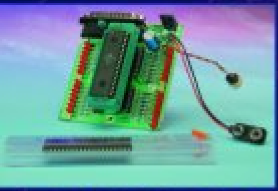
OCTOBER 2002
£3.20

ELEKTOR ELECTRONICS

THE ELECTRONICS & COMPUTER MAGAZINE

www.elektor-electronics.co.uk



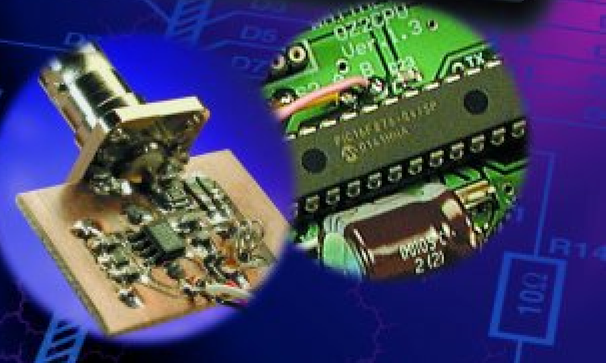
AT90S8535 Programmer

Sound Pressure Level Meter

1-GHz RF WATTMETER

with LCD readout

- *Car Diagnosis Systems*
- *USB Driver Programming*
- *Rugby MSF Alarm Clock*
- *Acoustic Voltmeter*

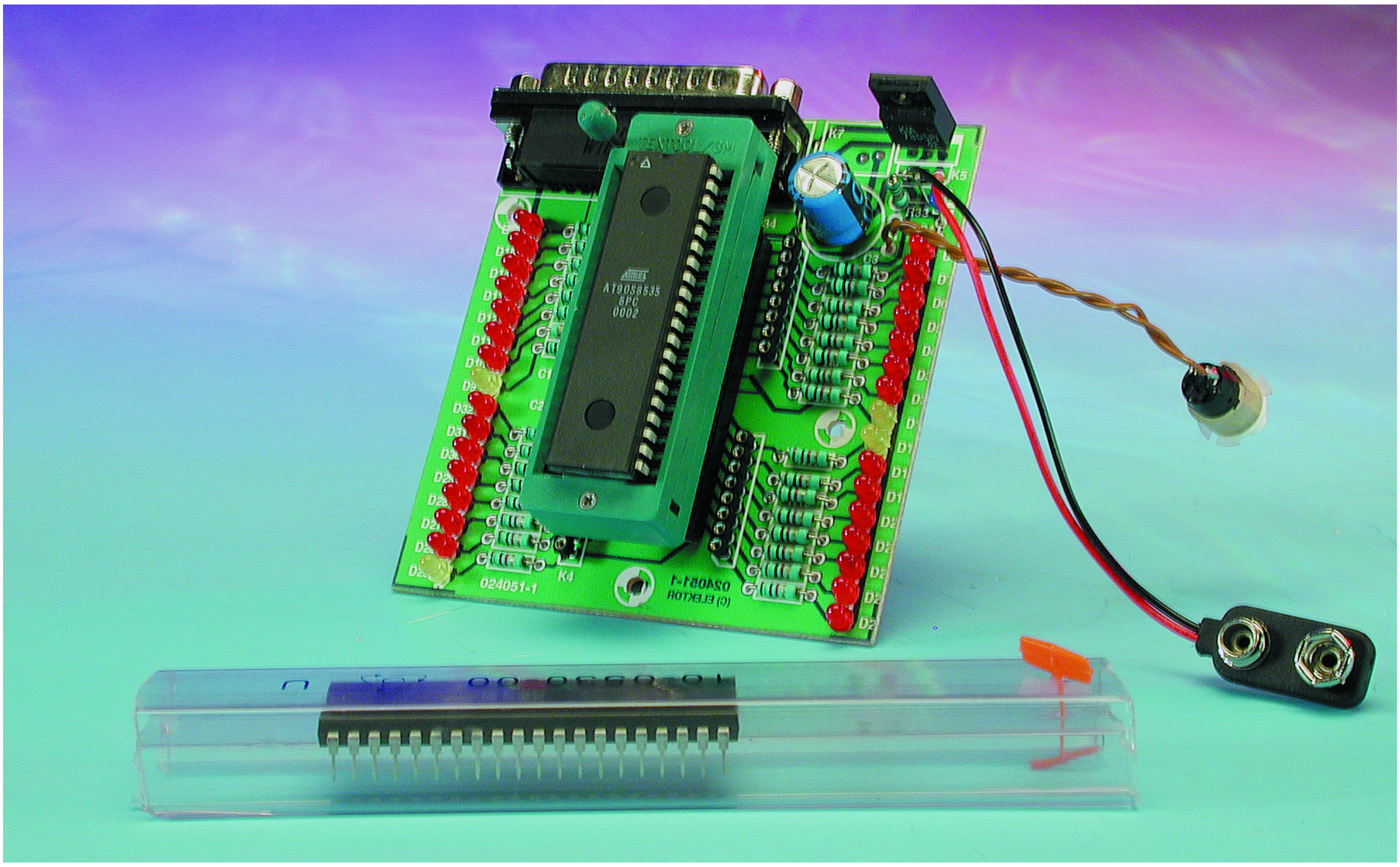


AT90S8535 Programmer

suitable for in-system use

Design by M. van Houwelingen

This programmer for Atmel processors can be easily built and is very cheap. It consists of little more than a ZIF socket, four connectors and 32 LEDs. The associated loader program can be downloaded free of charge.



Those of you who use a wide range of microcontrollers will have come across the inevitable drawback that they all need different programmers. When these have to be bought each time, the cost soon mounts up. Furthermore, most programmers have the disadvantage that the microcontroller first has to be removed from its circuit before it can be programmed.

By building the programmer yourself the whole exercise becomes easier as well as cheaper. The programmer described here can, in conjunction with the associated (free) software and a data cable, program AT90S/LSxxxx series chips from Atmel in-system. That means you no

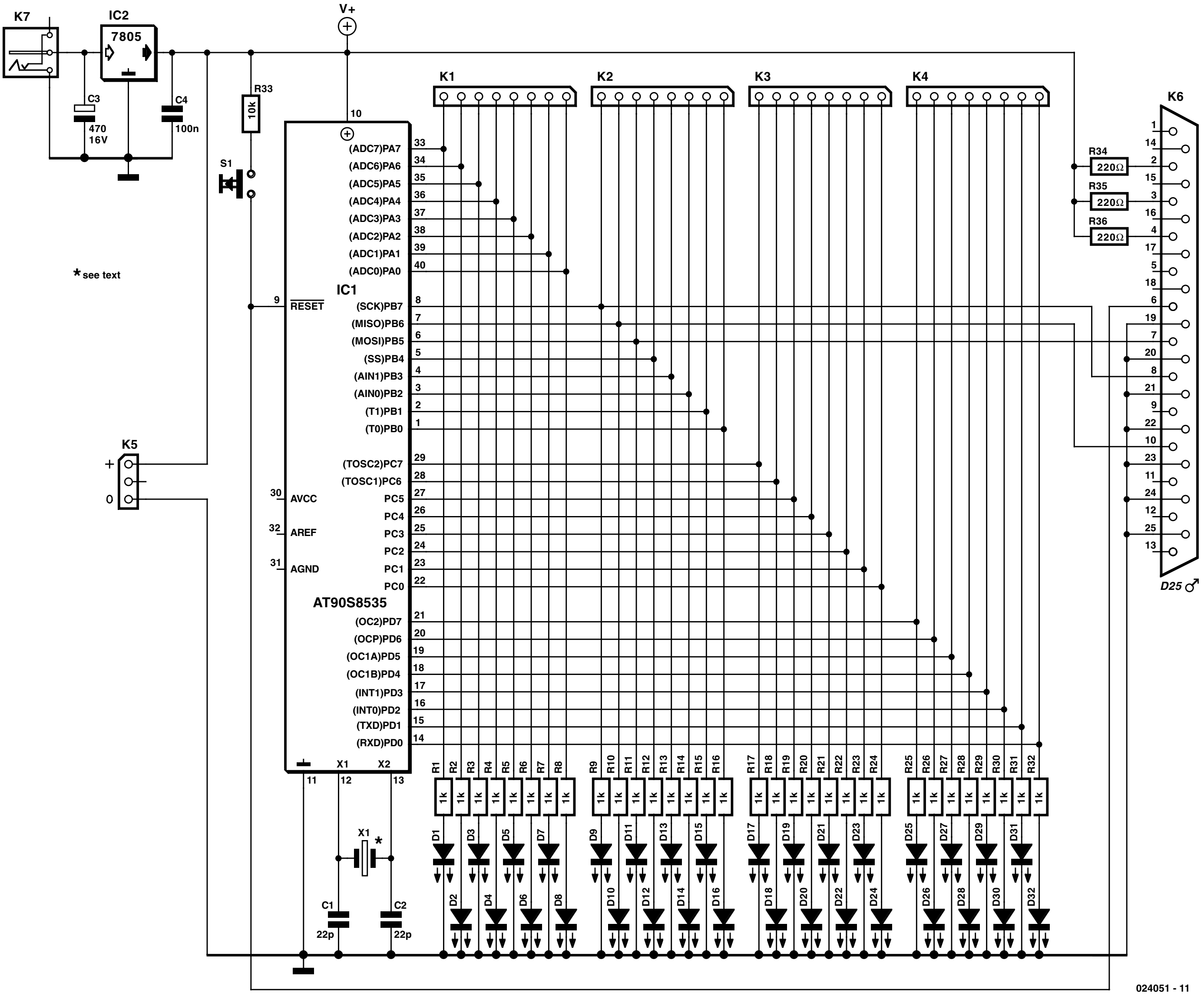


Figure 1.The circuit diagram of the combined tester/programmer. The external supply (via K7) is only required during testing.

longer have to remove the microcontroller form the circuit for programming. Moreover, it can be made very cheaply, since the minimum hardware requirement is just for a data cable used for transferring the program from the computer to the microcontroller. The main function of the circuit described here is to make the testing of programs easier. One look at the circuit diagram in **Figure 1** confirms that this is a very simple affair.

Software

The software that sends the HEX files to the microcontroller is based on the fact that these Atmel chips can be programmed serially. This happens via the MISO, MOSI and SCK pins that are found on all chips in the AT90S/LSxxxx series. The software also uses the RESET pin, GROUND and a supply via the parallel port. For further information regarding the programming technique you should refer to the Atmel datasheets.

Because only the above-mentioned pins are required for programming it should be a simple matter to provide the circuit containing the microcontroller with an extra connector for the data cable. The chip can then be easily programmed in-system.

One thing that should be kept in mind is that the parallel port can only supply a limited current. For this reason it is recommended that the microprocessor is powered by the onboard supply, rather than via the cable.

Circuit

As can be seen in **Figure 1**, the circuit consists of only a 40-pin ZIF socket for the AT90S8535, four connectors for connections to external circuitry and 32 LEDs with series resistors. The LEDs have been added so that the status of the outputs can be inspected during the execution of your own programs. There is also a reset button, as well as a 5 V regulator that takes its supply from a mains adapter (min. 8 V) connected to K7. The supply is only required when running the program, but not during the actual programming.

The frequency of crystal X1 should match the speed of the microcontroller, because there are 4 MHz and 8 MHz versions. The safest option is to choose a 4 MHz crystal, since this will work with both types of microcontroller.

PCB

The PCB for the tester /programmer is shown in **Figure 2**. It is clearly a straightforward circuit, which can be constructed very easily. Do make sure that the LEDs are high-efficiency types, since the values of the series resistors have been calculated for use with these.

Should you wish to use the circuit for programming a microcontroller with fewer pins, you will have to make an adapter that fits in the 40-pin socket. In this way it becomes possible to program the whole range in the AT90S/LSxxxx series. When building the adapter you should of course connect all I/O pins, so that the smaller chips can also be tested in this circuit.

COMPONENTS LIST

Resistors:
R1-R32 = 1kΩ
R33 = 10kΩ
R34,R35,R36 = 220Ω

Capacitors:
C1,C2 = 22pF
C3 = 470µF 16V radial
C4 = 100nF

Semiconductors:
D1-D7,D9-D15,D17-D23,D25-D31 = LED, 3mm, red (low-current)
D8,D16,D24,D32 = LED, 3mm, yellow (low-current)
IC2 = 7805

Miscellaneous:
IC1 = 40-way ZIF socket
K1-K4 = 8-pin header
K5 = 3-way header
K6 = 25-way sub-D plug (male), PCB mount, angled pins
K7 = mains adapter socket
S1 = pushbutton
X1 = 4MHz or 8MHz quartz crystal (see text)
PCB, order code **024051-1** (see Readers Services page)

And finally

The loader program is a freeware DOS version. It can be found on the website of the author at: www.student.tue.nl/E/m.j.v.houwelingen. The name of the program in question is FBPRG16.EXE. The website also contains other software that will prove useful during the development of your own programs.

(024051-1)

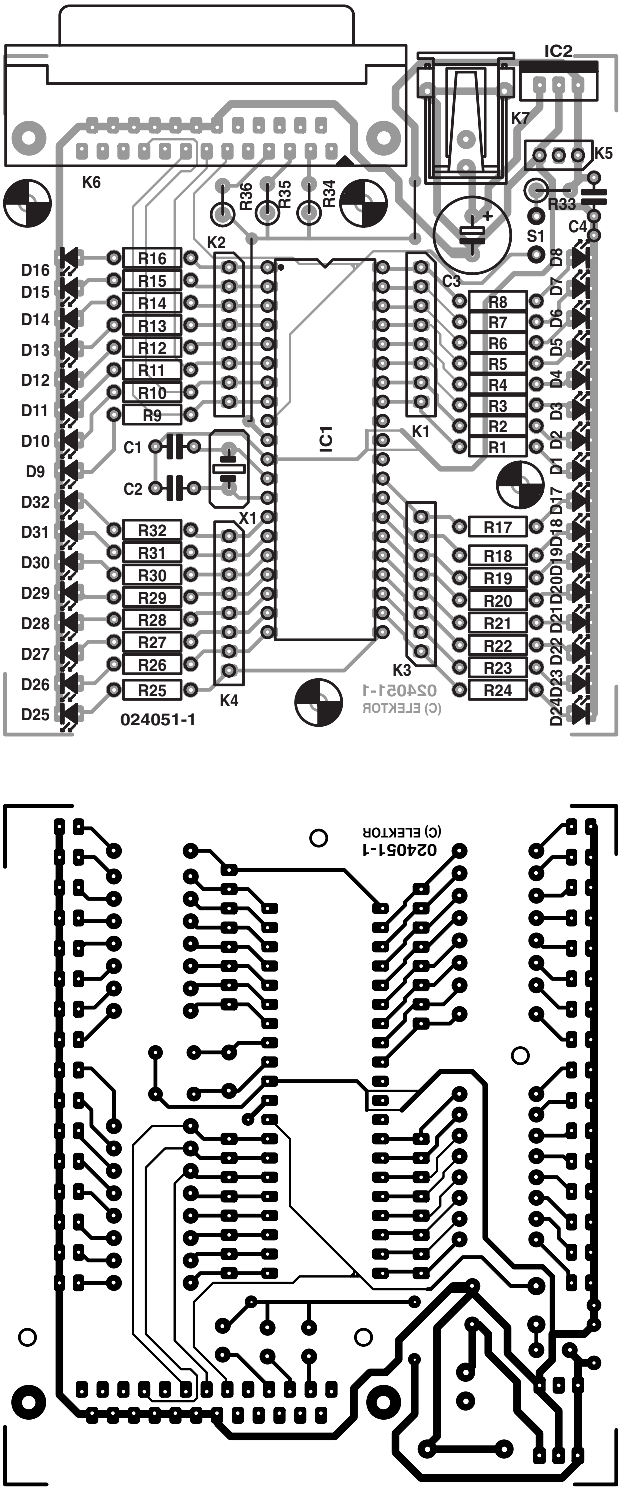


Figure 2. Populating the PCB shouldn't require more than an hour or so.

Digital RF Wattmeter with LC Display

for 1 kHz to 1 GHz

Design by Thomas Scherrer OZ2CPU

www.webx.dk

Any radio amateur knows the importance of an accurate RF power meter. A wattmeter can be used to measure gain in amplifiers, bandwidth in filters, field strength from antennas, transmitter power, SWR, return loss and many other things.

Specifications

Frequency coverage:	1 kHz to 500 MHz (calibrated) 1 kHz to 1000 MHz (uncalibrated, for relative power measurements only)
Nominal input impedance:	50 Ω
Input power range:	–60 dBm to +30 dBm (1 nanowatt to 1 watt).
Input power range using 50-dB attenuator:	up to 100 kwatts.
Dynamic range:	90 dB with good RF shielded case.
Resolution:	0.1 dBm (1 dBm on bar-graph)
Input return loss:	@300kHz: –35dB @100MHz: –27dB @500MHz: –25dB
Input SWR:	@ 300kHz: 1.036 @100MHz: 1.094 @500MHz: 1.12
Accuracy before calibration:	±1 dB from 1 MHz to 450 MHz.
After calibration:	±0.2 dB at each calibrated frequency.
DC Voltage Measurement:	0 to 20 volts.
DC Voltage Resolution:	20 mV
DC Voltage accuracy after calibration:	±20 mV
Power supply:	9 to 20 VDC
Current consumption:	with no LCD light: 30 mA; with normal LCD light:120 mA.

This RF wattmeter uses an AD8307 to measure the power level. The AD8307 front end circuit is both frequency compensated and optimised for return loss to give optimum input SWR over a wide frequency range.

A pre-programmed microcontroller type PIC16F876 with built-in 10-bit analogue to digital converters is used to convert the analogue voltage output from the AD8307 into digital values. Next, a set of lookup tables are used to convert the dBm values into RF voltage and RF power (watts). The readout of all values including a bar-graph appears on a large 20×2 LCD display with back light. There is also a DC voltmeter with minimum and maximum peak storage, plus many extra features, see the software description further on in this article.

The decibel milliwatt (dBm) unit

In RF technology 0 dBm represents 1 milliwatt into 50 Ω. Similarly, 0 dBW represents 1 watt into the same impedance.

So, +10 dBm = 10 mW; +20 dBm = 100 mW; +30 dBm = 0 dBW = 1 W and so on.

The term ‘dBm’ is used at any professional radio development, repair and servicing facility as well as by radio amateurs, to describe (relative) RF power levels. Some engineers are conversant with the dBm unit, others prefer ‘watts’ and still others like to talk about ‘RMS voltage’. This meter displays all three units at the same time...

About the AD8307

The AD8307 monolithic logarithmic amplifier from Analog Devices was first described in the article RF Decibel Meter, see *Elektor Electronics*, January. For reference purposes, the block diagram of this hugely successful IC is given in **Figure 1**. The AD8307 is a relatively low cost component — at least according to the datasheet — in practice, the author paid about €13 (approx. £8.50) for one off plus postage.

The author initially tried the DIL version of the AD8307 — this is also the type used in the previously mentioned article. Although it is easier to solder and use than the SMD chip, its longer connecting pins make it unusable for frequencies higher than about 100 MHz. Several experiments were carried out using this type before it was found that the SMD type could be used up to about 500 MHz. Again several experiments were set up, boards where made, input resistors and capacitors where changed to optimise the lot.

At frequencies above 300 MHz input power should not exceed +20 dBm (100 mW) to maintain accuracy. This is a documented weakness of the AD8307. Not a big problem, really, if you are aware of it — it is just a matter of using the right input attenuation, to reduce the actual RF power applied to the chip to a level below 20 dBm and so guarantee optimum accuracy.

The author is grateful to all the radio amateurs and skilled engineers who have contributed with their experience, good ideas, help with measurements, and lending out expensive equipment so that it was possible to construct this input circuit.

Circuit description

The circuit diagram of the RF Wattmeter is given in Figure 2. It consists of four sections, which will be discussed briefly below, leaving a bit more space in the article to ponder on the second main components — yes that’s the PIC16F876.

The RF voltage converter is draw as a separate unit around the AD8307. The ‘log’ equivalent of the voltage representing the RF power applied to socket K4 appears on K5 in the form of a step level between 0 and 2.5 V. The input resistor network is dimensioned for 50 Ω input impedance which is the *de facto* standard in RF work. The input network can handle power levels of up to 1 watts. C1 and

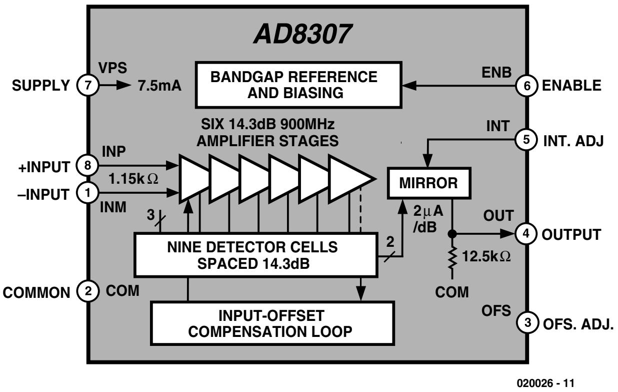


Figure 1. Block diagram of the AD8307 (courtesy Analog Devices).

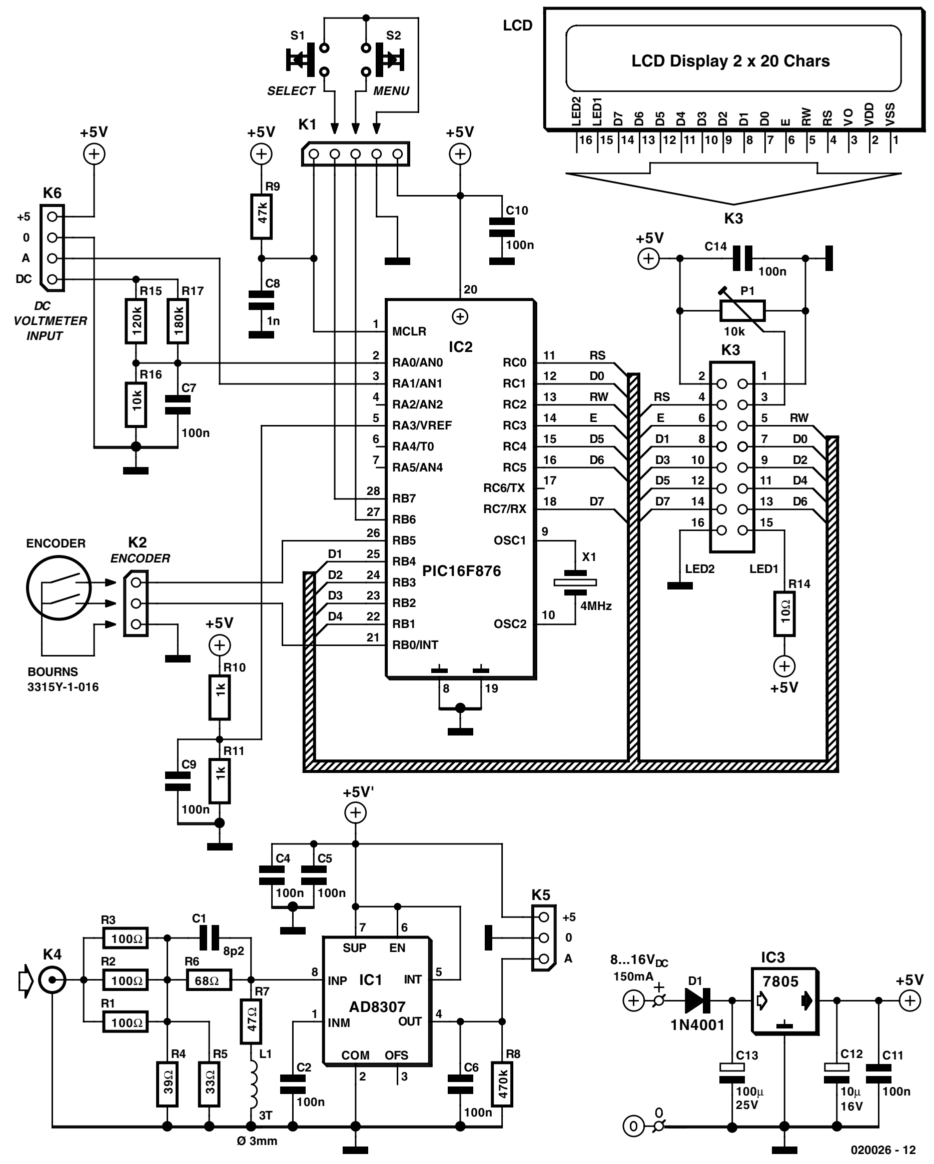


Figure 2. Circuit diagram of the RF Wattmeter. The two main components are the AD8307 in the input converter and the PIC16F876 in the controller section.

L1 serve to cancel stray capacitance and inductance and so optimise the input SWR for higher frequencies.

The second unit is the digital controller around IC2. This ‘black box’ runs software written by the author to handle the following functions:

process the output from the AD8307 into a format we earthlings can read and understand;

read the user controls (pushbuttons S1, S2 and the rotary encoder on K2);

drive an LC display, enabling it to present menus, values, etc.

The PIC is reset at power on by R9-C8. It is clocked by a 4-MHz ceramic resonator.

The third unit is the LCD. Here, a 2-line by 20 character type is used. Preset P1 is used to adjust the contrast.

The fourth unit is the power supply around IC3. Totally conventional in design, the supply should not require further discussing. Input power may be obtained from any old mains adaptor capable of sustaining a current demand of 150 mA or so at 8 to 16 VDC.

About the PIC16F876

The requirements for the digital section of this projects included a cheap micro controller with 10-bit analogue to digital converter (ADC), a cheap and simple programming interface and program memory of the Flash type for ease of software development and debugging. Furthermore, 4 digital inputs, 2 for the pushbuttons and 2 for the encoder, 7 outputs to the LC display in 4-bit mode or 11 outputs in 8-bit mode. The Microchip 16F873 and 16F876 PIC microcontrollers with 4 and 8 kwords of Flash program memory proved an excellent choice. Their price being almost the same, the author went for the larger 8 k type.

