SEPTEMBER 2001
£2.95

ELEKTOR ELECTRONICS

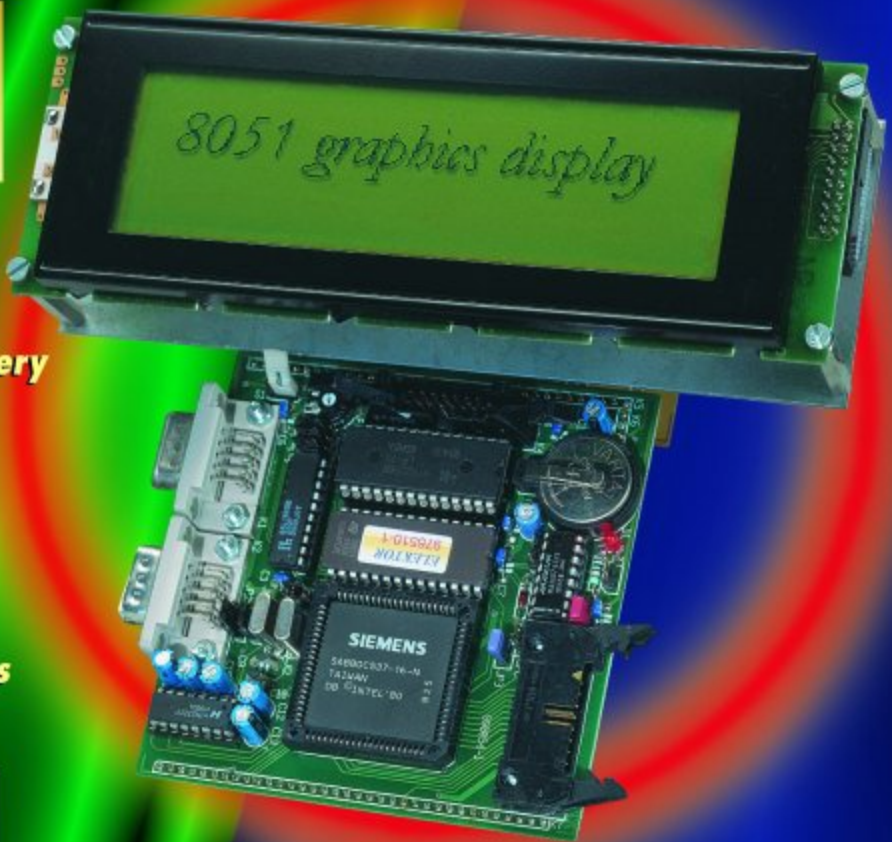
THE ELECTRONICS & COMPUTER MAGAZINE

www.elektor-electronics.co.uk

Personal Mini Webserver

Graphic LCD Module for 8051 Micros

Atmel AVR Micro Programmer



- Lead-Acid Battery Revitaliser
- MIDI-DMX Converter
- Video-SVHS Converter
- Servo on I^2C
- Li-ion Batteries

09
9 770268 451968

Capacitor ESR Tester

the good, the bad and the leaky…

Design by Flemming Jensen

How about an in-circuit capacitor tester to take the strain out of tracking down faulty capacitors? No need to solder out any capacitor, simply check it in-circuit, from thousands of microFarads down to a hundred nanoFarads. In most cases, parallel coils or low value resistors are no problem. Even shorted caps may be revealed in-circuit and polarity is irrelevant for the tester. High ESR? Replace!

The most significant property of a capacitor is its capacitance but besides that there's another important factor, namely the so called **ESR**, or Equivalent Series Resistance. An ideal capacitor is a purely reactive component with a 90-degree phase angle between current and voltage. In the real world, however, a capacitor needs to be modelled as an ideal capacitor in series with a resistor representing the losses introduced by the component. The equivalent circuit is shown in **Figure 1**. Sure, capacitance can be measured by means of a capacitance meter, which is pretty common nowadays, but unfortunately this test won't tell anything about the capacitor's quality — we need to know the ESR as well. Over time, electrolytics tend to dry out, which will raise their ESR and inevitably the voltage drop inside the capacitor. Evidently, the pure reactance X_c can not produce heat, due to the 90-degree phase shift between voltage and current, but the ESR can, and in switching circuits the resultant heat will cause a further degradation of the capacitor's quality i.e., a further rise in ESR. It's fairly common to find electrolytics that on the face of it have only lost just a few

How does ESR influence circuit behaviour?

In (fast) switching circuits, a low ESR may be crucial for proper circuit behaviour. For example, in a TV set, high capacitor-ESR may lead to inability to quit stand-by mode, incorrect picture height or width, synchronisation problems, interference or hum bars. In Switch Mode Power (SMPSUs) supplies, high ESR caps may lead to blown semiconductors, blown fuses or no start up. In power circuits, a rising ESR will make the capacitor warm up, leading to even higher ESR and eventually circuit breakdown. The usual method to troubleshoot these problems involves soldering out the capacitors, measure the capacitance and solder the good ones back in. A tedious task, but what's even worse, ailing capacitors often don't show a low capacitance, get soldered back in again and then the troubleshooting gets really time consuming.

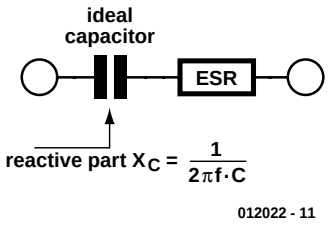


Figure 1. The most important property of a capacitor is its capacitance but beside that there's another important factor, namely the so-called ESR, or Equivalent Series Resistance.

percent of their rated capacity although their ESR is in the hundred Ohms range. Obviously, such a component acts as a load just running hot and wasting a lot of energy.

The measuring principle

The capacitor under test, C.U.T., is fed with a 100-kHz constant-current square wave signal. The ESR value is determined by measuring the AC voltage drop across the C.U.T. If the capacitance is high enough compared to the frequency, the voltage drop over the internal reactance is negligible and the drop is caused by the ESR only. This voltage is converted to DC and fed to the voltmeter section.

AC to DC conversion of a 100-kHz signal in the millivolts range presents a real design challenge. Furthermore, the conversion needs to be as linear as possible because we want to use an ordinary 200-mV DVM readout. It goes without saying that an ordinary diode rectifier will not suffice, and an active diode rectifier with opamps will have a hard time working at 100-kHz and a few millivolts. The solution we came up with is a **synchronous rectifier** — essentially a polarity changer controlled by the same generator that supplies the 100-kHz test signal. This circuit works surprisingly well and is cheap, too!

A simplified version of the circuit is shown in Figure 2. Here, the C.U.T. is assumed to be a 100-µF with an ESR of 10 Ω. As shown, the reactance is negligible and the ESR, which is purely resistive, is dominating.

Although the above principle works well, further reduction of the reactive influence is called for.

Figure 3 shows an example where the C.U.T. is 0.1-µF cap whose ESR is zero Ohms. As mentioned above we use a relatively high frequency to make the reactance negligible while enabling even the smallest electrolytics like 0.1-µF to be tested. For this it is necessary to reduce the influence of the beginning integration of the waveform even further. The ESR is zero and the reactance is 15 Ω. As can be seen, the integrated waveforms presented to the differential amplifier inputs result in a sawtooth centred around zero volts at the output. After integration, in the RC network that follows, a DC level of zero volts is fed to the voltmeter circuit. If the C.U.T. also represents an ESR of, say, 10 Ω, the sawtooth at the output will still have the same waveform. However, it will be DC-shifted in the positive direction by an amount representing

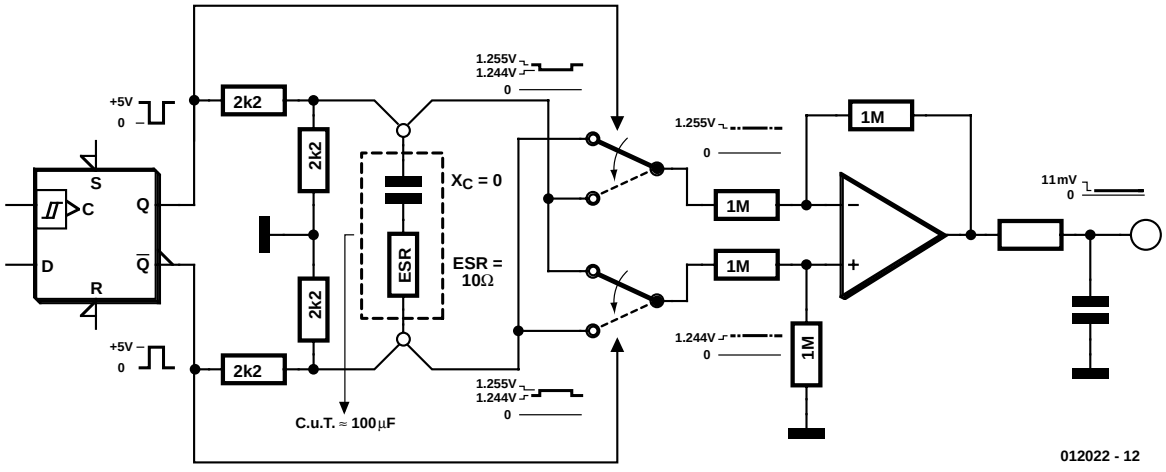


Figure 2. Illustrating the principle of operation. Assuming that the C.U.T. is a 100 µF capacitor with an ESR of 10 Ohms, the reactance is negligible and the ESR, which is pure Ohmic, is dominating.

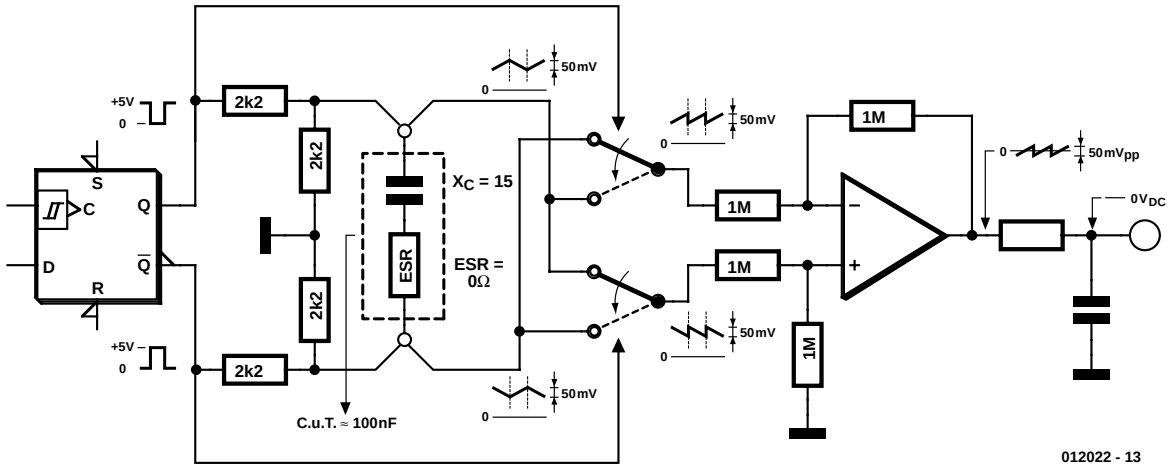


Figure 3. Second hypothetical test: C.U.T. is a 0.1-µF capacitor with an ESR of 0 Ω.

the ESR value. After integrating the sawtooth away, the output will give the proper reading of 10 Ω, excluding the 15 Ω reactance.

Low-ESR cap or shorted cap?

You may question if you're testing a low-ESR cap or simply a shorted one. A simple DC Ohms test is usually enough to decide this. No need to get out the multimeter — with a push of a button the ESR tester becomes a DC Ohmmeter and your display should change to a higher Ohm reading. If it doesn't, the odds are you have a shorted cap on your hands.

Some practical ESR values, please?

So how high will the ESR be then? Well, that depends on where the capacitor is used, the type, the make, the voltage rating, etc. A 2,200-µF reservoir capacitor with an ESR of 10 Ω may be fine in a linear power supply, while a 2,200-µF one having 1 Ω ESR may be grossly inadequate in a switch-mode PSU.

In general, if a large capacitor, as in this example, reads more than one Ohm, you should be suspicious and run a comparison on a similar component. But don't worry! It won't take long before you are able to distinguish bad caps from good ones. If you regularly are repairing SMPSUs, TV sets, monitors etc. you will soon appreciate the ESR tester.

Circuit diagram

Let's have a look at the circuit diagram of the Capacitor ESR Tester — see **Figure 4**. A 200-kHz square wave generator is built around IC1. This signal is divided in IC2.A which in fact constitutes our bipolar 100-KHz test signal generator. Series resistors R6 and R3-P3 on the Q and $\overline{Q}$ outputs of IC2.A give the generator a high output resistance compared to the low ESR, and, essentially, make the generator act as a 100-kHz, balanced, constant-current generator. The voltage drop across the C.U.T. is taken to IC3, four bilateral switches coupled as a controlled polarity changer, changing polarity in sympathy with the outputs of IC2.A. This enables IC3 to act as a (rudimentary) ADC. IC4.A, a differential amplifier, converts the differential signal into a single-ended signal, i.e., one which is referenced to ground. IC4.B amplifies the signal such that it can be applied to a 200-mV voltmeter. IC9 is the voltmeter IC, here the ICL7106 is used with an LCD, all in a standard configuration. The LM358 in position IC8 is a comparator that tells you when it's time to change the battery. IC7, finally, generates the negative supply rail for the circuit.

As shown in the circuit diagram, the test probes carry two screened wires each. Each probe carries a signal wire (e.g., 'A') and a measuring wire (e.g., 'B'). More about the probes under 'Construction'.

Construction

A compact printed circuit board was designed for the Capacitor ESR Tester by the Elektor labs. The resulting double-sided through-plated board design is shown in **Figure 5**. As appropriate for a test instrument, the board is designed such that all adjustment points are easily accessible, in this case, from the sides of the board (multiturn presets P1, P2, P4, P5) and from the top (preset P3).

Although the construction of the board follows standard practice (of which the main maxim is: *work carefully*), a few things should not be left unmentioned. Firstly, the circuit board has a screening ground plane at the component side, so care should be taken to avoid short-circuits by solder blobs or solder hairs between component terminals and the ground plane. Secondly, ascertain, check and double-check the polarity of any polarized component, in particular the tantalum capacitors in positions C3, C4 and C5. Tantalum capacitors when reverse polarised have a nasty habit of exploding and emitting hideous fumes. Finally, we recommend using sockets for all ICs (except IC9) and the LCD. The latter is easily made by cutting a 40-pin IC socket in two (lengthwise) and using the two 20-way socket strips.

Small holes should be drilled in the two long sides of the ABS case to allow P1, P2, P4 and P5 to be adjusted from the outside.

Regarding the probes, their basic construction is illustrated in **Figure 6**. These two wires are soldered together as close to the probe tip as possible. In this way the voltage

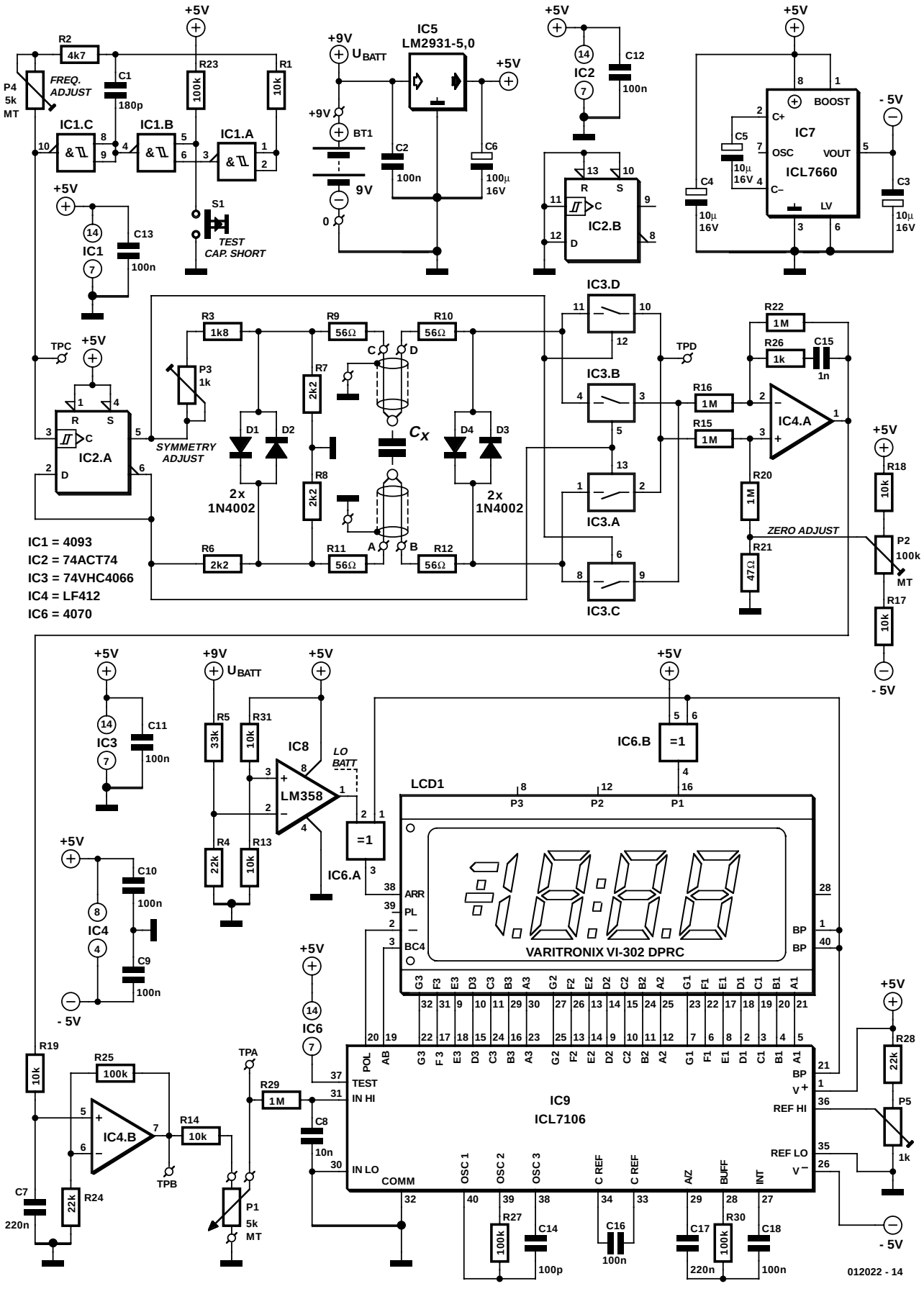
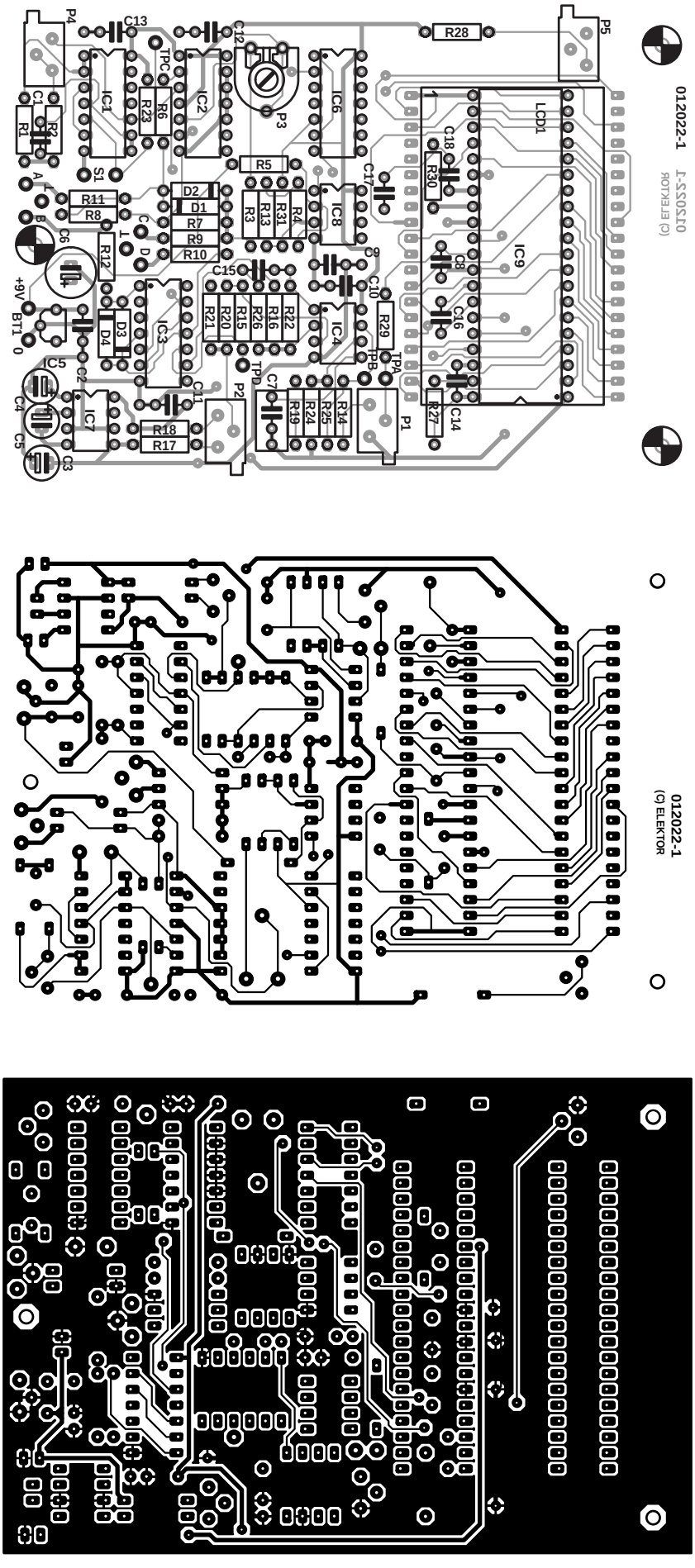


Figure 4. Circuit diagram of the Capacitor ESR Tester. C_x is the capacitor under test.



drop along the signal wire will not add to the measurement. The screening ensures that the test leads do not pick up noise, and that you maintain a stable zero adjustment.

The ESR Tester as an add-on

The most costly parts in the circuit are the display and the 7106 A-D converter. Money can be saved if you decide to use the ESR Tester as an add-on for an existing digital multimeter (DMM). Switch the multimeter's range selector to the 200.0 mV/DC position and connect the inputs to GND and the wiper of P1. You should not be tempted to supply the ESR Tester from the multimeter's battery. Remember, the ESR Tester has its output referenced to ground, so if you run it off the multimeter's battery the Tester will have its battery minus connected to the input common terminal, which is far from advisable. Use a separate battery for the ESR Tester to avoid any problems. Or if you really want to use just one battery, give the ESR Tester an add-on 9-volt battery, connecting the ESR Tester's regulated plus 5 volt to the plus terminal of the multimeter's battery connector and the ESR Tester's minus 5 volts to the multimeter's minus terminal.

A few words of warning

Though the ESR Tester has diode-protected inputs, it is still a good idea to discharge any largish capacitors you want to test. Some reservoir capacitors in power circuits contain so much energy that the protection circuit may burn out. If this should happen, the defective components are usually to be found in the protection circuit alone. The remedy should therefore be pretty straightforward and inexpensive.

Figure 5. Copper track layout and component mounting plan of the PCB designed for the instrument. Double-sided, through-plated board, available ready-made (see Readers Services).

COMPONENTS LIST

Resistors:
R1,R13,R14,R17,R18,R19, R31 = 10kΩ
R2 = 4kΩ7
R3 = 1kΩ8
R4,R24,R28 = 22kΩ
R5 = 33kΩ
R6,R7,R8 = 2kΩ2
R9-R12 = 56Ω
R15,R16,R20,R22,R29 = 1MΩ
R21 = 47Ω
R23,R25,R27,R30 = 100kΩ
R26 = 1kΩ
P1,P4 = 5kΩ multiturn preset, vertical mounting, side adjust (Bourns 3266X, Farnell #347-747)
P2 = 100kΩ multiturn preset, vertical mounting, side adjust (Bourns 3266X, Farnell #347-784)
P5 = 1kΩ multiturn preset, vertical mounting, side adjust (Bourns 3266X, Farnell #347-723)
P3 = 1kΩ preset, horizontal mounting

Capacitors:
C1 = 180pF
C2,C9-C13,C16,C18 = 100nF
C3,C4,C5 = 10µF 10V radial
C6 = 100µF 16V radial
C7 = 220nF
C8 = 10nF
C14=100pF
C15 = 1nF
C17 = 220nF

Semiconductors:
D1-D4 = 1N4002
IC1 = 4093
IC2 = 74ACT74 PC
IC3 = 74VHC4066
IC4 = LF412-CN
IC5 = LM2931-5,0
IC6 = 4070
IC7 = ICL7660
IC8 = LM358-N
IC9 = ICL7106-CP

Miscellaneous:
LCD1 = 3.5 Digit LCD with LO-BATT indicator, e.g., Varitronix VI-302 DPRC (Farnell #478-660)
S1 = pushbutton, 1 make contact
Battery holder
On/off switch
2 miniature probes, e.g., Hirschmann PRUF1 (Farnell #523-483)
Length of 2-core screened cable
ABS enclosure with LCD window and battery compartment, e.g. Multicomp type BC4, (Farnell # 645-758)
40-pin IC socket cut in half (see text)
PCB, order code 012022-1, see Readers service page or www.elektor-electronics.co.uk

ESR Tester adjustment

Before adjusting the instrument, be sure that you have a regulated plus 5 V from IC5 and minus 5 V from IC7. If you don't, you'll have to troubleshoot your circuit board.

1. Start with the voltmeter circuit. P1 should be disconnected at this point. Connect a known, accurate voltage source of less than 200 mV to point TPA (test point A) and adjust P5 until the LCD shows the right value. Remove the voltage source. Connect TPA to TPB, short the test leads together, and adjust P2 for a '000.0' reading. Remove the connection. Reconnect P1.

2. Connect a frequency counter or an oscilloscope between TPC and GND. Adjust P4 for 200 kHz counter reading or 5 µS period time on the oscilloscope.

Connect the test leads to a 10-Ohm resistor. Connect an oscilloscope (in AC mode) between point TPD and GND. Turn P3 (symmetry adjust) so that the two half cycles line up and produce a straight line. Adjust P1 for a '10.0' DVM reading.

If you do not have a counter or a 'scope available, turn P3 and P1 to the centre of their travel.

To ensure that the ESR Tester works properly you can connect different (known, good) capacitors in series with different resistors and have these simulate capacitor ESR.

Component considerations

The LF412 in position IC4 is a good choice for the differential amplifier. Since we are dealing with high frequency signals in the millivolts range, low drift, low offset and high bandwidth are crucial. Many different op-amps have been tested but most resulted in DC drift problems. The LF412 emerged as a good, low cost choice causing minimal drift.

IC5, then, is a 5-volt regulator that works just fine at a voltage drop less than 600 mV and so ensures long battery life. This regulator enables the circuit to keep working down to a battery voltage of less than 6 V. IC2, a 74ACT74, is capable of delivering enough current at 100 kHz to produce a nice clean square wave. IC3 is a high speed (VHC) version of the well known 4066. Compared to the common 4066, the effect of unwanted reactance is halved. For best performance the specified components should be used, but all in all, quite acceptable performance is achieved still if you use an ordinary 4066 for IC3, and a 74HCT74 for IC2.

