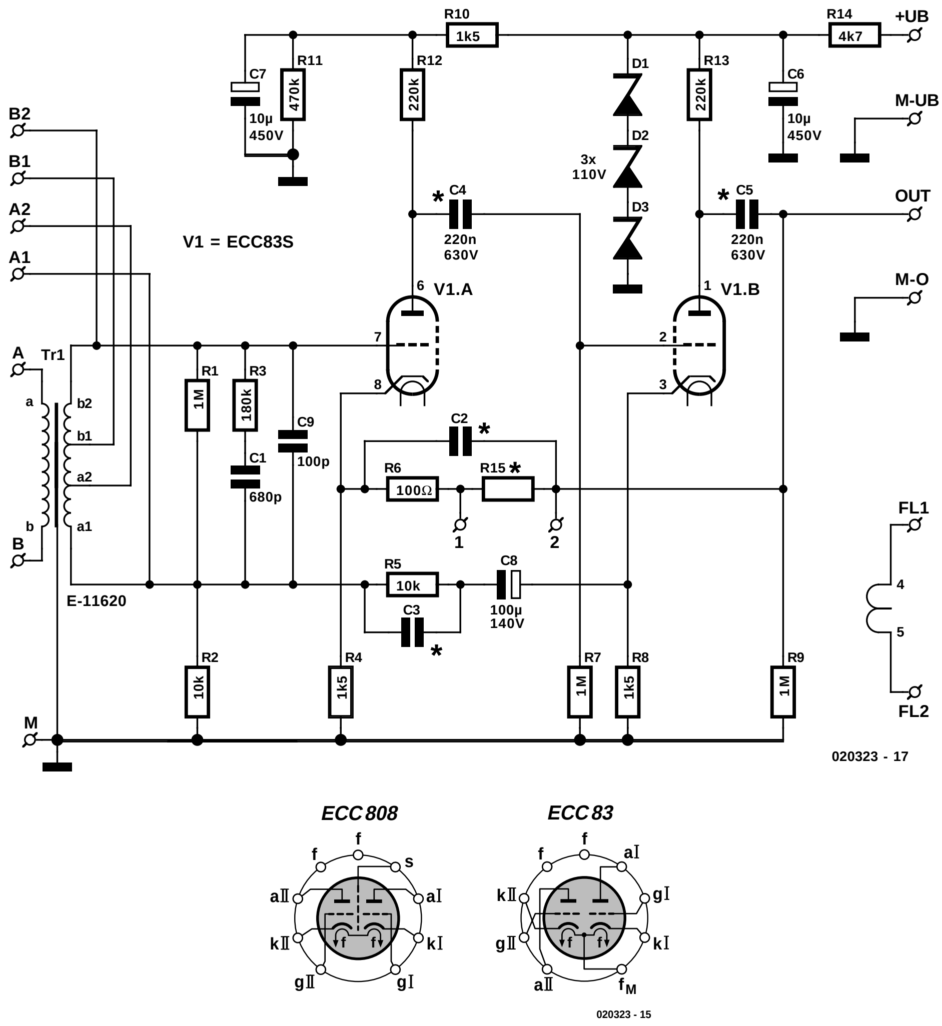


Figure 6. The final circuit of the microphone preamplifier, including the base diagrams for the two types of valves used.

f	filament	**aI**	anode I	**aII**	anode 2
gI	grid I	**gII**	grid 2	**kI**	cathode I
kII	cathode 2	**fM**	filament tap	**s**	screen

(as seen from the bottom viewing the pins)

COMPONENTS LIST

Resistors:
(metal film, 1% tolerance, 0.7 watts, unless otherwise noted)

R1 = 1MΩ
R2 = 10kΩ
R3 = 180kΩ
R4 = 1kΩ5
R5 = 10k
R6 = 100Ω
R7 = 1MΩ
R8 = 1kΩ5
R9 = 1MΩ
R10 = 1kΩ5
R11 = 470kΩ, metal oxide, 2% tolerance, 2W
R12,R13 = 220kΩ, metal oxide, 2% tolerance, 2W
R14 = 4kΩ7
R15 = see text and Table 2