The 16F876 has 5 analogue inputs with 10-bit resolution, representing a range of 0-1023 in discrete values when the input voltage goes from 0 to 5 volts. The DC signal from the AD8307 covers 0 to 2.5 volts for the entire operation range. To get full digital resolution the ADC inside the PIC could use an external positive reference for the full-scale voltage. This function is implemented by R10 and R11 which create a 2.5-volt reference from the 5-volt power supply rail. This voltage is not super critical, if there is a small error, it will be calibrated in software with a zero dBm offset point.

A good and simple programmer for this PIC was designed by Johann Aichinger and is called PROPIC, schematics and software may be downloaded at

http://jaichi.virtualave.net/

Many other PIC programmers support the 16F876 type, including ‘IC-PROG’ by Bonny Gijzen. This one also works fine and supports almost any IC that can be programmed. You’ll find it at

www.ic-prog.com

The wattmeter software was developed using the Microchip ‘MPLAB’ PIC programming suite in combination with an external C compiler called PICC from HI-Tech, their web site is at

www.htsoft.com

where you can download a free trial demo version. If you need to re-compile the software for this wattmeter, you can change the start up screen to display your name or callsign if you like. The wattmeter software source code files can be downloaded from the Free Downloads section of the *Elektor Electronics* website. The relevant file number is **020026-11** (October 2002). For those without access to the Internet (raise your hand please), the files are also supplied on floppy disk through our Readers Services.

Construction

An instrument like the present RF Wattmeter should be built with due consideration given to mechanical aspects as it will never function properly when screening is neglected.

To begin with, the double-sided and through plated circuit board shown in **Figure 3** should be cut in two to separate the controller section from the input board.

Input Board

This board (**Figure 4**) employs SMDs (surface mount devices) because small components better handle high frequencies. Read the following paragraphs if it is the first time you build up a board using SMDs.

If you are soldering with your right hand, first add a small amount of tin on the right-hand solder pad.

Place the component carefully with your left hand using tweezers, solder the right-hand side while holding the component perfectly straight. Now simply solder the left-hand pad to the left-hand side of the component.

To solder the AD8307, carefully and quickly pre-tin pad number 3 on the board. Then keep the IC pressed in place when applying heat to pin 3, joining it to the solder pad by pushing gently with the solder tip. If necessary re-align the IC and then solder the other pins.

The input board has specially designed shapes with generous pads that allows both 1206 and 0805 style SMD components to be fitted.

Inductor L1 consists of 3 turns of 0.5-mm enamelled copper wire (ECW). The internal diameter of this inductor is 3 mm and the turns are spaced about 0.5 mm apart.

The input board needs to be mounted directly onto the flange of a BNC socket with four M2.5 threaded mounting holes. Some BNC sockets have a 3-4 mm long Teflon® isolation piece around the centre pin. Cut this isolation away, and cut the

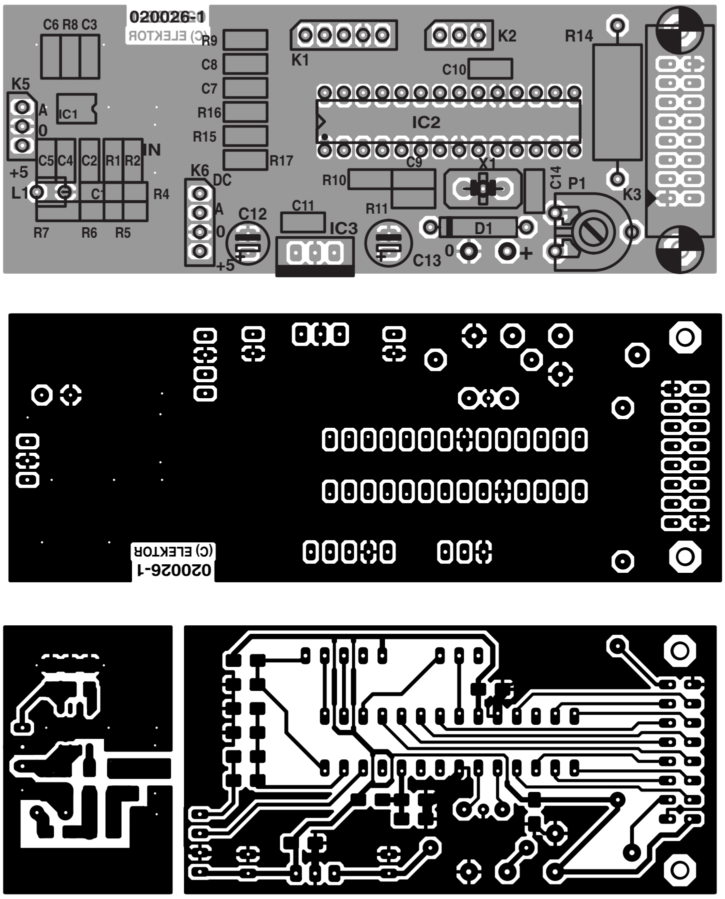


Figure 3. PCB design for the instrument. The board is double-sided, through plated and available ready-made through the Publishers' Readers Services.

COMPONENTS LIST

Resistors

SMD case 1206 or 0805:
R1,R2,R3 = 100Ω (R3 on top of R1/R2)
R4 = 39Ω
R5 = 33Ω
R6 = 68Ω
R7 = 47Ω
R8 = 470kΩ
R9 = 47kΩ
R10,R11 = 1kΩ
R15 = 120kΩ
R16 = 10kΩ
R17 = 180kΩ
R14 = 10Ω 1W
P1 = 10kΩ preset

Capacitors

SMD case 1206 or 0805:
C1 = 8pF2
C2-C7,C9,C10,C11,C14 = 100nF
C8 = 1nF
C12 = 10μF 16V radial
C13 = 100μF 25V radial

Inductor

L1 = 3 turns, 0.5mm dia. ECW (SWG #30), turns spaced at 0.5mm, internal dia. 3mm.

Semiconductors:

D1 = 1N4001
IC1 = AD8307AR (SMD)
IC2 = PIC16F876-04/SP, programmed, order code 020026-41, see Readers Services page
IC3 = 7805

Miscellaneous:

K1 = 5-way SIL pinheader
K2,K5 = 3-way SIL pinheader
K3 = 16-way SIL pinheader
K4 = BNC socket with flange
K6 = 4-way SIL pinheader
S1,S2 = pushbutton, 1 make contact, chassis mount
PC1,PC3,PC8,PC10,PC12 = solder pin
X1 = 4MHz ceramic resonator (3 pins)
LCD module with 2 lines of 20 characters, e.g., LM032L (PC2002LRS-BEA-C)
Rotary encoder type 3315Y-1-016 (Bourns)
Mains adapter socket, chassis mount
IC socket, 28 pins, narrow
PCB, order code 020026-1 (see Readers Services page).
Disk, source code files, order code 020026-11 (files also available from Free Downloads)

centre pin so that it is about 2 mm long.

Check that it is possible to solder the rear side of the flange. If not, remove any protective layer by scratching it with a tool so that soldering is possible. Now mount the input board, don't be economical on the solder tin, solder it all the way, including at the bottom side of the board.

Controller Board

An early prototype of this board is shown in **Figure 5**. If the PIC16F876 is mounted in an IC socket, you should use one of undisputed quality, say, one with turned pins. However, those of you with access to suitable PIC programming tools may also solder the PIC directly on to the controller board as the chip can be updated in-circuit with new software when available.

The ceramic resonator frequency is 4 MHz, this is not critical. If you don't want to use the optional serial output discussed in the 'menus' inset, use a 3-pin type with built in capacitors. A normal 2-pin version will also work fine.

The controller board is mounted on the rear side of the LCD module, using two standoffs with a length of about 10 mm or whatever you have available.

Four stand-offs are needed to mount the LC display to the case front panel.

Resistor R14 is a 1-watt type, which controls the LCD backlight level. A value of 10 Ohms will give normal light but if you want to use batteries to power this instrument it will be a good idea to increase the value of R14 to 20 Ohms or so, or maybe mount an switch so you can disable the light when the meter runs battery powered.

The 7805 voltage regulator will run a bit hot if the input voltage exceeds about 10 V. Either bend the regulator down to the case and secure it using an M3 screw, or mount a heatsink to its metal tab. A small click-on type will be enough.

Rotary encoder and switches

The rotary encoder is a cheap double contact type supplying Gray code. This encoder is used as a dial to select options from the menu and to change different settings, see the software description further on. If the encoder appears to turn the wrong way, simply reverse the wires at the two data output connections.

The two switches S1 and S2 must be push button types. S1 is used to pick options from the menus and S2 is the menu access button.

Initial testing (hardware)

At this point it is assumed that both boards are ready assembled. If the PIC is in a socket: remove it, if the PIC is soldered in the main board, a variable DC power supply should be used for the testing described below.

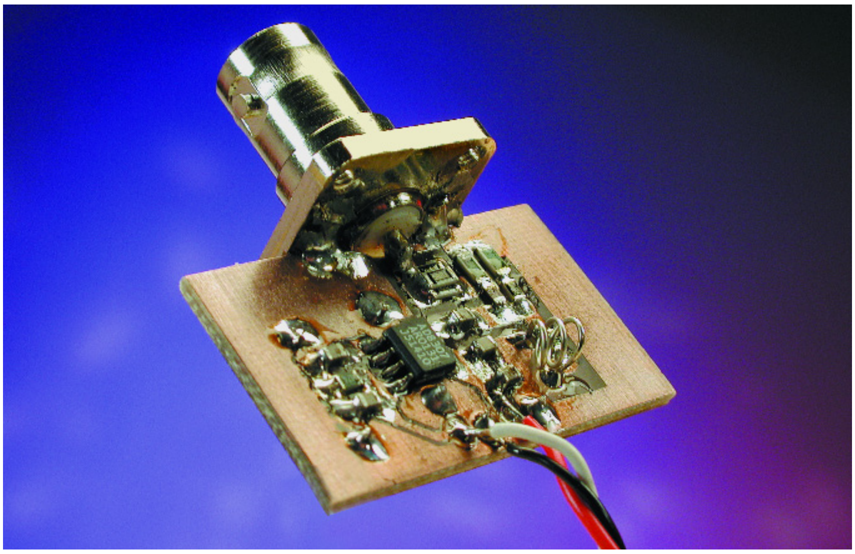


Figure 4. Input board with BNC flange socket attached by soldering along the edge.

Start with just the controller board, with the 7805 fitted and maybe also the PIC. The display and the input board are not yet connected.

Before proceeding you need to verify that the 7805 regulator is working okay. Supply the board with 9 volts, check for +5 V at the pad for the input board near C12. Check for +2.50 volts at IC1, pin 5.

Check for +5 V at IC1 pin 1. Switch off, connect the display, insert the PIC, and switch on again.

Turn P1 counter clockwise so that the display get zero volts on the adjust pin 3. This will yield maximum contrast so that you can see the LCD is alive. Adjust P1 for best contrast at your viewing angle. Check for green backlight in the display and you will see the display writing the RF Power Meter's welcome message (see inset). Touch the input A terminal with a finger, this will give a response on the display. Switch off.

Double check all the soldering on the AD8307 and the input board, then connect it to the main board using shielded audio-style cable or whatever you have to connect the signal from the input board to the controller board. Apply power again and check that current consumption is under 150mA.

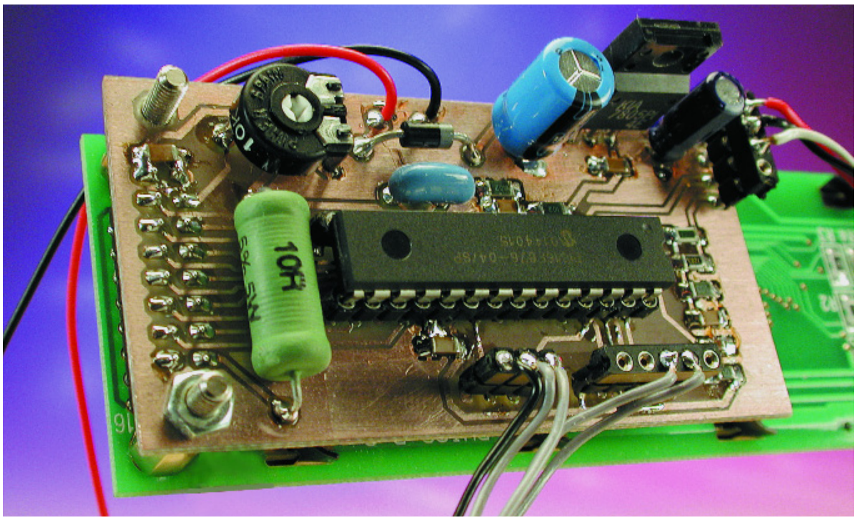
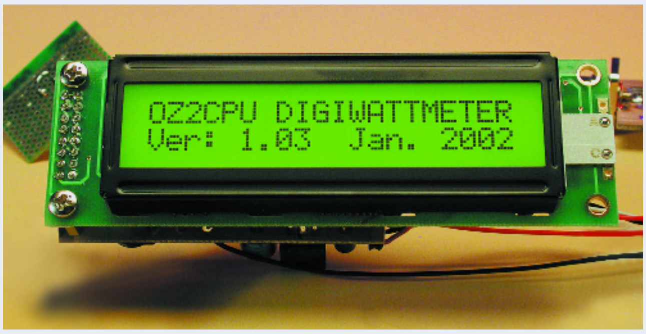


Figure 5. Controller board attached to back side of LCD module.

Using the menus (software v. 1.03)

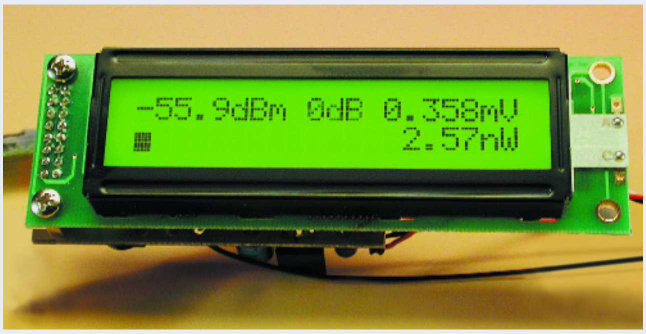
All intelligence vested in the instrument resides in the software developed by the author and Flashed into the PIC microcontroller. Those of you with access to suitable PIC programming software and equipment will care to know that the source code files for this project are available free of charge from the Publisher's website (Free Downloads, item 020026-11, October 2002). Ready-programmed PICs are also available under order code 020026-41.

The instrument's welcome screen should look like this:



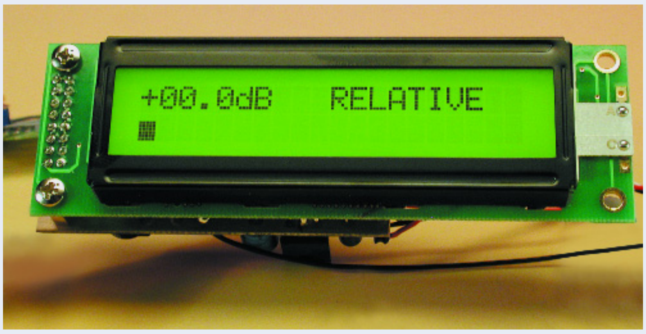
```
OZ2CPU DIGIWATTMETER
Ver: 1.03   Jan. 2002
```

The main start up screen shows:
dBm, status, RF-voltage Bar-graph, RF-power watts



```
-55.9dBm 0dB 0.358mV
▮              2.57nW
```

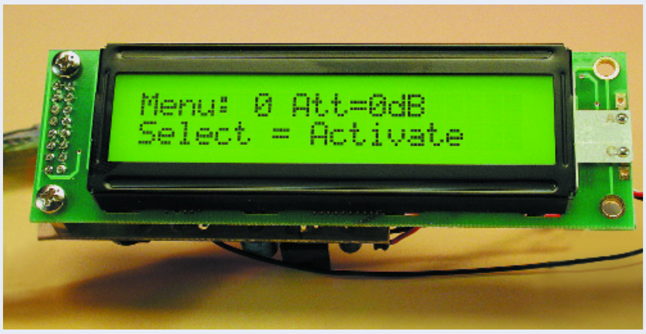
If no attenuation is used, the dBm readout goes from –63 dBm (noise floor) to +30 dBm (1 W).
The status readout shows the selected frequency band, and attenuation mode. Use the band dial to change between LF, HF, VHF, UHF and SHF calibration memories. It is suggested to calibrate the wattmeter's 0 dBm reference at: LF = 3.5MHz, HF = 14 MHz, VHF = 145 MHz, UHF = 430 MHz, SHF = 440 MHz. Of course, you can calibrate at your own favourite frequencies for best performance.
In the RF power meter menu, use the SELECT button to enter RELATIVE mode. In this mode dBm and bar-graph is shown, when entering this mode the dB read out is zeroed.



```
+00.0dB    RELATIVE
▮
```

The menu

To enter the menu / settings use the MENU button. When in a menu, use the rotary encoder to get the desired setting. At the right setting, use SELECT to activate, this is also shown on the display.



```
Menu: 0 Att=0dB
Select = Activate
```

Available menu entries:

0: 0 dB, no attenuator mounted, 1 W max.
1: –10 dB attenuator mounted, 10 W max.
2: –20 dB attenuator mounted, 100 W max.
3: –30 dB attenuator mounted, 1 kW max.
4: –40 dB attenuator mounted, 10 kW max.
5: –50 dB attenuator mounted, 100 kW max.
6: DC Voltmeter, actual and min. and max.
7: RF Power Meter, the default start up screen.
8: SSB PEP (peak envelope power) Wattmeter, with peak hold and variable decay.
9: Return loss with SWR readout, usable with a SWR bridge.
10: Calibrate 0 dBm in the selected band.
11: Read all calibration values.
12: Zero all calibration memories.
13: Display update delay 2-80 ms, peak hold and decay speed.
14: About Info, shows software version and so on.

DC voltmeter

Nothing gets burned if the input polarity is reversed. In the DC voltmeter screen, actual voltage, minimum and maximum are displayed. To reset min. and max. readings, press the Select button. The voltmeter can be used to monitor the battery voltage if a battery supply is used, or whatever you want to measure, but remember the input impedance is about 80 kΩ.

Extra features

Some enthusiasts requiring additional features may find a serial output a useful extra. On pin 17 the PIC outputs a serial datastream which may be converted to RS232 levels using a MAX232 IC in its usual configuration. The datastream produced by the PIC may be connected to a free RS232 port on your PC. Any communications or terminal emulation program like HyperTerminal should be able to read the datastream. The comms settings are: 38400 Baud, 8 bit ASCII, no parity, 1 stop bit. In short: 38K4 8 N 1.

If you do not have an RF test generator to calibrate and test with, put a normal resistor pin in the BNC input plug, and transmit with your VHF, UHF rig or whatever (PMR) radio you have available, the closer you take the radio to the wattmeter, while transmitting of course, the higher the readings on the instrument will become.

With this initial test completed okay, use, lend, or get access to a good RF signal generator. This will enable you to calibrate your new digital wattmeter and use it to for accurate measurements in your own shack. But first you need to build the electronics into a properly shielded cabinet. The calibration will be done later.

Cabinet? Home made or..

Sure, you want the cabinet to fit exactly to the LCD display, BNC, switches, DC connector and so on. So why not try to make a cabinet of unetched single-sided copper clad board? This material is cheap, easy to cut and drill.

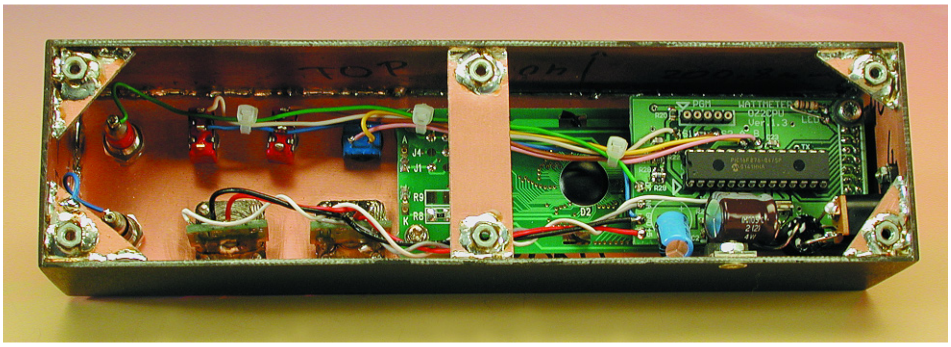


Figure 6. As a suggestion, a screened housing for the instrument may be made from pieces of one-sided copper clad board soldered together along the edges. This method of construction, popular among radio amateurs, even allows securing nuts to be provided for mounting the assembly onto a larger case.

Some extra care and time was spent on the rectangular clearance for the display. The sides of the case can easily be attached using soldering. The completed case may be painted black with spray paint, with white lettering added later using 'Seno' transfer symbols. A coat of clear lacquer spray was used to secure the fragile lettering. Such a cabinet costs nothing except time and paint. The author's version is shown in **Figure 6**.

Calibration of RF input and DC input

The RF meter is calibrated in software, by applying 0 dBm to the input terminal. First select the corresponding band memory using the dial encoder, then press Menu, choose Calibrate 0 dBm and press the select button.

The 0 dBm signal strength is saved in the meter's internal memory. From now on, it will measure within ± 0.2 dB in that frequency band. This is repeated for all the other bands you need. There are five band memories called LF, HF, VHF, UHF and SHF. Remember, you can calibrate at any frequency you need, at any time, over and over again.

The DC voltmeter is calibrated in hardware. You need to fine tune the voltage divider top resistor R15 in parallel with R17 and the bottom resistor R16. You'll find that an accurate DC digital voltmeter is needed besides a variable DC power supply.

Change the wattmeter display screen to DC Voltmeter mode using Menu -> Dial -> Select.

Apply 20.00 volts to the DC input and look at the LCD reading. If it displays a value smaller than 20.00, mount a 10 megohm resistor in parallel with the top resistor R17; if it reads more than 20.00, just mount it to the bottom resistor R16. Remember to disconnect the 20 volts source and the wattmeter power supply while soldering inside the instrument. If you short out R15 or R17 while 20 V is applied to the input DC terminal, you will need a new micro controller!

Caveats

RF voltages of a couple of volts, especially from UHF transmitters, can cause burns and other skin damage. **Never touch any terminal carrying transmitter output power**.

The AD8307 can not survive +5 V erroneously applied to its output pin, so be extra careful to double check your wires from the main board.

Be aware of the fact that **the input board is designed for input power levels not exceeding 1 watt**. If you apply more power, the reading on the display actually goes down! If you are careless and apply more than 1 watt to the instrument, then you might blow up your AD8307. Never rely on the input resistors to burn out first, because they can handle much more than specified (even for several minutes, and by that time your AD8307 will have given up the ghost). **So there is no input overload protection!**

If you want to test a transmitter, use an attenuator pad, directional coupler or a power tap. The author has a homemade 30 dB attenuator capable of handling 50 watts continuous RF power. Its frequency response is flat from DC to 700 MHz. None of his VHF or UHF transmitters can output more than 50 watts so no problem. However, an HF rig will require the use of a 50 dB 'tap' type attenuator which is also home made. This one allows measurements up to one kilowatt with no problems. More details may be gleaned from author's website.

Conclusion

The author has been using this wattmeter for over a year and found it usable for many purposes involving radio signals.

The marginally higher accuracy, larger frequency range and dynamic range of professional RF wattmeters comes at a cost of 20-40 times that of the instrument described here.

As a future extension, two input boards could be mounted with a software routine subtracting input B from input A, to display the forward and reflected power along with the calculated SWR value. This is just an idea, however, and actually the reason for the presence of two input BNC plugs called input A and B on the cabinet of the author's RF wattmeter.

(020026-1)

Sound Pressure Level (SPL) Meter

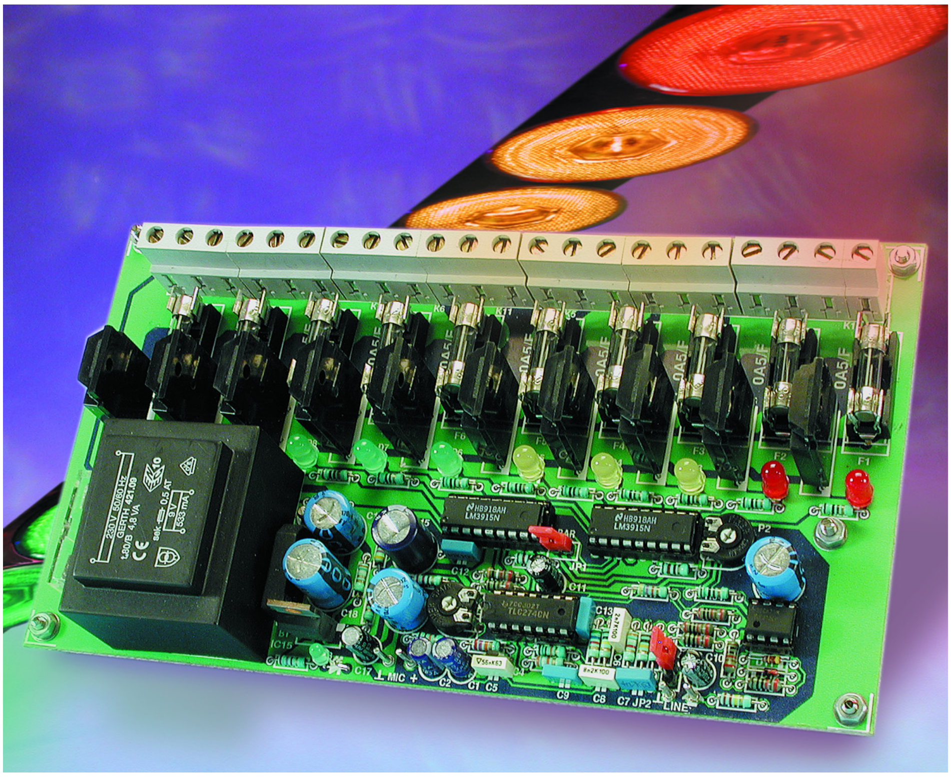
the king-sized model

Design by T. Giesberts

The measuring instrument described here distinguishes itself in one respect from all the others in its species. The display has been implemented in such a way that is – without exaggeration – not easily overlooked. This opens up entirely new possibilities for its application.