(012022-1)

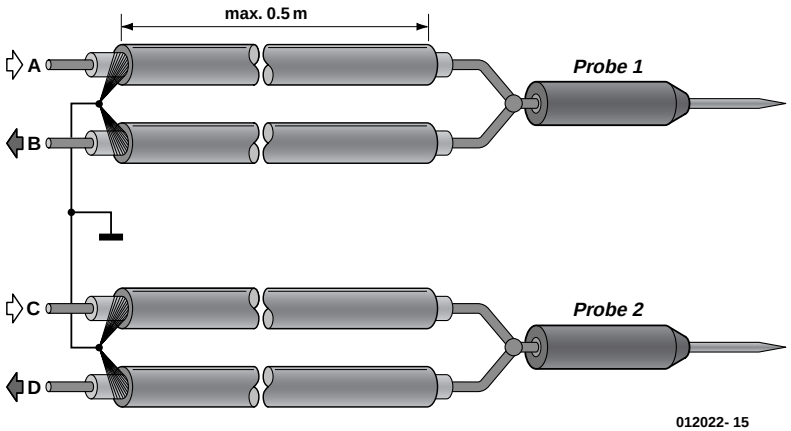


Figure 6. Here's how to make the 4-wire test leads between the probes and the instrument proper.

Related websites:

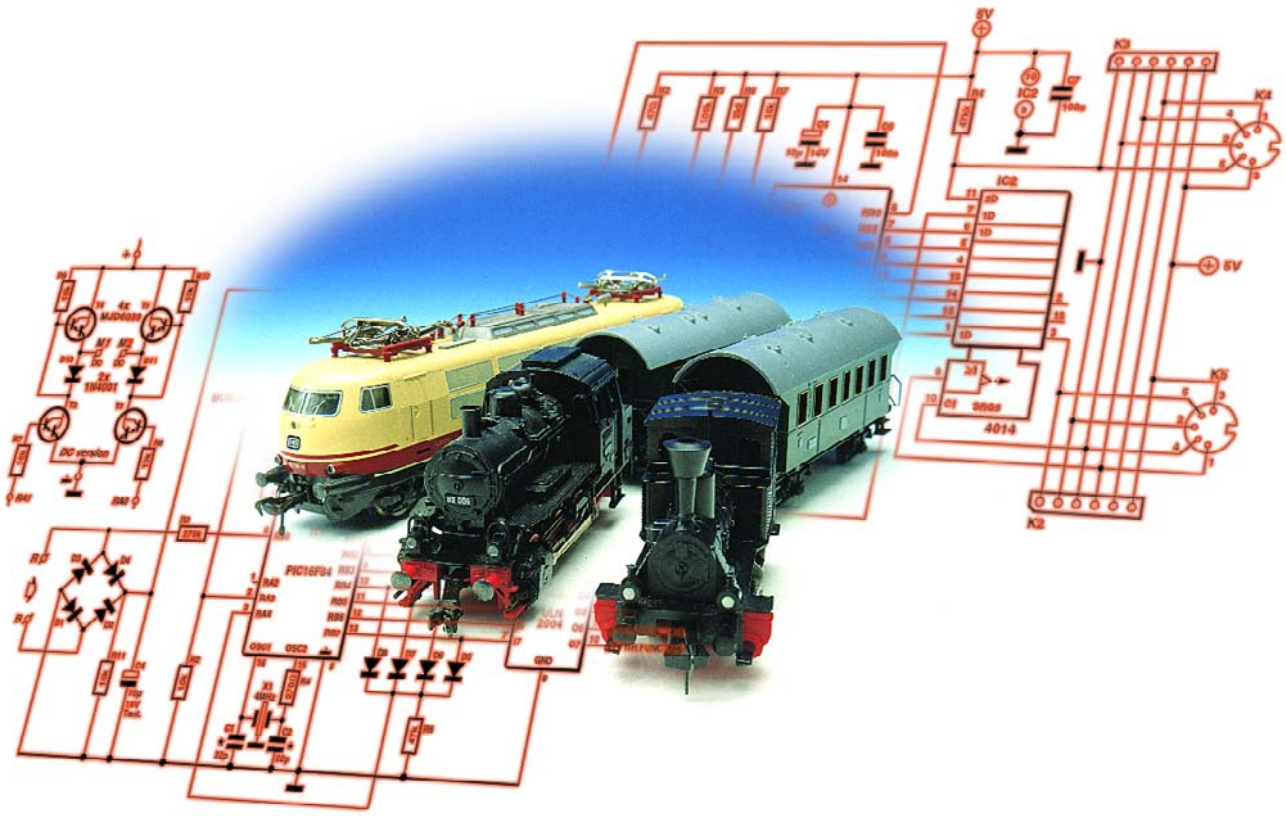
www.awiz.com/cwinfo.htm
www.flippers.com/esrkttxt.html

EEDTS Pro
Super Loco Decoder Upgrade

with a return signaller upgrade

Design by H. Prince

When the announcement was made for a new version of the PC software and controller for the popular EEDTS Pro model train control system it was inevitable that an upgrade for the super loco decoder would soon follow. The original super loco decoder was published in the October 1999 issue of Elektor Electronics. The version described here replaces the old one and offers several enhancements, such as a programmable speed table.



The new version of the super loco decoder (ESLD) doesn't just have extra functions to support the new PC software. Improvements have also been made to some of the original functions.

It now takes into account which type of motor is used in the loco. There are trains that use a.c. or d.c. motors. The properties of d.c. motors are different to those of a.c. motors, which causes d.c. locos to run too fast at lower speed settings. Because the use of d.c. motors is increasing, the speed table can now be adapted to suit the properties of the motor.

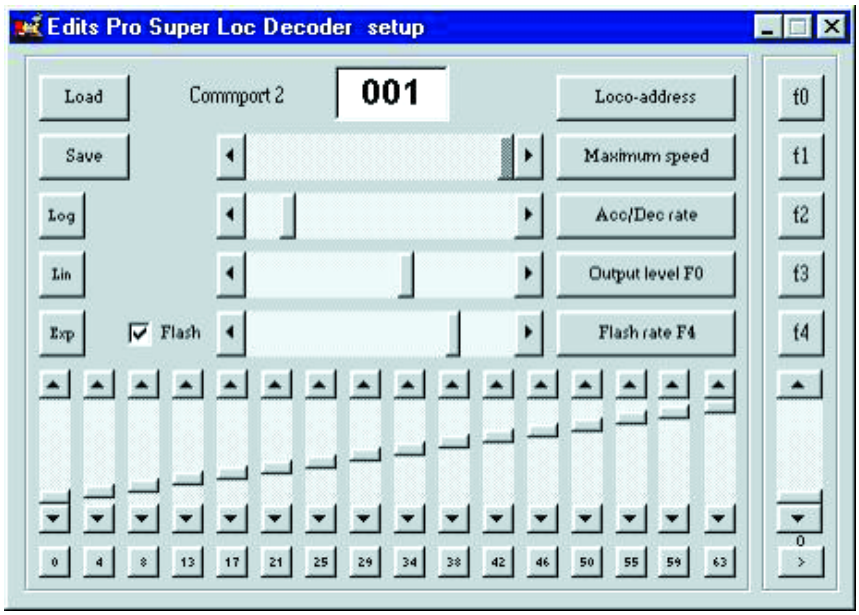


Figure 1. Screen dump of the program for setting up the new super loco decoder.

The speed table

The speed of the motor can be adjusted in 16 steps. Each position of the speed controller corresponds to a location in a table that is stored in the EEPROM of the decoder. This table contains values that determine how fast the loco runs. In principle, location 0 (de lowest position of the controller) contains the value 0 and location 15 contains the largest value of 0FFh.

The decoder was originally designed for use in Märklin locos with a.c. motors and the table has been defined such that a relatively high voltage is fed to the motor at lower settings of the controller.

Locos using a d.c. motor normally have a permanent magnet and higher gearing. This causes them to start sooner and makes them run too fast at low speed settings. This can be partially offset by reducing the maximum speed, but that is not an ideal solution. For this reason we have changed the firmware such that the contents of this table can be modified.

One of the original design criteria for programming the settings into the decoder was that it should be possible to do so using only simple tools. A constraint of this is that only 'trits' from the Motorola format (which the protocol is based on) can be used. Since two trits are used for the commands, the maximum number of possible commands is $3^2 = 9$. Of these, seven are used for common settings (**Table 1**). The other two are therefore available for the programming of the speed table. Since use is made of the standard Motorola format this can also be done with the hand-programmer described in the original article.

We realise that programming a complete table would be a time consuming job. That is why we've introduced a utility program that can be found on the EEDTS Pro website (www.gironet.nl/home/editspro) and on the Elektor website (www.elektor-electronics.co.uk, order code **020094-11**).

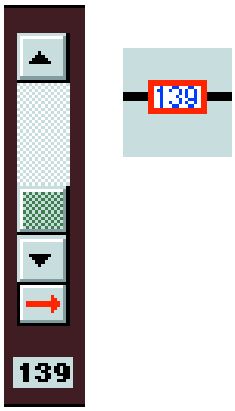


Figure 2. Speed control for a loco and the return button with an address of 139.

Programming

In order to give enthusiasts the chance to program the decoder with their own programmers, we'll explain how the table (as well as the other parameters) can be programmed. It should be well known that the information put on the rails consists of four trits for the loco address, one trit for the front lights and four trits for the speed data. Further details regarding the Motorola protocol can be found in various articles in *Elektor Electronics* in 1999.

For setting up an ESLD a fixed address (79) has been reserved, which obviously requires four trits. The other five trits are used to convey setup information, such as loco address, maximum speed, rates of acceleration and deceleration, etc. Each command has to be correctly received by the decoder a minimum of ten times before it is accepted. In practice you will find that when a command has been put on the rails for two seconds it will certainly have been accepted.

For completeness we have shown all possible settings in **Table 1**.

The procedure for programming the table is simple: first the address of the location that we wish to modify is programmed, and then the actual value is programmed in that location. This way the locations can be programmed one by one.

The software

To make life easier, a software application was written to help set up an ESLD. This can only be used in combination with the EEDTS Pro controller. Users of other types of controller will have to write their own program.

When the program is started, a window appears that contains all settings. Several of the buttons are familiar from the setup screen in EEDTS Pro: loco-address, maximum speed, acc/dec rate, brightness and flash rate. To the left of these are the accompanying slider controls.

Underneath these are the controls for setting up the table values. Each slider control corresponds to a location in the table and a setting of the speed controller. The setting for each slider control is shown in a button underneath it. When this button is clicked (with the left mouse button) the value of the corresponding control will be programmed into the memory of the decoder. This has to be done once for each of the locations.

Table 1. Programming the settings.

987654321	Trit
LLLL0	program loco address (LLLL = 1 - 255)
MMM01	program maximum speed (MMM = 0 - 15)
AAA0X	program rate of acceleration & deceleration (AAA = 0 - 15)
LLLL0	program loco address
SSS11	program brightness of lights for function 0 (SSS = 0 - 15)
KKK1X	program flashing rate for function 4 (KKK = 0 - 26)
LLLL0	program loco address
CCCX1	**new: program table address (CCC = 0 - 15)**
EEEXX	**new: program speed (EEE = 0 - 255)**

Where: trit 0 = 00 binary, trit X = 01 binary, trit 1=11 binary; trit 1 - 4 = loco address 79

Since the amount of information per loco has become rather large, and it's not possible to read information back from the decoder, the 'Save' and 'Load' load buttons have been added to store the data to the hard disc and subsequently retrieve it. The settings are stored in a file with a name the same as the number in the top input box and an extension of '.stp'. Underneath the 'Save' button are three buttons that have been pre-set with standard table values: logarithmic, linear and exponential. These can be used as a starting point for creating your own tables.

The program uses the same setup file (com.inf) for the COM port as EEDTS Pro and will therefore start up using the same port that EEDTS Pro is set up for, as long as the programs are in the same directory.

Another COM port can be selected by double clicking on the text. The software is available only in English.

Remember that when programming the decoder, all locos on the rails at that time will take the new settings. The programming is therefore not determined by the loco address on the screen.

On the far right are several buttons that are used to test the settings just made. The loco with the same address as that in the input box for the loco-address will respond to these. In this way the speed, the direction and the five functions can be operated.

Other enhancements.

The new version of the EEDTS Pro control software greatly extends the loco address range to 255. A consequence of this is that both the loco decoder as well as the address return signaller have to be made compatible with this. Although the ESLD was originally designed for an address range from 1 to 81, the old version could handle some addresses above 81. Since we were already making changes to it, we decided to make the new decoder recognise the full range of addresses.

Several users have managed to program a loco with an address of 79. The result of this is that the loco will run, but can never be set to another address. The new firmware blocks this address.

Return signaller

Obviously the loco address return signaller (Elektor Electronics January 2000) also has to recognise the higher addresses. In the original design the decoded data was placed on the bus as soon as the IR information was received. Only when the contact rail was closed would bit 7 be set high and the data would be read by EEDTS Pro. That means that 129 (81h) corresponds to loco address 1, 130 (82h) to address 2 etc. In the new version this is dealt with differently. The return signaller will only place the data onto the bus when the contact rail is closed. To keep the return signaller compatible with the previous version, the lower range remains unchanged, but is extended to 127 (0FFh) and the rest of the range (loco addresses 128-255, 00h-07Fh) has been added to that. A consequence of this is that loco-address 128 (80h) is seen by EEDTS Pro as address 00h, which is then recognised as a loco without infrared return signalling.

This is shown more clearly in **Table 2**, along with other restrictions.

We would like to point out that both versions of the ESLD and return signaller can be used simultaneously, even with the old version of EEDTS Pro. It is only with the new version of EEDTS Pro that the extended addressing range becomes available. Since only the firmware has changed, the PCBs for the decoder and the return signaller have remained the same.

(020094-1)

Tabel 2.

Addresses in the return signaller.

Loco addr.	Return button	Bit 7
Without IR	blank (00)	
0 (*1	not permitted	
1	1	1
⋮	⋮	⋮
78	78	1
79 (*1	not permitted	-
80	80	1
81	81	1
82	82	1
83	83	1
⋮	⋮	⋮
127	127	1
128 (*2	not permitted	0
129	129	0
⋮	⋮	⋮
254	254	0
255	255	0

(*1 This address is blocked in the ESLD
(*2 The loco will accept this address, but the return button will be inactive

Order codes for PICs:

Programmed PIC for the super loco decoder: 020094-41

Bulk packs are also available for 5, 10 and 20 pieces including PCB (refer to the Readers' Services pages)

Programmed PIC for the return signaller: 020095-41

The JTAG Interface

a standard test interface for ICs

By Paul Goossens

The testing of large logic ICs such as FPGAs, CPLDs, ASICs, etc. is very difficult and time consuming when conventional test probes are used. The measurement of internal signals that are not brought out to an IC pin is impossible in this way. A number of companies have therefore joined forces to find a common solution to these problems. The result of this is the JTAG interface (IEEE 1149.1).

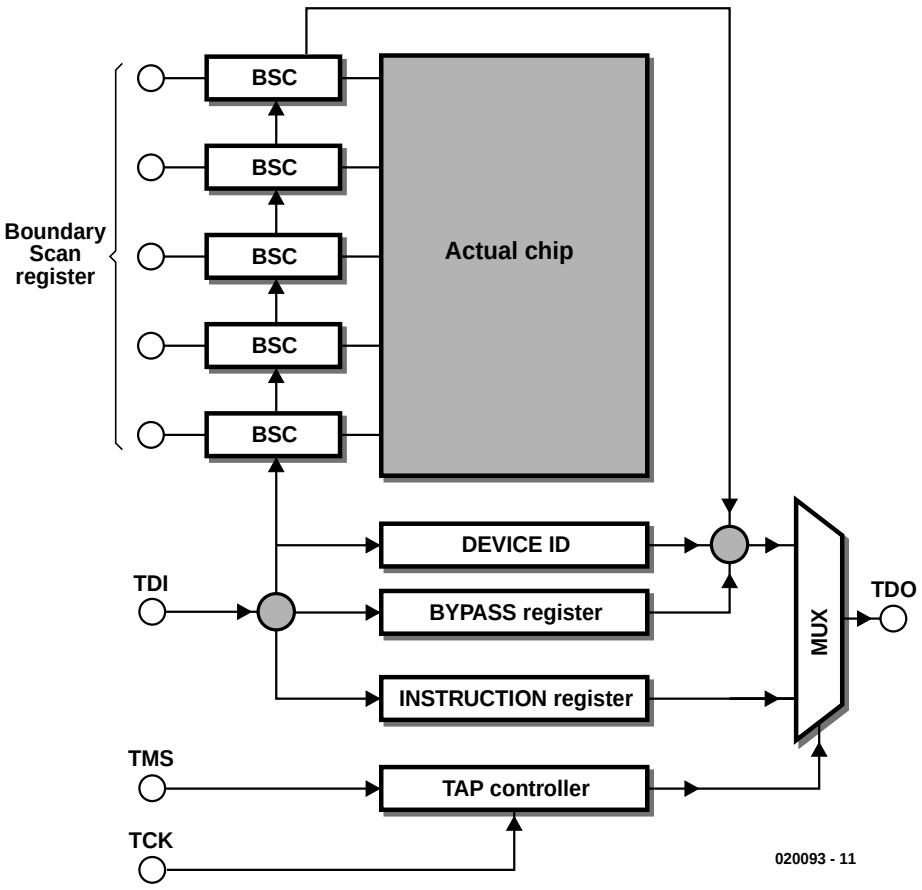


Figure 1. Simplified block diagram of the JTAG interface.

In the industry (and also increasingly in the electronics hobby world) more and more use is made of larger, complex ICs. The advantages of these ICs are clear: the PCB can be smaller, the current consumption is less, parts of the design can be easily re-used, etc. Unfortunately this development also has a few disadvantages. The testing of these ICs hasn't become any easier. In the case of SMD ICs the test probe has to be applied precisely in order to measure the signal at a specific pin and for BGA (ball grid array) ICs on a multi-layer PCB this becomes all but impossible. Apart from the fact that it is difficult and sometimes impossible to measure certain signals, it is also a very time consuming job to measure these signals with test probes. A number of IC manufacturers thought these problems warranted the setting up of the 'JTAG committee' (Joint Test Action Group). The tangible result of this cooperation is the special interface that we'll describe in this article.

The JTAG interface

The JTAG interface designed by the members of the committee has a

number of remarkable features. The most important of these are listed in **Table 1**.

The most remarkable feature of the interface is that, from the chip's point of view, it can function completely transparently and asynchronously if required. This means that the chip can be tested **in circuit**, without having any adverse effects on the functioning of the chip. Service engineers will certainly appreciate this feature. Apart from the standard features and functions, the chip manufacturer can extend the functionality of the JTAG interface. Examples of this would be the ISP programming of chips, the debugging of DSPs and microprocessors, etc.

Table 1. Features of the JTAG interface.

– Can work transparently
– Fully asynchronous
– Requires only four signals
– Can be cascaded
– Test patterns can be generated
– Internal signals can be tested/generated
– The manufacturer can extend the functionality

Block diagram

The structure of the JTAG interface is shown in **Figure 1**. At first sight it may appear very complex, but on closer inspection it will become easier to understand.

The four signals that control the JTAG circuit can be clearly seen. TCK is the clock signal for the JTAG logic. This signal is independent from the clock signal within the IC, so they operate asynchronously. All other signals of the JTAG interface operate synchronously to this clock signal. The TMS input drives the JTAG controller. This is used, amongst other things, to choose between the Instruction register and other registers.

Depending on the state of the JTAG controller and the contents of the Instruction register, a register will always be connected as a shift-register between TDI and TDO. Data can be stored into the register using TDI, while at the same time TDO can be used to read the contents of the register.

Boundary-scan register

Every JTAG compatible chip will have at least the following registers: Boundary-scan register, Instruction register, DEVICE ID register and BYPASS register. The controller is designed such that a register has to be activated before the contents of the register can be read.

The Boundary-scan register is the largest and most important register in the JTAG interface. This register can represent inputs and outputs of the IC, or it can have user-defined data loaded into it that is subsequently presented to the inputs of the chip, and also to provide the output pins with a user-defined signal.

The Boundary-scan register consists of a collection of separate memory cells and some logic to control these cells and the output. The block diagram in **Figure 2** represents a BSC (Boundary-Scan Cell) that controls an output pin. This is almost identical to the version for an input pin. The only difference between the two types of BSC is the way they are connected to the chip circuitry. When the input of the BSC is connected to an input pin, and the output is connected to the input circuitry of the chip itself then this BSC is capable of controlling the input.

The 'capture/scan' control line is used by the controller to select the data to the input of FF1. If the current state of the outputs has to be captured, the controller connects the output of the chip to the input of FF1 via MX1. At the next clock pulse the flip-flop will remember the current state of the output of the chip. For the data to be read out serially, all BSCs need to have their FF1s connected

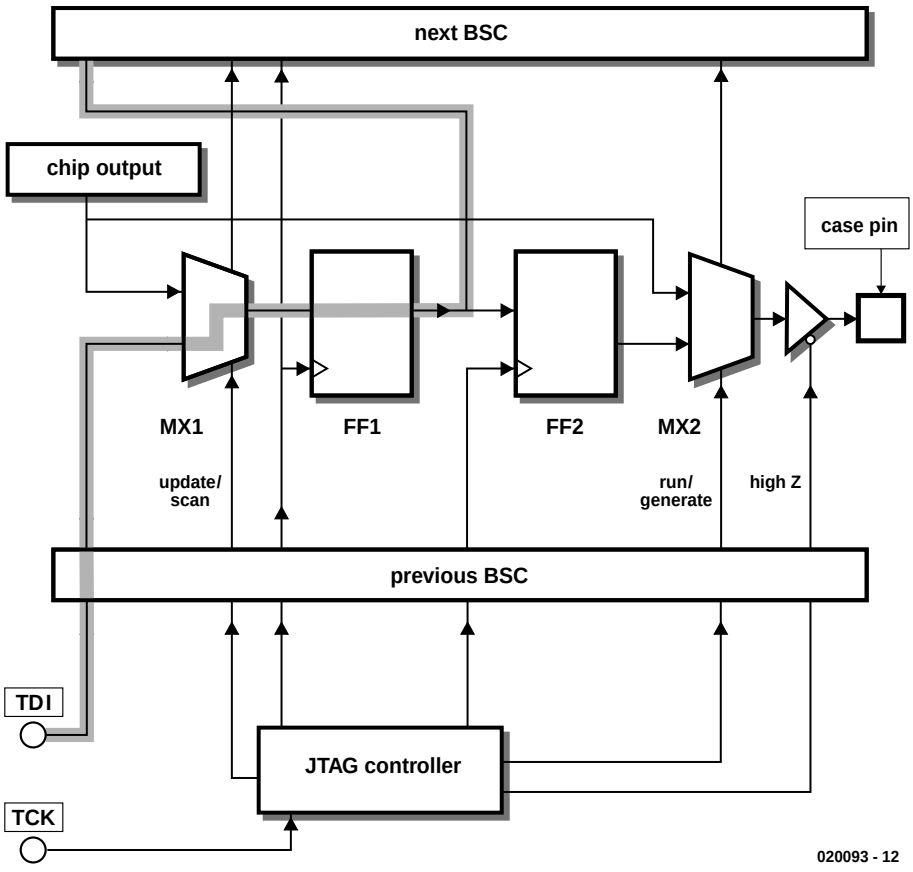


Figure 2. A Boundary-Scan Cell (BSC) at the output of an IC.

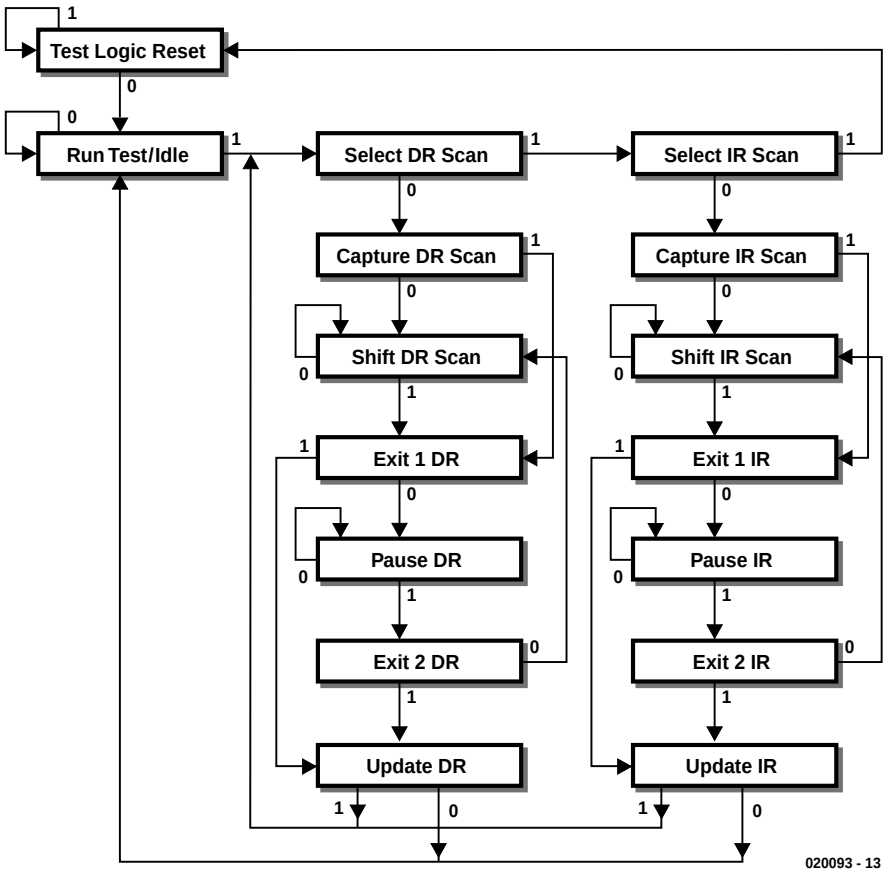


Figure 3. State diagram of the TAP controller.

in a chain, forming one long shift-register. This data path is shown shaded in **Figure 2**.

At the same time as reading the data, it is possible to load user-defined data into the shift-register. Once that has completed, the data can be clocked into FF2. MX2 is used to select either the output of the chip *or* the output of FF2 as the signal on the output pin of the chip. This allows the user to connect the output permanently to the output of the chip during testing, so that the chip appears to function normally to the outside world. Furthermore, this allows you to set the output of the BSC yourself. In this way we can generate signals for the outside world or (in the case of an input BSC) as inputs for the chip.

An interesting fact of the Boundary-scan register is that the connections aren't restricted to those signals that can be physically accessed on the chip. The manufacturer can freely connect internal signals to a BSC, so that these signals can also be measured even though they're not connected to an external output. This is especially useful with PLDs, where it becomes possible to measure internal product terms. In this way it is possible to check that certain functional blocks within the PLD are operating correctly.

Instruction register

The Instruction register is used to select the various functions of the JTAG interface. Apart from the standard functions, this register also controls those functions added by the manufacturer.