Capacitors:
C1 = 680pF ceramic
C2,C3 = only fitted when oscillation or RF noise is noted (approx. 10-47pF)
C4,C5 = 0.22µF 630V, MKS4, lead pitch 22.5mm
C6,C7 = 10µF 450V, lead pitch 5mm
C8 = 100µF 40V, lead pitch 5mm
C9 = 100pF ceramic

Semiconductors:
D1,D2,D3 = 110V zener diode, 1.3W

Miscellaneous:
R1 (Ü1) = E-11620
V1 (Rö1) = ECC83S, E83CC, 12AX7, ECC808 (see text)
1 valve socket, ceramic, PCB mount

Kits, special parts and PCBs available from

Experience Electronics
Weststrasse 1
D-89542 Herbrechtingen
Germany

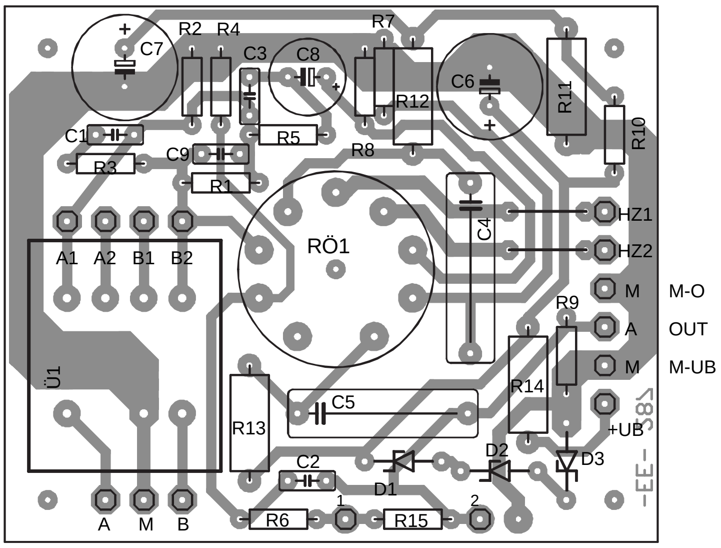
Internet: www.experience-electronics.de
E-Mail: experience.electronics@t-online.de

Tel.: +49 7324 5318
Fax: +49 7324 2553

Maximum input voltage

(as a function of gain, for 1 percent total harmonic distortion)

a_u	u_i	R15
25 dB	375 mV	0 Ω
30 dB	180 mV	11 kΩ
40 dB	180 mV	62 kΩ
50 dB	85 mV	173 kΩ