The most salient feature of this circuit is that the display consists of a panel made from 10 incandescent lamps. This immediately catches the eye, and at the same time it is obvious that we're dealing with a novel kind of measuring instrument. A typical sound pressure or VU meter makes use of a much more modest LED or moving coil indicator.

So what is the point of this oversized display? Has it been added just for fun? Well, yes and no. Obviously, a display such as this begs to be used in discos and with parties, where it can function as a light show to provide an optical setting for the music. We expect that this circuit will be popular in these circles.



But that is not the purpose for which this piece of electronics was originally designed. The most important application that we had in mind initially, was to provide a clear indication of the sound-pressure level in workshops, party halls and discos – in short, any place where extended exposure to excessive sound levels is likely to lead to hearing loss.

Design

The design of a sound pressure level (SPL) meter is in reality not all that difficult. In addition to a microphone, there is, strictly speaking, not much more required than an amplifier stage, a rectifier and a display. Also, to adapt the indication to the human hearing response, it is desirable to add an A-weighted filter, while in our case, we had to add lamp drivers as well.

A brief glance at the circuit shown in **Figure 1** makes the design of the circuit clear. A bird's eye perspective: the sound is picked up with the aid

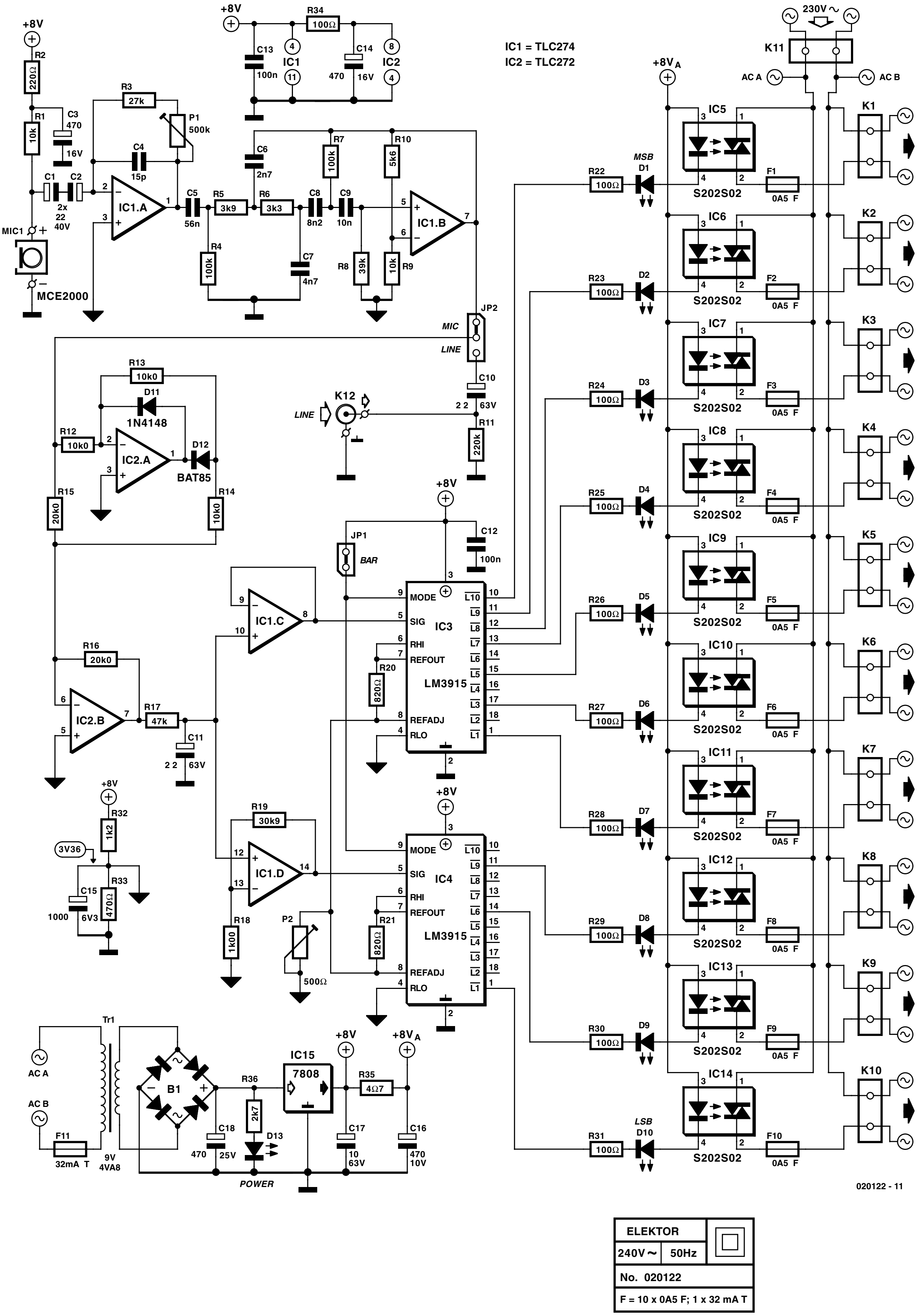


Figure 1. The circuit comprises a microphone amplifier, a filter, a rectifier and a display. The usual LED scale has been extended here with incandescent lamp drivers.

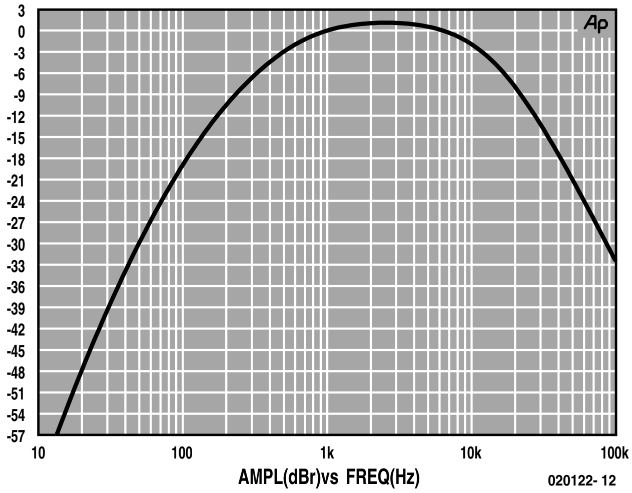


Figure 2. The transfer function of the A-weighted filter has a maximum at a frequency of around 3 kHz to mimic typical human hearing response.

of an electret microphone cartridge (MIC1), amplified (IC1a) and filtered (IC1b). This is followed by a full-wave rectifier (IC2) after which the signal is averaged (R17/C11) and buffered/amplified (IC1c/IC1d). The signal is then presented to two display drivers type LM3915 (IC3/IC4), which have been designed to drive LED indicators directly. Each of the total of ten LEDs is connected in series with an opto-triac (IC5 through IC14). It is the opto-triac's task to switch the lamps connected to K1 through to K10.

The sensitivity can be adjusted with P1. By changing JP2 to the other position it is possible to replace the microphone signal with a line signal that may be connected to K12. The latter option is likely to be used when the circuit is used as a light show in PA installations. If the meter is used in this way on a permanent basis, then the microphone and IC1 plus surrounding parts can be omitted.

We have now shed some light on the essence of the circuit. But, as usual, there are a number of details, which are worth mentioning. So we will now take a closer look at the various parts of the circuit.

Microphone amplifier

The microphone selected for this project is a small electret cartridge, type number MCE-2000 from Monacor (Monarch). Other comparable cartridges are likely to work without problems as well.

The electret microphone receives its supply voltage via R1. R2/C3 suppress any disturbances that may be present on the supply rail. Without the filter, these disturbances would be significantly amplified by IC1a. The amplification of the microphone signal is defined by the ratio of the output impedance of the electret cartridge and R1, as well as the value of P1 plus R3. The value of resistor R3 has been selected in such a way as to provide an adjustment range of about 26 dB for P1.

When making absolute sound pressure measurements, it is important to know what the calibration levels are. When the amplification of IC1a is set to maximum with P1, and P2 is set to '0', full-scale deflection corresponds to a level of about 90 dBA. This was very conscious decision, because this sound pressure marks the beginning of the danger zone for hearing loss.

A few more details. The (FET) output of the electret cartridge is at a constant potential via C1/C2. This improves the drive range and makes a slightly higher value of R1 possible. Opamp IC1a therefore operates as a current to voltage converter. C4 eliminates possible RF-disturbances at the highest amplification and limits the bandwidth to about 20 kHz.

The option of changing jumper JP2 to the other position and connecting a line signal to K12 has already been mentioned previously. The microphone and microphone amplifier are then ignored.

Filter

The so-called 'A-weighted filter' is used both in sound measurements as well as in audio technology when measuring the signal to noise ratio. The essence of this type of filter is to modify the measuring spectrum so that it corresponds to the characteristics of human hearing. The sensitivity of the ear reaches a maximum at around 3 kHz and drops quickly above and below this frequency. The A-weighted filter is therefore designed in such a way that its maximum also occurs at about 3 kHz. **Figure 2** shows the transfer function of the filter. To simplify the construction of the filter, only ordinary components from the E12-series have been used. Despite this, we have been able to match the transfer function to within about ±1 dB of the theoretical model.

Rectifier

To make a full-wave rectifier, a standard circuit with two opamps has been used. The 'negative' signal from the half-wave rectifier IC2a is amplified by a factor of two by IC2b and summed to the input signal to create a full-wave rectifier. A Schottky diode has been used for D12 because it has a lower threshold voltage, resulting in better linearity and a greater bandwidth. The disadvantage is that these diodes have a higher leakage current, and for this reason a normal diode has been used for D11.

R17 and C11 average the rectified signal with a time constant of about 0.1 second. This may well result in an indicator that reacts a little too quickly, but it does produce a better indication of the peaks. The time constant can be increased, if desired, by increasing the value of C11 (there is probably enough space on the PCB for a small 100 µF, 10 V electrolytic capacitor.

Display drivers

In order to drive the display, two ICs of the type LM3915 have been used. To dedicated *Elektor Electronics* readers this device will certainly be well known. If you are unfamiliar with this IC, suffice it to say that the LM3915 was designed to display a varying input signal directly on an LED bar graph. Internally, the IC has 10 comparators, the inverting inputs of which are tied together to the output of an internal buffer/amplifier. Each of the non-inverting comparator inputs is connected to a tap on a ladder-shaped potential divider, which generates the various reference voltages. The following applies to each comparator: as soon as the potential at the inverting input is higher than the reference voltage on the non-inverting input, the output level will go ‘low’ and the corresponding LED will come on. Why then, do we need two LM3915, when one IC would have been sufficient to drive 10 LEDs? There are two reasons. The first is that a simple rectifier is not very linear at the low end of the range and this can be corrected by selecting specific outputs. The second is that we also want the circuit to indicate low levels of sound pressure. **Table 1** shows the relationship between the theoretical value and the actually measured value on the prototype (measured at 1 kHz). The table also indicates that the sound-pressure meter has a range of 48 dB, which will be sufficient in most cases.

Stretching the scale over two LM3915 is made possible by amplifying the measured value found at C11 by an extra 30 dB for IC4. In all other respects the adjustments for IC3 and IC4 are equal. A simple virtual ground has been created with R32/R33/C15 for the entire circuit. This is also the lower reference value (RLO) for the potential dividers in both ICs. This voltage level was chosen such that, at full drive level, the output of IC2b will saturate symmetrically. The top of the potential dividers (RHI) is connected to the LM3915’s internal reference (REFOUT). It is possible that because of the small variations between individual devices (tolerance) the scales do not join exactly. This is of no consequence in this application.

The sensitivity of IC3 and IC4 is greatest when the voltage level at REFADJ is equal to the virtual ground potential (P2 at 0 Ω). This pin has been connected to both ICs so that this level can be adjusted with a potentiometer. It is possible, for example, to calibrate the circuit with P2 to a 2-V full-scale signal (the output of a CD player or similar). The internal reference voltage is then typically 1.28 V.

With a pure sine wave this equates to a sensitivity of 1.42 V_{rms} $(1.28 \times \pi) / 2\sqrt{2})$.

Sound pressure

We now have a fine measuring instrument, but what criteria do we use when making measurements? What is a normal sound pressure and when is there a risk of hearing loss?

The table below lists a number of typical values in dB(A). In this summary do not confuse the pain threshold with the level at which hearing loss may occur. The latter is typically at 80 dB(A). Employers are legally required to provide hearing protection when the sound pressure is continuously above this level in the working environment. At higher levels the risk of hearing loss increases rapidly; at a sound-pressure of 90 dB damage can occur after 1 hour, while at 95 dB(A) the time is reduced to just 15 minutes. The table makes it clear that for visitors of rock concerts it is no unnecessary luxury to provide for some kind of hearing protection!

120 dB(A)	pain threshold
110 dB(A)	live rock concert
100 dB(A)	bench saw
90 dB(A)	passing train
80 dB(A)	motorway traffic
70 dB(A)	vacuum cleaner
60 dB(A)	normal conversation
50 dB(A)	quiet street
40 dB(A)	quiet room
30 dB(A)	library
20 dB(A)	rustling leaves
0 dB(A)	hearing threshold

With the circuit described here, ‘full-scale’ corresponds to a level of 90 dB(A) when P1 is set to maximum amplification and P2 is set to 0 Ω. If you would like to change that, or if you have a different microphone cartridge, then the sensitivity of the instrument will have to be adjusted with the aid of a calibrated sound-pressure meter.

Lamp drivers

The trigger current for the opto-triacs that have been utilised here is 8 mA minimum and 50 mA maximum. To create a comfortable operating margin, the trigger current has been set to 16 mA and is therefore a little higher than strictly necessary (10 ? 1.28 V/R20). To make checking easier, an LED with series resistor is also connected to each of the trigger inputs, so it is possible to see if the drive to the opto-triacs is correct without having to connect any lamps. The resistors are mainly intended to limit the dissipation of the LM3915, but they also make it possible to check the trigger current.

A nice feature of the LM3915 is that pin 9 allows a selection to be made between bar- and dot-mode. This can be achieved by changing JP1. When two LM3915s are cascaded, and dot-mode is selected, it is normally necessary to use a trick to turn off the MSB LED of the ‘lower’ LM3915. But since this output is not actually used it is not necessary here.

Practical matters

The printed circuit board designed for the sound pressure meter is shown in **Figure 3**. Because of the 10 sets of screw terminals for the lamps and the same number of fuses and opto-triacs, the size of the PCB is quite large. This does not mean however, that it is a cumbersome size to deal with. On closer exami-

Table 1

Relationship between theoretical and actual sound pressure values.

LED	theoretical [dB]	measured [dB]	
D1	0	0	MSB
D2	-3	-3	
D3	-6	-6	
D4	-9	-9	
D5	-15	-16	
D6	-21	-22	
D7	-27	-28	
D8	-33	-33	
D9	-42	-40	
D10	-57	-48	LSB

nation, the number of parts that make up the circuit is quite reasonable and since the layout is very straightforward, even less experienced hobbyists should not encounter insurmountable problems. If you stick to what the PCB overlay and parts list prescribe and do not mix up any resistor values, it's unlikely anything goes wrong.

It is of course easiest to start with the low profile components followed by the taller ones. Finish by fitting

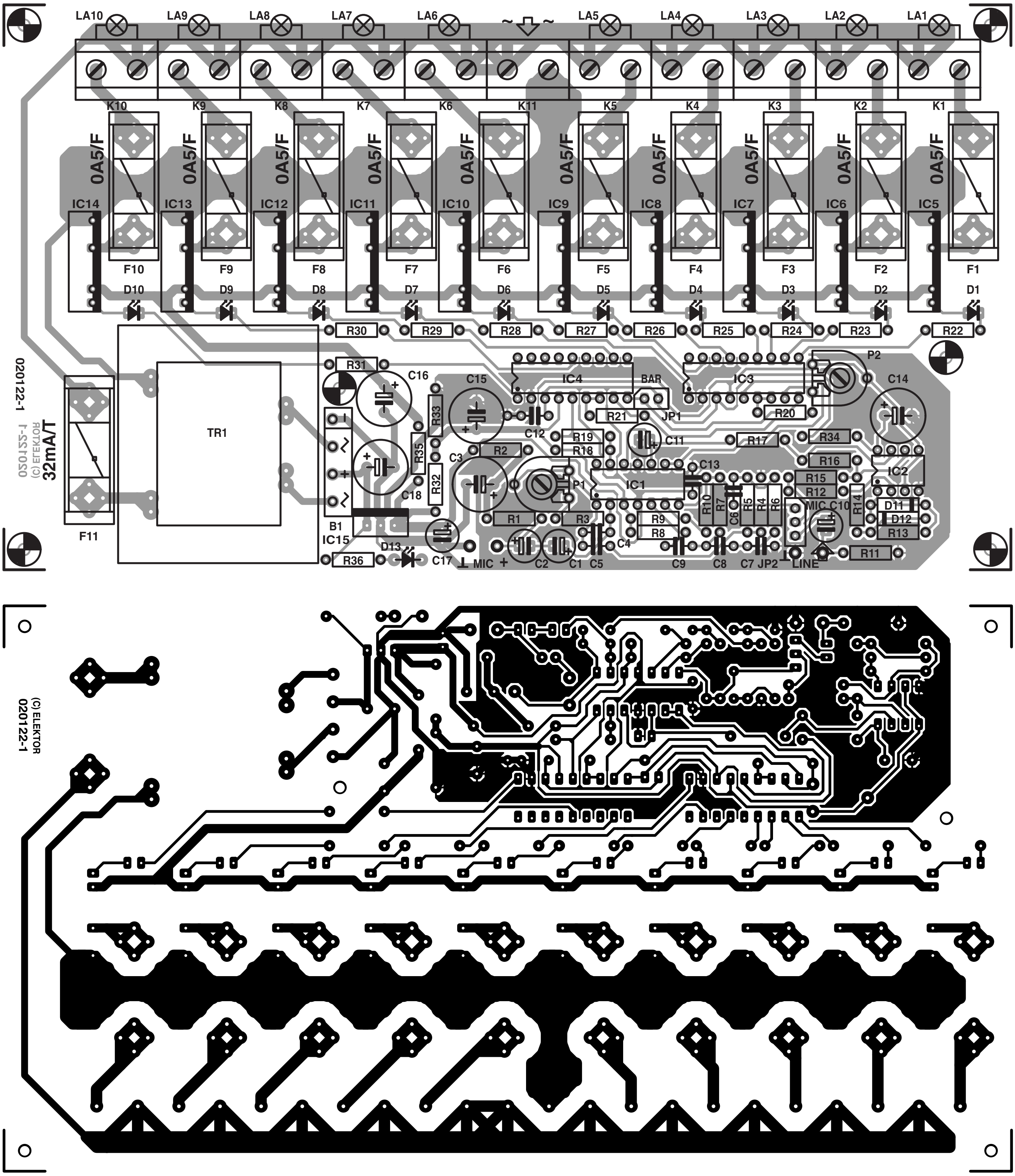


Figure 3. The PCB is dominated by the 10 opto-triacs, the fuses and the connectors for the lamps.

the screw terminals, opto-triacs and power transformer. The ICs are mounted in sockets.

The PCB layout has been designed for Class-II electrical isolation — the opto-triacs, too, easily conform to the standard (4000 V_{rms} isolation!).

In order to simplify the construction as much as possible, the power supply for the circuit, including the transformer, is also mounted on the PCB. As can be seen in the schematic of **Figure 1**, the power supply follows the standard recipe of 4.8-VA transformer, bridge rectifier (note the order of the connections), filter capacitor and 8-V regulator IC15. LED D13 acts as on/off indicator. To prevent any possible EMI problems, it is recommended to use a sizeable mains filter (minimum 5 A load rating!). Finally a few comments related to the enclosure for the circuit. There is no argument that with safety in mind, a plastic enclosure is preferable. Various enclosures can be found that are suitable for the size of this PCB. If you choose a version made from translucent plastic then the LEDs will still be visible from the outside. In the case of a non-transparent plastic case, it is also possible to lengthen the leads of the LEDs and mount these indicators on a panel visible from the outside. On the prototype, just for fun, we actually used different colours for the LEDs: red for D1 and D2, yellow (amber) for D3 to D5 and green for D6 to D10.

Pay close attention when fitting and wiring the circuit into an enclosure that no undesirable short-circuits can occur and provide a suitable mains entry connector or strain relief for the mains cable. To avoid an unnecessary search for the screw terminals for the mains cable, we will quickly mention here that it can be found exactly in the middle among the lamp connectors. The enclosure needs to be well ventilated if the circuit is to be used continuously at full signal and the maximum load is connected to each output (max. 100 W for each lamp!). This is because IC5 through IC14 dissipate quite a bit of heat when they have to switch a lot of power.

(020122-1)

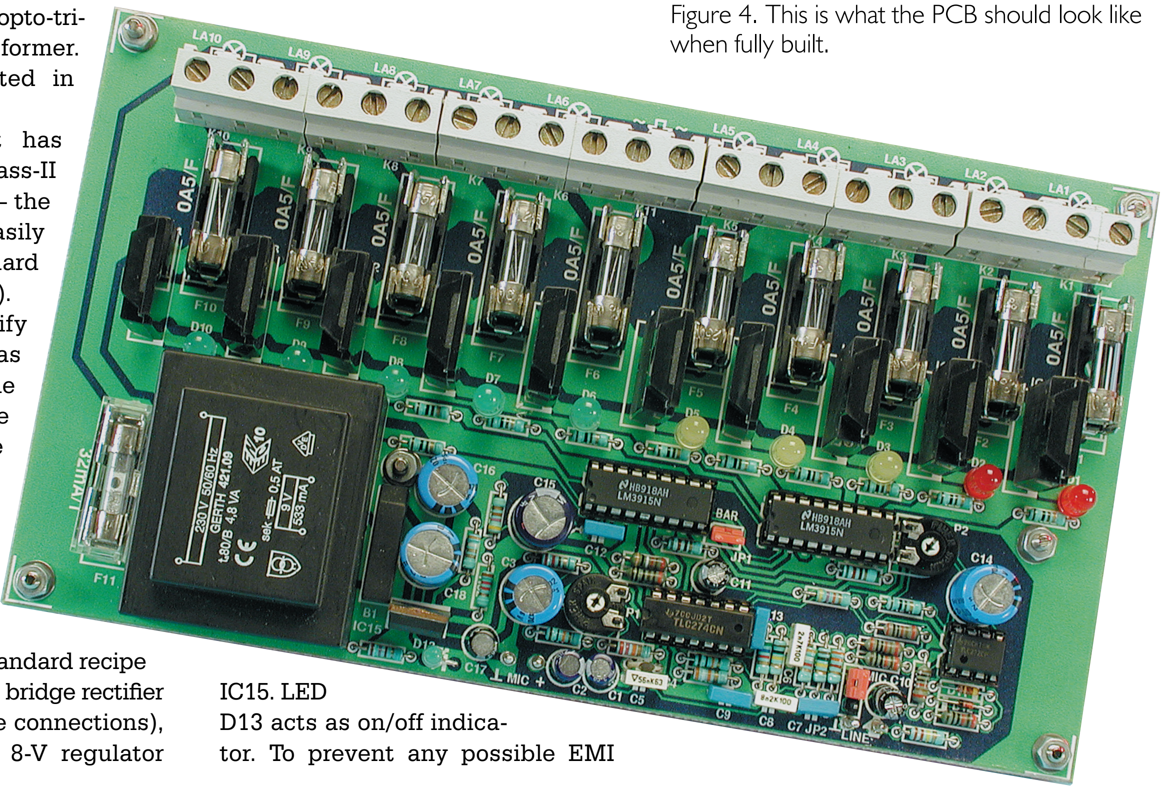


Figure 4. This is what the PCB should look like when fully built.

COMPONENTS LIST

Resistors:

R1,R9	= 10kΩ
R2	= 220Ω
R3	= 27kΩ
R4,R7	= 100kΩ
R5	= 3kΩ9
R6	= 3kΩ3
R8	= 39kΩ
R10	= 5kΩ6
R11	= 220kΩ
R12,R13,R14	= 10kΩ0
R15,R16	= 20kΩ0
R17	= 47kΩ
R18	= 1kΩ00
R19	= 30kΩ9
R20,R21	= 820Ω
R22-R31,R34	= 100Ω
R32	= 1kΩ2
R33	= 470Ω
R35	= 4Ω7
R36	= 2kΩ7
P1	= 500kΩ preset
P2	= 500Ω preset

Capacitors:

C1,C2	= 22µF 40V radial
C3,C14	= 470µF 16V radial
C4	= 15pF
C5	= 56nF, lead pitch 5mm
C6	= 2n7, lead pitch 5mm
C7	= 4n7, lead pitch 5mm
C8	= 8n2, lead pitch 5mm
C9	= 10 n, lead pitch 5mm
C10,C11	= 2µF2 63V radial
C12	= 100nF
C13	= 100nF, lead pitch 5mm
C15	= 1000µF 6V3 radial
C16	= 470µF 10V radial
C17	= 10µF 63V radial
C18	= 470µF 25V radial

Semiconductors:

D1-D10	= LED, red (no high-efficiency type)
D11	= 1N4148
D12	= BAT85
D13	= LED, green (high-efficiency)
IC1	= TLC274
IC2	= TLC272
IC3,IC4	= LM3915
IC5...IC14	= S202S02 (Sharp)
IC15	= 7808

Miscellaneous:

JP1	= 2-way pinheader w. jumper
JP2	= 3-way pinheader w. jumper
K1-K11	= 2-way PCB terminal block, lead pitch 7.5 mm
K12	= cinch socket, chassis mounting
Tr1	= 9V 4VA8 e.g., Gerth 421.09
B1	= B80C1500 in rectangular case (80V piv, 1.5A peak)
F1-F10	= fuse, 500mA, fast, with PCB mount holder
MIC1	= MCE2000 (Monacor/Monarch) or equivalent
F11	= fuse, 32 mA, slow, with PCB mount holder
PCB, order code **020122-1** (see Readers Services page)	

Rugby MSF Controlled Alarm Clock

By David Fletcher, BSc

dave_fletcher@bigfoot.com

A digital alarm clock which decodes and synchronises itself to the time signal picked up from the Rugby 'MSF' transmitter on 60 kHz.