Unfortunately the JTAG specification doesn't state which binary codes have to be loaded into the Instruction register for it to select certain functions. The same function can therefore have different opcodes on other chips. These opcodes are usually found in the data sheet for the relevant chip. **Table 2** contains all the instructions that are defined in the JTAG specification. Only the first three instructions are required for a chip to be JTAG compatible, the rest of the instructions may be implemented at will. Apart from the instructions in **Table 2**, the manufacturer may add other instructions that are not described in the JTAG specification. This makes the JTAG interface even more flexible.

EXTEST

In the EXTEST mode all chip connections are connected to BSCs. In this mode the user can generate a test signal and read it back to check, for example, for correct connections between ICs, when several ICs are present in the JTAG chain. It is also possible to directly drive any peripheral ICs that may be present. This mode is generally used by PCB manufacturers to check for shorts or open circuits on populated boards, without requiring a so-called 'Bed-Of-Nails'.

SAMPLE/PRELOAD

This mode is used for reading the contents of the Boundary-scan register or to load a test pattern into it prior to the execution of the EXTEST function.

BYPASS

When several ICs are cascaded (JTAG chain), but only one of these requires the use of its JTAG functions at a particular time, then it is sensible to give the BYPASS instruction to all other chips in the chain. In this mode the chip functions normally, but the JTAG interface selects the BYPASS register, which is only one bit long and which has no further function. In this way the total length of the shift-register is greatly reduced. The Boundary-scan registers can contain a very large number of bits (in practice they'll often have several hundred bits). By selecting the BYPASS register the JTAG interface will have to transmit a much-reduced number of bits for each instruction, which will increase the speed of testing.

INTEST

During an INTEST the chip is connected directly to the BSCs. This makes it possible to test the functionality of the chip. The Boundary-scan register can be set up with a defined input state after which the outputs of the chip can be read. This option is especially useful when testing prototypes with programmable logic, but also during repair work.

RUNBIST

This function causes a pre-programmed test to run within the chip. The result of the test (chip pass/fail)

Tabel 2. Instructions for the JTAG interface

INSTRUCTION	Chip mode	Selected Register	Application
EXTEST *Required*	TEST	Boundary-scan register	Generate and measure external signals.
SAMPLE/PRELOAD *Required*	NORMAL	Boundary-scan register	Read or write the Boundary-scan register.
BYPASS *Required*	NORMAL	Bypass register	Reduces the scan path.
INTEST *Optional*	TEST	Boundary-scan register	Internal test of the chip. The Boundary-scan register is connected directly to all inputs and outputs of the chip.
RUNBIST Optional	TEST	-	Executes the Built-In Self Test of the chip.
IDCODE Optional	NORMAL	Device ID	Read the Device ID of the chip.
USERCODE Optioneel	NORMAL	UserCode register	Programming of PLDs, extended Device ID.
CLAMP Optional	TEST	Bypass register	Sets output pins to defined levels and reduces the scan path.
HIGH Z Optional	TEST	Bypass register	All outputs of the IC are put in the High-Z state. Electronically, the IC is now isolated from the PCB.

is put on the TDO output. The RUNBIST function is mostly used for memory modules. If these memories were tested using the standard JTAG interface, they would first have every memory location set to a value, which is then read back and checked. Subsequently the inverse value would have to be stored and read again. This would have to be done for each memory location and would take a very long time (keep in mind that all communications are serial). The BIST can complete this significantly faster.

DEVICE ID
In this mode the DEVICE ID of each chip can be read via the JTAG interface. This information can be used to verify that the correct chips are mounted on the board and also to read the version numbers of the chips. This function is often used by programming software, letting it verify that the user has selected the correct device.

USERCODE
This mode selects an extra register that is optionally implemented by the manufacturer. This register usually contains extra DEVICE ID information. Some manufacturers use it in the programming of PLDs.

CLAMP
CLAMP is very similar to the BYPASS function, the difference being that the IC pins are now driven via the BSCs. This means that whilst the BYPASS register has been selected, the outputs of the chip will be held at fixed, defined levels.

HIGH-Z
This function is used to electrically isolate the chip from the outside world. All outputs are set to the High-Z state, so that this chip won't affect measurements made on other components.

Other functions
The functions described so far have all been defined in the JTAG specification. As we've said before, the manufactures are permitted to add their own functions, extending the capabilities of JTAG interface. These enhancements make it possible to set hardware breakpoints in certain DSPs, read or modify its registers or memory, execute instructions, etc.

The TAP controller

The main control circuit in the JTAG interface is the TAP (Test Access Port) controller. This controls the entire JTAG interface in conjunction with the instruction register and its instruction decoder. The TAP controller is a so-called state machine with only one input and that is the TMS input. The TMS signal determines the next state of the controller. The controller changes state on the rising edge of the TCK signal.

In order to drive the TAP controller you first need to know which of the 16 states the controller is in. Unfortunately it isn't possible to read this state directly, but a little trick can be used to put the TAP controller into its reset state, regardless of its current state. The state diagram has been designed in such a way that when five clock pulses are applied while TMS is set to '1', the controller will always be in the 'Test Logic Reset' state. This can easily be verified with the help of **Figure 3**: Choose any state on the diagram. From this state follow the arrow with a '1' to the next state. Then repeat this procedure another four times. Whatever the initial state, the final state will always be 'Test Logic Reset'. Some chips with JTAG controllers have an extra input, TRST. As soon as this is set 'low', the TAP controller will be set to the 'Test Logic Reset' state.

As we have already seen, the contents of the Instruction register determine the function that the JTAG interface has to execute. To this end the Instruction register has to be loaded with the opcode for the function that we want the JTAG interface to perform. To load the Instruction register, the TAP controller has to be set to the 'Shift-IR' state. The path to this state is always via the 'Capture-IR' state. In this state the Instruction register is loaded with a value that is burnt into the chip itself. In principle the actual value is not important.

Once the TAP controller is in its 'Shift-IR' state, the TDO output becomes active. It will now contain the LSB of the Instruction register. The data at the TDI input is shifted into the Instruction register at each rising edge, while the contents of the Instruction register are shifted on the falling edge to the outside world via TDO.

When the Instruction register has been loaded, the JTAG interface has to be told to act on the new contents. This happens in the 'Update-IR' state. In this state the contents of the Instruction register are recognised as the current instruction. This instruction also determines which of the other registers is selected as the data register, as shown in Table 2.

The reading and writing of a data register are similar to that of the Instruction register, except that we now use the 'Capture-DR' state to load the data register. 'Shift-DR' is used for reading and writing the data and 'Update-DR' is used to activate the data just written.

BSDL files

The JTAG specifications also include the so-called BSDL files. These are files that contain all relevant information for chips with a JTAG interface. They contain, for example, the opcodes for the JTAG instructions defined for a certain chip. They also contain the names of the chip pins used and the order in which they are stored in the Boundary-scan register, and of course the DEVICE ID of the chip concerned.

The BSDL files have the same structure as VHDL files. This makes it easy to use the information in BSDL files in VHDL code. Most of the programs designed to work with the JTAG interface are capable of reading BSDL files and interpreting their contents.

It would be too much to describe the structure of VHDL and BSDL files in this article, but by all means take a look at one of those files. Fortunately they contain only ASCII characters, so the information can be viewed in any standard word processor. You will find that much of the information in these files can be easily interpreted. The instructions, DEVICE-ID, names of the pins, etc. are clearly shown in separate sections.

(020093-1)

Note: a practical implementation of the principles discussed in this article may be found elsewhere in this issue in the 'Parallel JTAG interface' article.

Microcontroller Basics Course FAQs

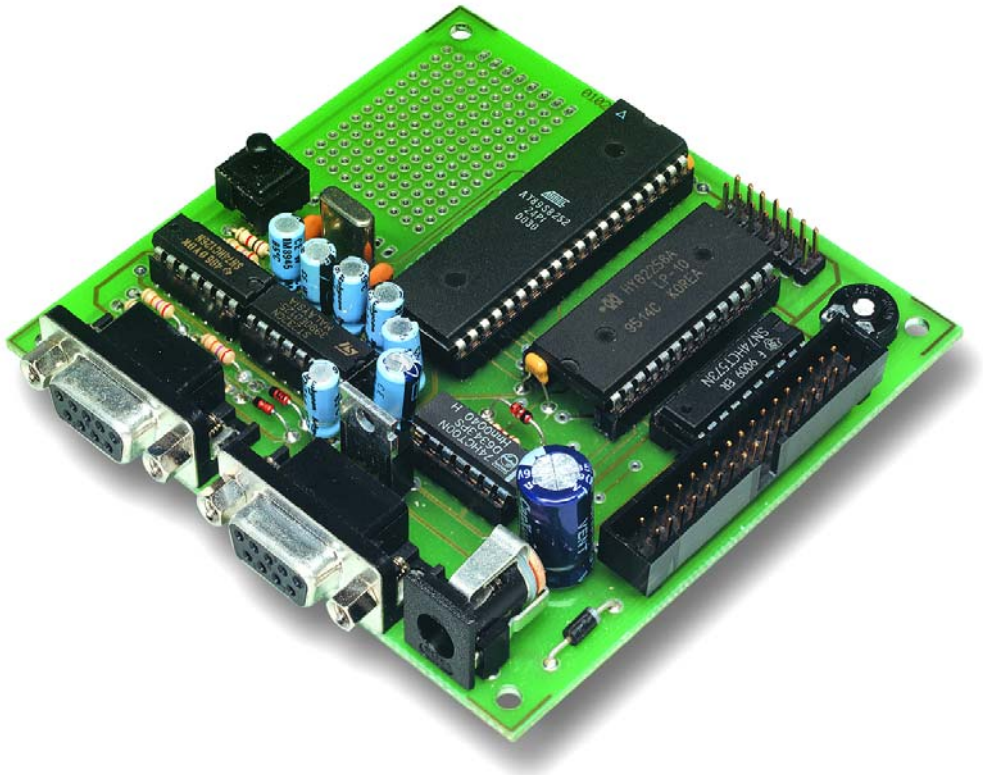
the most commonly asked questions

By K. Walraven and B. Kainka

We have received many questions from readers regarding our microcontroller course. The most interesting questions are summarised in this list of FAQs (frequently asked questions).

? *Where can I order the printed circuit board for the 89S8252 Flash Board? Can I also order it complete with the components?*

Only the printed circuit board can be ordered from *Elektor Electronics*. All other components must be purchased from regular electronics traders. You can order the circuit board via Internet or by telephoning us at *Elektor Electronics*. Some traders like C-I Electronics (www.dil.nl) can provide a complete construction kit. Elektor also sell a floppy disk (order number **010208-19**) containing all the sample software and necessary utility programs, such as TASM and Microflash. You can also download this software free of charge from our website at www.elektor-electronics.co.uk. Select Free Downloads, then June 2002, then file 010208-19. For some parts of the course, you will also have to download additional programs from the Internet.



? *With some of the course files, I can't find Tasm.exe, and also not the editor.*

The editor is called **TASMedit.exe**, and the exact name of the assembler is **Tasm.exe.** Both programs are stored in Tasm.zip. These programs

must thus be present on the diskette for the course and in the zip file that you can download from the *Elektor Electronics* site. You can see which files are supposed to be on the diskette or in the zip file by looking at the 'context.txt' file.

The assembler manual be separately unpacked using the command 'booz x tasmdoc.zoo'.

? *Where can I find the source code for Microflash, so I can see how this program works?*

All the source code that has been released is on the course diskette. If it is not there, it is not available.

? *There is a RAM IC on the AT89S8252 Flash Board. I wonder if this is really necessary. Can the board also be used without the RAM?*

If the software does not use external RAM, it can be omitted without any problems.

? *What do I have to do to use the microcontroller without the external RAM? If I omit the RAM IC, it looks like the microcontroller is still searching for it (activity on Port 0) and the program hangs. The same program code runs with no problems as soon as the RAM IC is back in its socket.*

Actually, everything should work OK without the external RAM. That is in any case true for the assembler. However, BASIC52 does need external RAM and first checks extensively to see if it is present. Rigel C also does not work without external RAM, but Bascom can easily manage without it.

? *Can I use a MAX202 instead of the MAX232?*

We think so, since it is pin-compatible with the MAX232. However, we have not tried this ourselves.

? *Is the 74HC126 actually necessary?*

Only for programming.

? *I've closely examined the schematic diagram of the Flash Board, but I can't find any indication of what sort of power supply the board needs. It's clear to me that it must be a mains adaptor that can deliver 12 V dc, but how much current must it supply? Maybe that's obvious to an experienced electronics technician, but a hobbyist cannot figure this out so easily using the schematic diagram. I think this actually should be stated in the components list.*

There we have to agree with you. The board needs around 50 mA, and 35 mA during programming. If LEDs or other devices are connected to the board, they will also draw some current. The smallest mains adapters you can buy nowadays deliver around 300 mA, so they will be perfectly satisfactory.

? *How should the board be connected to the computer?*

A female DB9 connector must be fitted to the board. The connecting cable should have a male DB9 connector on one end and a female DB9 connector on the other end. The pins of the two connectors should be wired 1:1, which means pin 1 to pin 1, pin 2 to pin 2 and so on. Pay careful attention to the pin numbers on the connectors, since the male and female connectors have mirror-image arrangements.

For programming you use connector K2, and for serial communications you use connector K1.

? *Despite many attempts, I can't manage to program the board!*

The cause is most likely that the protective Zener diodes on the programming inputs are 1.3-watt types. They have an excessively low Zener voltage at the rather low programming currents that are used, so the digital signals are not recognised. Replacing the diodes with 0.5-watt types will almost always clear up the problem.

? *I have the following problem with the AT89S8252 board: on Port 1 there is a separate LED connected to each bit line, always between V_{cc} and the port pin. I connected the LEDs directly without any series resistors, but they still give an excellent indication of the output levels.*

? *If I now try to reprogram the AT89S8252, it only responds briefly to 'Break' and then continues to execute the old program. The new program is only sent and subsequently executed if I open the V_{cc} line during programming. I would like to know what's wrong here.*

The answer here is the same as for the previous question. The LEDs prevent the programming signal levels from being greater than 2 V. This level is not high enough to be recognised if an HC type is used for IC2 (but an HCT type will work OK). This means that you shouldn't connect the LEDs without series resistors! For more information, see the question below about connecting LEDs.

? *Whenever I retrieve a program using the TASM assembler or write my own program and then click on the 'TASM' button, the assembler immediately translates the entire program, but this is always the first test program 'flash1.asm'. This happens even if I only change a numeric value (e.g. 0Fh to 10h). If I load flash2.asm and then press 'TASM', the assembler again translates the test program 'flash1.asm'.*

You are not the first person to experience this problem. It is essential to put TASM together with TASMedit and all other associated files in the same folder on the hard disk. If you do this, everything will be OK.

? *I have recently built the Flash Board with the Atmel microcontroller and have already generated a large number of programs in assembler. They all work fine.*

Now I am busy developing C programs for the same Flash Board. However, when I tried to compile the first simple sample C program from the April issue, the Reads51 C compiler reported two linker errors, namely 'external "InitSerialPort0" defined in a01.obj is not exported by any module' and then the same

again for the function putc defined in <Sio51.h>. What might be the cause of this problem?

These messages mean that the compiler cannot find the include files (such as sio51h). If you put copies of the desired files in the directory where the program to be compiled (a01c) is located, the compiler will be happy. Most of the include files are located in *reads51/include*.

? *Is there also software available to allow the Flash Board to be programmed under Linux?*

Absolutely. Albert van der Horst of the HCC Forth Users' Group wrote us the following on this subject:

'Here is a Flash program for Linux for the *Elektor Electronics* board in the December 2001 issue. It took a while, since we had to experiment a bit, particularly with resetting the board. It has also become somewhat more luxurious than the original program; among other things it has a test option that helps in tracking down cabling errors.'

'This program is GPL'ed, which means that it can be distributed with no restrictions as long as it is accompanied by the source code.'

You can find more information about this program on the Forth Users' Group website at www.forth.hccnet/nl/.

? *Which connections are suitable for driving something, and how should I do that?*

All of the lines on the board that are not used for something else can readily be used as inputs or outputs (which means P1.0 through P1.4 and P3.2 through P3.7). With the lines that are used for programming (P1.5 through P1.7), you can only connect something that will not disturb the programming process, which in practice means using a series resistance of at least 10 kΩ. If you don't use the serial interface, P3.0 and P3.1 are also free.

Similar considerations apply to using ports P0 and P2. If you do not use the RAM and LCD, you can omit the components and the ports are then freely available; otherwise they cannot be used.

If you use a connection as an input, you can connect anything you wish to it as long as the voltage does not go below 0 V or above 5 V.

If you use a connection as an output, it works as follows: when you output a logical '1', this is in practice equivalent to a resistance of around 50 kΩ to the positive supply line (100 µA). If you set the output to '0', it can sink a few mA to ground. To connect an LED, for instance, you don't connect it between the output and ground (since the LED could draw 100 µA with this arrangement), but instead between the output and +5 V. In order to limit the current to a few mA, a resistance of around 1.5 t it will be fully on at 2 mA.

If you short an output (or all of the outputs) to ground, it's not so bad since only 100 µA can flow. If you short an output to +5 V, around 5–25 mA will flow. This will not destroy the IC, but you should avoid this situation. If several outputs are shorted to +5 V at the same time, the IC can be destroyed by overheating. You should thus always use series resistors in this situation to limit the current to a maximum of 2 mA per output.

The values given above are rules of thumb. If you measure the actual currents, you will find that they can vary markedly from chip to chip.

? *I have downloaded the Reads51 C compiler, but even though my microcontroller works OK according to the tests the Flash Board cannot be programmed using Reads51, and I have to use microflash.exe instead. Reads51 also has tutorials in which it says that the board should be programmed using a 'MON/RUN' switch, but no such switch is present on our 89S8252 board.*

Am I doing something wrong, or is this simply not possible using Reads51?

Reads51 can only be used for programming with Rigel's own boards. You must in fact always use an indirect route, such as the Microflash program.

? *I am using the demo version of Bascom 51 in combination with the 89S8252 board. In this connection, I would like to know the answers to the following questions:*

– *Which type of programmer must I select for OPTION/COMPILER/MISC/PROGRAMMER to be able to work with the Flash Board?*

– *Is a special configuration necessary to allow the Flash Board to work with Bascom 51?*

Bascom 51 has not (yet) been adapted to our Flash Board. You have to take the output from Bascom 51 (the hex file) and use it as the input to TASM in order to program the Flash Board.

If you make a special connecting cable, it is probably actually possible to send data directly from Bascom to the Flash Board. For this you should use the parallel port of the PC and select the Sample Electronics programmer.

The connecting cable should be wired as follows:

DB25 pin	**µC pin (89S8252)**
2 (D0)6	(MOSI) (P1.5)
4 (D2)9	(RESET)
5 (D3)8	(SCK) (P1.7)
11 (BUSY)	7 (MISO) (P1.6)
18–25 (GND)	20 (GROUND)

P1.5–P.7 can be found on K4 (pins 6–8), GROUND is on K7 and RESET can be best tapped off from push-button S1.

For safety, it is a good idea to insert 220Ω series resistors in the data lines, but it will probably work fine without these resistors.

? *Why does the LCD not work in combination with Bascom 51?*

Bascomer way uses the data bus, just as with the Flash Board. However, Bascom assumes different connections to the LCD. If you swap lines A1 and A0 to the LCD, it will also work with Bascom.

? *Can I also drive the LCD using Bascom without swapping these lines?*

In order to support the standard LCD screen for the Flash Board without any soldering or swapping lines, the author of Bascom, Mark Alberts, has made a modification. It is included in the new demo version on the Bascom website (www.mcselec.com). The following simple program demonstrates this:

```
$regfile = "89s8252.dat"
Config Lcd = =16 * 2
$lcd = &H8000
$lcdrs = &H8002

Cls
Lcd "Test"
End
```

Can I also buy the 89S8252 board fully assembled?

No, as far as we are aware there are only retailers who offer construction kits.

However, there is a German company called AK-Modul-Bus that offers a similar board. It was also developed by the author of the course (B. Kainka) and it can be used very well with the course. The board is called 'ES52-Flash' and costs 99 Euros.

More information can be found on the Modul-Bus website (www.modul-bus.de).

However, there are some differences between this board and our board. The ES52-Flash board has only one serial port that is used for both downloading and communications. The associated download software is called 'Flash.exe' and is available on the Modul-Bus website.

Where can I find the original version of Intel's BASIC-52?

The original file, 'Basic52.hex', is one of the files present on the diskette for our course (diskette or download number **010208-19**).

I have the impression that there is a mistake in Listing 3 of Part 2 of the course (February issue, port characteristics and operations). The counter loop is initialised using r1 and then executed using r3.

A very good observation! So why does it still work? The reason is that r3 always leaves the loop with a value of zero and also starts again at zero the next time. As a consequence, the loop is executed 256 times instead of 255.

The correct version is thus:

```
;flash4.asm  port outputs
#include 8051.H
   .org 0000H

main     mov  a,#00
next     mov  P1,a     ;1
   mov   r1,#255       ;1
loop     djnz r1,loop  ;2 * 255
   inc   a     ;1
   sjmp  next ;2
   .end
```

I would like to have more information for the course instalments, for instance more programming examples and Basic source code programs.

You can find Basic source code programs, Reads51 programming examples and even more on the author's home page (home.t-online.de/home/b.kainka/basismi.htm). Most of these files are called mikro1.zip, mikro2.zip and so on.

My Flash Board sometimes makes strange computational errors in Basic. When computing log((1) I sometimes obtain a result of 0 and sometimes the board says 'bad argument' or 'divide by zero'. When computing exp(1) I sometimes obtain the correct value and sometimes a value of –2.7182. These errors occur when a program is executed in direct mode. I also have an old system with the original 8052AH BASIC and it works just fine! Is there a bug in the Basic version on your website or am I doing something wrong?

This could be a RAM error or a timing error. However, with a RAM error the consequences are usually worse. Besides this, we have to note that a whole lot of minor bugs were present in the original Intel Basic. However, that does not explain why you do obtain the proper results with your old system. Most of the errors will probably disappear if you use the upgraded version (BASIC5 2 V1.3).

The Microflash.exe programming software freezes whenever I try to reprogram the board while the flash2 program is running (the program that loads 0Fh and F0h alternately into P1). The 153-kHz signal on P1.6 can also be found on the CTS line of the PC, and evidently there is a timing problem that prevents the DTR line from putting the board into the Reset/Programming mode. As soon as the HEX or BIN button is clicked in Microflash, the program hangs. Everything works normally for loading the flash2 program. Could this have something to do with my PC? I am using Windows XP Professional. I can work around this problem using a trick, which is to briefly disconnect R1 and D1 and forcing a reset at that point using a pushbutton switch connected to V_{cc}, after which I select the program to be programmed in Microflash. I was able to load the very first test program using this technique.

After that I restored everything to its original state. Reprogramming flash1 went well, and reprogramming with flash2 also worked, but as soon as that happened I was again faced with the same problem.

It looks like something is going wrong with the download, which leads to your timing problems. Try using the ATMElisp program, which will probably work better under Windows XP. Besides this, you should have a good look at the Zener diodes used on the circuit board (see one of the earlier questions).

I have two questions regarding the Flash Board. I am currently programming an IDE tool for the Flash Board and I have run into a problem with the serial connection.

Although I can separately configure the DTR and RTS lines using

SetCommState(hport, SETDTR),

I do not know how I should do this with the TxD (MOSI) and CTS (MISO) lines. Do you have a tip for this?

Also, I would like to know whether I can have the microcontroller execute only one instruction and then stop, for debugging purposes. It is possible to set breakpoints?

If you use PORT.DLL, which has been used in a variety of other projects (and can be downloaded from the September 2000 page of the *Elektor Electronics* website, number **000074-11**), it is very easy to set the lines to the desired states.

For debugging you would have to use a monitor program, but we do not have any experience with this.

(020183-1)

Parallel JTAG Interface

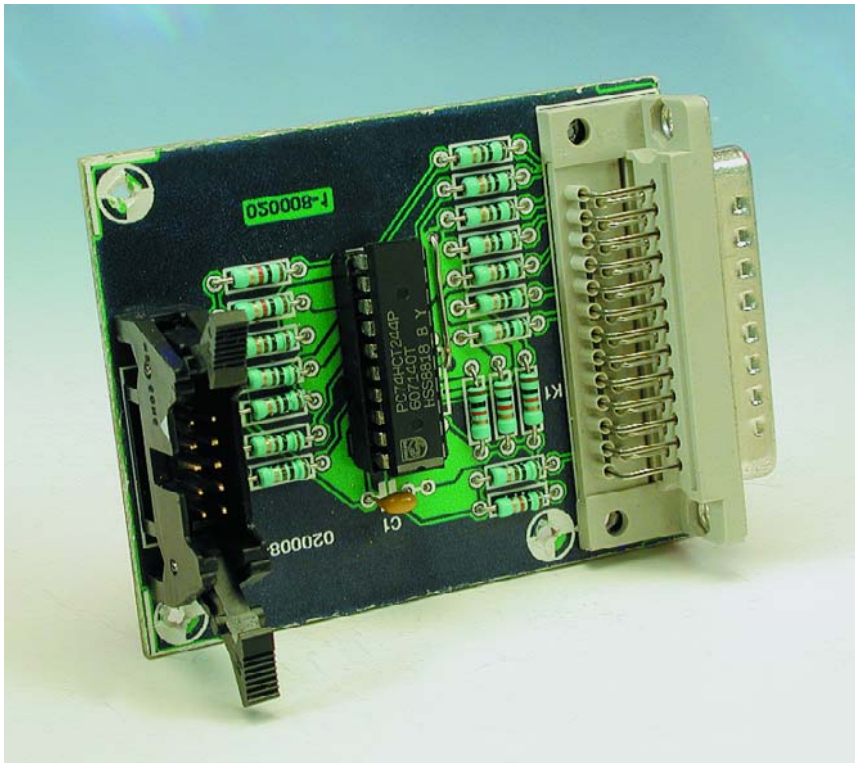
Design by P. Goossens

This circuit is for an interface between the parallel port of a PC and a so-called JTAG interface. This type of circuit was originally developed by a company called Altera, which referred to it as the 'Byteblaster'.

The version described here is compatible with the Byteblaster (electrically it is almost identical). This has the advantage that programs written for use with the Byteblaster can also be used with this interface.

The circuit itself is very simple. K1 is connected to the parallel port of the PC, whereas signals D0, D1 and D6 are brought to the JTAG connector via IC1. The output of the JTAG connector (TDO) is in turn fed back to the PC's BUSY signal via IC1. Pin 7 of the JTAG connector is used to provide an extra output. This signal is brought to the SELECT input of the parallel port via IC1. Some circuits make this signal active while they are in debug mode. This isn't really a requirement, but in some cases this can be useful (assuming that the software supports this feature). This signal could of course also be used to pass different status information back to the PC. IC1 is driven by the AUTOFEED output (active low) of the parallel port. As long as this is high the outputs of IC1 will be in a high-impedance state. This isolates the interface from the JTAG connector, so the target board isn't loaded by the JTAG interface.

The other signals are not specifically required by the JTAG interface. These signals are used by the software to check that the interface has been connected to the (correct) parallel port. Two tests have been made available for this. The first is formed by the direct connection between D5 and the ACK input (active low). The second is made by the connection of D7, via IC1, to the P.END input of the parallel port. This connection only exists when the signal AUTOFEED (active low) is active. These two detection methods are very easy to implement in software. Should a printer be connected to the port, it won't react by printing several pages of gibberish, since the STROBE signal (also active low) has not been used.