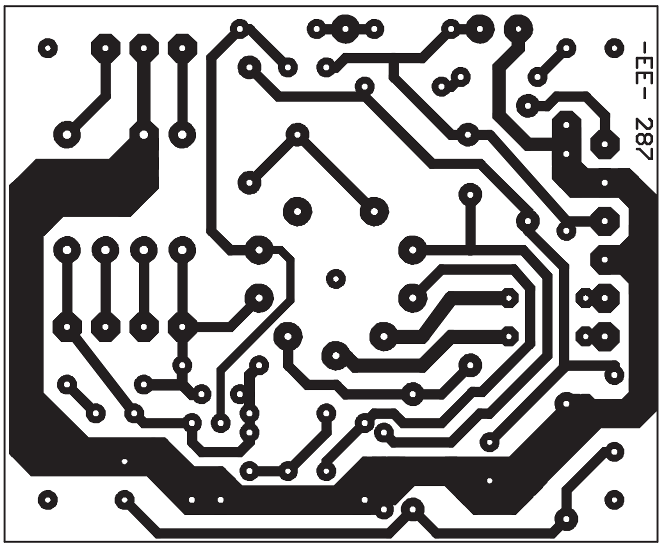
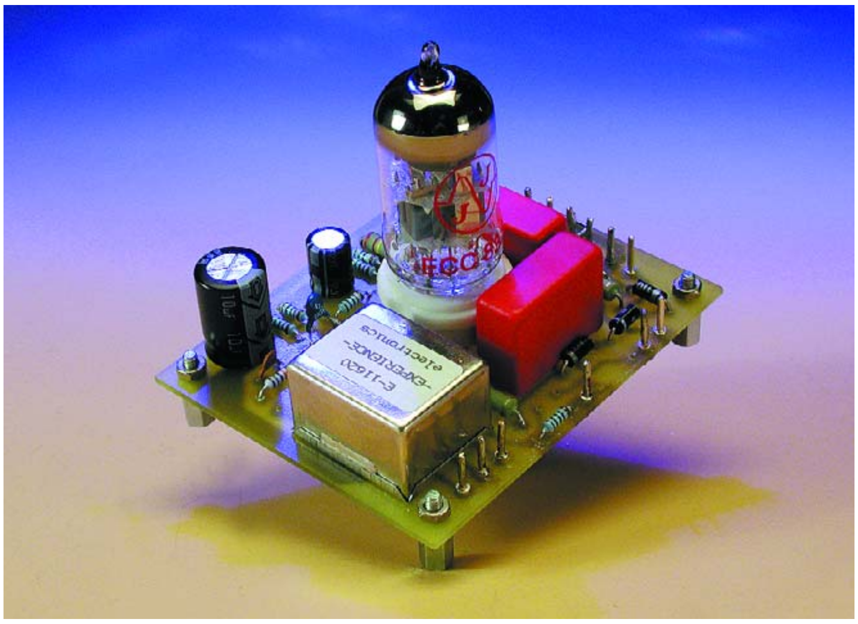
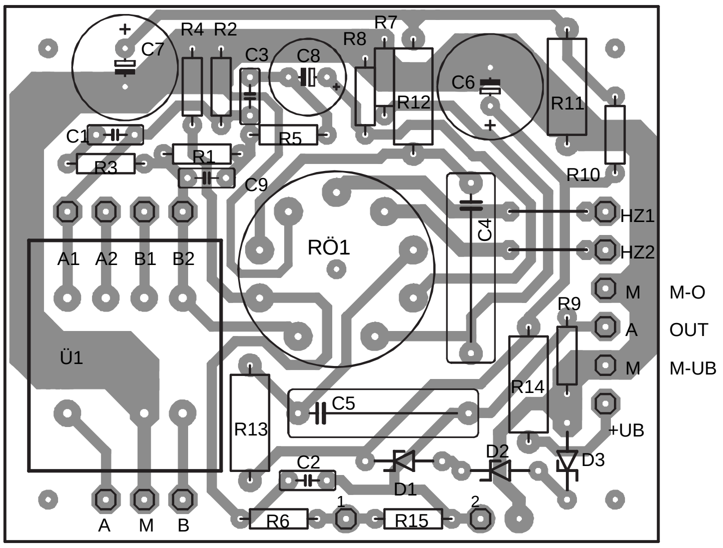


Figure 7. Circuit board layout for ECC83 (board available from Experience Electronics).





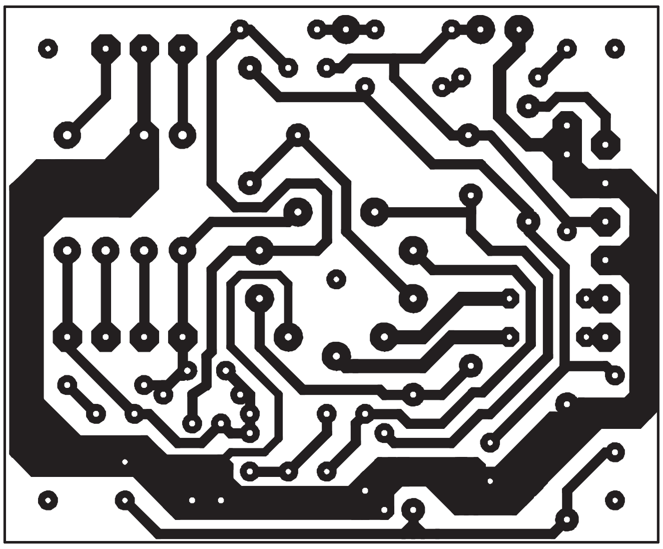


Figure 8. Circuit board layout for ECC808 (board available from Experience Electronics).