This is a project I have had in mind ever since I did my final year project at University. At that time, the 8080 and other microprocessors such as the 6502 were around, and would have been capable of doing the job. However, the necessary design tools were not available at reasonable cost for the home user, and neither was the modern PC. So the project went on hold for a good few years.

The design tools I have used along with my PC are:

Vutrax, which is a very capable printed circuit CAD program. So far as I know, Vutrax is unique in offering unlimited free use of the software for designs of up to 256 pins. This should be sufficient for most small home projects. See www.vutrax.co.uk for information and downloads.

The **Metalink** assembler, which is available free from http://www.metaice.com/

8052 simulator. Shareware program, available for download from www.8052.com

The clock has an alarm facility which I have not seen on any other unit, that enables any number of alarms to be set throughout the day. In addition, the days of the week on which each alarm operates are specified, so that it is very easy to make the alarm operate early during the week, and later (or not at all) during the weekend.

The display uses seven segment LEDs. These have the advantage over LCDs of readability in the dark without having to provide extra lighting, and in my design the brightness of the display automatically adjusts to the ambient lighting.

No battery is used in this design. I don't like batteries. They need to be monitored and replaced when exhausted, a flat battery can result in data loss when a memory device depends on having constant power, and they create a filthy mess inside equipment when they get forgotten about for too long and leak. All of the alarm settings in this clock are stored in a serial EEPROM, which is said to safely store data for many decades.

Power cuts in the UK are rare, but when they do occur this clock does of course lose all record of the current time. However when power returns the clock receives the MSF signal and automatically sets itself to the correct time again.

Occasionally, the MSF transmitter is switched off for maintenance. During these periods, the clock continues to keep time using its internal quartz crystal which, without any trimming, has an adequate accuracy over short periods.

The clock requires an external

MSF receiver. I have used the Galleon module to build a receiver, but any receiver which provides an open collector output which pulls low with the carrier ON and switches OFF with no carrier should suffice.

Procesor Board

The circuit diagram of the processor board is shown in **Figure 1**. The heart of the clock is an Atmel AT89S53 microcontroller. This takes as inputs the serial data stream from an MSF receiver via connector P1, and the push button controls, connected to ports P3.0-P3.3. P1.3 controls the alarm sounder. P0 controls the segments, and P2 controls the digits, of the multiplexed display.

P1 also provides power to the external MSF receiver, and the constructor can choose to supply either +5 V or the raw DC power. R1 is a fusible resistor for safety. If the external power is shorted to ground the current will be limited, and if the voltage is high enough, R1 will blow open circuit.

The output pin drivers of 8052 type microcontrollers are primarily pull down, with a very weak pull up, resulting in an output with characteristics similar to TTL. Hence the provision of RN1 and RN2. I have found it necessary to only fit RN2 with the AT89S53, but the PCB has a location for fitting them both should they be needed for different microcontrollers.

The AT89S53 has the In System Programming (ISP) facility. This is the function of P3, connected to port P1, which accepts the programming cable of the Equinox Technologies MicroISP series IV programmer. See www.equinox-tech.com for information on

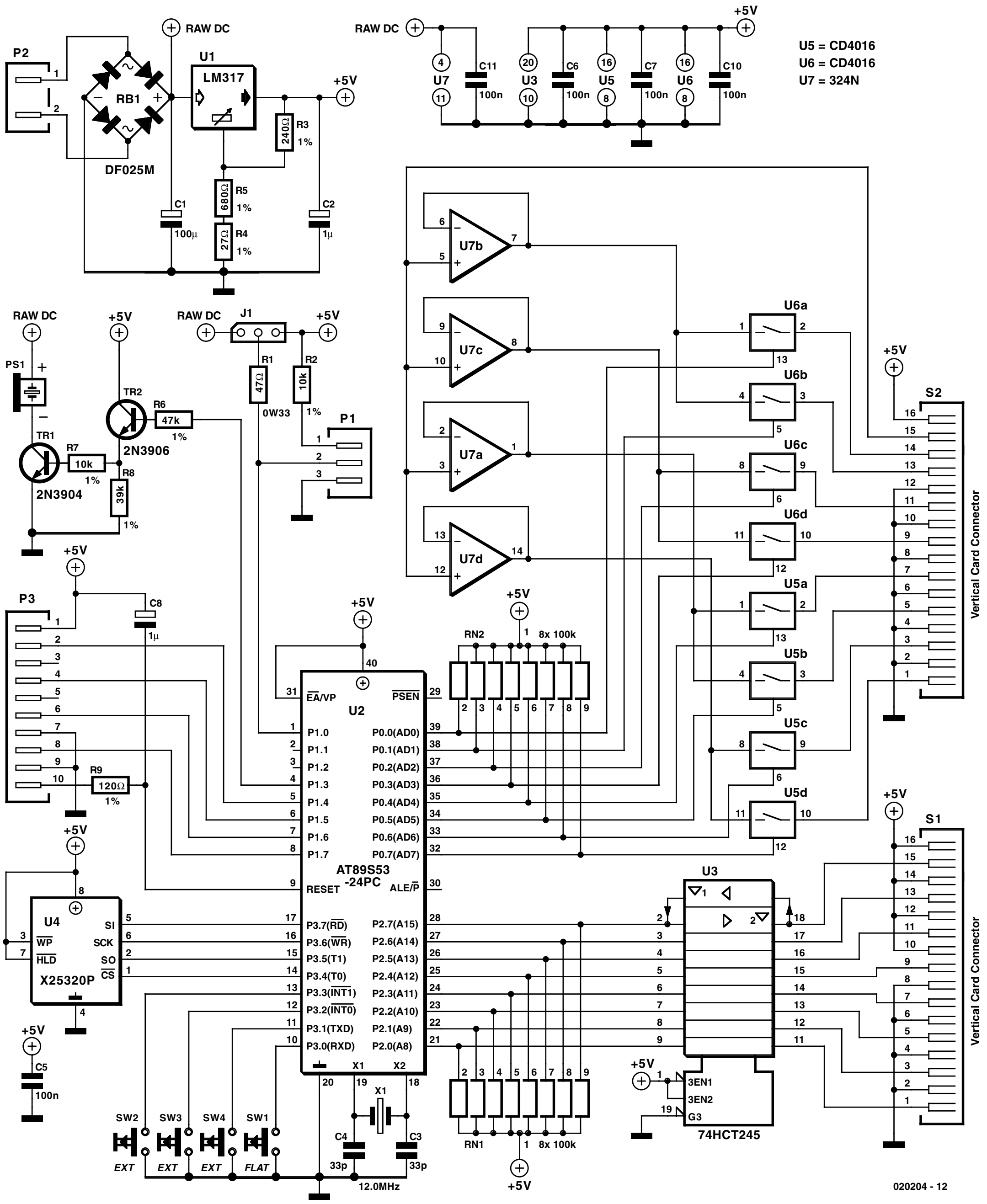


Figure 1. Circuit diagram of control board.

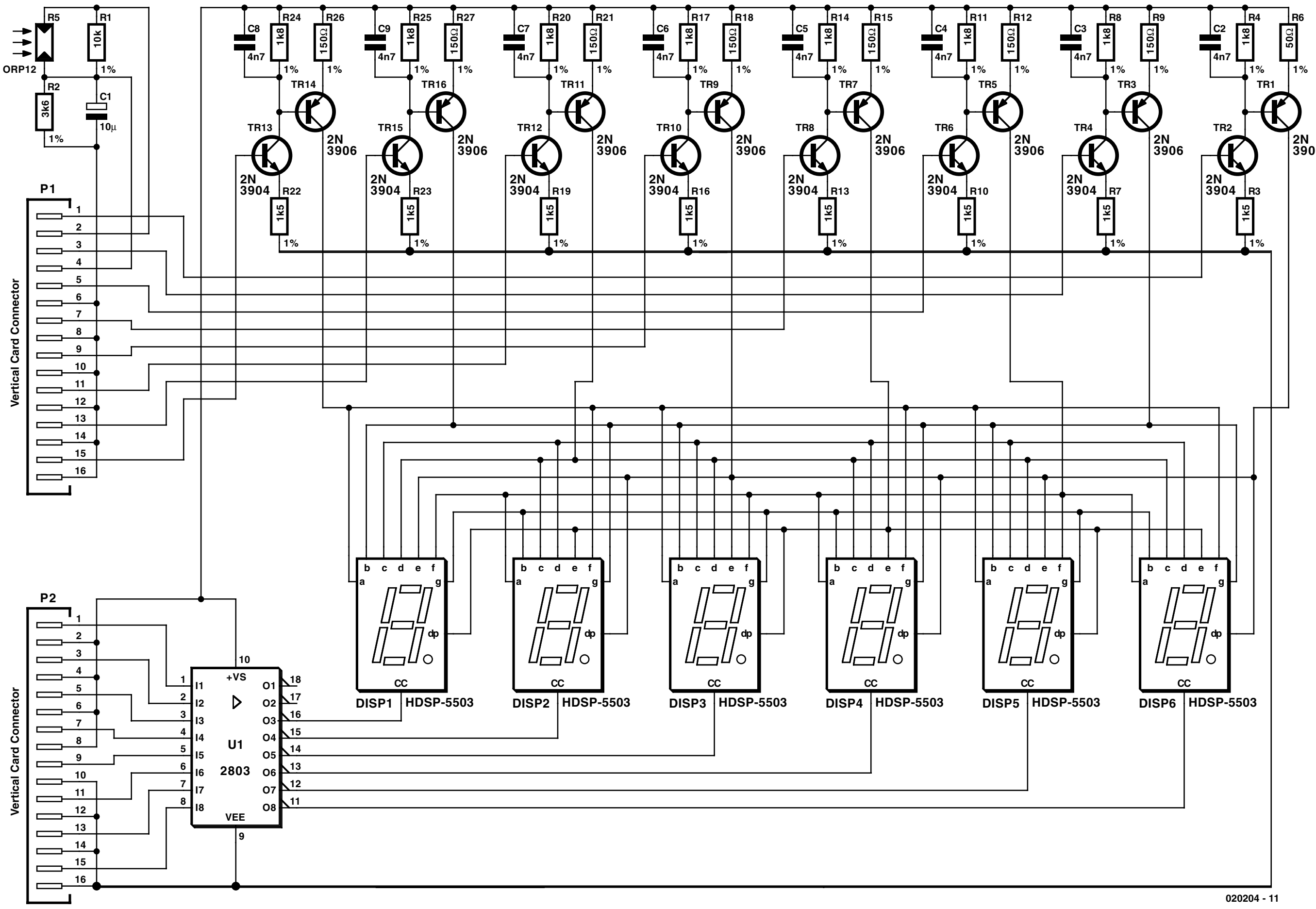


Figure 2. Circuit diagram of display board.

this product.

Very little RAM is available on board the microcontroller. There is enough for the variables and stack required by the software, but not for the alarm facilities I decided to implement. In any case, a non volatile memory was required, hence the choice of the X25320P serial EEPROM for U4.

The 5 volt power supply is provided by linear voltage regulator U1. Demand is low, as it only supplies the microcontroller, op-amp, the CMOS parts and the ambient light level sensor, so no heat sink is required. The current required by the displays is taken from the input side of the regulator. This unregulated supply is referred to as Raw DC.

Display Board

Figure 2 shows the circuit diagram. On the display board, the ambient light level is sensed by the voltage divider formed by R2 and R5. R1 prevents the minimum voltage of the divider from falling below approximately 1.3 volts, which keeps the display dimly lit in a completely dark room. C1 slows down the response of the voltage divider, to make certain it cannot react to fluorescent lights etc. Back on the processor board, the output of the voltage divider is buffered by U7 and selectively switched to the display segment drivers by U5 and U6, under the control of the microcontroller. Each of the segment drivers is a voltage controlled current source. Taking the first driver, when switched on, the voltage across R22 is the output voltage of the ambient light level sensor minus the 0.7 volts V_{be} dropped by TR13. For example, if we present 2.2 V at the base of TR13, 1.5 V will appear across R22, resulting in a current of 1 mA. This same current also flows through R24, producing a voltage at the base of TR14 of the Raw DC voltage minus 1.8 V. Again dropping 0.7 V, this time across the base-emitter junction of TR14, 1.1 V appears across R26, resulting in a display segment current flow of 7.3 mA.

Capacitor C8 limits the rate of change of voltage across R24, and therefore the rate of change of current in the display segment as it switches on and off, reducing RFI emissions.

Software Description

Note: This section should be read in conjunction with the *National Physical Laboratory MSF 60kHz Time and Date Code* specification, and the software listing (which is available as a free download from the *Elektor Electronics* website under number **020204-11.zip**). An extract of the generously commented source code is shown in **Listing 1**.

The software operates by continually executing a main program loop, which updates the display data according to data and flags which are set up by the interrupt routines.

The hardware timer 2 is set up in free running mode to run the T2

COMPONENTS LIST

Receiver board (Figure 3)

S1 = 3-way mini DIN socket
Galleon 60 kHz receiver module, Maplin # MK68Y
MSF antenna, Maplin # MK72P
Receiver PCB (available from author)

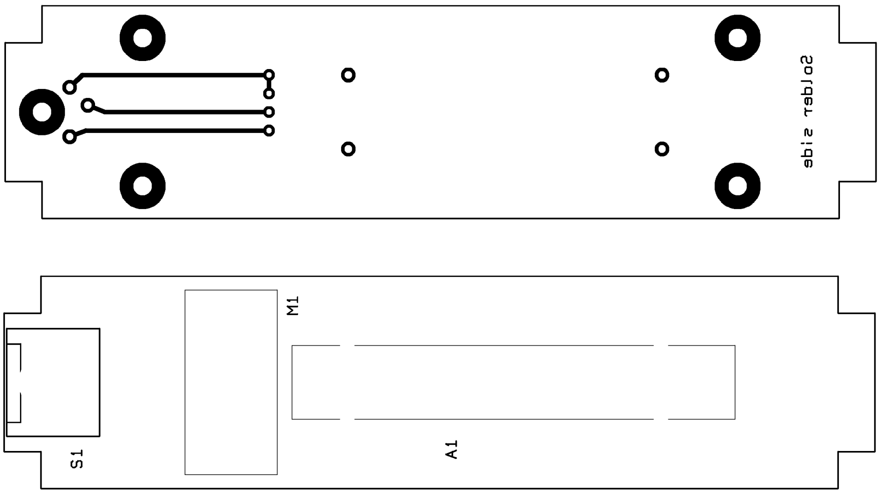


Figure 3. PCB artwork for receiver board (single-sided).

interrupt routine 100 times per second. In the absence of the MSF signal, it counts interrupts, and sets the new second flag at every 100th call. The main program loop increments its second, minutes, hours etc. counters every time the flag is set.

When the MSF signal is present, the T2 routine uses the MSFONCNT byte to synchronise itself to the start of the next second using the 700 ms second markers. It then counts the number of edges which occur during the second, and the time of the last edge, and uses these two pieces of information to decide which of the three bit pattern scenarios occurred during that second, to determine the contents of the bit A and bit B data streams during that second.

Two buffers are implemented in RAM for the bit A and bit B data streams. At the end of each second, the A and B data bits from the MSF signal are shifted into these buffers. A few additional lines of code at the end of the T2 routine perform the alarm cadence, when the alarm is been activated.

Each time the main program loop executes from the top, it checks the bit A buffer for the 01111110 end of minute marker from the MSF signal. This triggers the extraction of all of the time and date information from the bit A buffer, and the parity information from the bit B buffer. All data from the bit A stream are then checked for correct parity, converted to binary, and checked that they have sensible values, as double parity errors can sometimes result in a false parity check pass. When all checks have been passed, the GOOD_DATA flag is set, then the clock is updated from the MSF data when the MIN_MARK flag is set by the T2 routine as it detects the 500mS end of minute marker.

The clock display operates on three modes, normal time display, date display and alarm setting display, controlled by the Display Mode byte DM. The display is set up from the clock data as the last action of the main loop. Time and date displays are simple to set up, using a call to a subroutine for each of the three data items, hours minutes and seconds, or day month and year. The alarm setting display mode is much more complicated, interfacing via control flags with the push buttons routine.

Presses of the four push buttons are detected by the DEBOUNCE routine, which is called during every execution of the main loop. When any push button or any other mechanical switch is operated, the contacts do not close cleanly — they bounce many times over several milliseconds. Whenever the status of the push buttons has changed, which includes contact bounces, the DEBOUNCE routine restarts a counter. Only when the counter expires does the DEBOUNCE routine indicate a button state change and return the new state. Once that happens, the new button state is passed to the BUTTONS routine.

BUTTONS (Listing 1) performs various tasks, depending upon the current mode of the display and which button was pressed. It can stop the alarm sounding, toggle the time display to show 12 or 24 hour format, make the display show the date, put the clock into the alarm setting mode, and edit the alarm settings.

The alarm settings are stored in the serial EEPROM. Every minute of the day has a byte allocated in the serial EEPROM, and seven bits of every byte each represent a day of the week. Alarms are set by the BUTTONS routine, by turning ON the bits which represent the days on which an alarm is required, for a particular minute of the day.

Each time the main program loop increments the time to the next minute, it uses the time of day to calculate the corresponding alarm memory location in the serial EEPROM. This byte is read back into location ALARM_DAYS. The value of the DAYOFWK counter is used to generate a one bit mask which is ANDed with ALARM_DAYS, then if the result of this is not zero, the alarm is triggered and the alarm timeout counter started. The timeout is set at five minutes, after which the alarm automatically stops, unless the alarm stop button has already been pressed.

Construction

The printed circuit board layouts designed by the author are reproduced in the following illustrations:

Figure 3	receiver board, single-sided
Figure 4	processor board, double-sided
Figure 5	display board, double-sided

Ready-made PCBs are available from the author only — see information at the end of the article. Please note that this project has not been tested by the Elektor Electronics design laboratory.

Construction of the clock should be simple and straightforward, using the assembly

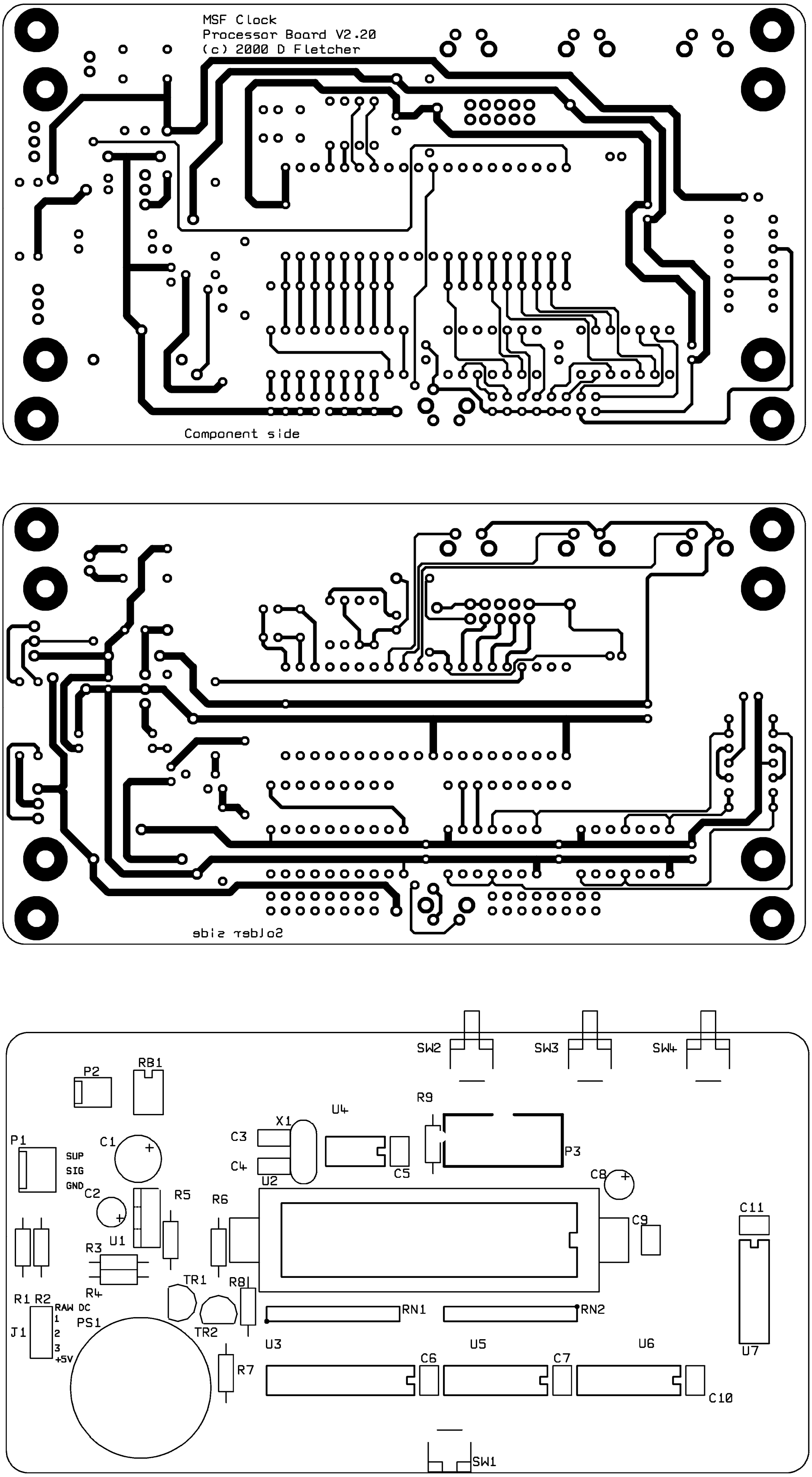


Figure 4. PCB artwork for control board (double-sided, through-plated).

COMPONENTS LIST

Processor board (Figure 4)

Resistors:

R1 = 47Ω fusible resistor
R2,R7 = 10kΩ
R3 = 240Ω
R4 = 27Ω
R5 = 680Ω
R6 = 47kΩ
R8 = 39kΩ
R9 = 120Ω
RN1,RN2 = 100kΩ

Capacitors:

C1 = 100µF 50V radial
C2,C8 = 1µF 63V radial
C3,C4 = 33pF ceramic
C5,C6,C10,C11 = 100nF

Semiconductors:

Tr1 = 2N3904
Tr2 = 2N3906
U1 = LM317 (TO220, vertical mounting)
U2 = AT89S53-24PC (programmed)
U3 = 74HCT245
U4 = X25320P (Xicor)
U5,U6 = CD4016
U7 = LM324N
RB1 = bridge rectifier DF005M (IR) (20V, 1A)

Miscellaneous:

S1,S2 = connector, Berg 76342-308
P3 = 10-way IDC header
J1 = 3-way SIL pinheader with jumper, 0.1 in. pitch
P2 = connector, Molex 22-27-2021
P1 = connector, Molex 22-27-2031
SW1 = pushbutton, Omron B3F-3100
SW2,SW3,SW4 = pushbutton, Omron B3F-3150
PS1 = piezo sounder
X1 = 12MHz quartz crystal
Lock/eject socket for microcontroller, Aries 40-C182-10 (Farnell # 177821)
Enclosure, type BM22W, Farnell #531870
Display filter, Farnell # 177145
Mains adaptor to suit (see text)
Processor PCB (available from author)

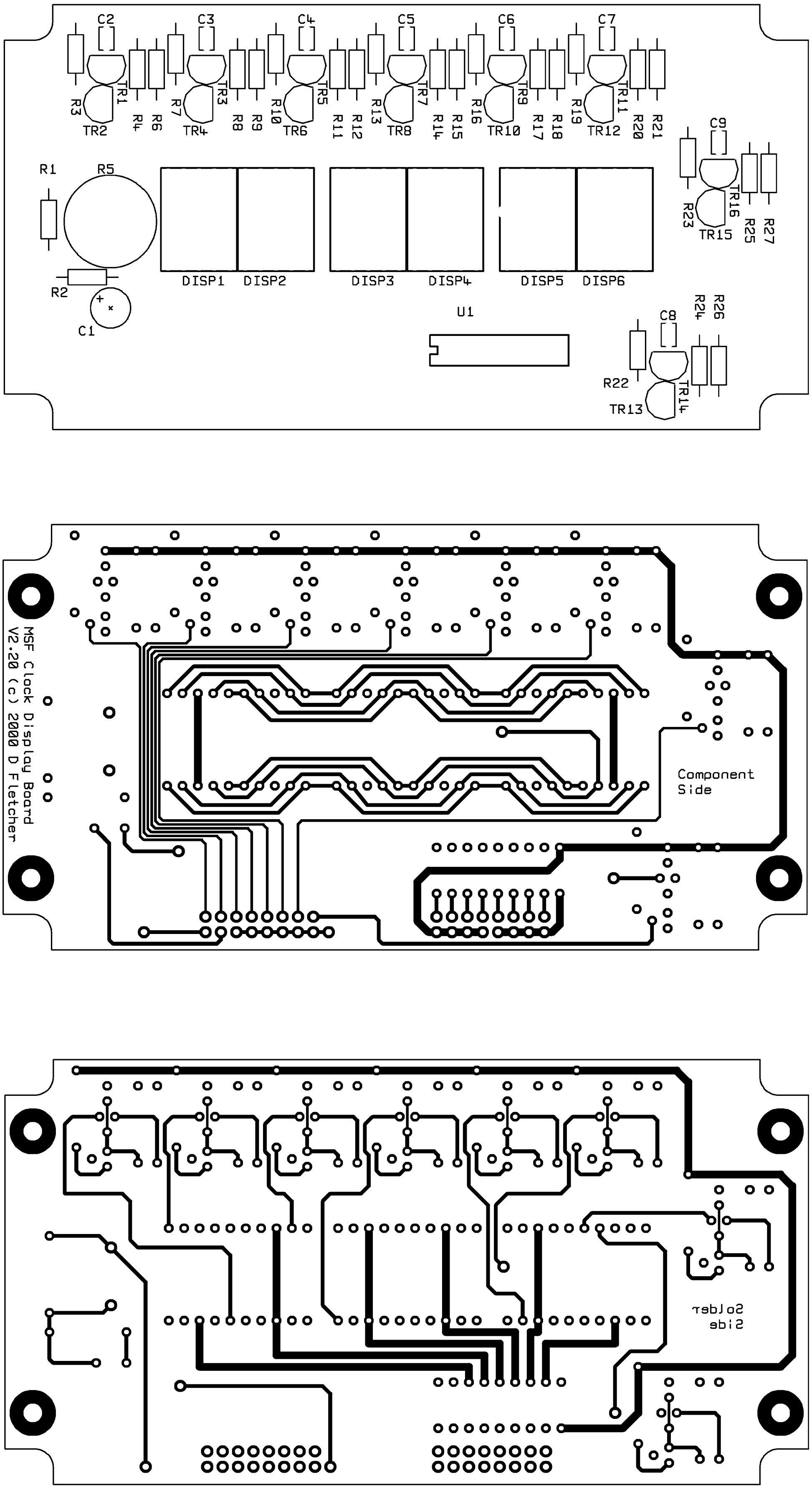


Figure 5. PCB artwork for display board (double-sided, through-plated).

parts list and the silk screen on the PCBs.