Software

Since the interface is designed to test, debug and/or program chips, the software used with it will be dependent on the chip that is connected. But we can still give you a few tips to help you get started quickly with this interface. You should first go to the website of Altera (www.altera.com) and download the software for programming Altera chips via the JTAG interface. You should also be able to find what is called a JAM player. Apart from a Windows version, there is also a Linux version available. The JAM player is in fact an interpreter with its own programming language, which has been specifically designed for use with the JTAG interface. This allows you to control

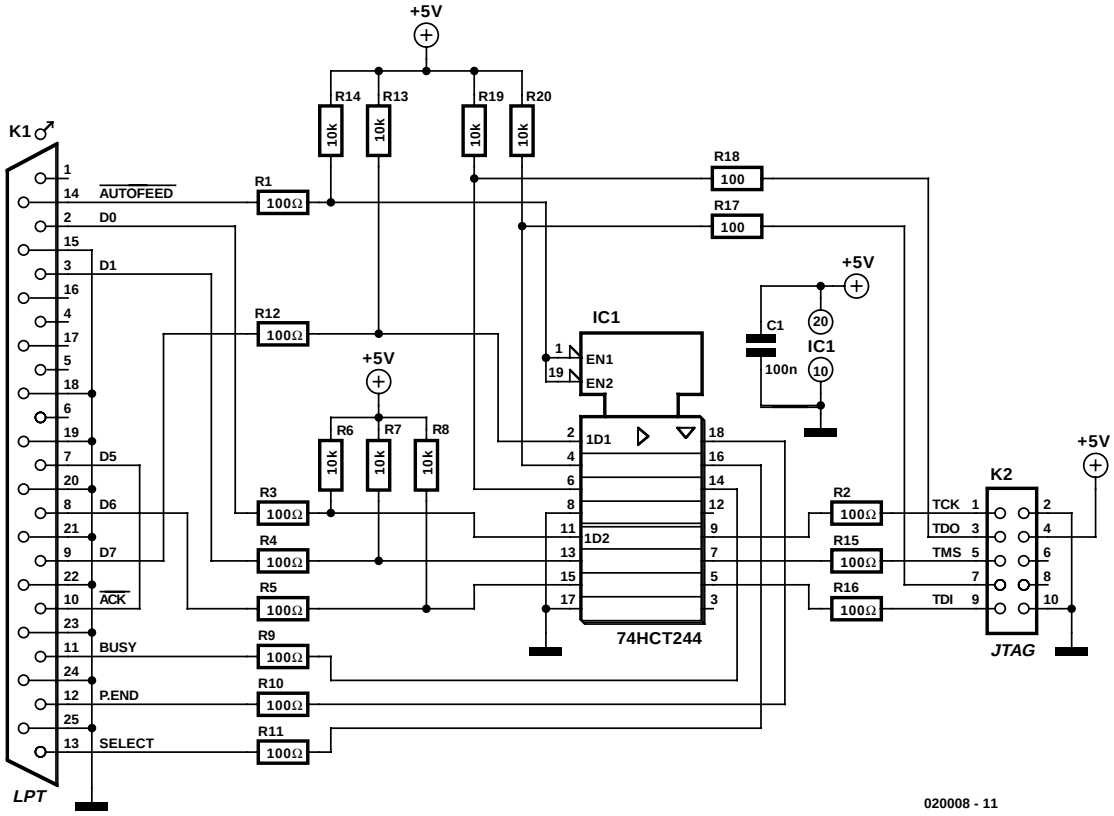


Figure 1. Circuit diagram of the JTAG Interface.

the JTAG interface in a simple manner, using special commands. Apart from the JAM player there are also conversion programs that change several types of file into a JAM format, so that the JAM player can then interpret these files.

There are various projects described on the Internet by people who write software for the JTAG interface. There is little point in giving a summary of the websites here because they are free web pages and regularly move to different servers. Any list will quickly become out of date. In this case a good Internet search engine like Google is the best starting point for finding suitable software.

(020008-1)

COMPONENTS LIST

Resistors:
R1-R5,R9-R12,R15-R18 = 100Ω
R6,R7,R8,R13,R14,R19,R20 = 10kΩ

Capacitors:
C1 = 100nF

Semiconductors:
IC1 = 74HCT244

Miscellaneous:
K1 = 25-way sub-D plug (male), PCB mount
K2 = 10-way boxheader
PCB, order code 020008-1 (see Readers Services pages)

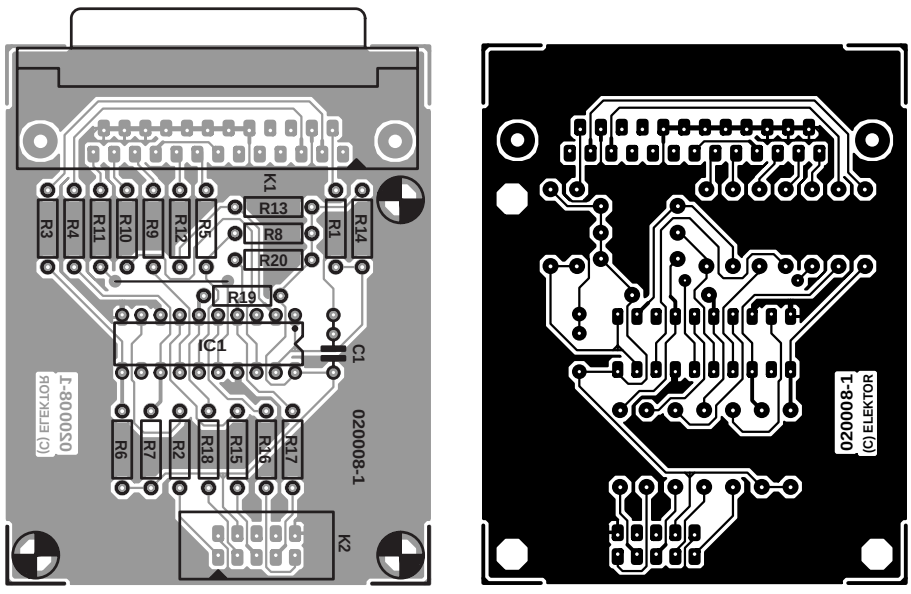


Figure 2. PCB design (board available ready-made).

High Voltage PSU

a universal supply for valve amplifiers

Design by Nils Gohr

This power supply unit (PSU) provides hum-free and stable operating voltages for valve preamplifiers and power stages. Negative bias voltage included!

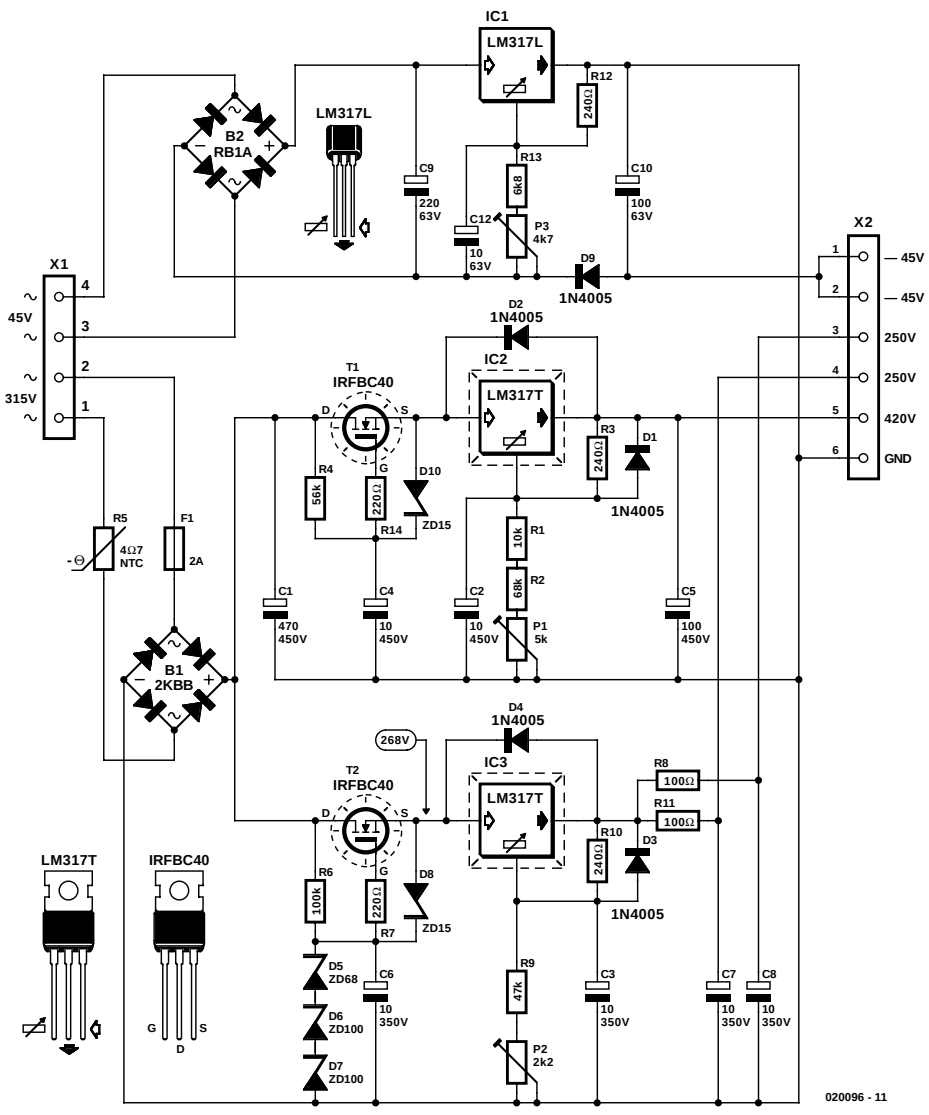


Figure 1. Three voltage regulator for three supply rails typically required in valve power amplifiers.

Editor's note:
The circuit is published as suggestion only. It has not been tested or constructed by Elektor's in-house design laboratory, hence the absence of an Elektor-style PCB layout and a parts list.

Most valve amplifier circuits require a high-voltage power supply. Here, such a circuit is proposed by one of our readers, Mr. Nils Gohr, who claims to own a working prototype.

The concept of the PSU is just fine for a power output stage using KT66 or KT88 power pentodes in push-pull or A/B mode requiring a relatively high plate voltage (here, 420 V) I combination with a grid bias voltage of –45 V for the quiescent current setting.

The preamplifier stage may be built around a double triode with two mutually decoupled 250-V anode voltages being available on the PSU.

Three voltages

The circuit requires a mains transformer with secondary windings

supplying 315 volts and 45 volts connected to the circuit via terminal block X1. Each output voltage has its own stabilizer. The resultant three regulators circuits are almost identical and based on the adjustable voltage regulator type LM317. The two high-voltage branches additionally feature soft start circuits based on FETs. These sub-circuits prevent the smoothing (reservoir) capacitors from being damaged by the current surge that occurs at switch on.

The raw direct voltage obtained from the rectifier on the 315-Vac winding is smoothed by C1 and applied to the voltage regulator via the FET. The direct voltage at the regulator input will be approximately 444 V. The target output voltage of 420 V is adjusted with the aid of preset P1 at the 'adjust' input of the IC. The 'adjust' voltage is always 1.2 V below the output voltage. Consequently, series network R1-R2-P1 drops about 420 V. Capacitor C2 is connected in parallel with this network to afford some buffering of the adjust voltage. Diode D1 counteracts the effect of C2 by ensuring that the adjust voltage can become positive with respect to the output voltage buffered by C5. In similar fashion, D2 protects the voltage regulator by preventing the output voltage exceeding the input voltage. All electrolytics in this section of the PSU must be rated at 450 volts!

The soft start is implemented using an R-C network R4-C4. Capacitor C4 will charge relatively slowly via R4, causing the current through the FET to increase proportionally. Diode D10 prevents the source-gate voltage from exceeding 15 volts.

With the exception of diodes D5, D6 and D7, the 250-V stabilisation is identical to the 420-V section discussed above. The three zener diodes keep the gate of FET T2, and with it the input of the voltage regulator, at about 268 V. Because most valve amplifiers have an input stage based on a double triode, two mutually decoupled 250-V rails are available at the output. The decoupling elements are R8-C8 an R11-C7.

Finally, there's a low-voltage section based around regulator IC1, whose task is to furnish a negative bias voltage for the input grid (G1) of the output valves. The current requirement on the output should not exceed a couple of milli-amps. Because the output voltage is to be negative with respect to the PSU and amplifier ground, a separate rectifier is required to enable the enable the regulator output potential to be connected to ground. Preset P3 should be adjusted for a lower reference potential of –45 V. Although the grid bias section of the supply is also based on an LM317, it should be noted that it differs from the two HV sections in that a low-current version in a TO92 case is used.

PCB layout

The other two regulator are housed in a TO220 case and are suitable for an output current of 1.5 A. Because of the anticipated heat dissipation, they should be mounted on heatsinks. The same applies to the FETs in the soft start circuits. On the PB designed by the author, an elegant solution has been found in that the two associated TO220 parts are mounted back to back on PCB-mount heatsinks type SK129 from Fischer (Dau Components). Unfortunately there are unequal potentials at the cooling fins (LM317: output; IRFBC40: drain), so that insulating washers are required.

The PCB layout only just meets the electrical safety requirement for tracks carrying voltages of the order of 400 V and more. The requirement is a track distance of at least 0.9 mm.

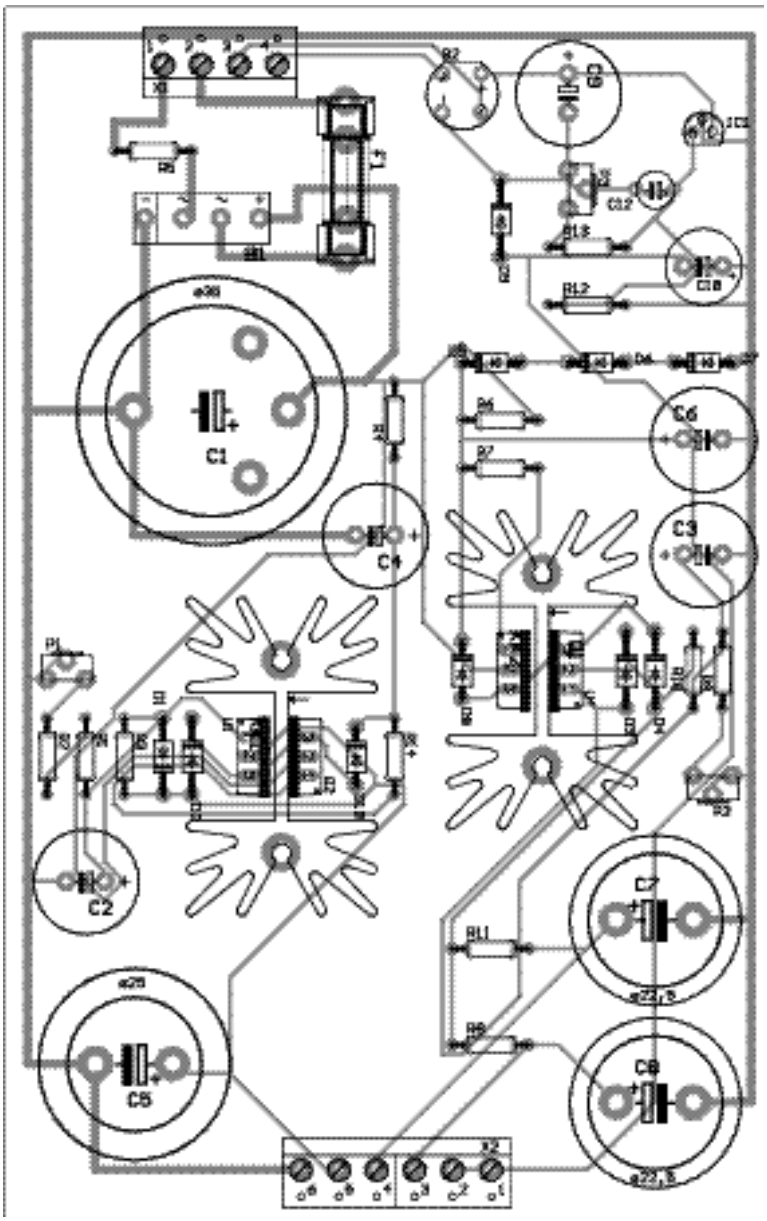
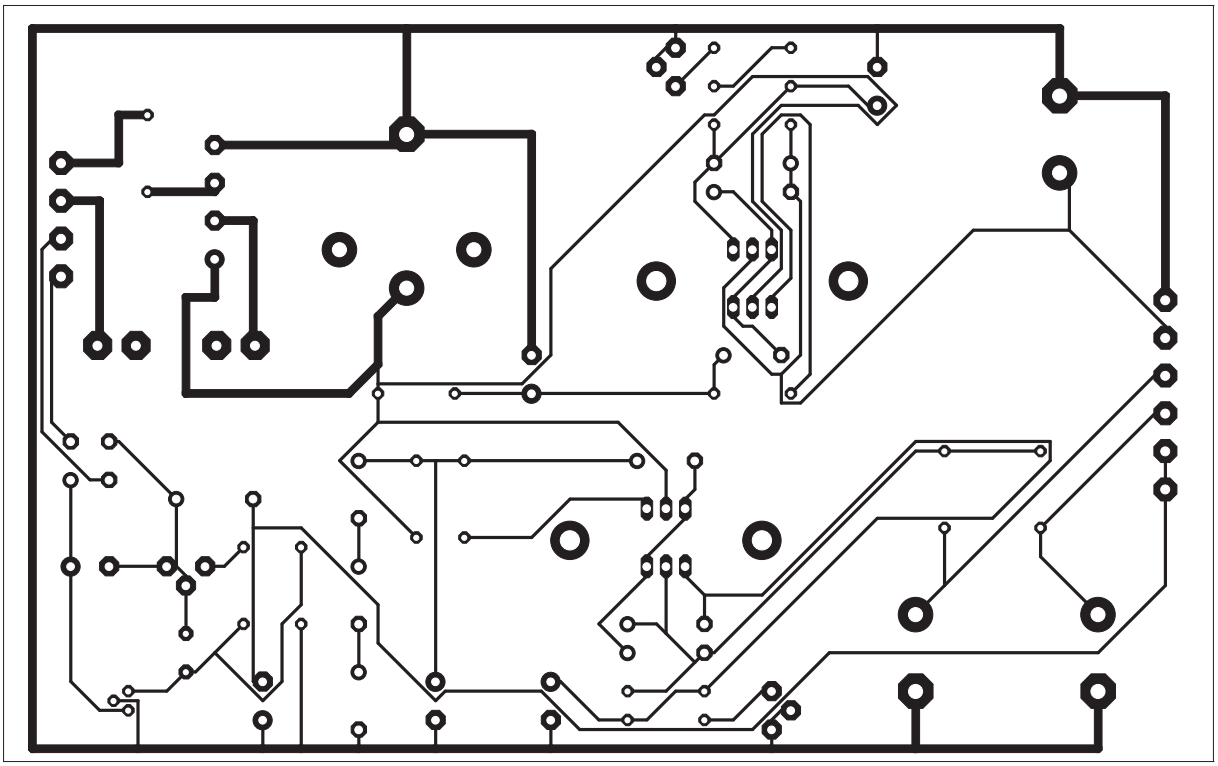


Figure 2. The generously laid out PCB designed by the author (not available ready-made through Readers Services).



Warning

This circuit carries dangerous and potentially lethal voltages and should only be built and used whilst in compliance with relevant electrical safety precautions.

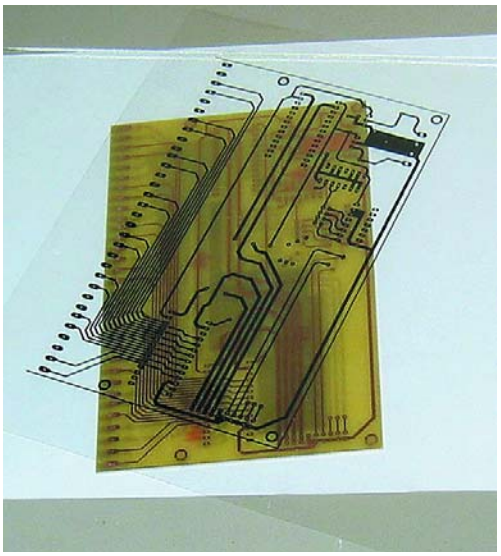
(020096-1)

Making Your Own PCBs

tips for circuit board fabrication

By Michael Möge

High-quality PCB layouts for hobby use, small lots or experimental setups can be generated using modern layout programs, some of which are available for free. However, this tidy work on the PC is followed by the unpleasant task of etching the circuit board.



A wide variety of etching kits and accessories are available in electronics speciality shops, but none of them is especially inexpensive and some are rather awkward to use. The difficulties start already with the layout film. Since even expensive overhead projector foils made with photocopiers or laser printers are not sufficiently opaque, it's necessary to superimpose at least two identical foils to expose the board. Copying the layout onto sheet film requires a small but fully equipped photographic darkroom.

Etching using ferric chloride or ammonium persulphate amounts to an exercise in patience and involves hotplates, a lot of mess and accepting a certain percentage of spoiled boards. Disposing of the chemicals is also a problem, and anyone who wants to achieve good etching results and doesn't want to risk poisoning himself should know what he's about. Chemicals require careful, well-considered handling.

Contact foil

Good-quality opaque foils can be produced using an ink-jet printer. Foils available from Conrad (ink-jet foil OH 3) have proven to be very good. They have a very fine coating, and thanks to the attached backing sheet they are drawn into the printer quite accurately. The foil, with the attached backing sheet, should be placed in the paper feed of the printer (such as an Epson Stylus Color 660 or Canon S450) with particular care being taken to ensure that the glued edge of the foil is positioned fully against the front feed stop and the side edge is tight against the fixed edge guide.

After being printed, the foil should be dried using a hair dryer and then printed again. Two passes through the printer are usually sufficient. It is important to ensure that the printed foil is adequately dried before being passed through the printer again – a bit of patience is necessary here. If the foil is carefully placed in the feeder, track widths as fine as 0.2 mm can be exactly superimposed in this manner. This yields a printout that is dark black and sufficiently opaque. After being allowed to dry for 24 hours, it can be used to expose the printed circuit board.

Foils from other manufacturers have too coarse a coating, cannot be drawn accurately into the printer due to strong warping or do not have backing sheets.

Exposure and development

An old sun lamp (1000-W mercury vapour lamp) is suitable for exposing the circuit board. The author has obtained good results with an exposure time of just under a minute with the lamp around 50 cm away from the board and pane of glass acting as a cover weight. The optimum exposure time depends on the light source, the type of mask used and

Printer settings

Paper:	photo quality glossy foil (not ink-jet foil!)
Colour:	black
User-defined settings:	fine (720 dpi), no halftone process, –25% brightness, +25% contrast

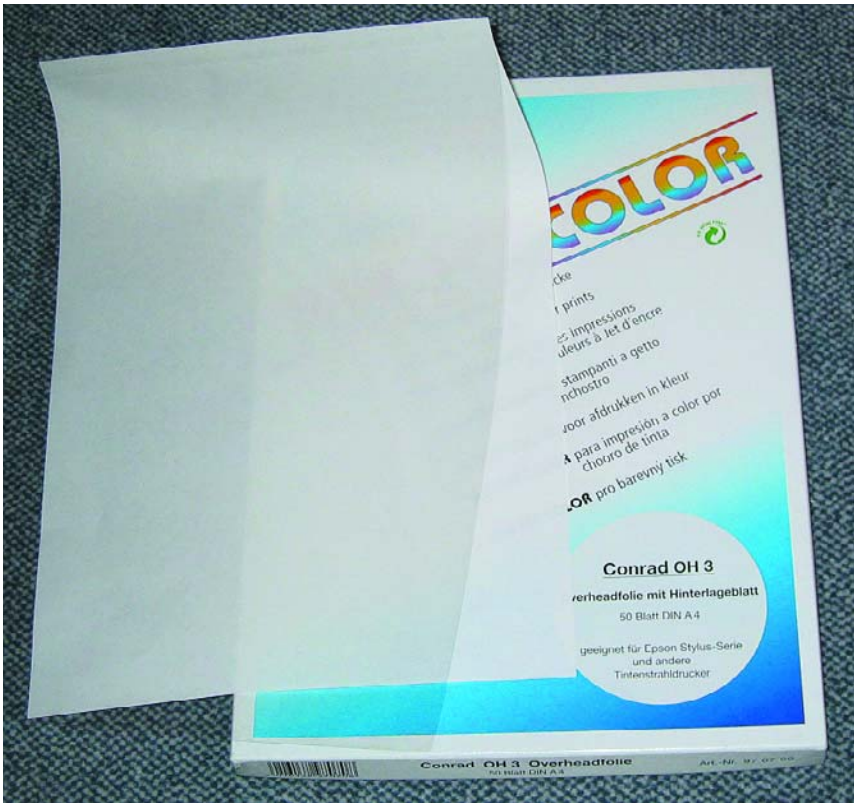


Figure 1. A suitable overhead foil has a fine surface structure, is free from warping and has a backing sheet.

Figure 2. The exposure process using a regular sun lamp.

the quality of the base material. Consequently, before actually making any PCBs you should expose and etch a test board made using a blank piece of circuit board material exposed in strips.

You can make strip exposures by covering the board with a sheet of cardboard and sliding it a bit to the side every 15 seconds while exposing.

Chat site tips

The following comments have been translated and edited from a 'thread' in the German-language section of a discussion forum at www.batronix.com. The thread contains a useful collection of suggestions and tips for the fabrication of PCBs by electronics hobbyists. The forum and other parts of the batronix website address the complete process of PCB fabrication, from purchasing chemicals to drilling.

Exposure

UV-C from an EPROM eraser is useless! Face tanners (with tubes) are perfectly OK. Use a black underlay, a (double-sided) film (sleeve) resting on two strips of board material and a sheet of glass on top (crystal glass from the glazier is best since it is more transparent to UV), and let the face tanner shine on the board for around two minutes.

Expose our boards using superactinic or UV-A light, which means a wavelength of 400 nm or more. UV-B is never present in professional PCB exposure equipment. You can tell the type of UV from the label on the fluorescent tubes. TL 20 W 05 is good, for example. Six of these per side (120 W) yield a guaranteed minimum exposure time of two minutes for our boards. Type XX yy W 08 or 09 tubes are just as good. The last two digits of the type number (e.g. '05', '08' or '09') give the wavelength. Types 08 and 09 are used in (face) tanners.

To obtain uniform brightness on the board, the distance between tubes should be fairly close to the distance between the tubes and the board, or reflectors should be used. Nitrophot lamps are OK, but they have to warm up first; optimum light yield is obtained only after around 15 minutes. They also have a rather long exposure time for our boards (seven minutes or more). And the longer the basic exposure time, the greater are the deviations from our specification (in absolute terms, 10 percent of 7 minutes is rather different from 10 percent of 2 minutes!). Construction lamps have impressive power ratings, but the heat generated in the mask and board can cause problems.

The advantage of a point source of light is that narrower tracks and track separations are possible. The best point source is the sun, but the weather is a problem. In May, an 5-minute exposure with a heavy glass cover sheet to ensure good contact gives excellent results (and costs almost nothing), but you're out of luck if it's raining or you have to make a board next November.