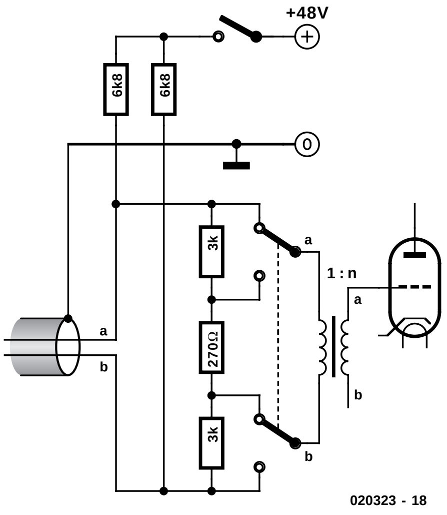
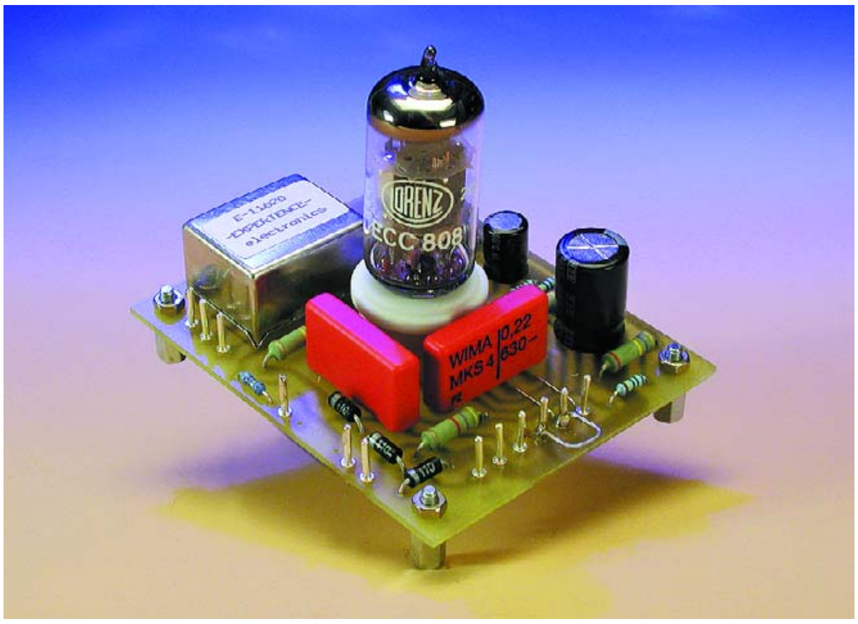


Figure 9. –30 dB input attenuator and connections for a phantom supply.



ments, since the circuit will deliver good results only if it is fitted into a fully screened metallic enclosure. The valves are heated using a 12.6-V dc voltage. The high voltage must be well smoothed. Suitable circuits have already been presented for the Valve Preamplifier (*Elektor Electronics*, June through September 2000 issues). Zener diodes D1–D3 must be used if several preamplifiers are powered from a single supply, or if the power supply has passive RC smoothing. The power supply output voltage must be 350 V.
Using a stabilised supply voltage provides the valves with well-defined operating conditions, which is beneficial since the gain of a triode more or less depends on the value of the supply voltage. An important point is that the negative terminal of the filament voltage must be connected to the negative terminal of the high voltage.

When choosing a valve type, you should pay attention to certain details. The measured performance values were achieved using an ECC83S, which is a cross between the ECC83 and the E83CC (military version). The noise figures of ECC83S valves are significantly better than those of standard ECC83 valves, so the ECC83S is clearly preferable. The ECC83 is also available with a variety of American designations, such as 12AX7, which exactly corresponds to the standard ECC83. The 12AX7A and 12AX7WA are versions with tighter tolerances, lower noise and lower microphonics, while the 7025 is the long-life version. An E83CC, or one of the equivalent American military versions with type numbers such as 6681, 6057 and 5751, can also be used if desired. Although these types are significantly more expensive, they have the advantage of being less microphonic and having longer service lives than the standard type.

The term ‘microphonic’ refers to the fact that mechanical vibrations, particularly in the control grid, can modulate a valve and lead to unpleasant noises or howling in an amplifier installation. This is thus not the place to cut costs in a good-quality microphone preamplifier.

The preamplifier should not be fitted into the same enclosure as the power supply, since otherwise electromagnetic interference and mechanical humming from the mains transformer can manifest themselves in an unpleasant manner. In some cases, it may be necessary to mount the circuit board elastically, for instance using rubber bushings. The circuit is designed such that it is not necessary to use selected valves.