A few points need to be made, however.

COMPONENTS LIST

Display board (Figure 5)

Resistors:
R1 = 10kΩ
R2 = 3kΩ6
R3,R7,R10,R13,R16,R19,R22,R23 = 1kΩ5
R4,R8,R11,R14,R17,R20,R24,R25 = 1kΩ8
R5 = ORP12
R6,R9,R12,R15,R18,R21,R26,R27 = 150Ω

Capacitors;
C1 = 10µF 25V
C2,C3,C4,C5,C6,C7,C8,C9 = 4nF7

Semiconductors:
TR1,TR3,TR5,TR7,TR9,TR11,TR14,TR16 = 2N3906
TR2,TR4,TR6,TR8,TR10,TR12,TR13,TR15 = 2N3904
DISP1,DISP2,DISP3,DISP4,DISP5,DISP6 = HDSP-5503
U1 = ULN2803A

Miscellaneous:
P1,P2 = connector, Harwin M20-9970806
Display PCB (available from author)

1. Keep the profile of the display board (Figure 5) as low as possible. If you have obtained the plated through printed circuits from the author, solder the 7-segment modules and U1 directly into the board and try not to let the tops of the transistors stand higher than the modules. The author's final prototype was constructed from home made PCBs which were not through plated. SIL turned pin socket strip was used for the display modules, enabling the shoulders of the pins to be soldered to the pads on the top side of the board. This raised the profile of the board, which required the plastic moulding to be removed from the pins of P1 and P2 in order to get the two boards close enough together to fit the clock into the case.

2) All components are fitted to the boards in normal fashion, **except** P1 and P2 on the display board, and S1 and S2 on the processor board, which must be fitted from the **reverse** side of the board.

3) RN1 on the processor board will probably not be needed so long as a 74HCT part is used for U7, but can be fitted if required.

4). It may be best to fit the microcontroller into a Lock/Eject socket, especially if the In System Programming facility is not used.

5) Before fitting the coloured tops to the push buttons, apply some water based felt pen ink to the ends of the buttons and slide the board into location in the case. Some of the ink will transfer to the case, marking the hole positions. Drill through from the inside and finish the holes to fit the button tops with a fine square file. Mark the edge of the case at the centre line of the secret button in the bottom of the case, and carry the mark to the location of the centre of the button. Drill a small hole here, 1 mm diameter should be fine.

6) Remember to make the display window in the case long enough to expose the LDR.

7) The Atmel AT89S53 microcontroller is recommended, because the In System Programming socket works with this device. However, other devices from the 8052 family may work perfectly well, so long as they can operate with the 12 MHz crystal and can be programmed by other means.

8). Sockets need to be fitted in the back of the case for the power inlet and receiver connection. The power connector will need to match the chosen power supply. 3-pin mini DIN connectors were used for the MSF receiver. Make these sockets up with stranded wire to housings matching P1 and P2 on the processor board.

Prototypes of the boards built by the author are shown in Figure 6 (receiver board), Figure 7 (controller board) and Figure 8 (display board).

Practical use

The clock uses for power a plug-in 'brick' mains power supply. A 12-V supply should be fine, but the voltage and polarity are not critical, and the supply can be either AC or DC, because the clock has its own bridge rectifier in the power input.

Plug in the MSF receiver, and connect the power supply whilst holding the secret button in the bottom of the case depressed with a bent open paper clip or similar. Holding the secret button whilst switching on initialises the serial EEPROM. Release the button and if all is working correctly the display will initially show HELLO after a short delay.

The decimal point of the last digit displays the raw serial signal from the MSF receiver. It should flash as the data bits are transmitted one pair per second. The clock requires most of a complete minute to receive the current time and date. When received without any detected errors, the display will change from HELLO to the correct time, and the decimal points will stop flashing.

Holding down the right button displays the date, in DDMMYY format.

Pressing the middle button toggles the display between 12 and 24 hour time formats.

To put the clock into alarm setting mode, again press the secret button in the bottom of the case. Now, the display changes, and the left button will now advance the hour setting, the middle button the minute setting. Advance the display to the required alarm setting time. To set the days of the week on which the

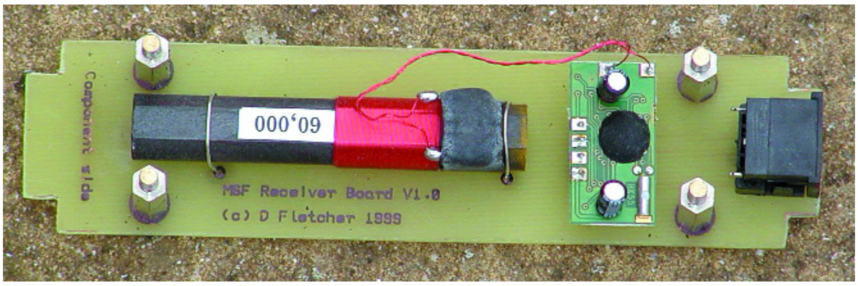


Figure 6. Completed receiver board with mini DIN socket fitted for connection to the control board.

Listing I. Button scan routine (extract).

```
                =1  1550    $INCLUDE(buttons.asm)
                =1  1551    ;Subroutine to process button presses
                =1  1552
                =1  1553    ;David Fletcher, April 1999
                =1  1554
                =1  1555    ;Example of how to use:-
                =1  1556    ;       PUSH    ACC             ;Dummy byte to reserve space on the stack
                =1  1557    ;       LCALL   DEBOUNCE
                =1  1558    ;       POP     ACC             ;Debounce return value
                =1  1559    ;       MOV     R0, A
                =1  1560    ;       XRL     A, #BUTTON_SAME ;Defined value, used by the debounce routine
                =1  1561                                    ;to tell calling code that the button state
                =1  1562                                    ;has not changed.
                =1  1563    ;       JZ      MN_KEYS_DONE
                =1  1564    ;       MOV     A, R0
                =1  1565    ;       PUSH    ACC
                =1  1566    ;       LCALL   BUTTONS
                =1  1567    ;       POP     ACC
                =1  1568
——              =1  1569            DSEG
                =1  1570
                =1  1571
                =1  1572
——              =1  1573            CSEG
04C8            =1  1574    BUTTONS:
04C8 C0E0       =1  1575            PUSH    ACC     ;Save all registers used by the routine
04CA E8         =1  1576            MOV     A, R0
04CB C0E0       =1  1577            PUSH    ACC
04CD C0F0       =1  1578            PUSH    B
04CF A881       =1  1579            MOV     R0, SP  ;Set up a pointer to the return parameter on the
stack
04D1 18         =1  1580            DEC     R0      ;which after the register saves now start six
04D2 18         =1  1581            DEC     R0      ;levels down
04D3 18         =1  1582            DEC     R0
04D4 18         =1  1583            DEC     R0
04D5 18         =1  1584            DEC     R0      ;R0 now points to the return parameter
                =1  1585
04D6 E54B       =1  1586            MOV     A, DM
04D8 B4012D     =1  1587            CJNE    A, #DM_TIME, PB_DM_NOT_TIME
                =1  1588
                =1  1589    ;
                =1  1590    ;Display Mode is TIME
                =1  1591    ;
04DB E6         =1  1592            MOV     A, @R0          ;Retrieve the new button state from the
stack
                =1  1593
04DC B40806     =1  1594            CJNE    A, #DATE_BUTT_MASK, PB_DMTIME_NOT_DATE
                =1  1595
                =1  1596            ;Deal with the date display button, only used when the display
                =1  1597            ;is in the normal time display mode. Makes the clock display the
                =1  1598            ;date whilst the button is held, then go back to time display
                =1  1599            ;when it is released.
                =1  1600    ;       MOV     C, MSF_ENABLE   ;Toggle signal ON/OFF each button push
                =1  1601    ;       CPL     C
                =1  1602    ;       MOV     MSF_ENABLE, C
                =1  1603
04DF 754B02     =1  1604            MOV     DM, #DM_DATE
04E2 020583     =1  1605            LJMP    PB_DONE
                =1  1606
04E5            =1  1607    PB_DMTIME_NOT_DATE:
                =1  1608
04E5 B40208     =1  1609            CJNE    A, #ALARM_BUTT_MASK, PB_DMTIME_NOT_ALARM
                =1  1610
                =1  1611            ;Clear the ALARM_ENABLE bit, to stop the noise!
04E8 C20B       =1  1612            CLR     ALARM_ENABLE
                =1  1613            ;Might as well zero the timeout counter, too
04EA 755200     =1  1614            MOV     ALARM_TOUT_CNT, #0
04ED 020583     =1  1615            LJMP    PB_DONE
                =1  1616
04F0            =1  1617    PB_DMTIME_NOT_ALARM:
```

```
                =1   1618
04F0 B40113     =1   1619            CJNE    A, #SEC_BUTT_MASK, PB_DMTIME_NOT_SECRET
                =1   1620
                =1   1621
                =1   1622            ;The secret button has been pressed, so set up the clock
                =1   1623            ;to adjust the alarm settings
04F3 754C00     =1   1624            MOV     ALARM_HOURS, #0
04F6 754D00     =1   1625            MOV     ALARM_MINS, #0
04F9 1207D5     =1   1626            LCALL   SEEP_RD_ALARM_DAYS_ED
```

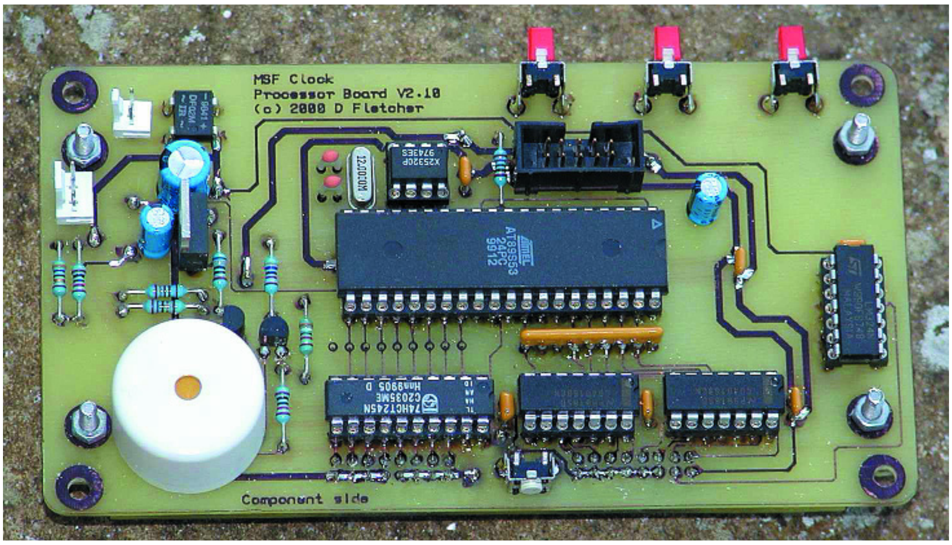


Figure 7. Control bard, ready for fitting into the enclosure. Note that the final version of the board supplied by the author may have slight improvements over the one shown here.

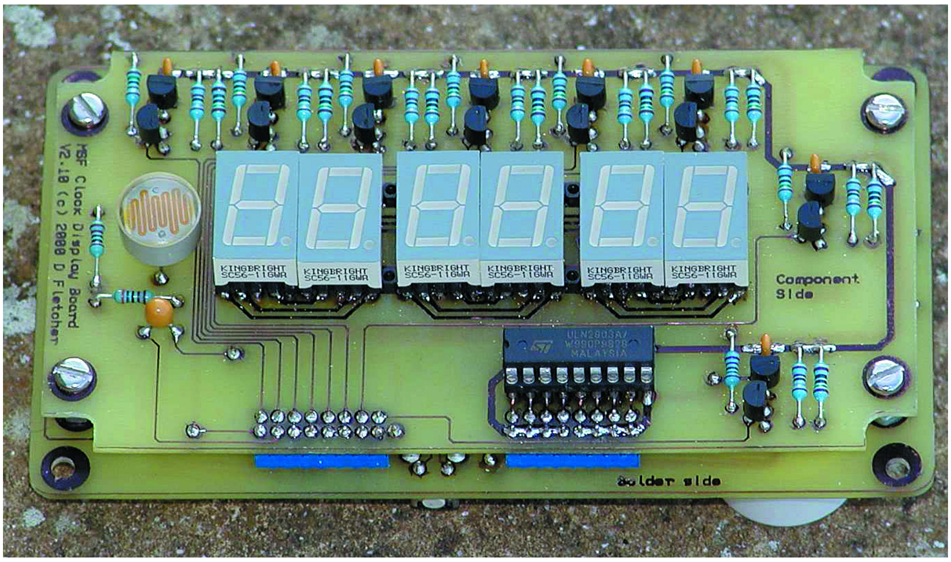


Figure 8. Finished display board mounted on top of the control board.

alarm will operate, hold down the right button. With the right button held down, the left button advances the fifth display digit through the days of the week, with

day 1 = Sunday;
day 2 = Monday;
day 3 = Tuesday;
etc.

At the required day, still holding down the right button, the middle button toggles the alarm on and off for that day.

0 = alarm off;
1 = alarm on.

When the alarm has been set for the required selection of days, release the right button. Another time can now be selected, or pressing the secret button writes the updated alarm settings into the EEPROM and returns the clock to normal time display mode.

When the alarm operates, it can be stopped by pressing the left button, or it will automatically stop after five minutes.

Hints & Conclusion

The MSF receiver is constructed, like an AM radio, using a ferrite rod antenna. This is directional, and the receiver will need to be rotated to achieve best reception.

This clock may be regarded as imperfect, due to the fact that in the circumstances of a power failure during an MSF signal break, the clock and its alarm settings will not operate until the signal returns. The design goal, however, was to produce a unit without any battery, and in the two years since it was constructed it has never once missed an alarm. So in all fairness it can be concluded that the design is a success.

If the mains supply in the UK was not as reliable as it is, the design might have had to include a battery operated real time clock module as a backup.

(020204-1)

Note:

The author can supply a set of factory made printed circuit boards. Price is expected to be £35.00 including VAT and postage. Please enquire by email only to *dave_fletcher@bigfoot.com*. The author's website is at *www.bigfoot.com/~dave_fletcher*.

USB Driver Programming (1)

a DIY developer's guide

By M. Müller and C. Ehmer

There are many advantages to connecting hardware via the Universal Serial Bus (USB), but there are also many obstacles to be overcome. In this series of articles, we provide you with a map to help you find your way along the stony path of developing USB applications.

DIY hardware for the PC is usually connected to the serial RS232 interface. After several decades of use, this interface is very well documented and easy to use. Of course, it also has certain disadvantages One of these is that you can only connect a device to it before booting the PC, and another is that powering an attached circuit from the serial interface is awkward (and only a small amount of power can be drawn this way).

The USB interface is free of these disadvantages. However, most users – including experienced PC hobbyists – know little or nothing about the USB interface. The objective of this series of article is to enable you to connect homemade USB devices to a PC. The key factor here is that almost everything you need can be downloaded from the Internet for free.

There are three elements to the development of a peripheral USB circuit:

1. The hardware and a program that runs in the USB IC.

2. A program that communicates with the USB IC and displays responses on the screen.

3. A device driver (SYS file) that provides communication between the USB port and the program.

All of this will be discussed one step at a time, but first we want to see what's available on the market. Many IC manufacturers offer USB ICs, but only a few companies provide good support. An outstanding example in this regard is Cypress Semiconductor (www.cypress.com).

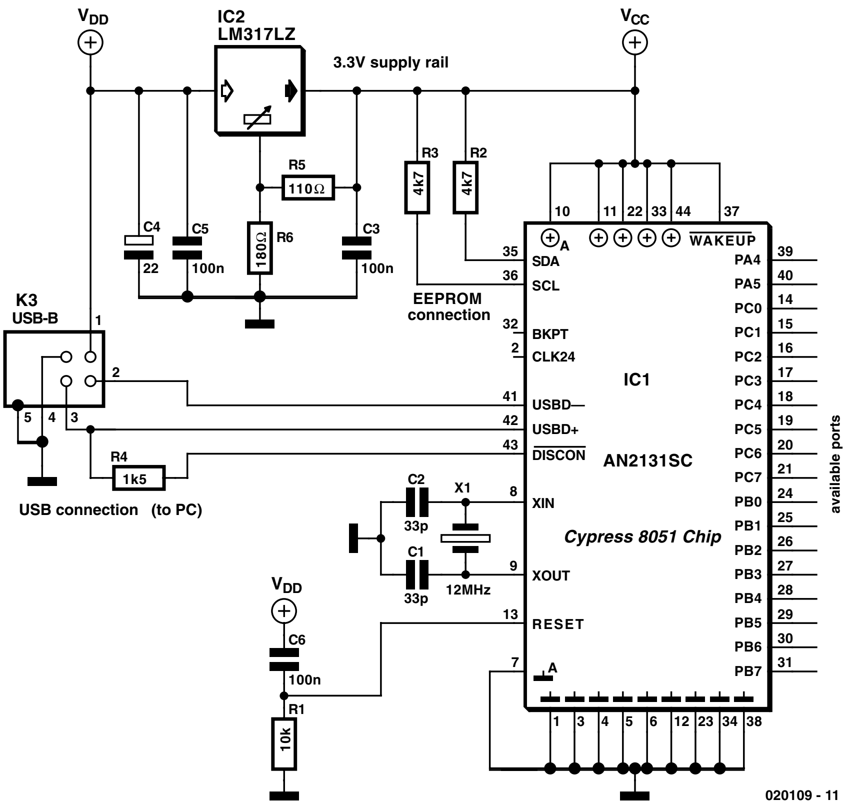


Figure 1. Minimal configuration for using the Cypress AN2131SC IC.

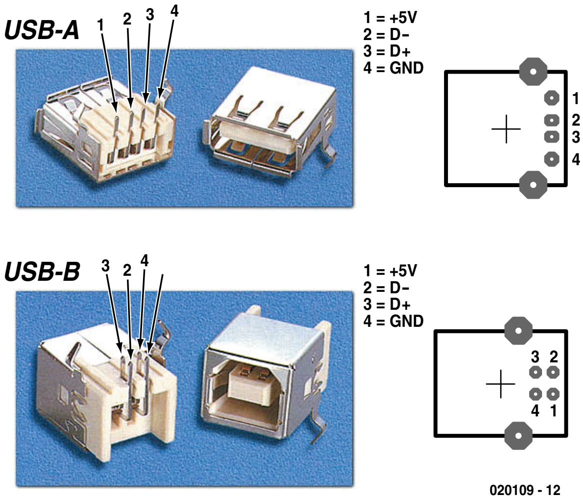


Figure 2. USB connector pin assignments.

Basic circuit for USB developers

Cypress makes our standard USB controller, the AN2131SC. The specifications of this microcontroller are provided in **Table 1**, and its data sheet can be downloaded from the Web at www.cypress.com/cfuploads/img/products/AN2131SC.pdf.

As can be seen from **Figure 1**, besides the USB microcontroller the USB connector is the second special component of the basic circuit. There are two versions, called 'Type A' and 'Type B'. The Type A version is always located on the PC or hub side, while the Type B version is always found in the peripheral device that logs in to the PC. Power can be drawn from a Type A connector, while a Type B connector can receive power. Consequently, we use exclusively Type B connectors (**Figure 2**) in our circuit. A USB cable is always wired 1:1.

A maximum of 500 mA can be drawn via the USB cable, but the Cypress IC must first report this level of current consumption, since if the PC does not have this much reserve power capacity it will automatically switch off the device. The AN2131SC IC automatically reports a value of 100 mA via its internal USB communications software. If the hardware needs more current, you must write your own software for USB communications.

The USB Controller

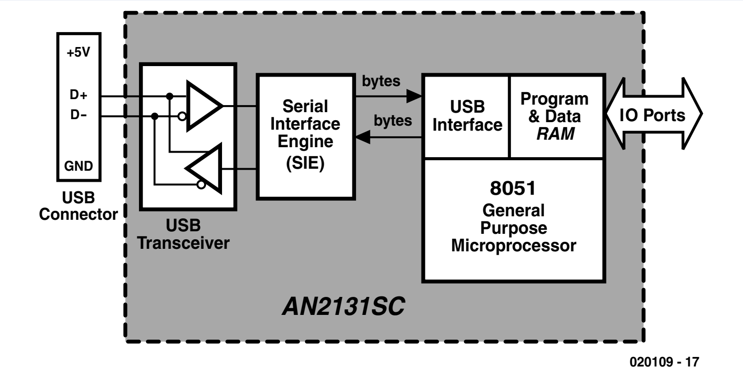
The AN2131SC single-chip microcontroller contains an 8051 processor core and a full-speed USB interface. The microcontroller has 8 KB of RAM, three counter/timers, two serial interfaces and interrupt inputs. A total of 18 freely programmable I/O pins are available.

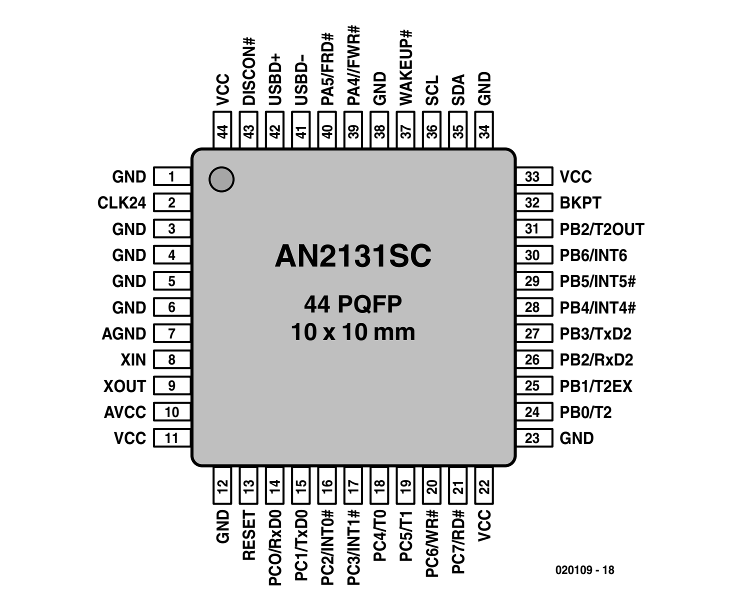
A serial EEPROM can be connected to act as program memory, or the IC can be operated using a program loaded into the RAM via USB. Practically no additional external components are necessary. The AN2131SC runs on 3.3 V, with the USB port clock frequency being derived from the 12-MHz oscillator frequency. A frequency doubler is used to generate a 24-MHz clock for the processor. Thanks to an optimised instruction set, machine instructions are executed at four times the usual rate.

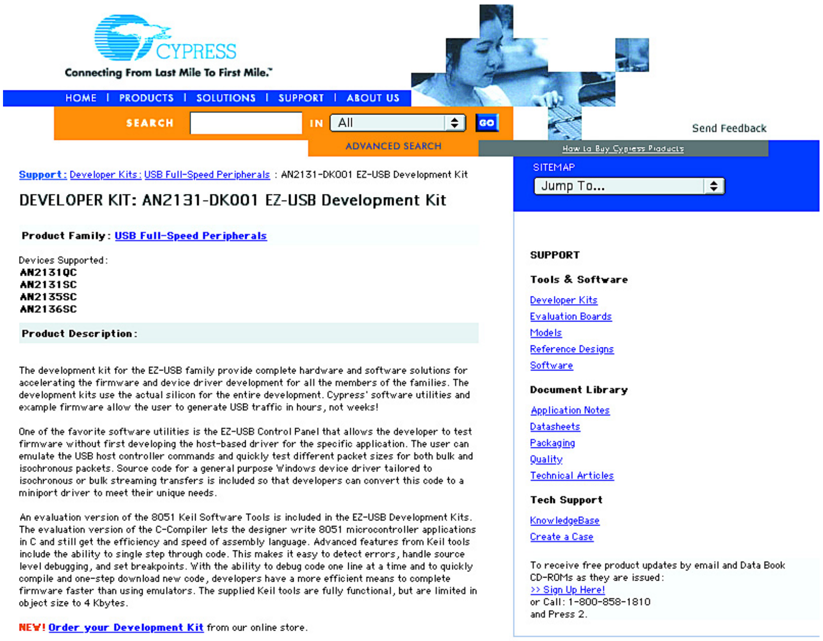
Timer 0 and Timer 1 are 8-bit timers, while Timer 2 is a 16-bit timer. The signal applied to a counter/timer input port must not exceed a frequency of 2 MHz. The serial interface can be operated at 9600 baud using Timer 0 or Timer 1. Up to 38,400 baud is possible using Timer 2. In order to achieve even higher baud rates, a different frequency must be applied to the timer input.

The IC can respond individually to five separate active-low or active-high interrupts. The I²C pins must be connected to 3.3 V via 4.7-kΩ resistors, and the Reset input must be wired to a POR network. After this, all that is necessary is to connect everything to a USB port so the PC can install a device driver.

Software updates are no longer required with the second version of the microcontroller, since with Active Software the appropriate program always runs in the USB IC. Using the built-in re-numeration feature, the USB interface can log in to the operating system with a new VID and PID.






CYPRESS
Connecting From Last Mile To First Mile.™
HOME | PRODUCTS | SOLUTIONS | SUPPORT | ABOUT US
SEARCH
IN All
GO
ADVANCED SEARCH
Send Feedback
How to Buy Cypress Products
SITEMAP
Jump To...