It's better to expose too long than too short, at least with

good board materials. With a good mask, too long an exposure does no harm. Another good tip is to make a step exposure by removing the protective foil a strip at a time, exposing each strip for X seconds. Strip n that is fully developed after (ideally) 40 s (maximum 60 s), plus one step for the film, gives the (minimum) exposure time = n x X (make a note of this time, it remains fairly constant).

Development

It's hard to make up for incorrect exposure during development. Bungard boards like a strong developer (13–30 g NaOH per litre), but please keep it at room temperature for safety (a splash in the eye could mean 'lights out' forever).

Store fresh developer in an old plastic container (for large quantities, use the same concentration and keep the container tightly closed, since the solution becomes weak on exposure to air due to the absorption of carbon dioxide). Only take out as much as you need for the job, and rinse it down the drain diluted with water. One-percent NaOH is what comes out of the back of your dishwasher!

If the exposure is too short, a film of incompletely exposed resist will remain on the copper. It sometimes looks like the structure of glass cloth, and during development the colour there will (by design!) turn to something between red-brown and violet. This film prevents the etchant from reaching the copper. As a test, briefly immerse the board in the etchant after it has been developed and rinsed with clear water. The copper in the developed areas of the board must immediately change colour. If it doesn't, the exposure was too short!

One remedy is to rinse the board with tap water, dry it carefully but thoroughly (to protect the exposure device), preferably using (compressed) air, and then expose the entire surface of the board again (the film can hardly be re-applied) for 20 percent of the original time. Develop it again and repeat the etching test. This procedure may sound complicated, but who wants to throw away a valuable board?

The idea is that about 80 percent of the original resist on top of the tracks will be left, which is sufficient for etching, but the thin 'barrier layer' on the rest of the copper will be removed – and the board will be saved!

By the way, our boards can always be exposed multiple times. You can 'misuse' this feature to go as far as creating a sort of solder mask and/or component overlay on the rear of the board. And for those of you who know what 'contrast gain' means, under ideal conditions our boards can be exposed and developed using a photocopy on writing paper, with a suitably short exposure time and a developer that is twice as strong as normal. Other tricks are also possible.

Etching

Problems in etching come from the exposure, or in rare cases from development. Here I must state very clearly that sodium and ammonium-persulphate are forbidden due to hazardous waste regulations. They etch extremely poorly and decompose if you just look at them (leave alone when you use them), and in terms of achievable track width alone they are catastrophic. The disposal costs are around ten times higher than for ferric chloride. The only reason they are used is because they do not foam (as does ferric chloride), so they can be used in shallow trays.

Etching time: at least 90 s for 35-μm copper with a reasonably fresh solution at 45 °C in a spray-etching machine. Maximum etching time: 180 s, after which the ferric chloride should be replaced; it will then contain at most around five times as much copper as NaPS. In any case, there will be at most 80 percent less hazardous waste exclusively due to the etched copper (but not the etchant itself). Once again, ferric chloride takes up five to ten times more copper per litre than NaPS and allows etching contours down to less than 0.1 mm to be obtained, all in one tenth of the time (using a spray process). It also doesn't eat holes in your clothes, and a spot remover (RX3) help against 'rust' spots.

There is only one etchant that is even more effective, but it is only for pros due to considerations of user safety: copper chloride in the form of a mixture of a lot of water, a little hydrochloric acid and even less hydrogen peroxide. Not for the home user!

Never store a partly used bag open! NaOH attracts atmospheric moisture like a magnet and forms a highly aggressive paste that can lead to loss of sight on contact. You should thus buy developer in special packages for one litre of water, dissolve the contents in a litre of water and keep the liquid in a closed container, or else buy 250-g cans and dole out the contents as needed with a measuring spoon, taking absolutely scrupulous care to keep the container closed.

According to the safety data, pure ferric chloride (granulate or pearls) is less dangerous to life and limb than NaOH. I don't want to over-generalise, but I dare say that an aqueous solution of developer in the intended concentration is fairly harmless (see the dishwasher example). Have you ever carefully read the warnings on dishwasher detergent blocks? Who uses the environmentally friendly developer SENO 4007 instead? And did you know that ferric chloride is used in large amounts to treat drinking water (but of course there's no copper present!)?

Dieter Bungard
Tech Support – Bungard Elektronik – www.bungard.de

ing the board. After the board has been etched, you can determine which exposure time gives the best results. Due to the UV radiation, you should wear protective glasses (or not look into the light)!

You can easily prepare the development bath yourself. Sodium hydroxide (NaOH) can be obtained from a druggist for a couple of Euros per kilogram. Dissolve 2 g of NaOH in 0.4 l of lukewarm water to make the developing bath. The boards can be developed at room temperature. The photoresist will colour the solution blue-green.

Etching

The best results are still obtained using the tried and true copper chloride process with hydrochloric acid and hydrogen peroxide. Although this process is used for industrial mass production, for home use it is potentially dangerous. When working with acids and alkalis, you must always wear suitable safety goggles.

In this regard, you should also read the comments of 'PCB guru' Ditmar Bungard (see the box).

A solution of 340 ml of 6-percent hydrochloric acid and 160 ml of 15-percent hydrogen peroxide can be used as an etching bath for approximately five Eurocard-sized PCBs. Etching takes around 10–15 minutes at room temperature.

Hydrochloric acid is sold by pharmacists, druggists and in 10-l containers by building supply shops (bricklayers use it to remove calcium

The chemistry in a nutshell

The main reactions are:

$H_2O_2 \rightarrow H_2O + O$ (accelerated by the presence of Cu)

$Cu + O \rightarrow CuO$

$CuO + 2\,HCl \rightarrow CuCl_2 + H_2O$

The possible secondary reactions are:

$2\,H_2O_2 \rightarrow 2\,H_2O + O_2$

$2\,HCl + H_2O_2 \rightarrow 2\,H_2O + Cl_2$

$H_2O_2 + Cl_2 \rightarrow 2\,HCl + O_2$

residues from walls). Hydrogen peroxide can also be obtained from pharmacists and druggists or from hairdressers.

The board must be rinsed thoroughly under running water between the developing bath and the etching bath. When the developed board is placed in the etching solution, the areas to be etched away immediately turn red and small bubbles form on the surface of the resist. The board should be moved gently in the etching solution for the full duration of the etching process. Air must not be blown into the solution, since this would cause premature decomposition of the hydrogen peroxide. Heating the bath is neither necessary nor beneficial.

The etching bath will become a light blue-green (from the copper chloride), but no sludge will be formed. If the etching rate drops, the solution can be rejuvenated with a little hydrogen peroxide. If the previously red copper areas turn distinctly whitish, the bath can be salvaged for a certain time by adding some hydrochloric acid.

After etching, the board must again be thoroughly rinsed under running water, but the rinse water must not be simply tipped down the drain. Instead, it must be collected and disposed of properly. The remaining resist can be wiped off using acetic acid ethyl ester (ethyl acetate), universal thinner (fast and cheap), spirit (ethyl alcohol) (somewhat slower) or acetone (very thorough). An even simpler technique is to use a pot scrubber to remove the resist.

The etching process produces copper chloride and water, with oxygen and small amounts of chorine being released. The primary reaction is the decomposition of hydrogen peroxide. A 'stale' etching solution can be regenerated quite well using 20-percent hydrogen peroxide (H_2O_2) or 20-percent hydrochloric acid (HCl). Higher degrees of dilution yield less satisfactory results. Concentrated chemicals should only be used by qualified experts.

A bath that is still usable can be stored in a loosely capped bottle (since hydrogen peroxide decomposes to form water, releasing oxygen in the process). When the bath is used again, it must be reactivated using hydrogen peroxide.

Disposal

A used etching bath should be allowed to outgas for several days. Hydrogen peroxide breaks down into water and oxygen. The bath will then contain only copper chloride, hydrochloric acid and water. This solution does not form any sludge or flocculent deposits, so it can be readily collected in a plastic container. A mixture of copper chloride and hydrochloric acid can be handed in to any hazardous waste collection depot as pure chemicals. Neutralising the acid with sodium hydroxide (lye) is not recommended, since it can be dangerous. Such a process produces salt water, copper sludge and various gasses. Due to the violence of the reaction, such a disposal procedure should be left to trained chemists.

Although the chemicals described here are used in relatively non-hazardous concentrations, even dilute acids can cause damage. It should be obvious that nobody would drink these solutions. However, for safety you should wear safety goggles and gloves. Splashes on the skin or clothing can easily be washed off or out, but splashes in the eye are a different story. Oxygen and small amounts of chlorine are released during etching. It should be clear that you must not eat, drink or smoke during such work. Suitable ventilation should be provided (opening a window is a good start).

The best results are obtained using Bungard PCB material. Although these boards are more expensive than 'no-name' products, they are worth the money. For very fine features they are indispensable, but boards with coarser features can be made using other products.

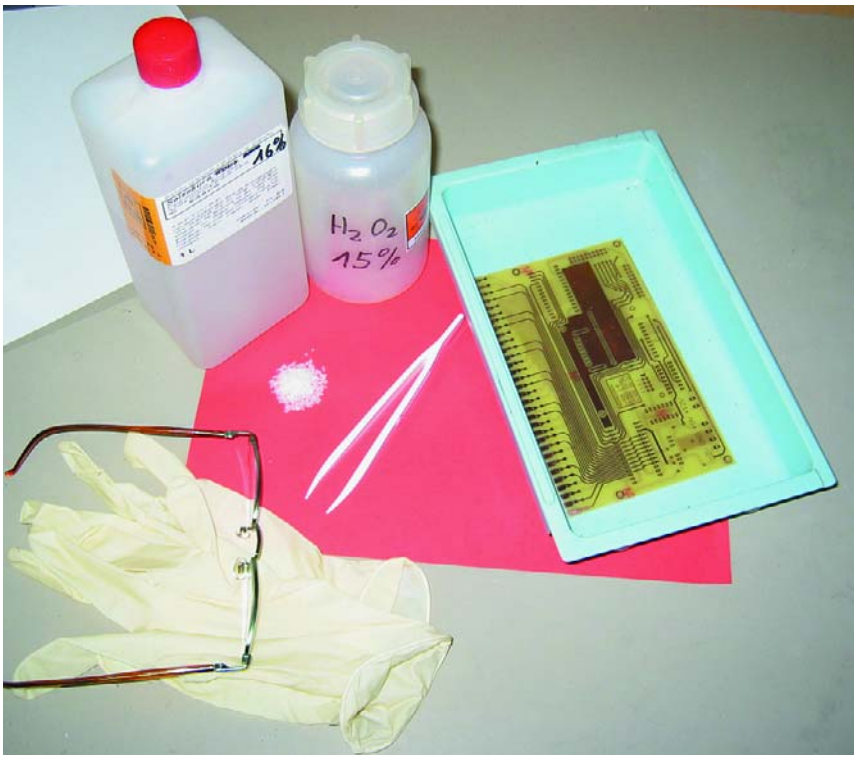


Figure 3. Etching equipment.

(020099-1)

High-Speed Controller Board (2)

part 2:circuit description and construction

Design by Luc Lemmens

In the June 2002 issue we introduced you to the DS89C420, the youngest member of the microcontroller family produced by our ever active friends at Dallas Semiconductor. This controller, although fully downward compatible with other devices from the 8051 series, is much faster and offers loads of extras. This month we continue where we left off by presenting a board which, helped by some free software from Dallas, forms the basis of a powerful yet affordable development system.

The 8051 series is still extremely popular with hobbyists as well as professionals. Since the introduction of this microcontroller by Intel in the late 1980's, a veritable galaxy of development tools, programs and subroutine libraries have appeared which are freely available to anyone with access to the Internet. Although the original controller has been extended with peripheral functions and its clock frequency raised by various manufacturers, the basic architecture of the 8051 core has remained virtually unchanged. With this controller, the frequency of the external clock oscillator is effectively divided by 12, simply because a machine cycle requires that number of clock cycles. Dallas Semiconductor for the first time overhauled the internal structure of the processor core in a successful move to eliminate the ÷12 clock prescaler. In a DS89C420, a clock cycle is the same as a machine cycle, so that the new controller is twelve times faster than its predecessors.

In part 1 of this article we already hinted at extra features offered by the C420 while remaining compatible with older applications. An 8051 or an 8032 may be replaced by a DS89C420 without problems — the new IC is pin compatible and it will faithfully run old software although some timing aspects may need to be looked at because of the much higher speed.

The availability of Flashable memory on the C420 makes the chip extremely well suited to developing, debugging and testing of new application software and hardware. The Flash memory may be programmed many times over, while a monitor program (present in a reserved area of the chip memory) allows a new program to be downloaded into the controller via the PC's serial port — all without the slightest problem or the need to pull chips etc. High time to see how what the Elektor High-Speed Controller Board looks like in practice.

Hardware description

The circuit diagram of the development board for the DS89C420 is given in **Figure 1**. Nothing unusual a first blush, you say, just another 8051 board? The apparent absence of EPROM is compensated for by the (invisible) 16-kByte Flash memory on board the C420. Although nothing should prevent you from extending the circuit with external program memory to a maximum size of 64 kBytes, it must be said that programming the board then becomes more difficult than with the standard configuration. The internal memory also has the inherent advantage of running at the full controller speed, plus you do not have to worry about timing problems when using a not so fast memory device (more about this further on).

The DS89C420 is fully pin compatible with other microcontrollers from the 8051 series, so we reckon its pin designations will be familiar to readers having seen (and built) some of our earlier projects based on the 8051 or its derivates. Only the clock oscillator circuit is slightly different if the controller is to work at

relatively high quartz crystal frequencies. As you may know, quartz crystals of up to about 25 MHz usually operate in fundamental frequency mode, while 'third overtone' is more common with crystals for higher frequencies. The latter configuration requires the fundamental-frequency mode of oscillation to be suppressed. If not, a quartz crystal specified for, say, 33 MHz will cheerfully oscillate at its fundamental frequency, 11 MHz. Hence an additional L-C combination (L1 and C18) has been added to the usual oscillator layout. That's right, the two added compo-

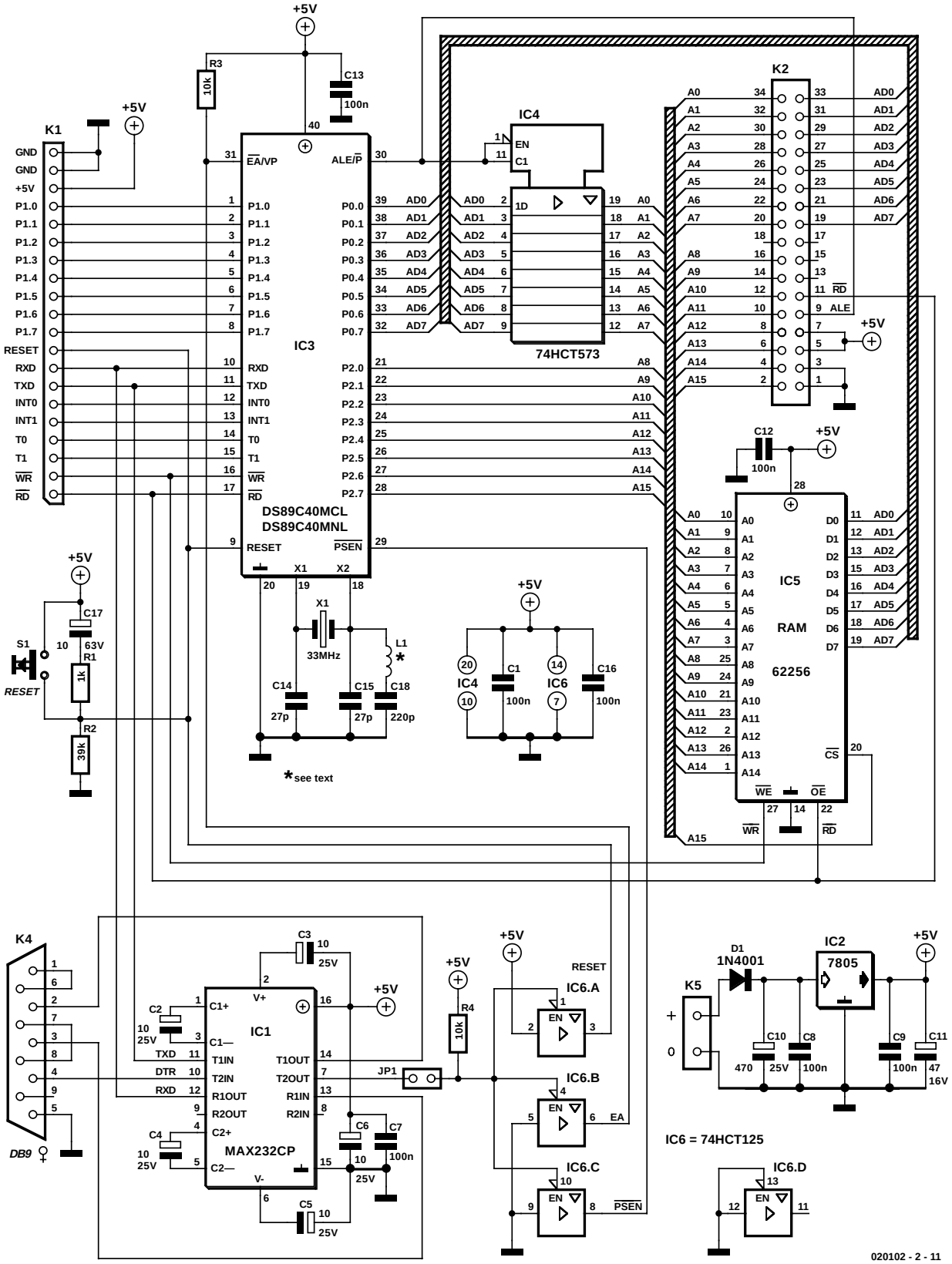
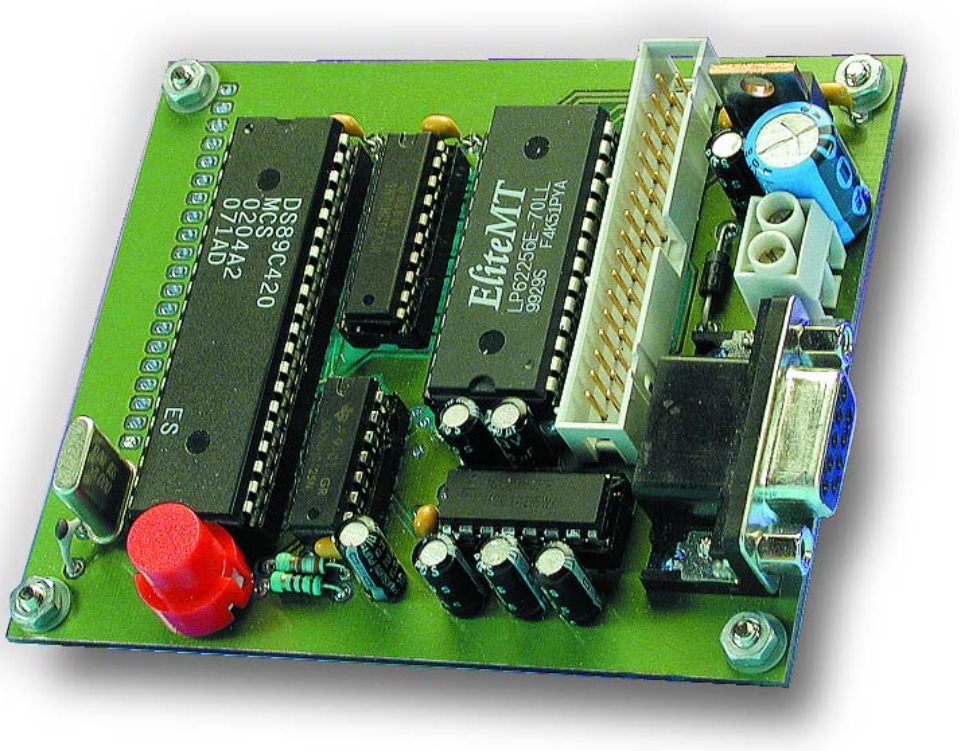


Figure 1. Circuit diagram of the DS89C420 development board.



nents force the crystal to oscillate at its third overtone (and your programs to run at maximum speed). **Table 1** lists the maximum value of L1 that may be employed with some commonly applied clock frequencies.

The address latch (IC4) and RAM (IC5) are standard for an 8051 application circuit. You will also see that most relevant control lines of the board are accessible for connection to the outside world via K1 and K2. These two connectors make it particularly easy to connect external hardware to the 'C420 development board.

A supply voltage of between 9 and 15 VDC may be connected to the board via K5. A cheap mains adaptor ('battery eliminator') is perfect for this job. The raw 9-15 V input voltage is stepped down and regulated to a clean +5 V supply rail by IC2. This voltage is used by the microcontroller and the other hardware. Diode D1 acts as a reverse polarity protection, preventing the circuit from taking damage if someone (not you, of course) accidentally connects the input supply voltage the wrong way around on K5.

Table 1.

Maximum value of L1 with various quartz crystal frequencies

Crystal frequency	L1 (max.)
20 MHz	12 mH
25 MHz	8.2 mH
32 MHz	4.7 mH

The reset circuit around C17, S1, R1 and R2 is also a classic. C17 and R2 provide a reliable reset signal to the microcontroller when the supply signal is switched on, as well as when S1 is pressed. Although R1 limits the current that flows through C17 when S1 is closed, it hardly affects the reset signal proper.

The serial interface using the MAX232 (IC1) has been applied in many other microcontroller circuits. Yet, this time there's a difference: the DTR (data terminal ready) pin on the RS232 interface is used to drive three 3-state buffers via IC1. When these buffers are activated (which happens when DTR on K4 goes High with pin 7 of IC1 consequently at Low), the microprocessor signals RESET, $\overline{EA}$ (external access) and $\overline{PSEN}$ (program storage enable) are switched to an unusual state. Note that $\overline{PSEN}$ is normally an output signal, so pulling it to a fixed logic level (0 V in this case), is remarkable.

The resultant combination of logic levels at the three controller pins prompts the 89C420 to start the internal monitor program which, among others, allows you to communicate with the controller via the serial port, erase the program memory, program it, as well as read and verify it. The monitor program is located in a special memory area inside the microcontroller, which is only addressed if the DS89C420 is switched into 'monitor' mode. In other words, the monitor is not run from the regular internal memory. It is therefore best described as a 'boot ROM' for the microcontroller, which is normally invisible but remains active as long as the three buffers IC6a, IC6b and IC6c are being activated.

Once the three buffer outputs are returned to high impedance, the 89C420 will behave like an ordinary microcontroller again, faithfully executing the instructions found in the program memory.

The monitor is switched off when jumper JP1 is removed. This may be necessary if the controller board communicates with a PC by means of an 'ordinary' terminal emulation program like HyperTerminal (found under Windows Accessories). Such programs may keep DTR High permanently, causing the microcontroller to remain stuck in monitor mode and unable to run your application program.

Construction

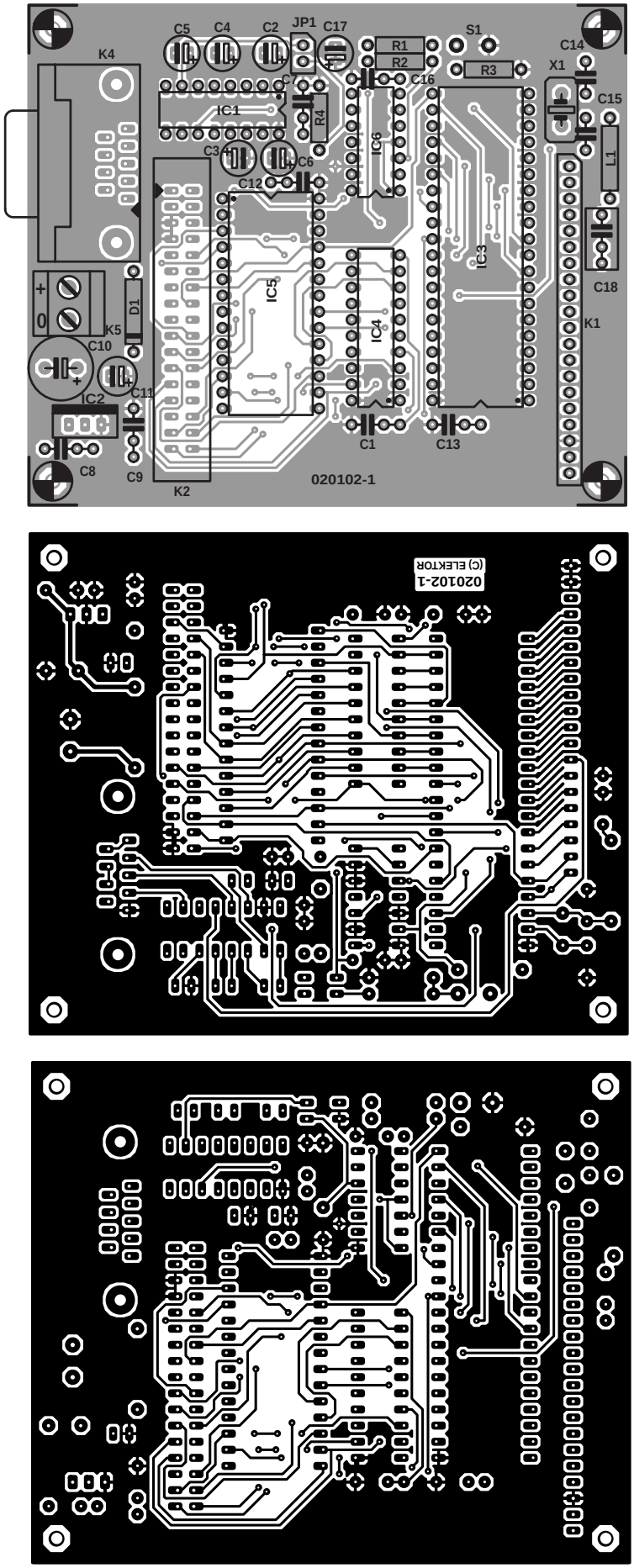
With our target audience in mind we can be brief about building the High-Speed Controller Board. Not much can go wrong if you use the double-sided, through-plated board shown in **Figure 2** (order code 020102-1) in combination with its silk screen component overlay and the information gleaned from the parts list. We recommend fitting the integrated circuits in sockets. Do observe the polarity of the diode and the electrolytic capacitors. JP1 may be a jumper but also a simple on/off switch.