There is yet another interesting option. The ECC808 valve was developed in response to the shortcomings of the standard ECC83 or its direct equivalent the 12AX7. The ECC83 and ECC808 are identical electrically, but the noise characteristics of the ECC808 are better by a factor of three, it is less sensitive to hum and it is significantly less microphonic. Its noise characteristics roughly match those of the ECC83S. In addition, it has a screen between the two triodes, which is of secondary importance in this application. The base arrangement is also different, with the control grid pins being located well away from the anode and heater pins. Consequently, and ECC808 cannot be used as a direct replacement for an ECC83. For this reason, we have also developed a second circuit board layout, as shown in **Figure 8**. The component values remain exactly the same, with the only difference being that the ECC808 requires a filament supply of 6.3 V dc at 0.34 A, instead of the 12.6 V at 0.15 A used for the ECC83. Unfortunately, the ECC808 is not exactly cheap, since it has become scarce. However, it represents an interesting alternative, and its price can be justified in a high-quality microphone preamplifier stage.

Interpreting the measured values

The measured values for the amplifier, which are shown in **Table 1**, require a little bit of interpretation. The open-loop gain, which is the gain when R15 is not fitted, is around 68 dB. If we want to allow a maximum gain of 60 dB, this leaves only 8 dB for negative feedback, which is not very much. Valves do not have high open-loop gains, unlike modern opamps. Consequently, it is recommended to select a gain in the range of 30 to 50 dB, since the best results with regard to harmonic distortion and frequency response will be obtained in this range.

The harmonic distortion measurements were made at 1 kHz and 80 Hz. As can be seen, the harmonic distortion increases at low frequencies, particularly odd harmonics. The influence of the input transformer can be seen here, since matching transformers generate predominantly this type of harmonic distortion components. The even harmonics can be attributed to the valves. The second-harmonic component has a pleasant sound that is typical of a good ‘valve sound’. An increase in harmonic distortion at low frequencies is not especially serious, since the ear is relatively insensitive in this range. Another thing that can be seen from the harmonic distortion values is that the total harmonic distortion at 1 kHz is greater than the average value of the individual harmonic distortion values. At this frequency, amplifier noise predominates. In this case, the measurement equipment cannot distinguish between harmonic distortion and noise, since it makes broadband measurements at frequencies above 1 kHz.

The noise values are to be understood as absolute voltage levels at the output of the amplifier. The input-referenced noise values are obtained by assuming a noise-free amplifier with a noise source at a certain level connected to its input. Three noise values are given: 20 Hz – 20 kHz, A-weighted and CCIR-486. The CCIR-486 filter is used with studio equipment. With this filter, instead of measuring the effective noise value, the rectified peak value is measured using a filter characteristic similar to that of an A-weighted filter, but with the noise components between 1 kHz and 12 kHz being significantly more heavily weighted. That is why it gives the worst noise value.

In order to correctly evaluate the amplifier, it is necessary to correctly interpret the measurements. If a 200-ohm metal-film resistor is connected to the inputs of the instrument, a level of around –118 dBm is measured using the CCIR-486 filter. A dynamic microphone with a source impedance of 200 Ω generates a noise voltage of –118 dBm. If we assume our amplifier to be noise-free and subtract its gain from the noise level measured at its output, we arrive at a value of –117.9 dBm (weighted using the CCIR-468 filter). This means that the amplifier is only 0.2 dB away from what is physically achievable (0 dBm = 775 mV, the standard studio level).

There is another important point to consider, namely the maximum input voltage. It must be borne in mind that the input transformer boosts the input level by a factor of 16. Thus, if a level of 10 mV is present at the transformer input, the voltage on the grid of the first valve is already 160 mV. Since the grid voltage is only around –1.2 V, the knee of the characteristic curve is reached fairly quickly. The maximum input level for 1 percent harmonic distortion depends on the gain. Several typical values are listed in Table 2. A value of 85 mV for a gain of 50 dB may not appear particularly high. However, a dynamic microphone has a nominal level of 2 mV. If the amplifier can handle 85 mV, there is still 18 dB of headroom.

If you want to use this amplifier with a relatively high input signal level, an input attenuator should be used as shown in **Figure 9**. With the indicated component values, the attenuation is approximately 30 dB. If you want to have an exact value, or if you want to modify the attenuation, you can adjust the value of the 270-Ω resistor. Figure 9 also shows how a 48-V phantom supply can be implemented.

(020323-1)