Support: Developer Kits: USB Full-Speed Peripherals : AN2131-DK001 EZ-USB Development Kit

DEVELOPER KIT: AN2131-DK001 EZ-USB Development Kit

Product Family: USB Full-Speed Peripherals

Devices Supported:
AN2131QC
AN2131SC
AN2135SC
AN2136SC

Product Description:

The development kit for the EZ-USB family provide complete hardware and software solutions for accelerating the firmware and device driver development for all the members of the families. The development kits use the actual silicon for the entire development. Cypress' software utilities and example firmware allow the user to generate USB traffic in hours, not weeks!

One of the favorite software utilities is the EZ-USB Control Panel that allows the developer to test firmware without first developing the host-based driver for the specific application. The user can emulate the USB host controller commands and quickly test different packet sizes for both bulk and isochronous packets. Source code for a general purpose Windows device driver tailored to isochronous or bulk streaming transfers is included so that developers can convert this code to a miniport driver to meet their unique needs.

An evaluation version of the 8051 Keil Software Tools is included in the EZ-USB Development Kits. The evaluation version of the C-Compiler lets the designer write 8051 microcontroller applications in C and still get the efficiency and speed of assembly language. Advanced features from Keil tools include the ability to single step through code. This makes it easy to detect errors, handle source level debugging, and set breakpoints. With the ability to debug code one line at a time and to quickly compile and one-step download new code, developers have a more efficient means to complete firmware faster than using emulators. The supplied Keil tools are fully functional, but are limited in object size to 4 Kbytes.

NEW! Order your Development Kit from our online store.

SUPPORT

Tools & Software

Developer Kits
Evaluation Boards
Models
Reference Designs
Software

Document Library

Application Notes
Datasheets
Packaging
Quality
Technical Articles

Tech Support

KnowledgeBase
Create a Case

To receive free product updates by email and Data Book CD-ROMs as they are issued:
>> Sign Up Here!
or Call: 1-800-858-1810
and Press 2.

Figure 3. Download page for the Developer's Kit.

A serial EEPROM can be connected to the SDA and SCL lines of the Cypress IC. Either the ID or an ID and a program can be stored in this EEPROM. If no program is stored in the EEPROM, the USB portion and the processor operate fully independently. All communications take place via the RAM. The processor can be placed in the Reset mode using USB driver commands. After this, a program can be loaded into the RAM starting at address 0. Furthermore, the USB driver has full access to the entire RAM region, even while the 8051 core is running. It's not necessary to write even a single line of code for this, since the USB portion of the microcontroller provides this capability on its own.

If you want to make a circuit for yourself, this basic circuit can serve as a good starting point. However, you should carefully consider how you use the ports. Although the port pins are freely programmable, they can also be used for timer inputs, interrupts and even two serial interfaces. If you connect ICs that have interrupt outputs, you should definitely use the interrupts. This makes programming significantly easier, even using C.

Tools from the Net

Before you can get started with actual programming, you will need a few programs and tools from the Internet, as follows:

1. The Cypress EZUSB Development Kit contains a C compiler from Kiel along with code templates for generating USB device drivers. In the local search engine, enter the IC designation and then go to the AN2131-DK001 Developer Kit (**Figure 3**). Here you can download the *EZ-USB Family Complete Tools*. The current version (July 2002) is V2.52.701, which weighs in at 63,153 kB.

2. Also download the *EZ-USB Technical Reference Manual* from the same site.

3. Finally, you will need BinTerm, which is available at no charge from the author's home page (www.mmvisual.de). You can use BinTerm to download the generated C program to the Cypress IC via the USB interface and to control the entire RAM region. The necessary device drivers are included in BinTerm.

Install the development kit, taking care to avoid making any changes to the installation path. Select the 'Custom' installation option, since the Keil C compiler is only installed with this option. When installing the Keil uVision2 C compiler, you must not change any of the installation directories, since otherwise the sample programs (which use absolute path names) will not run properly.

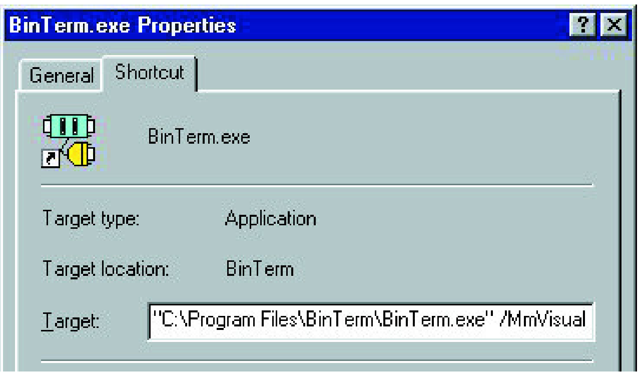
You can consult www.keil.com for additional tips on using the Keil program. One important note is that the evaluation demo from the Keil home page does not work correctly with the Cypress IC. Other compilers, such as Rigel51, generate code that is seven times as large as that produced by the Keil compiler.

Many software developers swear by assembly-language programming. A full assembly-language program is very awkward to work with, but it is easily possible to incorporate specific assembly-language routines in the C code. Code written in assembly language can only be used for a particular type of processor, while C code can be used for all other types of processors with only minor modifications.

It's not necessary to install BinTerm; all you have to do is create a new directory (folder) named BinTerm under C:\Programs and simply copy all the files into it. Drag the BinTerm program icon onto the desktop to create a shortcut. Then click on the shortcut icon with the right-hand mouse button and select 'Properties'. Modify the link by adding the startup parameter '/MmVisual' to the 'Target' field. This will cause an additional USB-Test tab to appear in the BinTerm window (see **Figure 4**).

C code for the AN2131SC

Now you're ready for the real work. Start Keil uVision2, which is the C environment for the 8051. Create a



BinTerm.exe

Target type: Application

Target location: BinTerm

Target: "C:\Program Files\BinTerm\BinTerm.exe" /MmVisual

Figure 4. Entering the additional startup parameter.

new project under Project → New Project. Enter the file name 'RamTest' and confirm it with 'OK'. The next step is to select the processor (Select Device for Target): select Cypress Semiconductor → EZUSB AN21XX → OK. In the Project Window (the left-hand window), you will see that the program has created 'Target 1' with Source Group 1.

In Target 1, open the popup menu by clicking with the right-hand mouse button and then click on 'Options for Target Target 1'. A convenient entry dialog for setting all of the compiler and linker options will appear. Select the 'Output' tab and switch on the 'Create Hex File' checkbox, then close the dialogue using 'OK'.

In order to be able to write code, you must first create a new, empty file using the menu command File → New. Next, use File → Save As to store the file in the current project directory with the name 'eMain.c'. Then use File → Close to close the file.

The file is now in the project, but it still has to be appended. To do this, select the entry 'Source Group 1' in 'Project Window', press the right-hand mouse button and select 'Add Files to Group Source Group 1'. The file eMain.c will be proposed, and you can add it to the project with 'Add'. You can now close the dialogue with 'Close'.

A new entry named 'eMain.c' will now appear under Source Group 1. A double click opens the editor, which now knows that it is dealing with a C source code and enables syntax highlighting. A major disadvantage of the C language is that the compiler distinguishes between upper- and lower-case characters.

Now you can generate a small program for the Cypress IC by using the editor to enter the code lines shown in **Listing 1**.

Press F1 to generate the project. The compiler should report '0 Errors, 0 Warnings' if the code has been correctly typed in.

How the program works

The first two lines generate two variables with fixed memory assignments. The processor starts the program using the function *main*. The endless loop *while(1)* { } is executed forever (until the next Reset). The 'meaningful task' of this routine is to copy the content of memory address 0x100 to address 0x101.

When the project is generated using F1, several files are created in the project directory. The file with the extension '.hex' can be loaded into the Cypress IC using BinTerm and then started.

Start BinTerm with the option '/MmVisual', using the shortcut icon that you have previously set up on your desktop. Select the USB-Test tab. You can control the entire Cypress IC using this program function. Plug in a USB device using the Cypress IC. The device name for the hardware must be selected according to whether you use a USB data spy or a Cypress IC with no EEPROM.

At this point, BinTerm can generate a driver diskette containing the device driver for the USB device. When the new USB device is plugged in, the operating system will attempt to install a device driver that enables the BinTerm software to communicate with the Cypress USB IC.

Built-in communications between BinTerm and the USB device must be disabled by switching on the 'stop query' checkbox located under 'controlling BinTerm'. You will also see a button with three dots next to the button labelled 'send new program.'. Click on this button to open the file 'RamTxt.hex' generated by the compiler and download it to the USB IC. Finally, click on the 'start USB program' button to cause the Cypress IC to start running.

Testing the 8051 processor

The processor can be tested by checking the memory region. This is done by entering and displaying numerical values in hexadecimal form, as shown in **Figure 5**. The option 'write in the address' can be used to write a byte directly to the RAM while the 8051 processor is running. In the example shown, the byte 0x02 will be written to address 0x100. The byte is written when the 'Send' button is activated. The option 'read from address' can be used to specify the start address for reading a number of bytes (in this example, three bytes starting at address 0x100). The 'auto-read every second' option causes the displayed value to be automatically updated.

The small example program has now been loaded into the Cypress IC and started. As you can see, the IC works perfectly. The byte in address 0x100 is copied to address 0x101, just as desired. But how is it possible for BinTerm to communicate with the Cypress IC, even though you haven't written a single line of code to drive the USB interface and haven't even set the parameters for any sort of special register?

The answer is that the USB portion of the Cypress IC works completely independently, together with the Cypress device driver, just as though a second microcontroller were managing the USB interface. Furthermore, this independent portion has full access to the RAM region of the 8051 at all times. The processor can even be placed in a Reset state and restarted via register 0x7F92. All RAM addresses can be written and read at all times, and we have taken advantage of this property. Even if you don't know anything at all about C and microcontrollers, you have just programmed one!

The Keil home page has a large number of sample programs for downloading. You can also search the Internet, where you are sure to find something you can use.

Listing 1.

```
xdata unsigned char BYTE_IN _at_   0x0100; // Byte arriving (RAM address)
xdata unsigned char BYTE_OUT       _at_   0x0101; // Byte to be written (RAM address)
void main()
{ while (1)   // endless loop
  {    BYTE_OUT = BYTE_IN;  // Copy byte from BYTE_IN to BYTE_OUT
  }
}
```

Can my own program also be so simple?

The sample programs have been generated using Delphi, which is by far the most efficient programming environment for professional use and guarantees the easy use of Visual Basic, classes and C type testing. Delphi can even be downloaded for free via the Internet from www.borland.com.

Now we come to the practical part. First you have to understand the hierarchy from the USB device to your program. You have a device that you connect to the USB interface. Based on the coding of the resistors attached to the D+ and D– data leads, the PC recognises the speed of the USB device. The Windows device driver then asks the device for its Vendor ID and Product ID. If the device is already known to Windows, the manufacturer-specific device driver is loaded. Otherwise, Windows searches its driver database for a suitable device driver and offers to install it. The manufacturer-specific device driver never communicates directly with the actual USB device, but only with the Windows driver. For each USB interface, Windows can manage one hub and several devices connected to this hub. The application program can only create a link to the USB device via the manufacturer-specific device driver. Windows provides the commands 'CreateFile' and 'DeviceIoControl' in Kernel32.dll for this purpose.

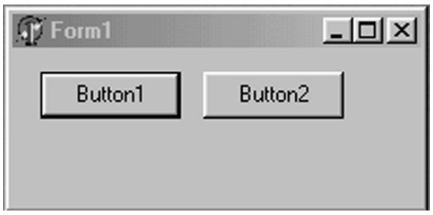
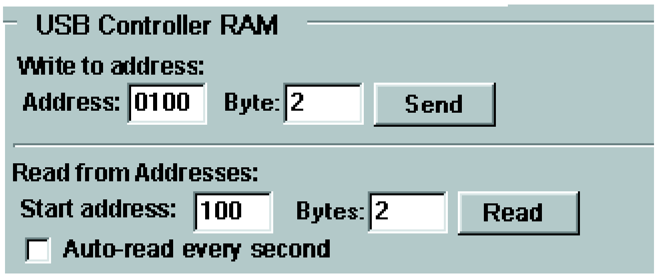


Figure 6. A form with two command buttons.



USB Controller RAM

Write to address:

Address: 0100 Byte: 2 Send

Read from Addresses:

Start address: 100 Bytes: 2 Read

□ Auto-read every second

Figure 5. Checking the RAM region.

To produce a program using Delphi, you first generate a new application using File → New → Application. An empty form will appear. Place two 'TButton' buttons in this form. Add the code shown in **Listing 2** between the end of the first User section and the beginning of the first Type section in the program.

The USB device is operated using the program RamTest.hex. If the device is unplugged, the program must be reloaded using BinTerm before the device can again be used. BinTerm can run in the background using RAM query to check the operation of the device by monitoring its RAM. BinTerm will not block the USB device.

Start the Delphi program from the Delphi compiler by pressing F9. When command button 'Button1' is activated, Byte 1 is written to RAM address 0x100 in the USB device. The value in address 0x101 can be read and displayed using 'Button 2' (see **Figure 6**).

Data in the USB device

The function 'WrRAM' can write data to the RAM of the Cypress IC. The function 'RdRAM' reads a byte from the specified address. The device name (which is correct when operating without EEPROM) is stored in the variable 'pGeraet' ('pDevice'). If a BinTerm USB data spy adapter is being used, you must use the name '\\.\binterm-0' instead of '\\.\ezusb-0'. The command 'CreateFile' generates a virtual file that creates a link to the device driver. The command 'DeviceIoControl' enables direct communication with the device driver. A record of type *_VENDOR_REQUEST_IN* is used for run-time monitoring. The command 'CloseHandle' terminates the link to the device driver. Since the link is closed after each request, it is possible for multiple programs to apparently access the hardware concurrently, without the device being aware of this.

By the way, if you rename the above-mentioned functions, all of this will also work with Visual Basic or C++.

(020109-1)

The second article in this series will deal with producing USB device drivers, which does not require a detailed understanding of C!

Listing 2.

```
type _VENDOR_REQUEST_IN = record
  bRequest : Byte;
  wValue : Word;
  wIndex : Word;
  wLength: Word;
  direction : Byte;
  bData : Byte;
end;

Insert the following code snippet at the line containing “implementation”

Function WrRAM(Adr: Word; Dat: Byte): Boolean; // Write byte to RAM address
```

```
var USBDeviceHandle: THandle;
  USBTemplateHandle: THandle;
  nBytes: DWord;
  MyRequest: _VENDOR_REQUEST_IN;
  USBError: Boolean;
  pGeraet: PChar;
begin
  pGeraet := PChar(‘\\.\ezusb-0’); // using BinTerm: ‘\\.\binterm-0’
  USBError := False;
  myRequest.bRequest := $A0;
  myRequest.wValue := Adr;
  myRequest.wIndex := 0;
  myRequest.wLength := 1;
  myRequest.direction := 0;
  myRequest.bData := Dat;
  USBTemplateHandle := 0;
  USBDeviceHandle := CreateFile (pGeraet, Generic_write, File_Share_write, nil, open_existing,
                                     0, USBTemplateHandle);
  If USBDeviceHandle = INVALID_HANDLE_VALUE Then
       USBError := True;
  Result := DeviceIoControl(USBDeviceHandle, $00222014, @myRequest, 10, nil, 0, nBytes, nil);
  CloseHandle (USBDeviceHandle);
end;

function RdRAM(Adr: Word): Byte; // Read byte from RAM address
var USBDeviceHandle: THandle;
  USBTemplateHandle: THandle;
  nBytes: DWord;
  MyRequest: _VENDOR_REQUEST_IN;
  Buffer: Array [1..2] of Byte;
  USBError: Boolean;
  pGeraet: PChar;
begin
  pGeraet := PChar(‘\\.\ezusb-0’); // using BinTerm: ‘\\.\binterm-0’
  USBError := False;
  USBTemplateHandle := 0;
  myRequest.bRequest := $A0;
  myRequest.wValue := Adr;
  myRequest.wIndex := 0;
  myRequest.wLength := 1;
  myRequest.direction := 1;     // Read
  myRequest.bData := $00;
  USBDeviceHandle := CreateFile (pGeraet, Generic_write, File_Share_write, nil, open_existing,
                                     0, USBTemplateHandle);
  If USBDeviceHandle = INVALID_HANDLE_VALUE Then
       USBError := True;
  DeviceIoControl(USBDeviceHandle,$00222014,@myRequest,10,@Buffer,SizeOf(Buffer),nBytes,nil);
  CloseHandle (USBDeviceHandle);
  RdRAM := Buffer[1];
end;

Double-click on the 1st key and complete the code with this:

procedure TForm1.Button1Click(Sender: TObject);
begin
  WrRAM($100, 1);
end;

The same for the 2nd key:

procedure TForm1.Button2Click(Sender: TObject);
begin
  Button2.Caption := IntToHex(RdRAM($101), 2);
end;
```

Acoustic Voltmeter

hear what you measure

Design by B. Kainka

With the test probe secure (you hope) on the resistor leg, turning your head to view the voltmeter indication is not always a good idea. While your eyes focus to see the measured value on the meter scale, the probe may wander off, cause a short circuit and maybe destroy a component or two. No such problems with this month's Mini Project.

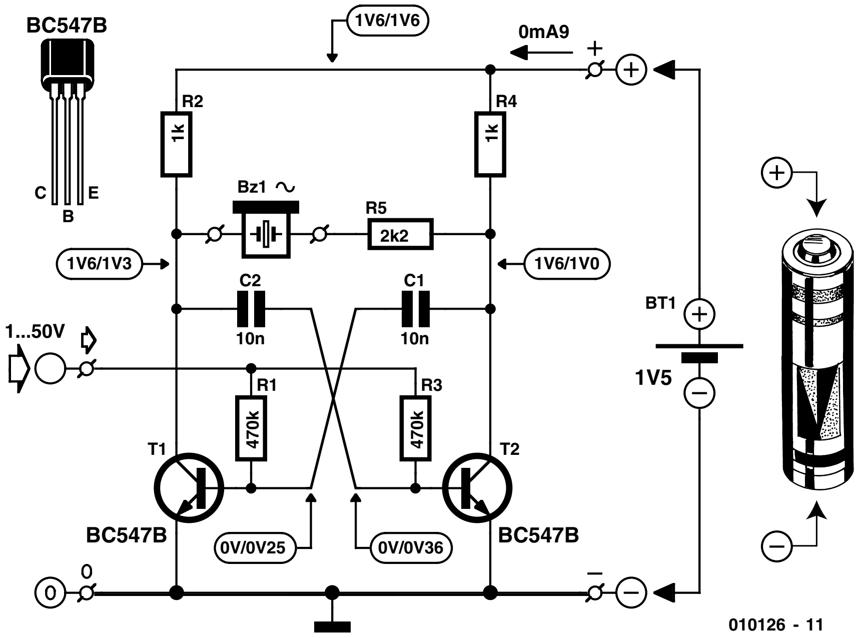


Figure 1. A simple multivibrator supplying its output signal into a passive buzzer.

The little circuit described in this month's instalment of **Mini Projects** effectively converts an electrical voltage into an acoustic (audible) signal. The higher the voltage you measure, the higher the frequency you will hear. Although this type of indication is not as accurate as the trusty voltmeter with a moving-coil meter or a digital readout, the clear advantage is that you can remain concentrated on the measurement location in the circuit without risking as much as glance at the indicating instrument. With some practice, you will soon achieve surprising accuracy using the 'acoustic' method.

Multivibrator

As you can see from **Figure 1**, the circuit consists of a simple multivibrator in which the base current to both transistors is furnished by the circuit being 'probed into'. A piezo ceramic sounder (a.k.a. 'passive buzzer' or 'ac buzzer') converts the rectangular voltage produced by the Acoustic Voltmeter into an audible signal. At a measured voltage of about 10 V, the frequency produced is about 1 kHz. From about 2 V onwards, a linear relation exists between measured voltage and frequency produced.

The input voltage range of the meter extends to well over 50 V. Negative voltages do not produce a frequency. They do not damage the circuit either, being clamped to a safe level of about –9 V by the internal zener diodes at the base terminal of the transistors. An on/off switch is superfluous as without an input voltage there is no current flow.

Alternating voltages (VAC) may also be measured without problems. These produce a rather special output sound, the frequency produced being modulated by that of the measured alternating voltage. This also enables unsteady direct voltages to be identified spot on, as well as slowly fluctuating or heavily rippled ones.

Practice!

The Acoustic Voltmeter is quickly and easily built using the spacious circuit board shown in **Figure 2**.

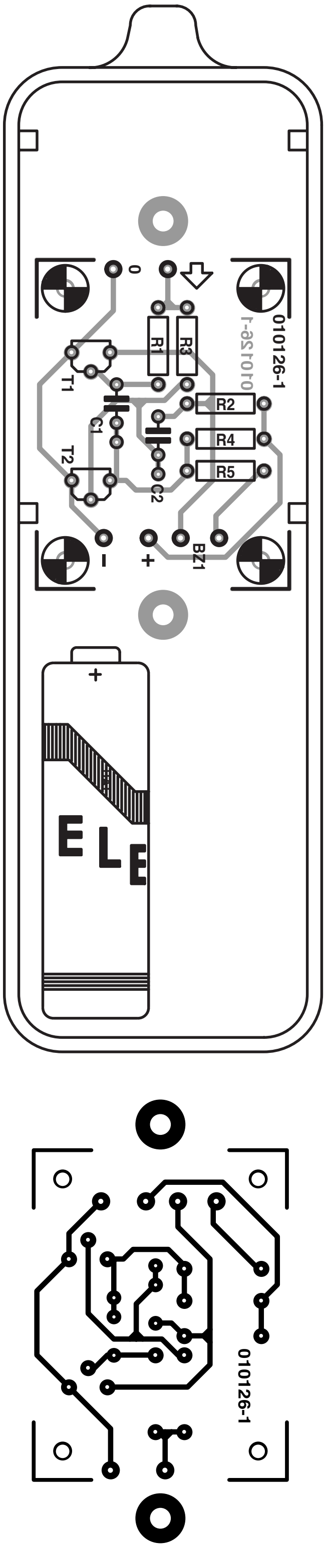


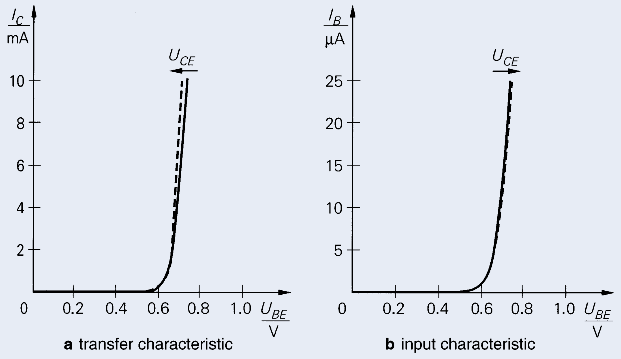
Figure 2. Copper track layout of the PCB designed for the Acoustic Voltmeter (not available ready-made).

A matter of gain

Theoretically, the voltages at the base and collector terminals of the transistors in this circuit should be identical. Although this is actually the case when the circuit is not active (first values), different values are found once everything starts working (second value). An explanation for this apparent oddity is given below.

A small-signal transistor like the BC547 is characterised by its current gain, H_{FE}, which expresses the ratio of collector current I_c to base current I_b, as in

$$H_{FE} = I_c / I_b$$



a transfer characteristic

b input characteristic

This factor is a measure of the transistor's typical static current gain. By contrast, the dynamic current gain (a measure of the current change ratio), is expressed as ß or h_{FE}. In the case of most small-signal transistors like the BC547, H_{FE} equals h_{FE} for all practical purposes.

Unfortunately it is not possible for semiconductor manufacturers to produce transistors to exact gain specifications. For example, the datasheet of the BC547 loosely indicates a gain of '110-800' at a constant collector current. Fortunately, this rather unwieldy range is divided into three sub-classes of gain, each identified by the suffix in the transistor type number. When using a BC547A, designers should expect a gain of 110-220, while a BC547B and a BC547C are marked by gain ranges of 220-450 and 420-800 respectively. Even when using two B547B's in this circuit, grave differences may occur as a result of the exact gain figure of each transistor. In spite of this, the differences are tolerable even if they result in… different measured values!

Apart from the transistors and the battery, there are no polarized components. The buzzer, too, may be fitted either way around. The completed circuit may be built into a largish probe-style case. The PCB is secured in the case using a glue gun or double-sided tape. The probe tip may be connected to the circuit input via a short piece of wire.

The buzzer is not exactly loud, being operated at just 1.5 volts. If necessary, a few small holes should be drilled in the probe case where the buzzer is secured to the inside.

You'll soon find that the actual use of our little instrument requires some getting used to. Many comparative measurements will be required against the values indicated by an ordinary voltmeter. After some time, you should be able to achieve 'acoustic resolution' of about one volt. Alternating voltages require further practicing before you are ale to tell their approximate level and frequency. Master Class users will also be able to recognize the waveform!

(010126-1)

COMPONENTS LIST

Resistors:

R1,R3 = 470kΩ
R2,R4 = 1kΩ
R5 = 2kΩ2

Capacitors:

C1,C2 = 10nF

Semiconductors:

T1,T2 = BC547B

Miscellaneous:

BZ1 = passive (AC) buzzer
Bt1 = AA (R6/UM3) battery with holder
Probe case with probe tip

Car Diagnosis Systems

Second-generation systems using the OBD-2 standard interface

by G. Müller

Car diagnosis systems have until now been manufacturer-specific and generally designed so that only authorised garage personnel can gain access to the data. This is now changing with the introduction of the standardised OBD-2 system, which has become an EU standard (EOBD) for all newly-developed cars. In this article we describe the interface in detail, and in a future article we will present a do-it-yourself EOBD diagnosis adapter.

Factors such as improved safety and greater comfort, not to mention environmental friendliness, have made modern vehicles more and more complex. As manufacturers update their ranges ever more frequently, pressures on servicing costs grow, and too little time is available for training of garage personnel, car diagnosis systems are becoming essential in order to guarantee rapid, safe and economical testing, servicing and repairs.