Software

The monitor program enables the microcontroller to communicate with the outside world. At the PC side, you will also need some software to be able to program or read the DS89C420. The Dallas Semiconductor ftp site at

ftp://ftp.dalsemi.com/pub/microcontroller

offers two free programs for that purpose: **Microcontroller Toolkit** (MTK-beta.zip) and **Loader420**



(loader420.zip). Whilst **Figure 3** shows the two application programs side by side on one screen, only one of them is required in practice. Although the programs are comparable as far as functionality is concerned, we favour Loader420 which has been specially written for our microcontroller. MTK is also suitable for other microcontrollers from Dallas Semiconductor, hence it has more settings and options which makes it less easy to use compared with Loader420. Besides, MTK gave us a couple of Blue Screens when the scroll windows became too crowded. This was noticed with the example program we'll be discussing further on, where external RAM is being read, and the content and address of each memory location is being shown on the display. When the window fills up with data, MTK eventually crashes and causes Windows to pop up the infamous Blue Screen ('fatal

COMPONENTS LIST

Resistors:
R1 = 1kΩ
R2 = 39kΩ
R3,R4 = 10kΩ

Capacitors:
C1,C7,C8,C9,C12,C13,C16 = 100nF
C2-C6,C17 = 10µF 25V radial
C10 = 470µF 25V radial
C11 = 47µF 16V radial
C14,C15 = 27pF
C18 = 220pF

Inductors:
L1 = see text

Semiconductors:
D1 = 1N4001
IC1 = MAX232CP
IC2 = 7805
IC3 = DS89C420-MCL or -MNL (DIP40, Dallas Semiconductor/Maxim)
IC4 = 74HCT573
IC5 = 62256 (8 x 32k, speed min. 100ns)
IC6 = 74HCT125

Miscellaneous:
JP1 = jumper
K1 = 20-way SIL header
K2 = 34-way boxheader
K4 = 9-way sub-D socket (female), angled pins, PCB mounting
K5 = 2-way PCB terminal block, lead pitch 5mm
S1 = pushbutton with make contact
X1 = 33MHz quartz crystal
PCB, order code 020102-1 (see Readers Services page)

Figure 2. Double-sided through-plated printed circuit board designed for the High-Speed Controller Board (available ready-made).

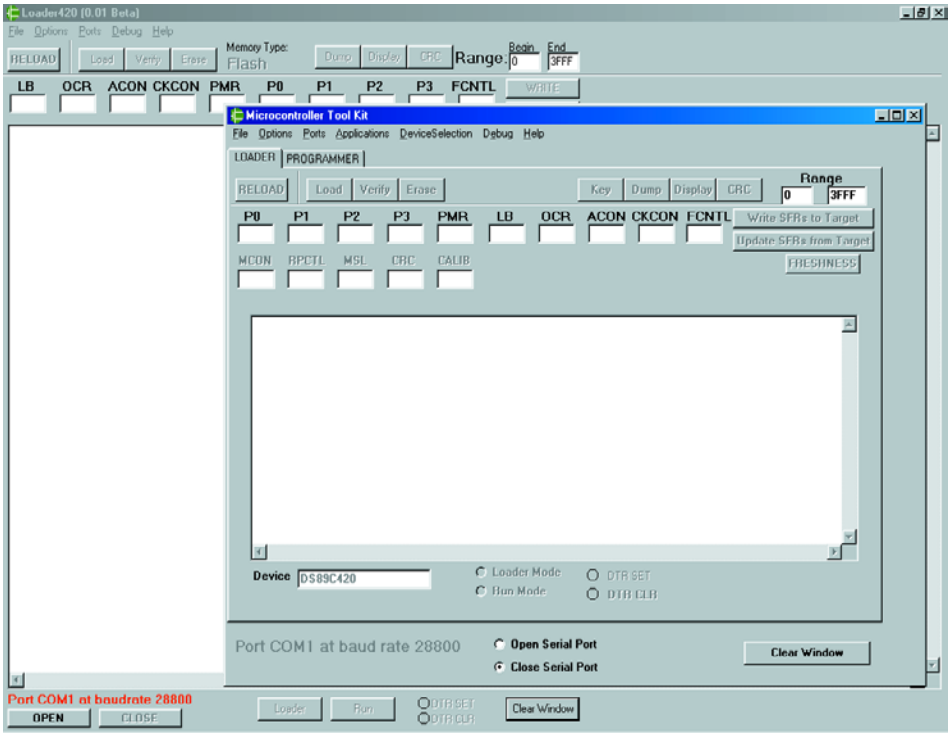


Figure 3. Both PC program utilities on one screen.

exception error'). Not very encouraging, of course, and enough reason for us to abandon MTK and continue working with Loader420.

Connecting up and initial testing

Connect the completed DS89C420 board to a free serial port on your PC, using a 9-way non-crossed sub-D extension cable (**not** a zero-modem cable). The controller board may be powered by just about any cheap 12-VDC mans adaptor. Next, run Loader420 on the PC. The first time it is run, Loader420 will select COM1: as its serial port. If necessary, this selection may be modified in the menu *Ports/Select Port Settings*. There, you will also be able to change the baudrate used on the serial link. When you quit Loader420, your communications settings are saved, so you don't have to enter them again during the next session.

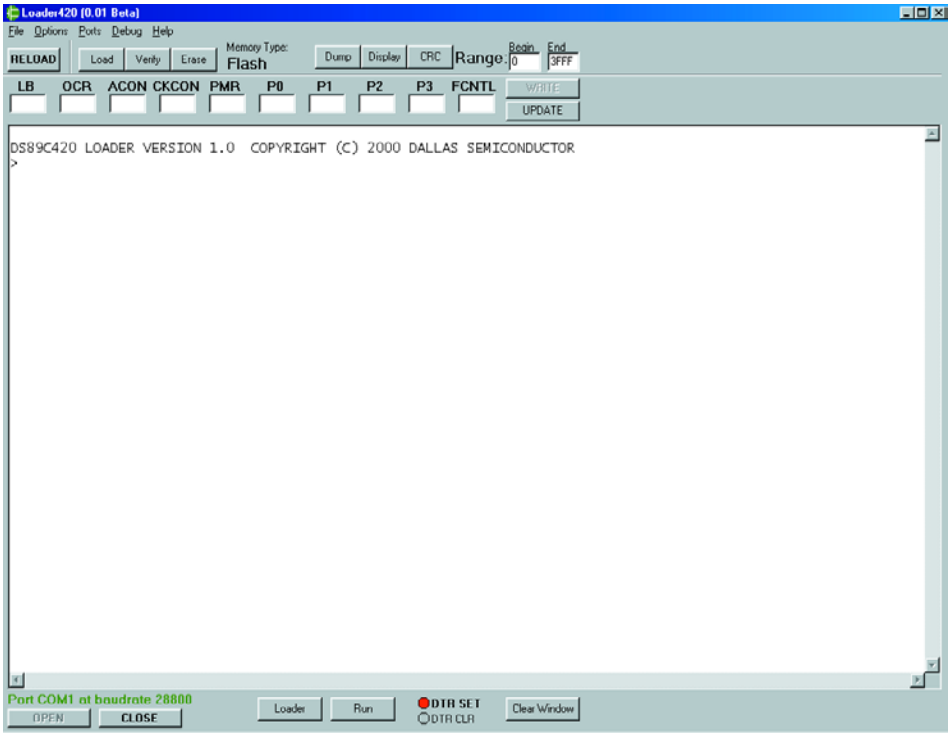


Figure 4. Loader420 in action. Note the command prompt.

It should be noted that the monitor program is generally called Serial Bootstrap Loader in the DS89C420 documentation. To preclude confusion with the Loader420 utility running on the PC, we will continue to use the term 'monitor'.

Not surprisingly, the serial channel is opened by clicking on the *Open* button. This, however does not yet establish communication with the microcontroller. In fact, that does not happen until you click on the *Loader* button. The DTR line on the serial port is then pulled High and the previously mentioned buffer triplet switches the controller into monitor mode. The DS89C420 will respond by producing a command prompt as illustrated in **Figure 4**. The microcontroller will automatically determine the speed (baudrate) of the communication with the PC. In fact, the monitor program will attempt a couple of baudrates by trial-and-error adaptation of the Timer1 reload value. If no prompt appears on your PC screen, communication with the controller is apparently not possible. In that unfortunate case, check your board for possible construction errors (with special attention given to the polarity of the electrolytics around the MAX232), the power supply, the physical link to the PC, and make sure the proper communication port is being used. If everything appears to be in order and the DS89C420 still fails to communicate after clicking the *Loader* button, you will be forced to choose a different baudrate on your PC. If you do not like calculations, simply try out different settings until the DS89C420 greets you with its command prompt. This trial and error process is simple but does require the serial port to be closed using the *Close* button, the new baudrate to be set and the port to be opened again by clicking on *Loader*. If you do not follow this cycle, the old setting will continue to be used and you are none the wiser.

Readers who are happy to use their pocket calculator will care to know that the monitor program is capable of detecting the baudrate (within limits), and 'adjust' the microcontroller for the corresponding settings. The monitor employs the microcontroller's serial port 0 in Mode 1 (asynchronous communications, 1 start bit, 8 data bits, no parity, 1 stop bit, full-duplex), with Timer1 in 8-bit auto-reload mode and PCON.7 at logic High. What it boils down is that the relation between the baudrate, crystal frequency and Reload value may be expressed as follows:

$$\text{Serial\_Loader\_BaudRate} = \frac{\text{CrystalFrequency}}{192 \times (256 - \text{Timer Reload})}$$

(timer-reload values attempted by the Loader:
FF, FE, FD, FC, FB, FA, F8, F6, F5, F4, F3, F0, EC, EA, E8, E6, E0, DD, D8, D4, D0, CC, C0, BA, B0, A8, A0, 98, 80, 60, 40).

The monitor program will attempt to use the indicated numbers for Timer1 Reload values, in order to find out which of these enables successful communication with the PC.

Let's get cracking!

Having overcome any problems you may have experienced with the serial communication, the command prompt produced by the DS89C420 invites you to get down to some serious programming. Although the monitor program will recognize commands typed behind the prompt (see the section *Command Line Interface*, page 183 *ff* of the DS89C420 User's Manual), Loader420 is a convenient alternative in that it makes all instructions accessible via menus and buttons. Simplicity and ease of use start here!

Loader420 has a number of functions to inspect and edit memory areas, registers and general DS89C420 settings. However, its fundamental function will be programming the on-chip Flash memory, with the ultimate aim of making the processor run our very own programs. Loader420 expects a file that complies with the industry standard

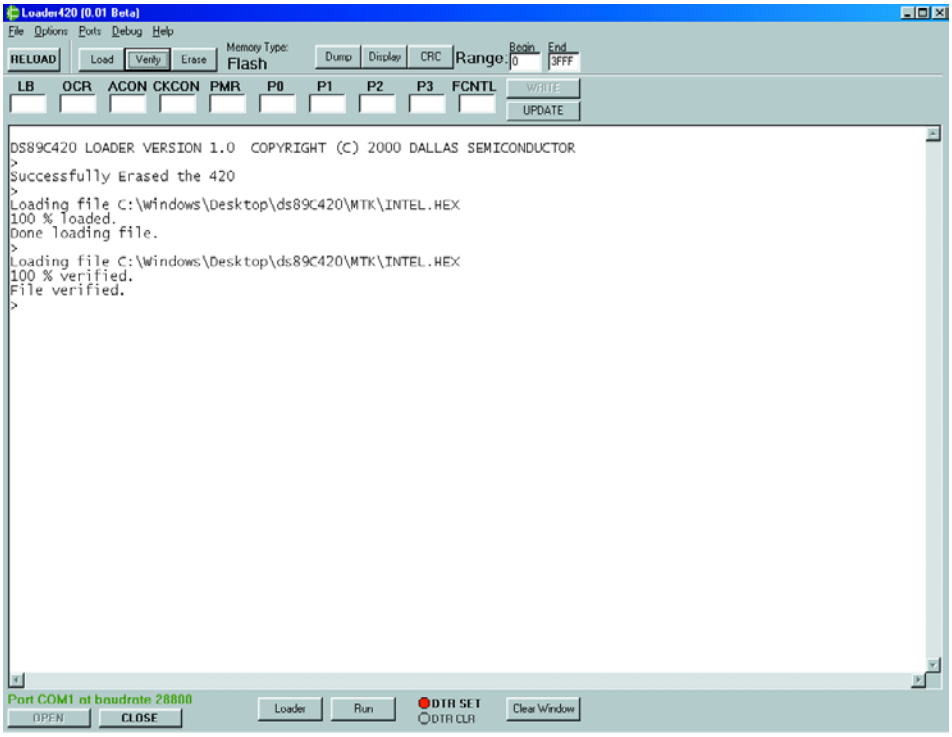


Figure 5. The example application has been loaded into Flash memory.

Intel Hex format, which is supported by most, if not all, assemblers and compilers. For our experiments we made use of an old assembler published as long ago as 1992 as part of our *Short Course 8051/8032 Microcontrollers and Assembler*. This assembler is still available on disk from our Readers Service under order number **1661**.

If an unwanted program is already present in the Flash memory of the DS89C420, it has to be removed first using *Erase* before a new application can be loaded. If you click on *Load*, a dialogue window pops up that enables you to select the desired HEX file. Loader420 will program this file into the microcontroller, after which the object code can be executed by clicking on *Run*. This effectively causes DTR to be forced Low, causing the microcontroller to quit monitor mode and start running the object code found in its Flash memory, starting at address 0H (i.e., the reset vector).

A simple application: RAM testing

The development board as discussed here has no LEDs or other indicators that allow a simple test program to run a functional check on the hardware. The program described below writes into the external RAM, reads back the memory location last written to, and writes the address and its contents back onto your display. **Listing 1** shows the simple RAM test program which also indicates the proper functioning of the communications between the processor and the external RAM device.

As already mentioned in part 1 of this article, the speed of the DS89C420 is such that timing problems may arise in read or write operations to and from external memory devices. Fortunately, the controller allows external peripherals to be addressed at a relatively low speed. Three bits in the CKCON register can be used to 'stretch' a memory access cycle, and with it, the length of the signals $\overline{RD}$ $\overline{WR}$, ALE and $\overline{PSEN}$, thus allowing slower external memory devices to be used without problems. Once these bits are set up properly, everything is handled automatically and should not worry the programmer. The process is described at length in a separate application note AN26, *Ultra High-Speed Microcontroller Memory Interfacing*, which may be found at
http://dbserv.maxim-ic.com/appnotes.cfm?appnote_number=954.
With most modern RAM ICs having access times of 100 ns and less (like the one suggested for use in position IC5) no problems are anticipated.

Listing I. Simple RAM-test.

```
; ****** FILE RAMTEST.A51 ************************************
; Register addresses
DPL     EQU   082H      ; DPTR, 2 registers
DPH     EQU   083H      ;
PSW     EQU   0D0H      ; program status word
ACC     EQU   0E0H      ; accumulator
PCON    EQU   087H
TCON    EQU   088H
TMOD    EQU   089H
TL1     EQU   08BH      ; Timer1 registers
TH1     EQU   08DH
SCON    EQU   098H
SBUF    EQU   099H

DPS     EQU   086H      ; Data Pointer Select, SFR in DS89C420

; reload for Timer1, baudrate generator
V24SPD EQU 256-5        ; speed for V24

;
        ORG   00H     ; program runs from 00H (reset vector)
        AJMP  START
        ORG   030H
CNT1    DS    1
        ORG   200H
START   MOV   PSW,#0          ; reset register banks
        MOV   PCON,#080H      ; SMOD=1
        MOV   TMOD,#22H       ; modes are timer
        MOV   TH1,#V24SPD     ; preload value
        MOV   TL1,#V24SPD
        SETB  TCON.6          ; start counter
        MOV   SCON,#052H      ; mode 1 , Enable receiver=10H

  MOV   A,#0
        MOV   DPH,A
        MOV   DPL,A
;The Data Pointer Select bits are set to use both DPTR's
;bit 5: toggle select, auto toggle between DPTR's after MOVX
;bit 4: AID, auto increment DPTR after MOVX
;bit 0: SEL is set to select DPTR1, DPTR0 is already initialised
        MOV   A,DPS
        ORL   A,#00110001B
        MOV   DPS,A
;now initialise DPTR1
        MOV   A, #0
        MOV   DPH,A
        MOV   DPL,A

LOOP    MOV   A,DPH ;#0AAH
        MOVX  @DPTR,A       ; Write using DPTR1
        MOV     A,DPH
        ACALL BYTE
        MOV     A,DPL
        ACALL BYTE
        MOV     A,#':'
        ACALL SND
        MOVX  A,@DPTR       ; Read using DPTR0
        ACALL BYTE
```

The example program writes and reads in external RAM with using not much more than the (expected) instruction MOVX. The length of the program is mostly due to the incorporation of routines for the serial communication between the DS89C420 chip and Loader420 running on the PC. The program may be divided into two parts:

- configuring the serial port on the DS89C420 to enable it to communicate with the PC;
- translating addresses and data into ASCII format so that they make sense when shown on a PC display.

Both pieces of code have been derived from the monitor program written by M. Ohsmann for the previously mentioned MCS51 course of 1992. The routines necessary for the second function may be found towards the end of the listing, starting at the label BYTE.

Serial port configuration

The 89C420 being twelve times faster than its predecessors (at the same clock frequency) has no effect on the speed at which the internal timers are clocked (using standard settings and after a reset). That's because the ÷12 prescaler is used for, among others, downward compatibility with older processors of serial port configuration routines. Although the presence and use of the prescaler is nothing new since souped-up 8051 derivates have appeared, a short explanation may be in order to get it all to work properly.

Using the settings defined from the label START onwards, the relation between quartz crystal frequency, baud rate and Reload value may be expressed as:

Baudrate = crystal frequency / [(12 x 16) x (256–Reload)]

Assuming a crystal frequency of 27 MHz is used, and a baud rate of 28,800 bits/s is required on the DS89C420 (Loader420 setting), you'll find that the Reload value works out at 5 (Listing 1, definition of V24SPD). Obviously, this line is subject to editing if you use a different crystal fre-

quency or a different baud rate in Loader420.

Second datapointer

Possibly a bit too much of a good thing in this simple application program, use is made of the extra 16-bit datapointer inside the DS89C420. Pointer DPTR0, which is available on any MCS51 controller, is used for the address we read from, while DPTR1 points to the address to be written to. Besides, the active datapointer is automatically incremented after the MOVX instruction. Next, the alternate datapointer is selected (Toggle Select bit). The high byte of DPTR1 is written into external RAM.

After address 7FFFH has been written to and read back into the processor, all memory locations in IC5 have been dealt with and the program enters the endless loop called FOREVER.

Conclusion

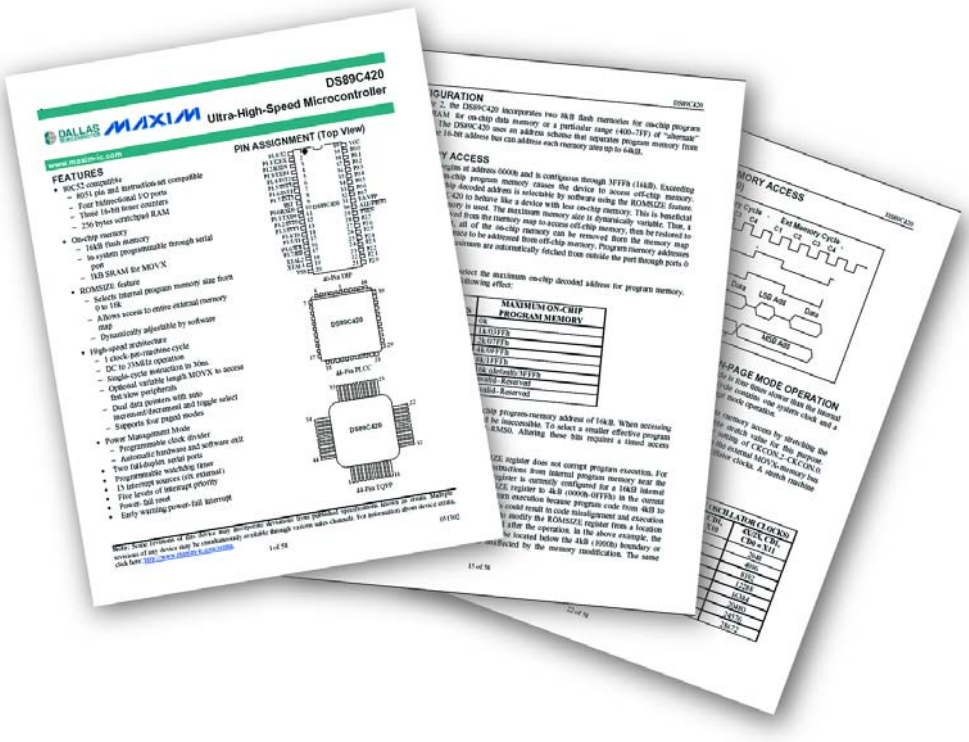
The DS9C420 is (1) a superb microcontroller, (2) 100% compatible with the 'known good' MCS51 controllers we've come to love so much, (3) 12 times faster than any of its predecessors and (4) sports internal Flash memory for easy reprogramming. These and other features make the microcontroller board described here eminently suited to developing new, exciting applications for 8051(-like) controllers.

The number of additional features found on the DS89C420 is such that they can not be discussed in the limited space of these two article instalments. Those of you who want to 'know it all' are advised to obtain the datasheets, user's manual and application notes that may be found on the Maxim/Dallas Semiconductor website.

(020102-2)

```
        ACALL CRLF
        MOV   A, DPH
        CJNE A,#80H,LOOP
FOREVER SJMP FOREVER

BYTE   PUSH  ACC        ; send BYTE hexadecimal , destroy ACC only
       SWAP  A
       LCALL NIBBLE
       POP   ACC
NIBBLE ANL   A,#0FH
       ADD   A,#246
       JC    HEXOUT
       ADD   A,#58
       SJMP  SND
HEXOUT ADD   A,#65
       SJMP  SND
;
CRLF   MOV   A,#13       ; send a CRLF
       LCALL SND
       MOV   A,#10
       LCALL SND
       RET
;
SND    JNB   SCON.1,SND ; must not destroy any reg
       CLR   SCON.1
       MOV   SBUF,A
       CJNE  A,#10,OK2
WAITCR MOV   CNT1,#100  ; wait for slow scrolling terminals
LOP1   MOV   A,#255
LOP2   DJNZ  ACC,LOP2
       DJNZ  CNT1,LOP1
       MOV   A,#10
OK2    RET
        END
```



Literature

DS89C420 datasheet:

http://pdfserv.maxim-ic.com/arpdf/DS89C420.pdf DS89C420

User's Manual:

http://pdfserv.maxim-ic.com/arpdf/Design/89c420_userguide.pdf

Continuity Tester

a component-friendly design

By H. Barteling

A continuity tester is a useful bit of kit for any electronics enthusiast but many of the off-the-shelf models you can buy suffer from one serious disadvantage: they can end up destroying the component under test!

Most silicon semiconductor junctions start to conduct when the forward voltage across the junction exceeds about 0.6 V, but other devices such as Schottky diodes conduct with a much lower forward voltage. If insufficient care has been taken in the design of a continuity tester to reduce the measuring current flow then the component probed can become electrically stressed and may fail. The design presented here produces a measuring voltage of just 5 mV into a test resistance of 10 Ω. In theory the measuring voltage can rise to a maximum of approximately 0.8 V (the battery voltage, VB, less V_{be}, the voltage drop across the base-emitter junction of one transistor) but the measuring current will always be limited to safe value of 0.6 mA to avoid damage to the component under test.

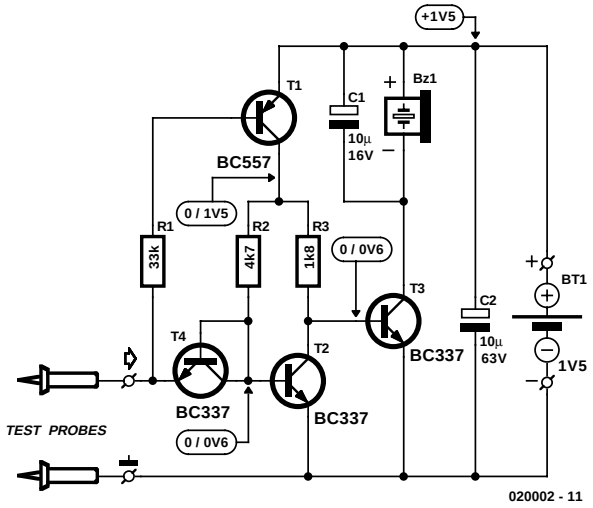


Figure 1. Circuit diagram for the current mirror continuity tester.

COMPONENTS LIST

Resistors:
R1 = 33kΩ
R2 = 4kΩ7
R3 = 1kΩ8

Capacitors:
C1,C2 = 10µF 16V radial

Semiconductors:
T1 = BC557
T2,T3,T4 = BC337 or BC347

Miscellaneous:
BT = AAA 1.5V battery with holder
Bz1 = DC buzzer for 5-6 V
Length of PVC electrical wiring pipe (used as case)
PCB, order code **020002-1** (layout available as a free download from www.elektor-electronics.co.uk)

A single 1.5 V battery powers the complete circuit and when there is no conducting path between the probes so little current flows that an on/off switch is unnecessary. A 5-V piezo buzzer is used to indicate when continuity is detected (operation at 1.5 V was found to be not a problem for the buzzer).

The current mirror

Figure 1 shows the circuit diagram for the continuity tester. Transistors T4 and T2 together with resistors R2 and R3 are connected in a current mirror configuration. The current mirror is a basic building block of electronic design and deserves a

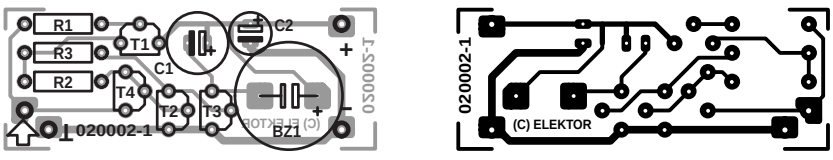


Figure 2. The PCB fits neatly into a small length of plastic conduit.

Reflections on the current-mirror

The current-mirror configuration is a fundamental circuit in electronic design and is a variant of the constant current source circuit. **Figure A** shows a constant current source circuit where the output current I_o is a function of the base-emitter voltage V_{be} of the transistor and the emitter resistor R_e expressed in the formula:

$$I_o = (V_B - V_{be})/R_e$$

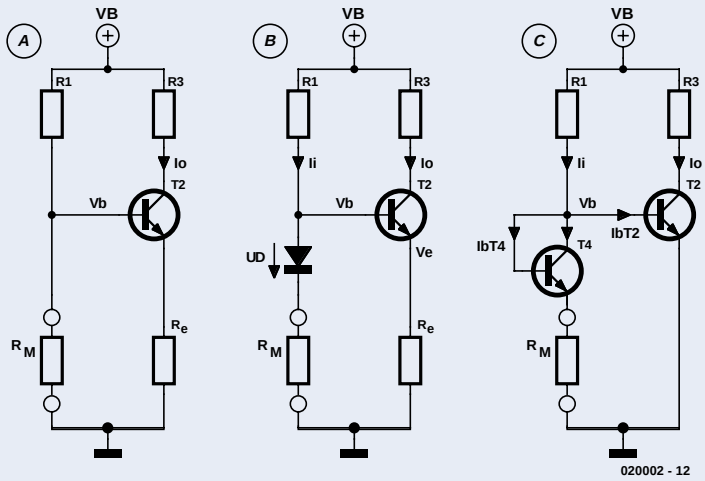
This current source is not too stable because it ignores the fact that the V_{be} for the transistor varies with temperature by a coefficient of –2 mV/°C. If we could somehow vary the base voltage V_b with temperature to compensate exactly for this drift in the transistor characteristics then the current source would be more stable. The standard method of achieving this is to include another p-n junction (it could either be a silicon diode or better still a transistor with its base and collector shorted together) in the voltage generator that produces V_B. This ensures that V_B will track with V_{be} so that their difference ($V_B - V_{be}$) will be constant as the temperature changes. The constant current generator shown in **Figure B** includes this modification and is very close to our final current mirror circuit. The output current I_o is proportional to the input current I_i and the scale factor of this current is given by R_M/R_e so that

$$I_o \approx R_M/R_e \times I_e$$

The current flowing into the base of T2 is so small it can be ignored. The circuit can also function without an emitter resistor (with a loss of accuracy, however). **Figure C** is now the basic circuit of our continuity tester where the input and output current are:

$$I_o \approx I_i$$

R_M is the measured resistance between the probes and will be either an open-circuit (no input current will flow) or a short-circuit, in which case the input current will be governed by the value of R1 and the supply (battery) voltage V_B.



closer look.