Computers, which have been used in cars for some time now, can also be used to diagnose the various systems and functions they control. In order to protect their authorised garages from those not attached to any particular make of car, each manufacturer has so far developed its own proprietary system. One or two standards have emerged, but only as a result of the effect of cost pressures on the supply industries. This has started to change with the national introduction of environmental legislation.

Although less densely populated, the United States was the first to take steps to reduce air pollution from exhaust gases, and still plays a leading role. In particular, the state of California has been prominent in the field of environmental legislation. For a long time European car manufacturers fought against fitting all their cars with catalytic converters, even though they had long been forced to fit them on cars for the American market. Technological advances, such as more powerful computer systems and the development of suitable sensors, combined with ever more stringent legal requirements on car exhaust gases, have led to a soon-to-be-introduced requirement for a certain proportion of cars in California to be so-called zero-emission vehicles.

Whereas here the state has passed the costs onto the driver through differential taxation for 'dirty' and 'clean' vehicles, in the United States the legislation puts the burden on the manufacturer. Manufacturers have been obliged to arrange the diagnosis systems in the car so that the driver is required to take the car into the garage even when the exhaust gas thresholds are briefly exceeded between regular services or testing. The system is called 'On-Board Diagnosis', abbreviated to OBD. In the event of excessive exhaust gases or failures in the vehicle's safety systems, a yellow 'Check Engine' lamp lights (**Figure 1**). The light is sometimes labelled 'Service Engine Soon' or 'Malfunction Indicator Lamp' (abbreviated MIL). This system has been a requirement in California since 1988, and was adopted by all the other states of America in 1994. In 1996, after further development, it was replaced by the standardised OBD-2 (or OBD-II)

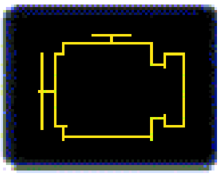


Figure 1. 'Check Engine' lamp symbol (source: author).

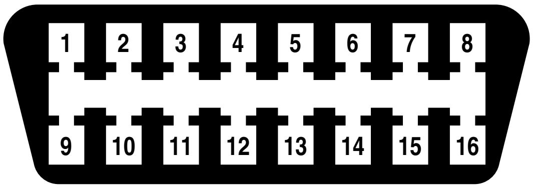


Figure 2. The 16-way diagnosis connector used in OBD-2 (source: author).

system, which is now in widespread use across manufacturers.

In an endeavour to encourage open competition, as well as in the interests of environmental protection, the European Union has issued guidelines (98/69/EU, dated 13/10/1998 [1]) for further reduction in exhaust gas levels requiring the installation of on-board diagnosis systems by all manufacturers according to the American standards. Manufacturers will be obliged for the first time to fit standardised diagnosis interfaces and connectors to allow the connection of suitable diagnosis equipment. Manufacturers will also be forced to publish details of significant parts of their diagnosis systems, which have until now been proprietary. The EU guidelines apply to spark-ignition engines (i.e., petrol engines) registered in 2000 and after, and to self-ignition engines (i.e., diesel engines) registered in 2003 and after.

OBD-2 diagnosis connector

The requirements stipulate that the diagnosis connector for OBD-2, or EOBD (European On-Board Diagnosis system), must be situated in the passenger compartment near the driver's seat. This is in contrast to previous systems where the connector has been in the engine compartment. Usually the connector will be below the instrument panel or in the central console, sometimes covered by a flap, sometimes behind the ashtray.

OBD-2 employs a 16-way connector (**Figure 2**), although not all the ways are fitted with pins (see **Table 1**).

Table 2 shows the various protocols of the bus system. It will be clear that there is little agreement between manufacturers. Among the American companies, Ford use a PWM (pulse width modulation) protocol and General Motors use a VPW (variable pulse width modulation) protocol, both to the SAE J1850 standard, while the European and Far Eastern manufacturers generally use the KWP (key word protocol) according to ISO 9141-2.

Significance

What does the new diagnosis system mean for the driver? Whereas previously most of the intelligence

Table 1

Standardised pins on the OBD-2 connector

Pin 2	J-1850 bus +
Pin 4	Chassis ground
Pin 5	Signal ground
Pin 6	CAN high (J-2284)
Pin 7	ISO 9141-2 K output
Pin 10	J-1850 bus
Pin 14	CAN low (J-2284)
Pin 15	ISO 9141-2 L output
Pin 16	Battery positive

Table 2

Real-time data readable in mode 01 (not all data is available on all vehicles):

Fuel system 1 and 2 status
Calculated load value
Engine coolant temperature
Short-term fuel-air mixture proportion bank 1
Long-term fuel-air mixture proportion bank 1
Short-term fuel-air mixture proportion bank 2
Long-term fuel-air mixture proportion bank 2
Fuel pressure (measured)
Inlet pressure
Engine speed
Road speed
Ignition timing advance for cylinder 1
Inlet air temperature
Mass air flow (MAF) sensor reading
Absolute throttle valve position
Commanded secondary air status
Installed Lambda probe locations, where probe 1 is nearest engine
Bank 1 sensor 1 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 1 sensor 2 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 1 sensor 3 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 1 sensor 4 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 2 sensor 1 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 2 sensor 2 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 2 sensor 3 Lambda probe reading and calculated short-term fuel-air mixture proportion
Bank 2 sensor 4 Lambda probe reading and calculated short-term fuel-air mixture proportion
OBD requirements, model dependent
Auxiliary input status

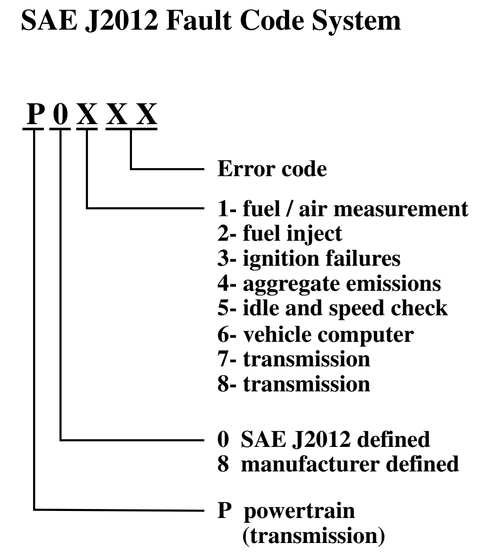


Figure 3. Fault codes according to SAE 2012 (source: author).

resided in the computer in the garage connected to the diagnosis system, this is now being transferred to the ever-more-powerful on-board computer. This means that an economical read-out device is all that is required to find the exact location of a fault in the car's systems. This does not help the ordinary driver, since in general he will not be able to sort out the problem on his own. However, it does mean that he will perhaps be able to drive to the nearest small garage or workshop instead of first making an appointment at an authorised repair shop, which might not always be feasible on a long holiday journey, for example. At least he can be spared to worry of not knowing what the illuminated service lamp means: all the manual will say is 'drive to the nearest repair shop'.

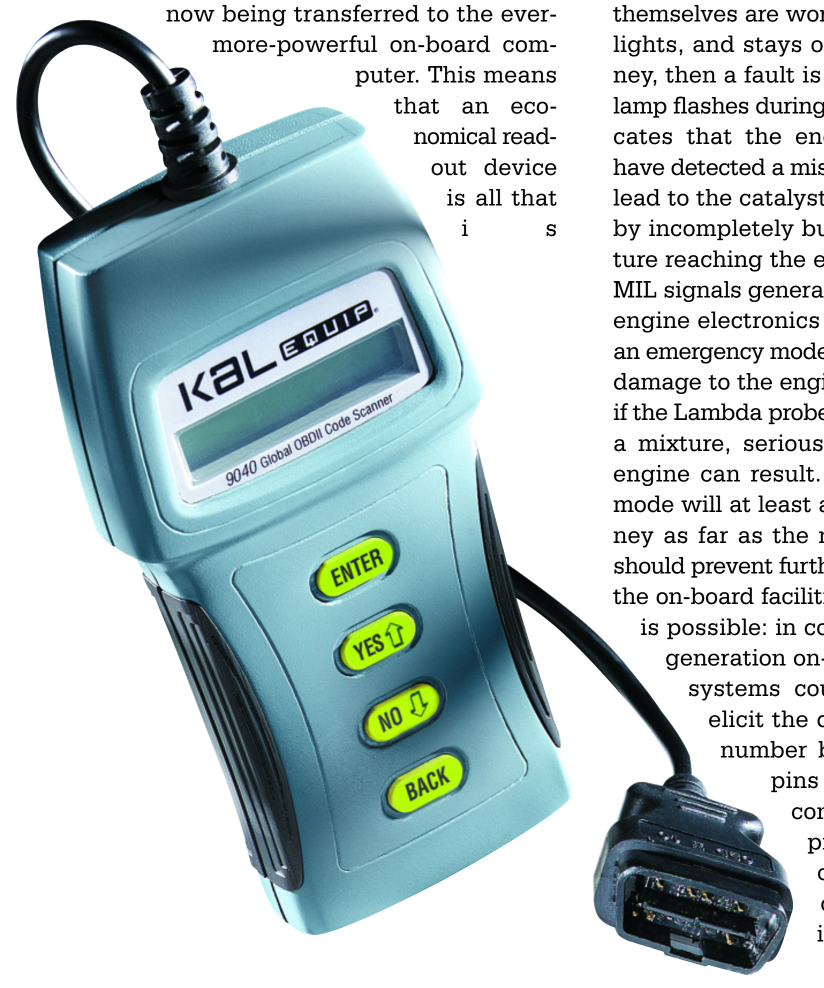


Figure 4. Typical handheld device for reading out fault codes (source: Actron Manufacturing Company, Cleveland, Ohio).

Fault conditions

When and why do the fault indicator lamps light? Along with the other indicator lamps, the Malfunction Indicator Lamp (henceforth MIL) lights when the ignition key is turned and go out when the engine starts: this tests that the lamps themselves are working. If the lamp lights, and stays on, during a journey, then a fault is indicated. If the lamp flashes during a journey it indicates that the engine electronics have detected a misfiring, which can lead to the catalyst being damaged by incompletely burnt fuel-air mixture reaching the exhaust. The two MIL signals generally mean that the engine electronics has switched to an emergency mode to try to prevent damage to the engine. For example, if the Lambda probes detect too lean a mixture, serious damage to the engine can result. The emergency mode will at least allow a safe journey as far as the next garage and should prevent further damage. With the on-board facilities nothing more is possible: in contrast, the first-generation on-board diagnosis systems could be made to elicit the display of a fault number by bridging two pins of the diagnosis connector, or by pressing a button on the central console; but this is no longer possible with OBD-2.

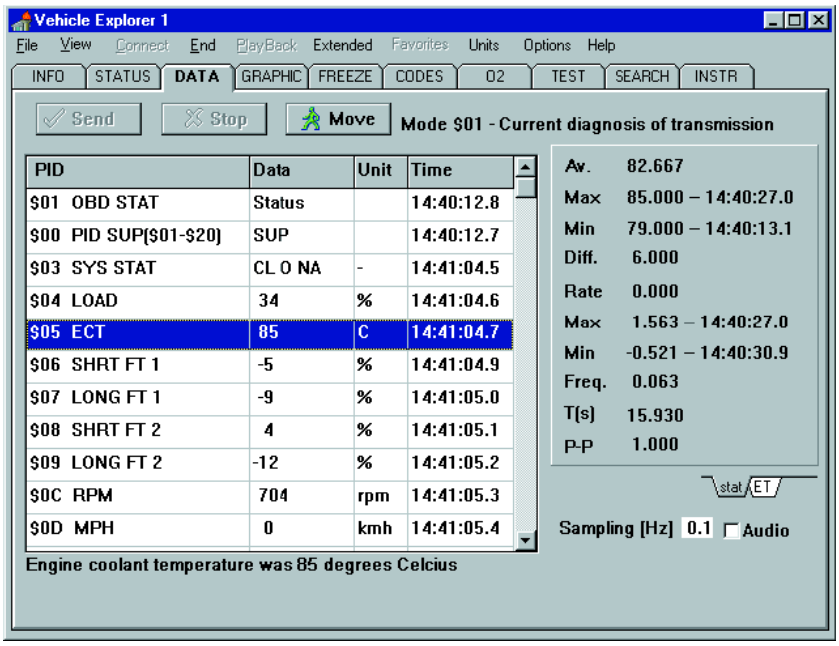
With luck, the MIL will suddenly extinguish after driving a little further. The standard stipulates that intermittent faults, which do not recur within a preset number of journeys, shall not light (or shall reset) the MIL. Nevertheless, a block of data is stored in the system containing all the data from the important sensors at the moment when the fault is detected: a so-called 'freeze frame'. This freeze-frame data can be read out at the next service to give an indication as to the cause of the problem, even when the fault has not recurred for some time. It may be the case that the fault only occurs occasionally or only under specific engine load conditions.

Fault codes

The format of the five-character fault codes specified by SAE J2012 and ISO DIM 15031-6 is shown in **Figure 3**. The OBD-2 standard defines the particular meanings shown for those fault codes that cause the MIL to light. As well as the initial letter 'P' for 'Powertrain' we can also have 'B' for 'Body', 'C' for 'Chassis' and 'U' for network faults. If the second character is '0' then we have a standard fault code. The digits '1' and '2' are used for manufacturer-defined fault codes, within which the manufacturer must adhere strictly to the provisions of the standard. The digit '3' is reserved for future expansion of the standard. The third character defines the system being monitored according to the scheme given in **Figure 3**. Finally, the last two characters give information about the fault: plain text explanations corresponding to these numbers can be looked up in the standard. So, for example, the code 'P0133' indicates that Lambda probe sensor 1 in bank 1 is responding too slowly.

Reading data out

How can we go about reading data out of the system? As we have already seen in **Table 2**, not only are there three different communications protocols that might be available on the OBD-2 connector according to the vehicle manufacturer, but we also have the problem that none of


Vehicle Explorer 1
File View Connect End PlayBack Extended Favorites Units Options Help
INFO STATUS DATA GRAPHIC FREEZE CODES 02 TEST SEARCH INSTR
Send
Stop
Move
stat
ET
Audio

Mode $01 - Current diagnosis of transmission

PID	Data	Unit	Time
$01 OBD STAT	Status		14:40:12.8
$00 PID SUP($01-$20)	SUP		14:40:12.7
$03 SYS STAT	CL O NA	-	14:41:04.5
$04 LOAD	34	%	14:41:04.6
$05 ECT	85	C	14:41:04.7
$06 SHRT FT 1	-5	%	14:41:04.9
$07 LONG FT 1	-9	%	14:41:05.0
$08 SHRT FT 2	4	%	14:41:05.1
$09 LONG FT 2	-12	%	14:41:05.2
$0C RPM	704	rpm	14:41:05.3
$0D MPH	0	kmh	14:41:05.4

Av.	82.667
Max	85.000 – 14:40:27.0
Min	79.000 – 14:40:13.1
Diff.	6.000
Rate	0.000
Max	1.563 – 14:40:27.0
Min	-0.521 – 14:40:30.9
Freq.	0.063
T[s]	15.930
P-P	1.000

Sampling [Hz] 0.1 □ Audio

Engine coolant temperature was 85 degrees Celcius

Figure 5. It is possible to use a dedicated program such as ‘Vehicle Explorer’ to read fault codes out using a PC.

these protocols is compatible with protocols commonly found on computers, such as the handshake protocol of the serial interface.

As well as the expensive analysis equipment that vehicle manufacturers supply to their authorised repair shops, an increasing number of economical devices is becoming available from third party suppliers. These are either self-contained handheld devices which are generally only capable of reading out the fault codes, or consist of an adapter cable which offers many more facilities via dedicated PC-based software. The OBD-2 connector has much more to offer than just fault codes.

A representative example of the handheld type of equipment is the Kal KM9040 (**Figure 4**). As with most devices in this category, it is also capable of erasing fault codes from the car’s computer. Prices in this category range from around three hundred to fifteen hundred pounds.

PC-based software solutions (**Figure 5**), employing an adapter cable (**Figure 6**) are also available for palmtop computers. Low-cost systems are available from around one hundred pounds. It is thus within most people’s means to poke around inside their car’s computer. The PC-based solutions generally offer access to the full range of data available from the OBD-2 connector. According to the standard, the system operates in nine modes:

01 gives real-time data
02 gives so-called ‘freeze frame’ data
03 gives fault codes
04 erases fault codes and stored values
05 gives the results of the Lambda probes self-test
06 gives the self-test results of systems that are not continuously monitored
07 gives the self-test results of systems that are continuously monitored
08 dedicated control mode
09 vehicle data request: information such as software version and VIN (vehicle identification number)

Not every vehicle supports every mode. Within each mode the vehicle computer provides a parameter identification (PID) as a hexadecimal number. For example, sending the byte sequence 01 00 asks which requests the vehicle computer can respond to and which data can be provided. After the mode has been selected, in this case 01, PID 00 will determine which further commands are understood in this mode and will be replied to.

We are now already in the middle of the details of the protocol, which are not important to the user of ready-made software products. The relatively simple format of the OBD-2 protocol, as we have illustrated, does however mean that it is possible for a programmer with a little experience to write a suitable program himself. All that is needed is for the OBD-2 communication protocol (41.6 kHz in the case of PWM, 10.4 kHz for VPW and ISO 9141) to be translated into a format suitable for the serial interface of a PC. Suitable ready-made adapter cables are also available, as are pre-programmed microcontrollers (for example from the PIC family) which allow do-it-yourself construction of an

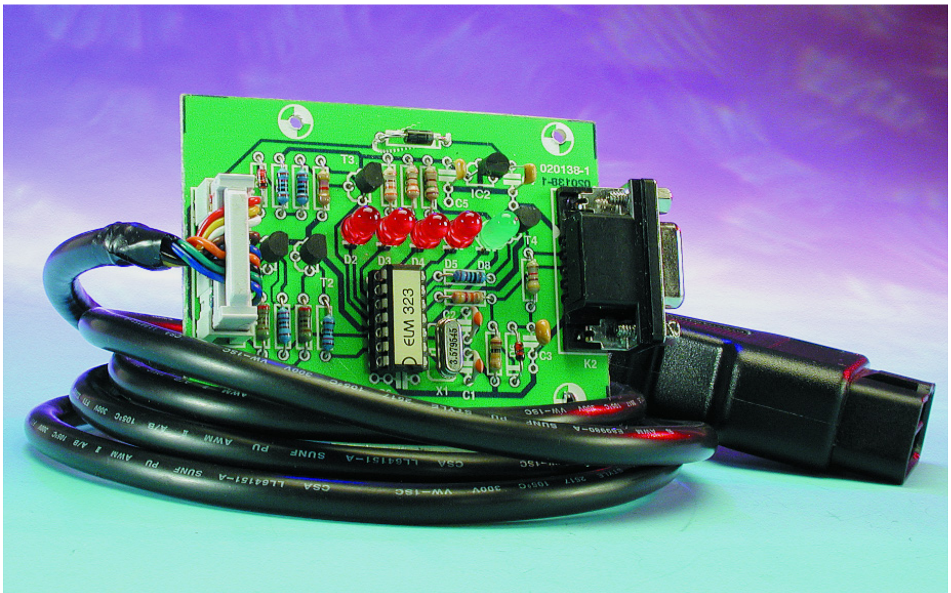


Figure 6. To use the PC-based software a suitable hardware adapter cable is required, which includes a microcontroller to convert between the two interfaces (source: Onboard Diagnostics).

adapter with a simple additional circuit (**Figure 7**). We will be describing such a project in an article in the near future.

Is it worth it?

Anyone who has been through the expensive process of having a new Lambda probe fitted might wonder whether the problem was really the probe or just a loose connection. Almost every garage survey — whether carried out by a motoring organisation or one of the many specialist magazines — has found that complete sensor systems have been replaced when the problem was just a simple, obvious cable fault. A diagnosis tool can thus pay for itself quickly.

Table 3 shows the real-time data that can be read in mode 01. Not all values are available on all vehicles: for example, not all vehicles have four Lambda probes. The calculated load value is not the power directly available from the engine, but rather a dimensionless quantity calculated as the instantaneous air flow rate divided by the maximum air flow rate corrected for altitude. The values can be continuously stored on a laptop and subsequently played back. PC software can present the data visually as graphs (**Figure 8**) or as instrument readings. Sometimes it is possible to set triggers which cause data recording to start or stop when certain values go above or below set thresholds. Usually the freeze-frame data block referred to above (mode 02) will store the same information.

In mode 04 a fault code that has occurred can be cleared. This is to be done with caution, since the operation will also clear the so-called calibration data from the vehicle's computer. Calibration data is information gathered by the vehicle's computer system itself to compensate for component tolerances in the sensors as well as production variations and wear in the engine. If these data values are cleared they will automatically be regenerated over the next journeys. This means that the car should not be taken for its MOT immediately after clearing the fault code and hence also the calibration data.

Finally a brief word to anyone hoping to tune the vehicle computer to get more power out of their car. The EU guideline states that manufacturers should take steps to ensure that 'reprogrammable computer codes or operating parameters should be protected against unauthorised access'. This is generally done using cryptographic processing and encapsulation of the relevant chips in the engine electronics. PC-based OBD-2 systems do not include cryptographic processing.

(020138-1)

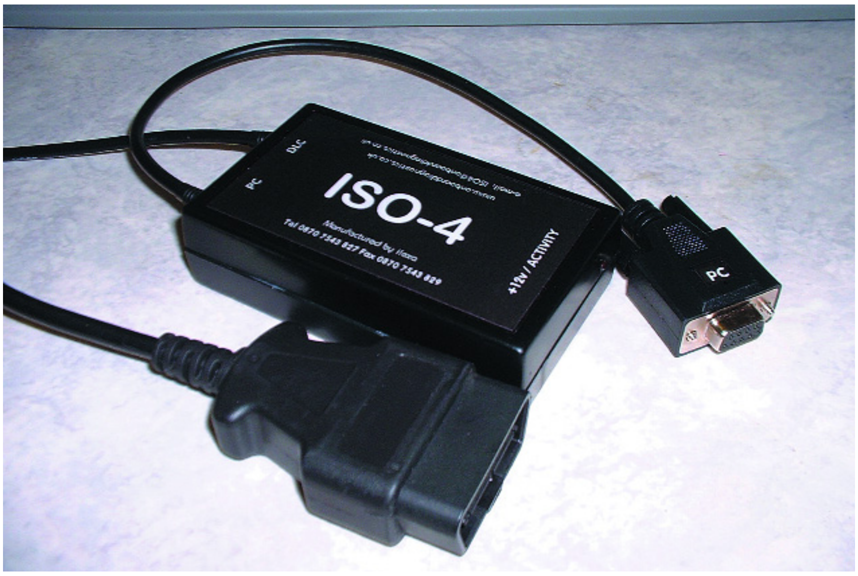


Figure 7. Elektor version of an OBD-2-to-RS232 converter. This design employs a PIC microcontroller.

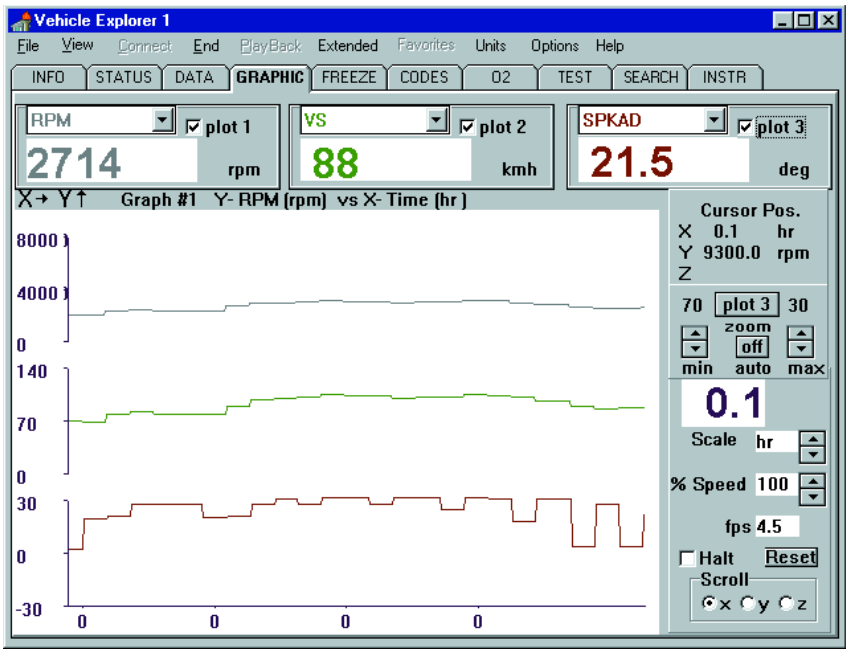
References and Web links:

[1] **EU Guidelines EC70/220, EC98/69, EC99/102, ECE R83**
http://europa.eu.int/index_de.htm

[2] **ISO and SAE standards**
www.beuth.de

[3] **Third-party OBD-2 diagnosis equipment suppliers**

www.actron.com
www.auterraweb.com
www.autoxray.com
www.abcwc.net/accounts/quanta/
www.eichstaedt-electronics.com
www.obd2.com
www.obd-2.com
www.obd-2.de
www.kalequip.com
www.mavericktechnology.co.uk
www.snapon.com


Vehicle Explorer 1
File View Connect End PlayBack Extended Favorites Units Options Help
INFO STATUS DATA GRAPHIC FREEZE CODES O2 TEST SEARCH INSTR
X→ Y↑
8000
4000
0
140
70
0
30
0
-30
0
0
0
0

RPM ☑ plot 1 2714 rpm
VS ☑ plot 2 88 kmh
SPKAD ☑ plot 3 21.5 deg
Graph #1 Y- RPM (rpm) vs X- Time (hr)
Cursor Pos.
X 0.1 hr
Y 9300.0 rpm
Z
70 plot 3 30
zoom off
min auto max
0.1
Scale hr
% Speed 100
fps 4.5
□ Halt Reset
Scroll ● x ○ y ○ z

Figure 8. The PC-based programs can present the data visually as charts or as an instrument display.