Transistors have two p-n junctions, one between the base and emitter and another between the base and collector. If we connect the base and collector leads together then one of these p-n junctions is shorted out and the transistor is left with only one junction… that's right, it has turned into a diode! If this diode were made from one of a pair of matched transistors then its forward conduction characteristics will be identical to the base-emitter characteristics of its transistor twin. When a current is passed through the diode a voltage will be developed across it corresponding to the current. If this voltage is applied to the base of its matched transistor twin, its collector current will be the same (mirrored) as the current through the diode.

To analyse the circuit operation we can firstly assume that a low impedance or short circuit exists between the test probes. In this case transistor T1 will conduct and supply power to the current mirror via R2 and R3. The emitter of T4 will be very close to ground potential so current will flow through R2 and T4 to ground. The voltage produced at the base of T2 will cause the same level of current to pass through the collector of T2. The values of resistors R2 and R3 are not the same so that more current will flow through R3. A portion of this current will pass through T2 but the voltage at the collector of T2 will be sufficient to cause T3 to conduct and switch on the buzzer.

If we now connect a resistor of a few hundred ohms between the probes, the current through this external resistance is supplied partly by T1 (approximately 30 µA) and partly by R2 and T4 (approximately 500 µA). The additional voltage drop across the external resistor will be added to the base voltage at T2 causing it to fully conduct and turn off transistor T3 and the buzzer.

Finally, when the probes are open-circuit, no current will flow into the base of T1 so it will not conduct and the supply to the current mirror circuit and buzzer will be switched off.

In practice

So from the theory, if the circuit is to operate correctly, it is important the two transistors used in the current mirror should be the same. This means that they should have matching characteristics, not only the type of transistor but also the same gain category (usually denoted by A, B or C suffix for the transistor) and if possible from the same production batch (same batch number or production date). The current mirror configuration is so common in high quality amplifier and instrument designs that matched pairs are available supplied in the same package outline. This ensures not only that the properties of both transistors are as alike as possible but also that they are in good thermal contact so that the characteristics of both transistors track together over temperature. Such accuracy is not necessary for our design here. Resistor R3 can be adjusted to ensure that the piezo buzzer only sounds when continuity is detected between the probes.

Construction and testing

Fitting the components to the PCB shown in **Figure 2** should present no great problems; it is only advisable not to fit R3 too closely to

the board initially because it may need to be changed. Once the cables, test probes and battery have been fitted you can proceed with the set-up. The voltage levels shown on the circuit diagram refer to the voltage levels that can be measured when the when the buzzer is respectively silent or buzzing. Fit leads for the test probes and wire the battery to the circuit. Touching the test probes together should cause the buzzer to sound but if this does not happen then experiment by soldering a resistor in parallel with R3 to reduce its value. The combined resistance of R3 should not be less than 1 kΩ.

The buzzer should be silent when the resistance between the probes is greater than about 10 Ω. Connect a resistor of this value between the test probes and if the buzzer sounds then unsolder one leg of R3 and add an additional resistor in series with it. The final value of R3 is not too important as long as it is less than about 22 kΩ. Don't be in too much of a hurry to test the unit after soldering because some of the components are sensitive to heat and correct circuit operation is not guaranteed until the components have returned to room temperature.

If you have no success by changing the value of R3 then try swapping transistors T2 and T4, if this does not do the trick then try exchanging these transistors for other examples that you may have.

When you are happy that the tester is working correctly, pop the PCB and battery into a small length of plastic tubing (plastic conduit) and fit a cork to both ends to secure the PCB, not forgetting to make an opening to allow the leads to pass outside to the probes.

(020002-1)

Firmware Update for EPROM Emulator

add MCS51 compatibility to our 27C256 Emulator

By G. van Breugel

Those of you who happily use the 'old' MCS51 assembler will be pleased to know that its dedicated file format can now be accepted by the popular 27C256 Emulator published in the January 2001 issue of Elektor. All you need is the free software update described in this article.

An EPROM emulator is likely to remain a useful tools for quite some time to come, because it allows users to debug and correct programs intended to run from EPROM, without going through the tedious and time consuming process of erasing and reprogramming an EPROM after any small change to the target program. The emulator's alternative to this drudgery is simple: edit the source code file, assemble, download and then restart the target system. The January 2001 issue of *Elektor Electronics* contains an article describing how to build such an EPROM emulator yourself. Although primarily intended for the popular 27C256, this emulator will also handle the 27C64, 27C128 and, if necessary, the 27C512.

This EPROM emulator was a 'hit' straight away, and has been built by

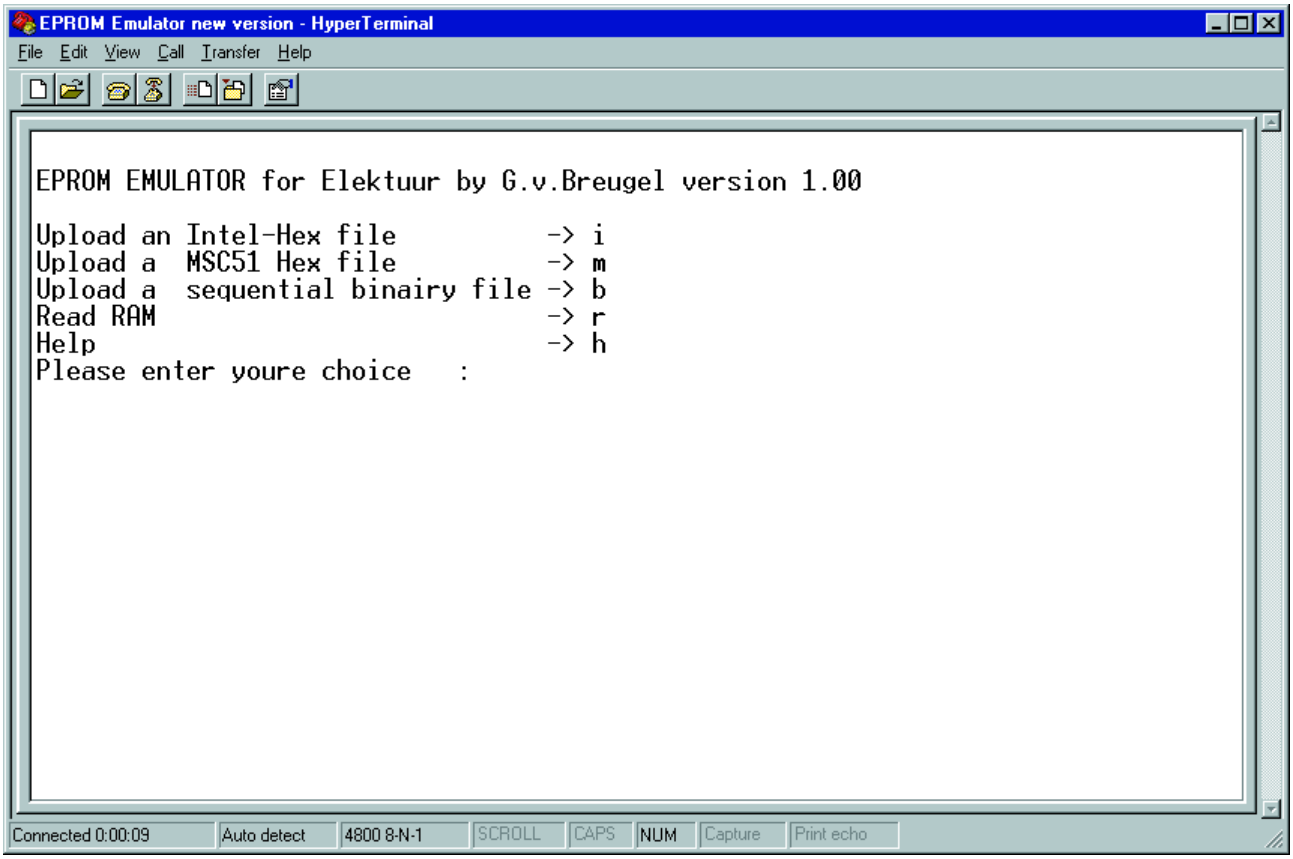


Figure 1. The updated firmware will present a menu with four options.

many readers. 'Old hands' at the art of software development, however, found the design lacking in just one aspect: it was only suitable for the industry standard Intel Hex file format. You may know, there are several other, equally useful file formats commonly used to program EPROMs. One such format is 'MCS51' produced by the assembler wit the same name. Its main disadvantage is that it is only supported by the EMON51 monitor program which belongs with the incredibly popular **80C32 Single-Board Computer** published in the May 1991 issue.

Readers' feedback indicates that many of you are still happily using the above mentioned MCS51 assembler and would like to see its rather special file format implemented on the 27C256 Emulator. The author has spent a few hours of his spare time to see how this could be done. The result of his efforts is an updated version of the control software for the EPROM emulator. Fortunately, hardware modifications were not necessary, so you can leave your soldering iron switched off.

What's new?

After launching the new software you'll be shown a menu with four options (**Figure 1**):

– Upload an Intel Hex file
– Upload an MCS51 hex file
– Upload a sequential binary file
– Read RAM

A small help function is also available. The above options should be self-explanatory. Once you've selected the desired file format, the object code file may be sent to the EPROM Emulator in ASCII.

In addition to Intel Hex, the new software also supports binary files and Elektor's MCS51 format.

To reap the benefits of the updated emulator firmware, you can program your own processor using the Hex file 024107-11 which can be downloaded free of charge from our website www.elektor-electronics.co.uk (Free Downloads, September 2002). Those of you without access to the Internet or a suitable programmer will need to buy a ready-programmed controller which is available under number **024107-41** from Readers Services.

As soon as the microcontroller with the new firmware has been installed in the circuit again, HyperTerminal may be run, just as with eh previous version (HyperTerminal comes as part of your Windows operating system). Alternatively, you may use any other terminal emulation program. The communication parameters remain at **4800,N,8,1**.

After resetting the 27C256 Emulator, the menu that allows te Emulator to be controlled should appear on your monitor.

For the sake of completeness we should repeat that the 'Elektor' Hex format is marked 'MCS51'.

Conclusion

The update is a breeze to install and should work spot on. In the (rare) case of problems, these may be solved quickly and efficiently with the aid of the Help function.

(024107-1)

Software for Old Computers

useful stuff for vintage PCs

By Harry Baggen

A computer that's too slow for today's programs may be found in many homes and certainly in any office. Once the pride of their owners, the old number cruncher is often abandoned and left to gather dust in the attic. Throwing an old PC away seems wasteful, but what can you do with it?

Today's computer users are spoilt by processors operating at gigahertz frequencies, hard disks offering gigabytes of storage capacity and high resolution displays capable of showing millions of colours. And yet, it is only a few years ago when we were all perfectly happy with what can be summarized as 'a lot less'.

Unfortunately, but also inevitably, software makers closely follow the latest hardware developments, producing programs that will only run on modern, fast PCs sporting lots of RAM memory and hard disk capacity.

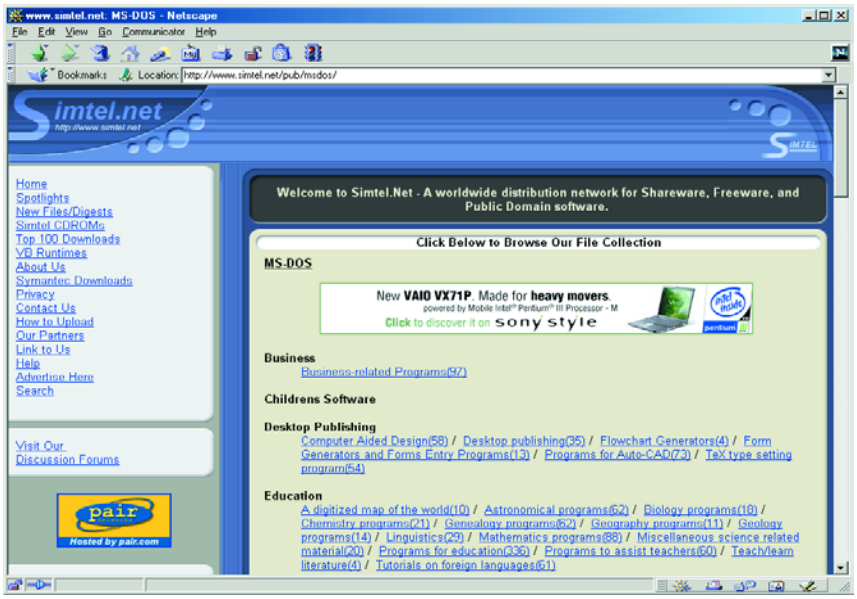
Those of you who would like to fetch the old PC from the loft and start using it again for, say, low-end word processing or the odd spreadsheet are sure to face some quite unexpected problems. To begin with, you need to resort to programs that are suitable for such a slow beast, and that rules out Windows and Word versions using the extensive graphics user interface. The original programs from the old days are often lost, leaving you stumped for software!

Fortunately, the Internet has a lot of vintage software available that is perfectly suited to older computers, giving them a new lease of life!

Free DOS

Older computers, particularly the ones from the 'command prompt' age (mid-1980's) should not be burdened with even de simplest versions of Windows. In practice, they are best loaded with nothing more than DOS for an operating system, of which the advantages include an incredibly fast booting process, almost direct access to the hardware and the reassuring thought of being able to write one's own programs in Q-BASIC or GW-BASIC for the occasional measurement and control application.

Several DOS versions may be found on the Internet. Some are remnants of the old DOS age, while others have been written as new applications by enthusiastic PC users. **DR-DOS** [1] may be familiar with many of our readers as a once fierce competitor of MS-DOS. DR-DOS, the PC operating system originally developed by Digital Research (hence 'DR'), is currently in its seventh release. It is supplied free of charge for private use only. An example of a freshly developed DOS is called **FreeDOS** [2]. Apparently it has a large group of followers (pioneered by Jim Hall) who are constantly refining the program, just as with today's Linux. By the way, there is, of course, no objection against running Linux on an old PC, provided you omit the graphics shell.



If you think that these DOS versions are just relics from the pre-Internet age, you may be wrong. Most of these operating systems are updated from time to time and closely follow all the latest trends in PC technology. For example, they will happily support the latest crazes like FAT32, NTFS and USB equipment.

Programs galore

There's an abundance of DOS programs available on various websites we came across while researching this month's *Electronics Online* subject. Usually, on these sites you'll find collections of old shareware and freeware, which, nonetheless, contain a fair amount of really useful programs.

Simtel is a network (or, if you like, 'ring') that's been busy collecting all kinds of software for many years. Their website contains a section devoted to DOS programs only [3], all neatly arranged in categories. **Free Software for DOS** [4] contains about 700 free programs for DOS. Here, too, the website makers have taken the trouble to cluster programs into groups with relevant names. There is even a search engine available so any program can be found easily. In the **Garbo PC Archives** [5] the available DOS programs have been sub-compiled into about 150 'class' pages. You may spend quite some time browsing this area because a lot has been brought together here. The **File Library** [6] also has a vast collection of DOS programs. There are 9 main categories, each divided into sub-categories. At **Interesting DOS Programs** [7] it's news rather than quantity. Here, the latest DOS programs are presented, complete with the associated links. If, after visiting these websites, you are still hungry for more DOS programs, have a look at **DOS-Only** [8] — the name pretty much says it all.

Programming languages

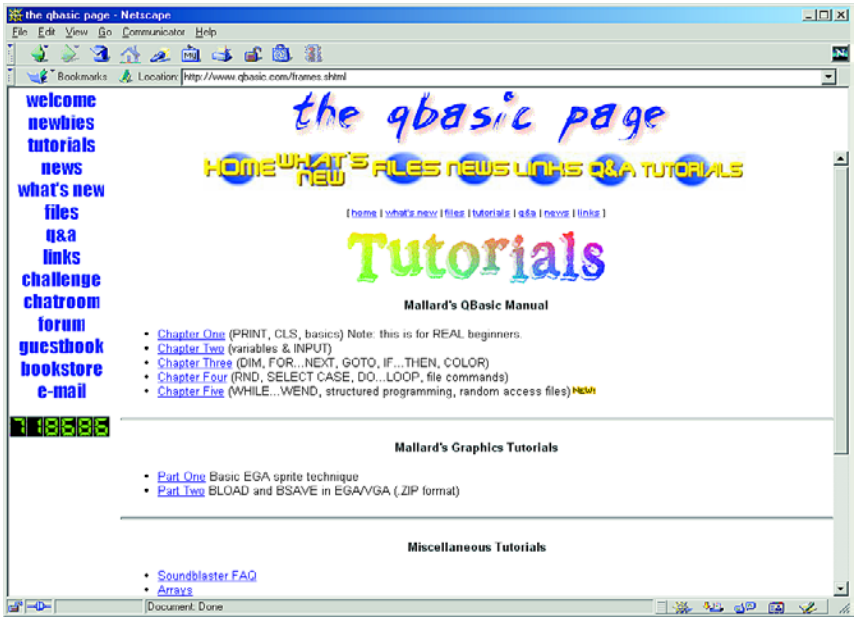
In the old days, Microsoft supplied the BASIC interpreter GW-BASIC as part of the MS-DOS package. This was later superseded by Q-BASIC. Most of today's Windows releases still contain a version of Q-BASIC. If you don't believe us, run a 'Find' on your Windows CD-ROM!

Lots of people are still actively using Q-BASIC, and they are on the Internet, too. The **Qbasic Page** [9] is a good starting point for those of you interested in taking this up. The ultimate overview of basic websites is, however, **Connors House: QBasic links** [10], where no fewer than 570 links have been brought together, all covering the subject in one way or another.

Turbo Pascal is another popular programming language from the days of DOS. At the **Turbo Pascal Programmer's Page** [11] we found a useful overview of compilers, utilities and programs related to this once immensely popular compiler.

If you find these higher programming languages interesting but feel the need to brush up your knowledge of them, simply visit the **QuickBASIC/Turbo Pascal Info Base** [12] for step by step guidance to writing your own programs.

(025053-1)



Internet Addresses

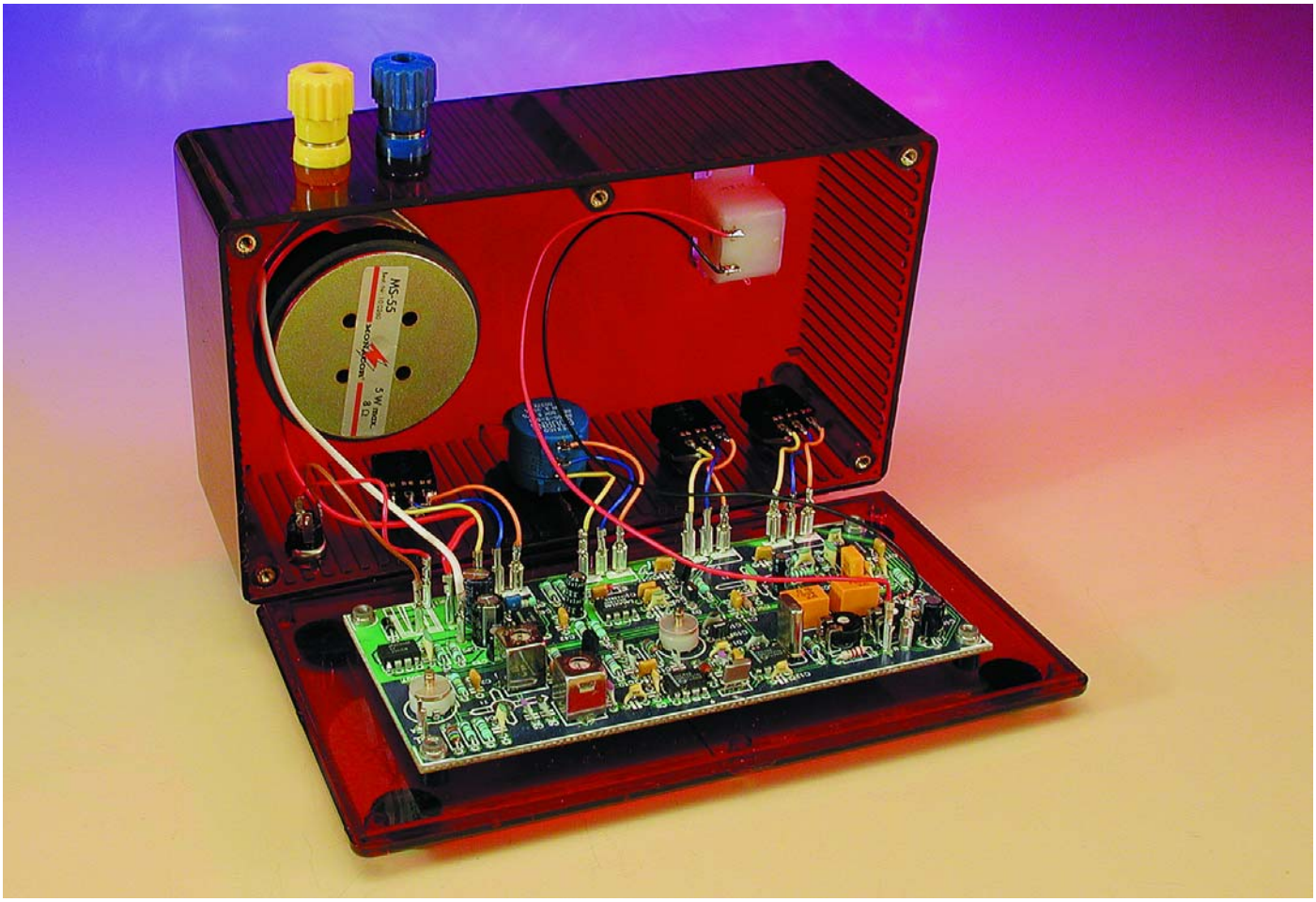
[1] www.drdos.com/
www.drdos.net/
www.drdos.org/
[2] www.freedos.org/
[3] www.simtel.net/pub/msdos/
[4] http://members.cox.net/dos/
[5] http://garbo.uwasa.fi/pc/
[6] www.filelibrary.com/categories/DOS.shtml3
[7] www.opus.co.tt/dave/index.htm
[8] http://dosonly.net/
[9] www.qbasic.com
[10] http://members.tripod.com/~connorr/qb.htm
[11] www.devq.net/pascal/
[12] http://infobase.hypermart.net/

20-Metres Band Receiver

an unusual double-conversion superhet

Design by G. Baars g_baars@hotmail.com

This SSB/CW receiver for the popular 20-m amateur radio band boasts a number of interesting features. Designed as a double-conversion superheterodyne with a direct-conversion demodulator, it offers adjustable IF bandwidth and is remarkably simple to adjust. What's more, the single-board concept makes the receiver easy to build, even by relative newcomers to the radio hobby.



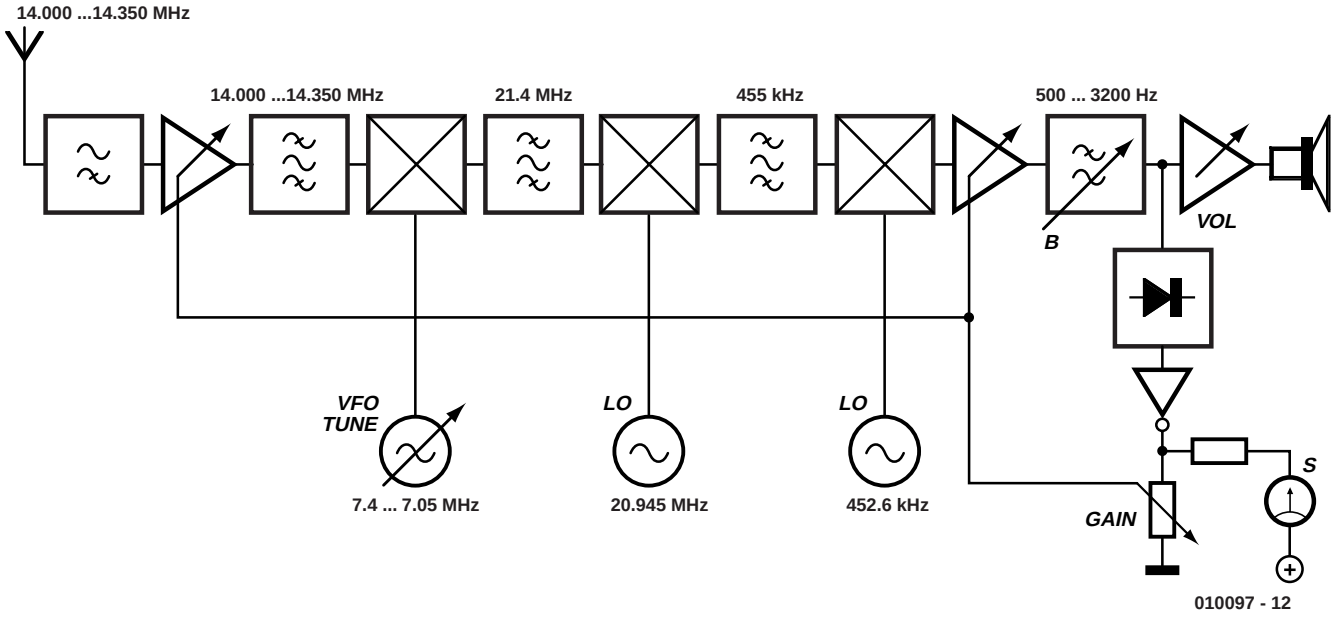


Figure 1. Block diagram of the 20-m band receiver. You are looking at a superheterodyne with three local oscillators.

A number of design aspects should be considered when blueprinting a 20-m band SSB receiver that is going to be reproducible. Besides the obvious aspect "receiver performance" there are other essential considerations like availability and price of the components used. Such considerations immediately rule out the use of, for example, a professional SSB filter which would cost you £30 or more. Furthermore, the concept of the receiver should warrant easy adjustment without special test equipment, allowing those without access to a frequency meter and an oscilloscope to build the project with a fair chance of success.

The receiver described in this article complies with the above design criteria while also offering an excellent price/performance ratio. Let's shortlist some of its features:

– Frequency range 14 – 14.350 MHz (20-m amateur radio band)
– Suitable for SSB, CW, RTTY, fax, SSTV
– Double-conversion superhet with direct-conversion (DC) demodulator
– Sensitivity approx 0.5 µV
– Selectivity adjustable between 0.1 and 2.8 kHz
– Adjustable gain (range 45 dB)
– Image rejection >65 dB
– Audio power up to 0.5 W into 8 Ω
– Max. power consumption 1.5 watts

To this we should add that the receiver is not at all difficult to adjust, and there are no special requirements with regard to the antenna.

Block diagram

The general design of the receiver is best explained with reference to the block diagram in **Figure 1**.

The antenna (RF) signal (14.000 MHz to 14.350 MHz) is taken through a high-pass filter and then amplified. Then follows a filter to suppress image signals, out-of-band and other spurious components. The RF signal is then mixed with the VFO (local oscillator, LO1) which is used to tune the receiver.

With receiver stability in mind, a high VFO frequency is best avoided. That is why a first IF (intermediate frequency) of 21.4 MHz is used in this receiver. This requires a VFO range of 'only' 7.4 to 7.05 MHz. As an additional advantage, image signals then occur at 28.8 to 28.45 MHz, which is sufficiently far from the desired input frequency to allow the use of a relatively simple filter.

The first mixer is followed by a 21.4-MHz bandpass whose main function is to get rid of the unwanted product which is generated in the second mixer when mixing down to 455 kHz with the aid of LO2. This unwanted product occurs at 20.490 MHz (20.945 – 0.455) and is rigorously suppressed by a steep bandpass filter.

Behind the second mixer we have a 455-kHz IF signal available. Normally, this would be amplified and then applied to a product detector. Unfortunately, most (affordable) 455-kHz filters have a bandwidth which is rather too large for our application, even when two such filters are cascaded as in this design. A alternative was found in mixing the 455-kHz IF signal down to AF using a third, frequency-adjustable, oscillator (LO3). This method of mixing the IF signal with a LO3 signal that results in an AF 'beat' signal enables the receiver bandwidth to be effectively determined by the bandwidth of the AF amplifier. Next, an adjustable receiver bandwidth is obtained simply by incorporating a variable filter in the audio section.

The method of using a mixer/LO combination to mix RF down to AF in one go is generally known as 'direct conversion'. In most direct conversion receivers, 'RF' means 'antenna frequency' so the image frequency is inevitably just 3 kHz or so removed from the desired frequency — and this is usually taken for granted. In our design, however, 'RF' means 'second IF' and the 455-kHz filter between the second and third mixer is sufficiently narrow to guarantee adequate suppression of the image signal. In fact, the image rejection may be increased by making the LO3 frequency a bit lower than usual. The underlying principle is as follows:

The –3 dB roll-off points of the 455-kHz filter are at 453 kHz and 456.4 kHz. The LO3 frequency is just below that of the down-converted IF signal, and the image signal is further down, at equal distance. Now, the ingenious bit is to make the lowest frequency of the received signal coincide with the lower limit of the filter. In this way, the distance between the lowest passband frequency and the image frequency is maximized. Consequently, image

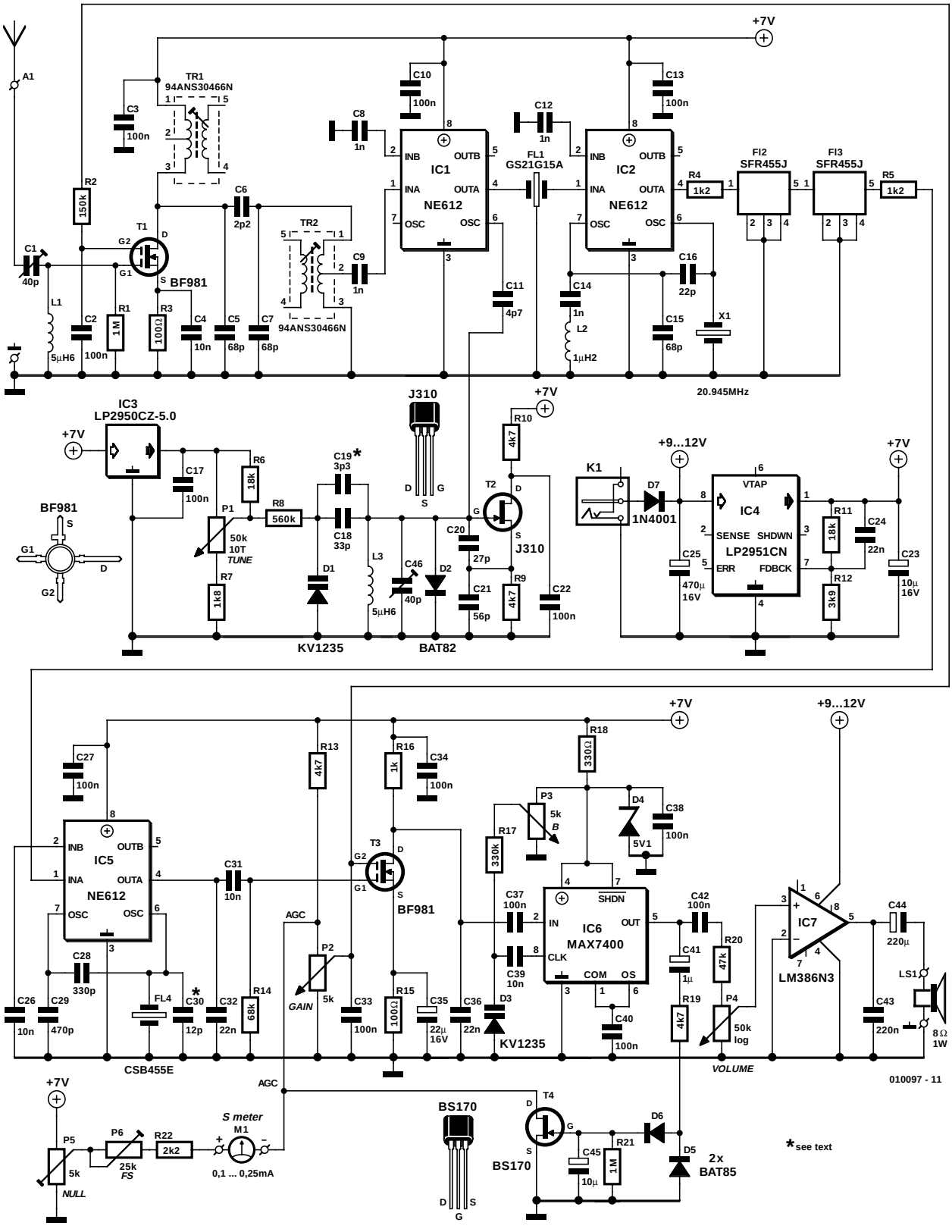


Figure 2. By using a mix of discrete components and integrated circuits, the author was able to keep the circuit relatively simple.

rejection is maximized.

In order to make this work, a LO3 frequency of 452.6 kHz has to be used (453 kHz – 400 Hz). This causes the image product to occur at a distance of at least 800 Hz from the filter passband where it is adequately suppressed. It could be argued that the present receiver is a triple-conversion superheterodyne but the designation 'double-conversion superhet with a direct-conversion demodulator' would seem equally legitimate.

Returning to the block diagram of the receiver, we have arrived at the audio section. This begins with an adjustable-gain amplifier whose main function is to suppress high-level interference. Next, there's a low-pass filter which determines the final selectivity of the receiver. The roll-off is adjustable between 500 Hz and 3200 Hz, which, after subtraction of the above mentioned lower limit of 400 Hz, results in a selectivity range of 100 Hz to 2.8 kHz. Here, a sharp eight-order filter is used which achieves more than 40 dB suppression at 1.25 times the roll-off frequency. As a matter of course, this filter is also a great when it comes to suppressing noise and other interference.

That almost completes our tour of the block diagram. What remains to be mentioned is the receiver's AF power amplifier and the AGC (automatic gain control). The first is a standard design throughout and the latter serves to reduce the gain of the two amplifier stages as a function of received signal strength, ensuring a reasonably steady audio level even if the received signal strength varies considerably.

VFO stability

For an SSB (single-sideband) receiver, a stable VFO is an absolute requirement. A number of practical tests indicated that the Colpitts oscillator used here requires about 3.5 minutes to warm up, and then exhibits a frequency drift of no more than 25 Hz within 5 minutes. Such stability is ample for SSB and CW reception, and obviates constant retuning.

The actual frequency stability of the VFO can not be measured until about 5 hours after the receiver has been built up, because the PCB requires quite some time to cool down and stabilize mechanically. VFO stability may be improved by purposely ageing the board. Package the PCB moist-free in a plastic (freezer) bag and leave it in the deep-freeze compartment of your refrigerator for a couple of hours. Take the bag out of the freezer and allow the PCB (still in the bag) to warm up to room temperature for an hour or so. Then put the bag on a (hot) central heating radiator. Allow it to cool down again to room temperature and then put the bag in the freezer again. Three such rounds will remove all mechanical stresses from the PCB and guarantee optimum VFO stability.

Circuit diagram — RF sections

Figure 2 shows the practical realisation of the block diagram.

Using high-pass filter C1-L1 d the input signal is cleaned of any unwanted components such as block-busting signals from nearby medium-wave transmitters. An additional function is antenna matching.

The RF input stage comprises a dual-gate MOSFET (T1) which couples high gain to a low noise figure. By making the voltage at gate 2 variable, the gain of T1 is adjustable across a range of about 20 dB. The gate is connected to the AGC circuit via resistor R2.

The critically coupled bandpass filter consisting of Tr1, Tr2, C5, C6 and C7 has a virtually flat response across a frequency range of about 500 kHz. The filter also suppresses image signals at 28.800 – 28.450 MHz by about 65 dB.

The first mixer is an NE612 (IC1) in a configuration that results in about 17 dB conversion gain.

The VFO is a Colpitts oscillator built around a FET type J310 (T2). The VFO is tuned by varicap D1. The supply voltage for the varicap tuning has an additional 5-V regulator of the low-drop type (IC3). Tuning is accomplished with a ten-turn pot, P1.

The first mixer is followed by a 21.4-MHz ceramic filter (FL1). Then follows another NE612 (IC2) which handles the frequency conversion to 455 kHz. The associated LO employs the oscillator components on board the NE612. Here, the oscillator is configured in overtone mode with quartz crystal X1 acting as the frequency-determining element.

The second mixer supplies the sum as well as the difference of the input frequency (21.4 MHz) and the LO frequency (20.94 MHz). Because we only want the difference frequency, the mixer is followed by a 455-kHz bandpas built from two ceramic filters type SFR455J (FL2 and FL3) connected in series ('cascaded'). The bandwidth of this combination amounts to about 3.4 kHz.

The third mixer is another NE612 (IC5) whose internal oscillator is tuned to about 452.6 kHz with the aid of ceramic resonator FL4.

Circuit diagram — AF section

Any RF residue at the output of IC5 is removed by C32. Next, the AF signal is amplified by T3. Here again, a BF981 is used because it allows adjustable gain to be implemented in a simple way. Control P2 allows the gain of T1 and T3 to be adjusted. Besides, the gain of the two MOSFETs in the receiver is automatically reduced (at large RF signal levels) by the AGC circuit built around T4, D5 and D6. The AGC voltage is derived from the audio signal at the output of IC6. In general, SSB reception will be optimum at minimum gain (turn down P2 as far as possible). This way you will reduce the interference (and possible blocking) caused by nearby transmitters and atmospheric noise.

Next, we will discuss one of the most important elements of the receiver: the variable low-pass filter. Here, a MAX7400 (IC6) is used. The MAX7400 is an integrated 8^{th}-order elliptical filter IC using switched capacitors. The roll-off frequency (hence the receiver selectivity) is adjusted by varying the voltage at pin 8. That is accomplished using varicap diode D3 and a direct voltage taken from from the wiper of P3. In this way, P3 enables the optimum receiver bandwidth to be set between 100 Hz and about 2.8 kHz to match any mode and modulation form you want to listen to (or decode on your PC).

Two more circuit parts are left to discuss: the S meter and the AF power amplifier.

The S (signal strength) meter (M1) is controlled by the AGC voltage at the drain of T4. The moving-coil instrument will only start to deflect at signals of a couple of μV, so it will not respond to extremely weak signals. Full-scale deflection (FSD) is reached at an RF sig-

nal level of about 200 μV. Meter zero and meter FSD may be adjusted on P5 and P6 respectively.

The AF amplifier is built around the familiar LM386N3 integrated circuit which supplies up to 0.7 watts of audio power. Because the LM386 boasts a supply ripple rejection of 50 dB, it may be powered from the receiver's unregulated supply voltage. If desired, headphones may be used instead of a small loudspeaker. Provided P4 is turned up a bit, the AF signal level will be sufficient to drive a PC's serial port line, directly or via an interface, for the decoding of facsimile, SSTV or CW signals on a computer running JVFAX or Hamcomm.

Power Supply

Depending on the audio volume set on P4, the receiver will consume 30 to 150 mA at a supply voltage of 9 volts. Because the circuit has an on-board voltage regulator (IC4), a low-cost mains adaptor (a.k.a. battery eliminator) with an output voltage of 9 to 12 VDC is perfect for the job — and safe, too! Even the smallest of these adaptors are usually capable of supplying 300 mA and more. Of course, you are welcome to build your own dedicated power supply for the receiver — simply use a 9-volt PCB mount transformer, a bridge rectifier and a reservoir capacitor of 470 μF/25 V. Do observe the relevant electrical safety precautions when building such a supply!

Construction

Elektor labs have once again succeeded in designing a beautiful PCB, see the copper track layout and component mounting plan in **Figure 3**. As you can see, the entire receiver, including all controls and the supply connection, is accommodated on the board. What's more, despite it being

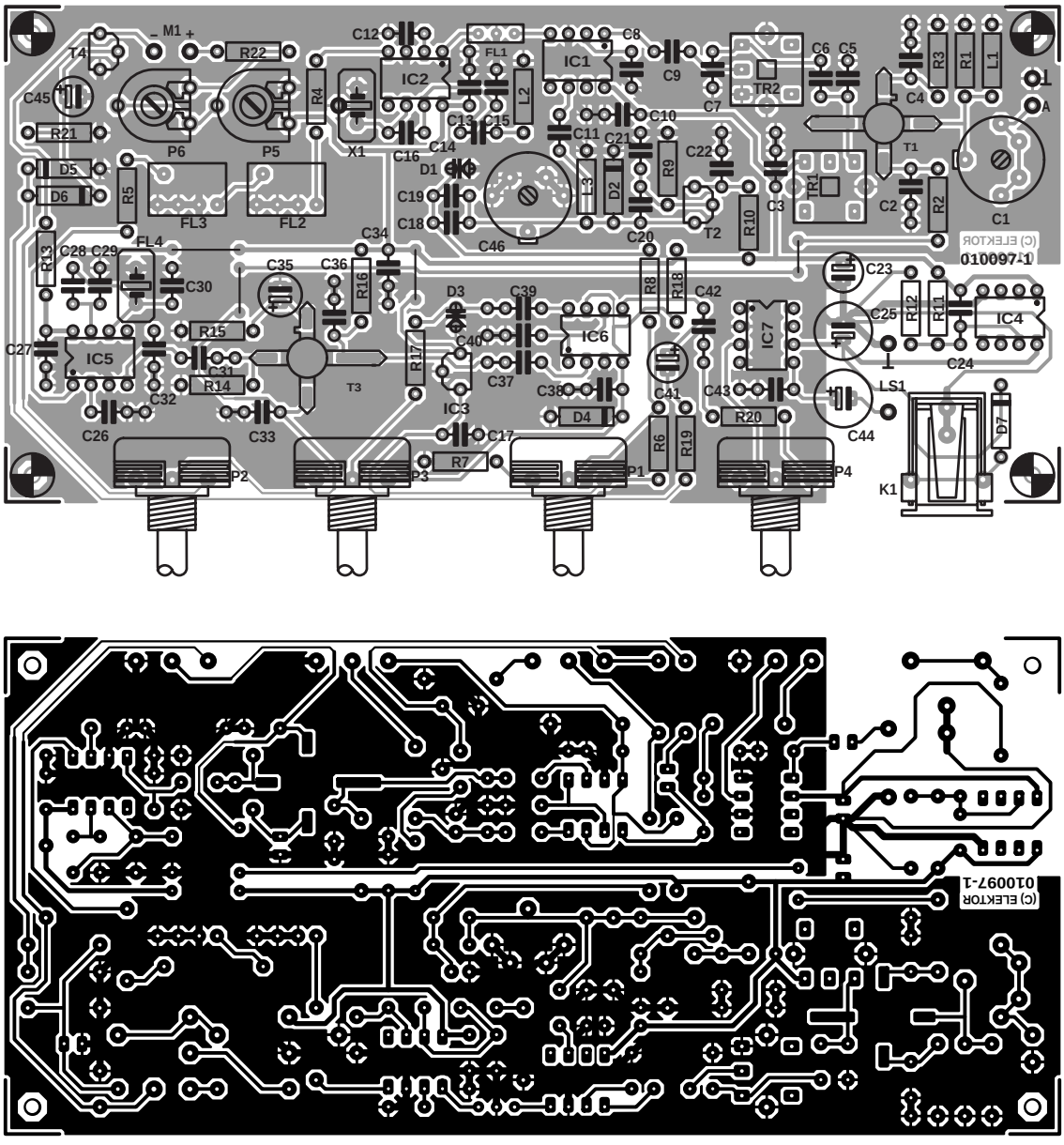


Figure 3. Copper track layout and component mounting plan for the PCB designed for the 2-m receiver (board available ready-made through Readers Services).

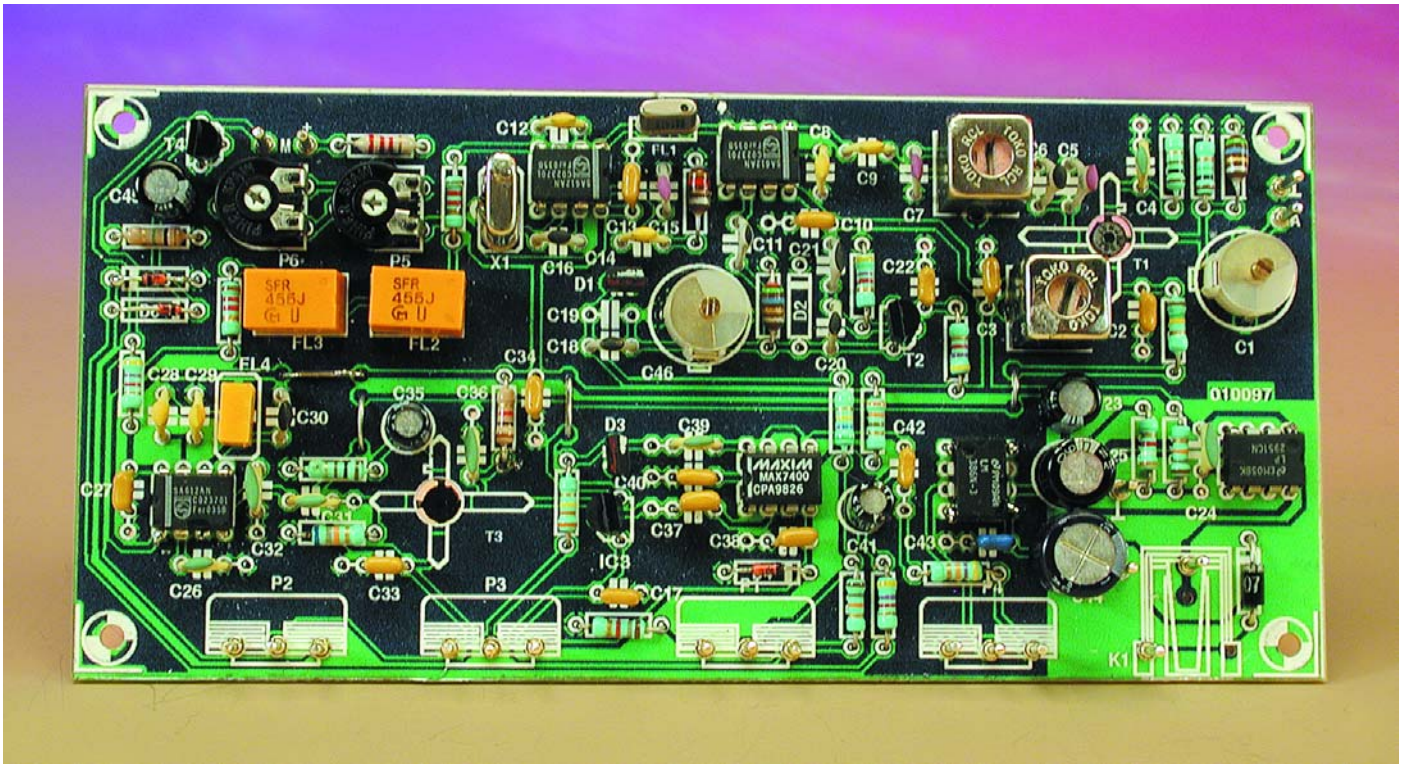


Figure 4. Prototype of the receiver built and tested in the Elektor labs. Note that MOSFETsT1 and T3 are mounted at the underside of the board!

single sided, the PCB has a minimum number of wire links — just four!

The connection points for the antenna, loudspeaker and moving-coil meter M1 are clearly labelled on the board, so little can go wrong in that department. Of course, the four potentiometers may also be mounted off the board and

COMPONENTS LIST

Resistors:
R1,R21 = 1MΩ
R2 = 150kΩ
R3,R15 = 100Ω
R4,R5 = 1kΩ2
R6,R11 = 18kΩ
R7 = 1kΩ8
R8 = 560kΩ
R9,R10,R13,R19 = 4kΩ7
R12 = 3kΩ9
R14 = 68kΩ
R16 = 1kΩ
R17 = 330kΩ
R18 = 330Ω
R20 = 47kΩ
R22 = 2kΩ2
P1 = 50kΩ 10-turn cermet
P2,P3 = 5kΩ linear
P4 = 50kΩ
P5 = 5kΩ preset
P6 = 25kΩ preset

Capacitors:
C1,C46 = 40pF trimmer
C2,C3,C10,C13,C17,C22,C27,C33,C34, C37,C38,C40,C42 = 100nF
C4,C26,C31,C39 = 10nF
C5,C7,C15 = 68pF
C6 = 2pF2
C8,C9,C12,C14 = 1nF
C11 = 4pF7
C16 = 22pF
C18 = 33pF
C19 = 3pF3 *
C20 = 27pF
C21 = 56pF
C23 = 10µF 16V radial
C24,C32,C36 = 22nF
C25 = 470µF 16V radial
C28 = 330pF
C29 = 470pF
C30 = 12pF
C35 = 22µF 16V radial
C41 = 1µF 16V radial
C43 = 220nF
C44 = 220µF 16V radial
C45 = 10µF 16 V radial

Semiconductors:
D1,D3 = KV1235 (1 V/500 pF, 8 V/25 pF)
D2 = not used (do not mount)
D4 = zener diode 5V1, 400mW
D5,D6 = BAT85
D7 = 1N4001
IC1,IC2,IC5 = NE612 or SA612AN
IC3 = LP2950CZ-5.0 (possibly 78L05)
IC4 = LP2951CN
IC6 = MAX7400
IC7 = LM386N3
T1,T3 = BF981
T2 = J310
T4 = BS170

Miscellaneous:
FL1 = GS21G15A or GS21G15B
FL4 = CSB455E
Fl2,Fl3 = SFR455J
L1,L3 = 5µH6 miniature choke
L2 = 1µH2 miniature choke
LS1 = 8 W/1W loudspeaker
M1 = 0.1 – 0.25 mA moving-coil meter
Tr1,Tr2 = 94AES30466N or 94ANS30466N
X1 = 20.945MHz quartz crystal (parallel resonance, 3rd overtone, 2x20pF)
K1 = mains adaptor socket, PCB mount
PCB, order code **010097-1** (see Readers Services page)

connected to it by short wires. In that case, volume pot P4 requires screened cable.

Although RF circuits are reputedly a tad difficult to build, that does not apply to this receiver. Anyone with some experience in building electronic circuits will not have the slightest trouble with this particular board. As a notable detail, the metal case of quartz crystal X1 has to be connected to ground by a short wire.

Unusually (at least to the uninitiated), MOSFETs T1 and T3 have to be mounted at the solder side of the board.

Although a component labelled D2 is included in the circuit diagram, it is not actually used. In our prototype of the receiver, the VFO stability was better without D2.

If you stick to the parts list and the component overlay, take your time and solder carefully, there is little that stands in the way of success.

A good way to check if your construction skills are any good is to compare your receiver board with the prototype shown **Figure 4**.

You are free to choose any suitable enclosure for the receiver. Although a metal case is preferred for this kind of circuit, we found that our prototype gave good performance and little sign of interference or unwanted radiation when fitted in an ABS case.

Adjustment

Do not yet connect an antenna. Turn up the volume pot a little. Turn tuning pot P1 7.5 turns starting from the lowest setting. Also starting from the minimum setting, adjust VFO trimmer C46 until a whistle is heard.

Connect a not too long antenna to the receiver input and adjust input trimmer C1 to maximum capacitance (full mesh). Turn the cores in Tr1 and Tr2 fully down, and then back up one turn. Next, alternately adjust the cores for highest noise output. Finally, peak C1 for maximum noise output.

The S-meter circuit has to be adjusted without an antenna connected to the receiver. Turn P6 to mid-travel and then adjust P5 for zero deflection on the meter. Use a short piece of wire to temporarily take the drain of T4 to ground, then adjust P6 for minimum meter deflection. Remove the short circuit. If the meter does not return to zero, repeat the above steps a few times.

Two more details need to be discussed which we feel are relevant, although strictly speaking they do not really belong to the adjustment procedure.

The circuit diagram shows that board space has been reserved for an additional capacitor C19 in parallel with C18. In most cases, C19 will not be required. However, those of you in possession of, or with access to, an accurate frequency meter or an RF signal generator may care to 'tweak' the receiver's tuning range by adding a padding capacitor of which the value is determined by trial an error.

Similar considerations apply to padding capacitor C30 in parallel with FL4. This capacitor may be used to cope with the tolerance on the oscillation frequency of the ceramic resonator. Only if the receiver seems to sound a bit dull (lack of treble), C30 may be given a slightly lower value.

Practical use

After switching on the receiver, allow the VFO to warm up for a couple of minutes. Initially, set the bandwidth and gain controls (P3 and P2 respectively) to maximum.

When tuning to a station using P1, you'll find that you need to set the 'pitch' first. It is convenient at that point to reduce the gain somewhat to a level at which the signal sounds clear. Next, carefully turn the tuning control again for best reception.

The next step is to reduce the bandwidth until the received signal just remains undistorted. In this way, noise and interference (QRM) may be considerably reduced. When listening to a CW station, the bandwidth may be vastly reduced. If, for instance, the dots and spaces are reproduced at 700 Hz or so, the receiver bandwidth may be reduced to as little as 300 Hz. Having marked the best setting of P3 for various modes you will be able to get optimum results in no time at all when tuning across the 20-m band.

Antenna

At times, signals in the 20-m amateur radio band may be so strong that a 10-cm wire is enough for reasonable reception. However, for 'normal' listening you'll find that the receiver performs admirably with a wire of about two metres length connected to the antenna input. Even without special propagation conditions, this (indoor) wire should pick up stations within a range of 3,000 kilometres or so. Under favourable weather conditions, the 20-m band will burst open and swarm with activity, and you'll be surprised how much can be picked up with a truly modest antenna.

Best results are obtained when using a long wire antenna with a length of about 10 metres, hung up outside, well removed from sources of electrical interference. On the down side, such an antenna may easily overload the receiver to the extent that QRM becomes so high the AGC starts to work when no signal is heard! In such cases, a variable attenuator may have to be inserted in the antenna line.

Finally, note that C1 requires readjustment after any change to the antenna or the length of it.

(010097-1)