Motorcycle Theft Alarm

with dual sensors, tamper protection and remote control

Design by G. Visschers

This alarm system uses two PIC16F84 processors and is relatively easy to build and install. The cost is very low compared to ready-made systems, while its effectiveness leaves nothing to be desired!

Recently we have seen in the media that there has been a substantial increase in the number of motorcycle and scooter thefts. Large gangs operate together and steal certain makes on order. High up on the list of these criminals are the exclusive models of Ducati and Harley-Davidson, but the more expensive BMWs are also targeted, as is the £20,000 Honda Goldwing. Mind you, it's not the case that owners of less expensive machines are immune from theft, since many are stolen in that category as well. It is a sad state of affairs, but nevertheless true.

The police have recently had more success in tackling these particular crimes, but for those who care about their expensive two-wheeler a good alarm system remains a necessity.

Unfortunately, a reliable theft

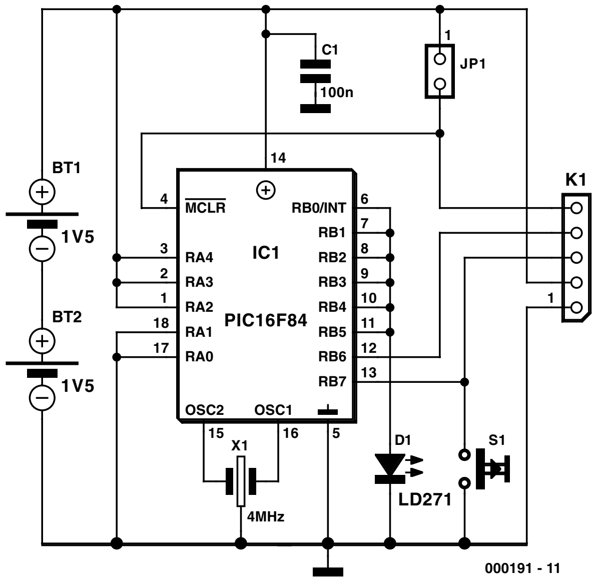


Figure 1. The remote control consists of little more than a (programmed) PIC controller and an IR LED.

alarm is anything but cheap. A decent alarm costs in the region of £250 and on top of that comes the installation charge. When faced with these amounts many people would hesitate to proceed and wonder if there isn't a cheaper alternative.

Of course it is possible to find systems that cost less, but the lower price often reflects a lower reliability. After these deliberations an electronics engineer would soon consider a DIY design. A well-designed alarm needn't be very complex from an electronics point of view, while building and installing it yourself has the advantage that the installation can be adapted optimally for the bike or scooter in question.

The biggest advantage however is the price: it is approximately a quarter of the previously mentioned £250!

Requirements

So what are the requirements of a reliable theft alarm for a bike or scooter?

It should mainly satisfy the following four points:

- The circuit should have a sensitive sensor that reacts to any movement. The infrared and ultrasonic sensors used in house and car alarms are no good for bikes. There really is a need for a type of sensor that can detect movements or vibrations mechanically.
- A conspicuous optical and acoustic alarm signal is obviously a must. This is fortunately not very difficult to implement.
- The alarm should be adequately protected against tampering. This is not so simple and requires more thought.
- And of course only the owner should be able set and clear the alarm with its correct code, preferably with the use of a remote control.

Design overview

What design issues are there when we put the previous wish list into practice and how is it implemented?

To start with we incorporated two different types of movement sensor. That may seem like overkill, but it costs very little and it's better to be safe than sorry. A pair of standard mercury tilt switches detects tilting movements of the bike. On top of that we're using a modified piezo transducer as a vibration sensor; this is particularly effective and can be made so sensitive that it reacts to a slight knock to the bike.

And then there is the tamper protection. This consists of two sections as well. An LDR (light dependent resistor) triggers the alarm when some miscreant removes the buddy seat or opens the compartment where the alarm is mounted. An extra fuse is used to prevent the alarm from malfunctioning when somebody shorts the supply of the indicators.

When the alarm is triggered the indicators of the bike start flashing and a compact but very loud piezo siren goes off. A small infrared transmitter controls the alarm and that obviously needs a matching infrared receiver module to pick up the signal.

At the hart of the circuit is a PIC16F84, containing a simple program. At this point a lot of readers may wonder if this circuit could do without a programmed microcontroller. And of course this would have been possible. But it is a fact that the circuit has to perform many functions: the signals from the sensors have to be interpreted, the IR signal from the remote control has to be detected and then arm or disarm the alarm, it has to generate control signals for the indicators and siren, by law the alarm has to be turned off after sounding for 20 seconds, when it should revert back to its armed state, and finally, the alarm may only go off a maximum of three times after it has been armed. This is a big list and one that requires an elaborate circuit if no microcontroller is used. And besides, there isn't much spare room available in a bike or scooter.

Since the issue of space is even more important for the remote control, we have gratefully made use of the services provided by a PIC16F84, this time in an SMD package. This controller has an easy job, since it only needs to produce the right 36 kHz pulsed signal for the IR transmitter LED.

Both PICs are available ready programmed via our Readers Services, but for those of you who would like to program them themselves we have made the source and HEX files available on our website free of charge under number **000191-11**.

Miniature IR transmitter

By now it's about time to take a closer look at the circuit diagrams. To start with we'll have a look at the remote control, since this circuit contains very few components apart from the small SMD PIC. This circuit is shown in **Figure 1**. Resonator X1 takes care of the required 4 MHz clock signal, D1 is the infrared transmitter LED and S1 functions as the start switch. Connector K7 has been added for the programming of the PIC, since this would otherwise be rather tricky with a SMD package. Before the PIC can be programmed jumper JP1 should be removed.

The operation of the program stored in IC1

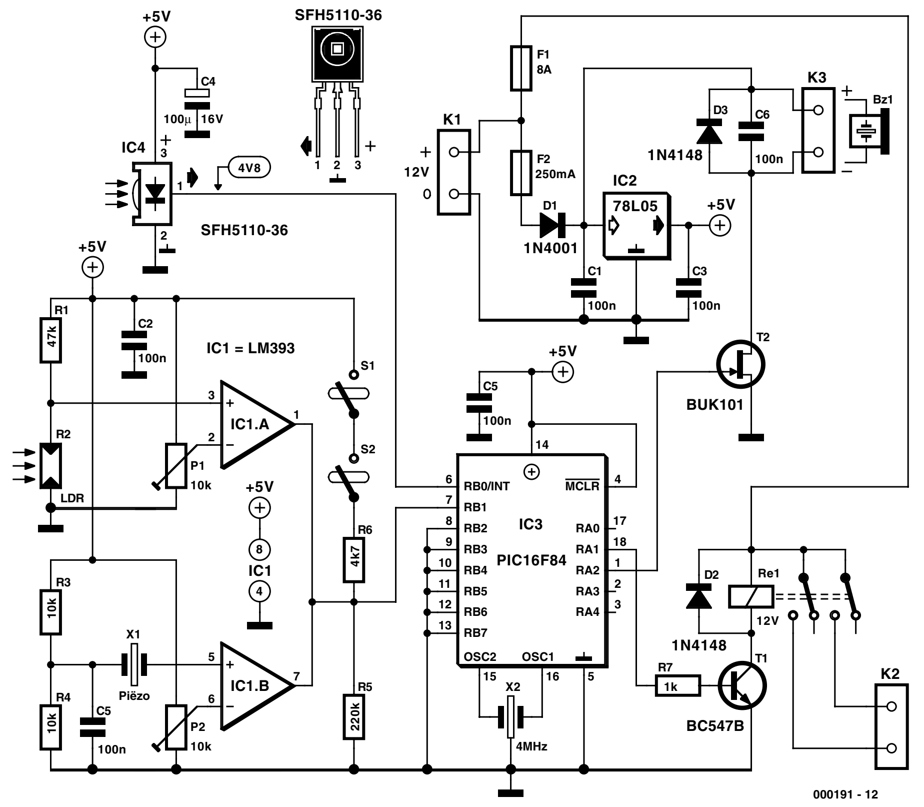


Figure 2. A programmed PIC is also found at the heart of the alarm. On the left are the IR receiver and sensors; on the right are the alarm indicators.

is very simple. When S1 is pressed the PIC comes out of its sleep-mode, after which it transmits the fixed code as a series of pulses via outputs RB0 to RB5. The PIC then reverts back to its sleep-mode. The IR code is transmitted to the receiver in the alarm by D1. The reason for the parallel connection of RB0 to RB5 is to provide sufficient current to drive the LED.

Although the supply voltage to the PIC16F84 should be at least 5 V according to the datasheets, in practice it appears that it can function reliably down to 2.8 V. For this reason the power is supplied by two 1.5 V button cells connected in series. Since the current consumption of the PIC in sleep-mode is negligible, a pair of button cells should last for at least a year.

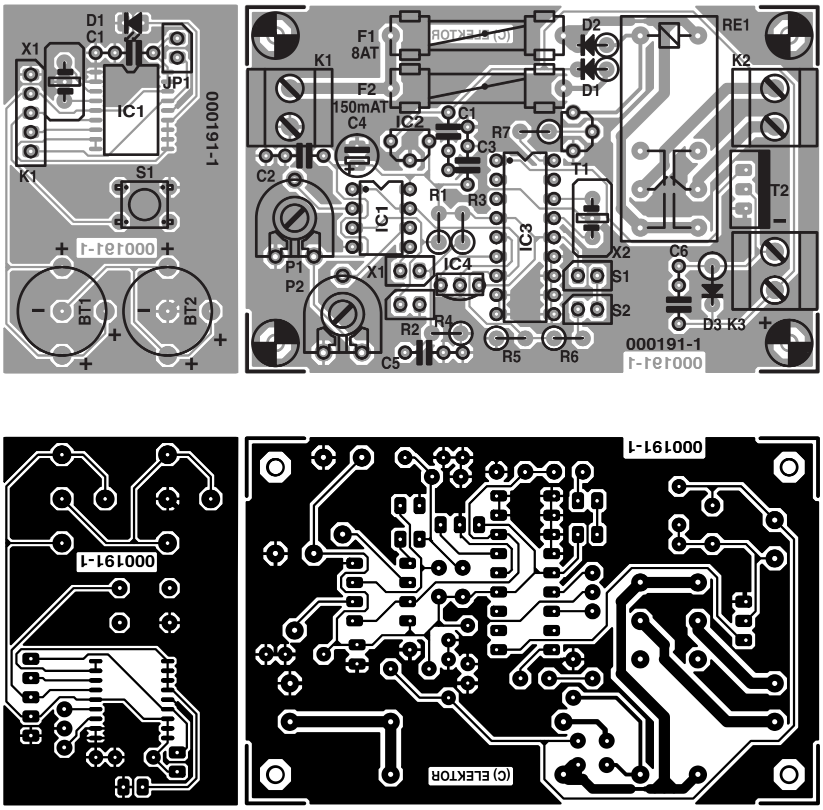


Figure 3. The transmitter and alarm come as one PCB and should be cut apart with a saw.

COMPONENTS LIST
Transmitter

Capacitors:
C1 = 100nF

Semiconductors:
D1 = IR LED, e.g., CQW13 (Conrad Electronics # 184551-89)
IC1 = PIC16F84-04/SO, programmed, order code **000191-41**

Miscellaneous:
JP1 = 2-way pinheader with jumper
K1 = 5-way pinheader
S1 = miniature pushbutton
X1 = 3-pin 4-MHz ceramic resonator
BT1,BT2 = 1.5V button cell with PCB mount holder 16×1.6 mm (e.g., CR1616)
Enclosure: Teko 11121 (Conrad Electronics # 543287)
PCB: see under 'alarm'

COMPONENTS LIST
Alarm

Resistors:
R1 = 47kΩ
R2 = LDR
R4,R4 = 10kΩ
R5 = 220kΩ
R6 = 4kΩ7
R7 = 1kΩ

Alarm circuit in detail

The circuit diagram of the actual alarm is shown in **Figure 2**. At first sight it is clear that the use of the PIC controller has kept the circuit nice and simple.

The microcontroller can be found at the hart of the circuit as IC3. Apart from the 4 MHz resonator (X2) that provides the clock signal, there is little more to be said about this because the on-chip software performs all tasks. These tasks have been listed previously. Worth mentioning is that two (unused) outputs can be used for faultfinding: RA0 reflects the status of the alarm ('0' when in standby, '1' when armed) and RA3 becomes a '1' when an IR signal is detected.

The integrated IR receiver IC4, which is found in the top-left of the circuit diagram, detects the signal from the transmitter. This should obviously be mounted in such a way that it can 'see' the transmitter and it should be connected to the PCB via a short cable. The received signal is passed on to input RB0 of the PIC, which validates the code and toggles the state of the alarm: so 'standby' changes to 'armed' and the next press of the transmitter button changes it from 'armed' to 'standby'. Two flashes of the indicators indicate each change in status. When the alarm reverts to standby the number of extra flashes indicate how many times the alarm has gone off during the previous armed period.

P1,P2 = 10kΩ

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

Semiconductors:
D1 = 1N4001
D2,D3 = 1N4148
T1 = BC547
T2 = BUK101
IC1 = LM393
IC2 = 78L05
IC3 = PIC16F84-04/P, programmed, order code **000191-42**
IC4 = SFH5110-36

Miscellaneous:
K1,K2,K3 = 2-way PCB terminal block, lead pitch 5mm
RE1 = 12V relay, 2 make contacts, 8A, (e.g., Schrack RT424012)
S1,S2 = mercury tilt switch (Conrad Electronics # 700452)
BZ1 = 12V siren, e.g., Alecto
X1 = piezo element (Conrad Electronics # 712930)
X2 = 3-pin 4-MHz resonator
F1 = fuse, 8A(T) (slow), with holder
F2 = fuse, 250 mA(T) (slow), with holder
PCB (alarm & transmitter): order code **000191-1** (see Readers Service page)
Disk, IC source code files, order code **000191-11** or Free Download

Below IC4 in the circuit diagram we find the sensor section, consisting of the mercury tilt switches (S1/S2) and two comparators (IC1ab), which take the outputs of the LDR and piezo transducer respectively. The circuit has been designed such that input RB1 becomes low when there is an alarm condition. When the bike is moved, switch S1 and/or S2 becomes open circuit and RB1 is pulled low by R6. When light falls onto LDR R2 or piezo transducer X1 produces a voltage due to vibrations, the open collector output of IC1a or IC1b respectively starts to conduct, also pulling RB1 low. The sensitivity of the LDR can be adjusted with P1 and that of the piezo transducer with P2.

When the alarm is triggered, output RA2 turns FET T2 on, which activates the powerful piezo siren connected to K3. The FET used for T2 is a type that can be driven directly by an output from a TTL compatible IC, and can switch a maximum of 5 A. Since the siren requires only about 140 mA, the FET operates well within its limits. A FET has been chosen here rather than a relay because they are about the same price, but the FET takes up much less space.

In an alarm condition RA1 turns on relay Re1 via T1, which powers the bike's indicators that are connected to K2. At each change in the alarm's status (from standby to armed and vice versa) RA1 outputs a short signal and flashes the indicators twice (plus the number of times that the alarm has gone off). D2 and D3 suppress any possible induced voltages.

The power supply consists of just a 5 V regulator (IC2), which obtains its input voltage from the bike or scooter battery via connector K1. D1 prevents any damage from occurring to the circuit when the supply is connected with a reversed polarity. As mentioned earlier, two fuses have been used to improve the protection offered by the alarm. When an inventive thief thinks of opening an indicator and shorting the supply, F1 will blow but F2 will be unaffected. The circuit remains working in this case and the alarm is still noticeable because of the siren.

And now here is a reassuring statement regarding the supply: The whole circuit takes only 12 mA, both in standby as well as armed mode. So there is no need to worry that the battery will be discharged quickly.

Two circuits on one PCB

Since the alarm circuit and the remote control always complement one another, they are supplied as one PCB (**Figure 3**). This will have to be cut into two before use. That leaves you with two compact PCBs, with the transmitter having the approximate size of a small matchbox.

The transmitter PCB only contains the bare minimum of parts, so the construction shouldn't take vary long. It's only the SMD IC that requires accurate soldering and a soldering iron with a very fine tip. For beginners it is easiest to position the IC exactly and then keep it in place with a drop of glue. Then carefully solder one pin, check the position again and then solder the remaining pins. Excess solder can be removed later using desolder braid. Check carefully that no unwanted solder bridges remain between the IC pins! If you can't get hold of the right battery holders for the button cells then you could always make your own simple frames from a few short lengths of thick tinned copper wire. That seems to work nearly as well. The button cells should be placed in their holders with the plus side facing upwards.

The construction of the main PCB shouldn't give rise to any problems either. The artwork for the components shown in **Figure 3** is clear enough and the various connections are unambiguously marked. So as far as the 'normal' components are concerned, you shouldn't have any problems. Most of the work for the main PCB will involve those parts that are not mounted directly on the board, but are external and are connected via short cables; this will be covered a little later.

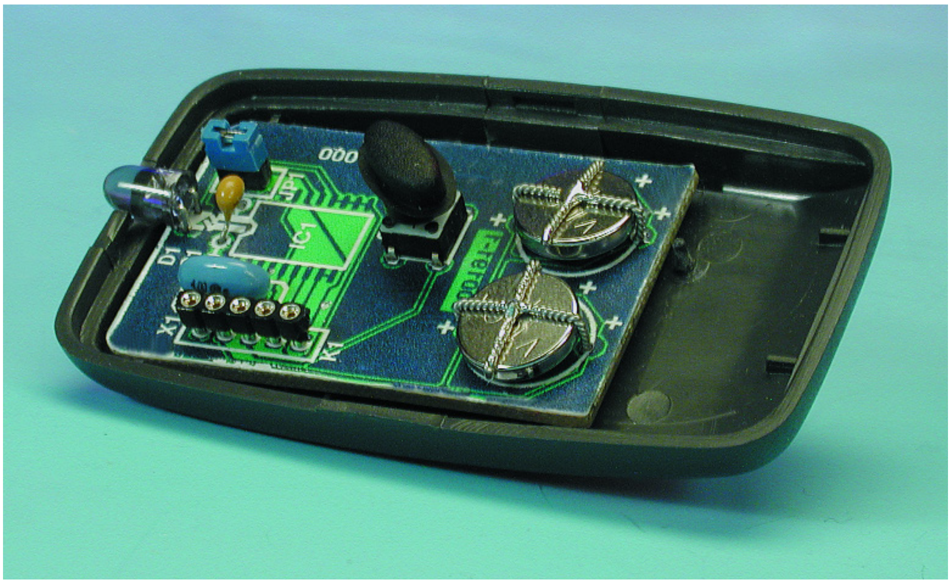


Figure 4. This small case is perfect for the transmitter PCB.

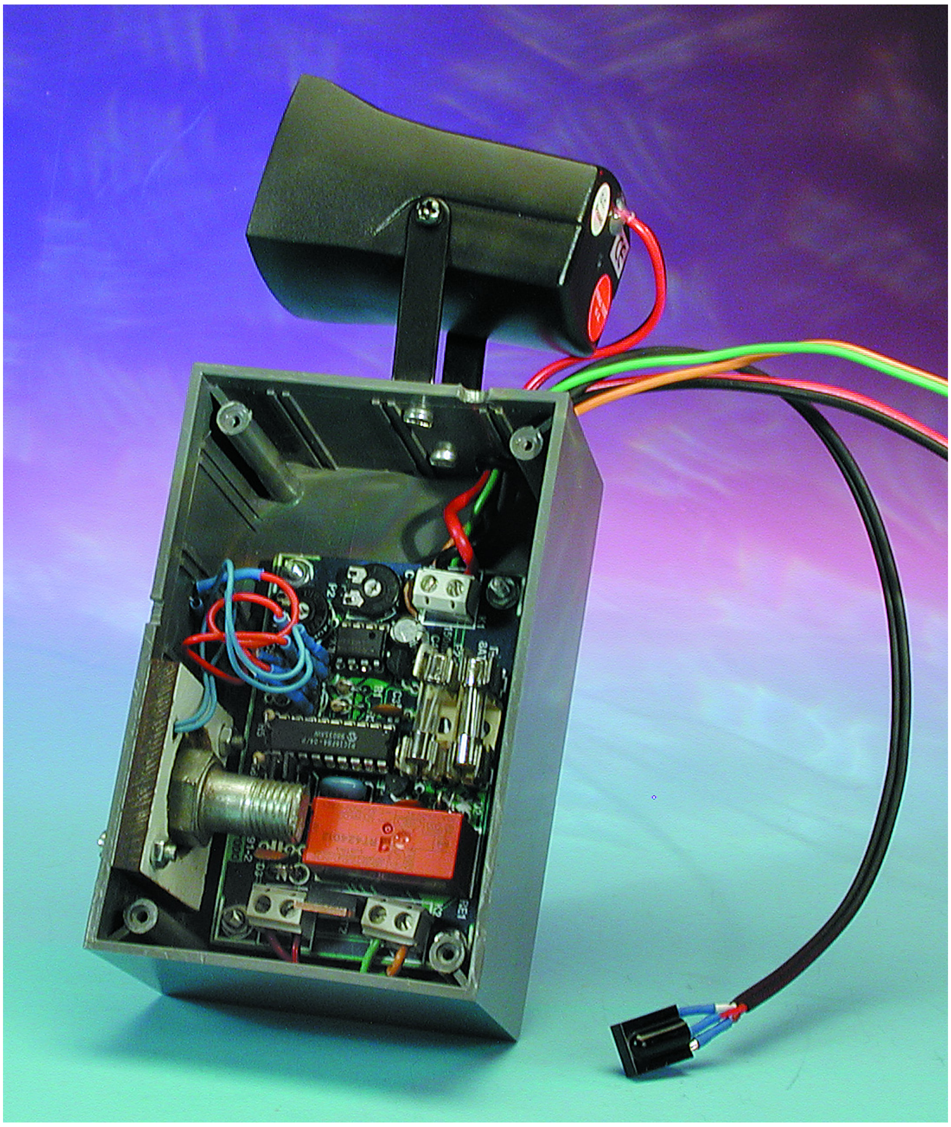


Figure 5. A view inside the prototype of the alarm. The way in which it is installed depends very much on the type of bike or scooter.

Installation

The transmitter PCB is so small that it could easily fit inside a small matchbox, including the button cells. There are also better and stronger cases that have been specifically designed for use as remote controls. For the prototype we used a case made by Teko, which can be obtained from Conrad (**Figure 4**). At the front you only need to make a small hole for D1. The box comes with a ready-made oval cutout along with a 'soft-touch' button that fits exactly above switch S1. Mounting the PCB into this box will barely take quarter of an hour.

The alarm PCB comes next. The way that is installed depends very much on the space available in the bike. Usually that can be found underneath the buddy seat, although the amount of space varies strongly from model to model. It could even be possible to leave out the case altogether and use a large piece of heat-shrinkable tubing round the PCB for insulation, and place that directly in the tool compartment of the bike. We have used a standard case to house our prototype (**Figure 5**) and we'd like to make a few important remarks regarding the construction and wiring.

It is obviously important to check that the polarity of the supply, IR receiver (IC4) and the siren are correct. Instead of the SFH5110-36 IR receiver used here, it is possible to use a SFH505; the only difference is that the connections are turned through 180° — so take care! When connecting the indicators to K2 you should take care that the left and right indicators remain isolated from each other, as otherwise they would both flash during normal operation.

IC4 should of course be mounted in such a way that the IR signal of the transmitter can reach it. A small hole will have to be made for this at the back of the bike; a slightly recessed position stops the IR receiver from 'seeing' signals that are not intended for it. A two-core shielded (audio) cable should be used to connect IC4 to the board, thereby minimising any possible interference.

The mercury tilt switches (S1 and

S2) should be mounted in such a way that they are closed when the bike or scooter is in its usual parking position (i.e., using the side or centre stand). It can be quite fiddly to find the best position for these.

LDR R2 has been mounted to the side of the case in our prototype. It is important that it is mounted in such a way that it can 'see the light' when the buddy seat is lifted, or the compartment that houses the alarm is opened.

This only leaves the mounting of the piezo transducer (X1).

Vibration sensor

The (cheap) piezo transducer used here turned out to be a very effective vibration sensor, so much so that in practice you could consider leaving out the mercury tilt switches altogether. It did however require a small modification. The transducer is glued to the side of the case (only glue along the edge, giving the active part the freedom to move!). A small weight of about 60 grams is then glued to the other side, pointing out sideways. The inertia of this weight increases the deformation of the piezo crystal that is caused by vibrations. This increases the sensitivity so much that this modification should be considered the most important part of the alarm. A 2.5 cm long M10 bolt seemed to work very well in practice.

If you want to keep everything tidy you could first make a plastic holder for the piezo transducer, using a drill and cutter. This can then be mounted to the alarm case using two screws (**Figure 6**). Since we had to experiment a fair amount on the prototype we decided on this practical solution.

Adjustments

Once the complete circuit has been constructed and the sensors, indicators and siren have been connected, you should first turn P1 and P2 fully counter clockwise before applying power from the battery via K1.

When the supply is connected for the first time the alarm system will be in its standby state. When the correct code from the remote control is received by the PIC16F84 and verified, the indicators will flash twice and the alarm will be armed. You can verify this state by looking at output RA0 of IC3: this should now be high.

P1 and P2 are then carefully turned clockwise until the required sensitivity has been set. Take care that you don't turn P2 too far because the piezo transducer can be made so sensitive that the alarm is almost triggered just by looking at the bike or scooter!

(000191-1)

Change the code?

In practice it is of course not very secure if all constructors of this alarm use the same infrared code to turn the alarm on and off. For those of you who know how to program, or have a friend who could help them, this can be rectified easily. You can easily change the fixed code into a personal one. For this you'll need an assembler and PIC programmer.

The source code files for both PIC controllers can be freely downloaded from the 'Free Downloads' section of our website (www.elektor-electronics.co.uk). The file is called **000191-11.zip** and the codes for the ready-programmed PICs are 000191-41 and 000191-42.

In both files the lines containing 'code 1' and 'code 2' should be searched for. The current values of 'F2' and '45' can then be changed to any value in the range '00' to 'FF'. Do make sure that both controllers have the same values! The modified programs are then assembled and finally burnt in the PIC controllers with a programmer.

A suitable DIY programmer for the PIC16F84 can be found on page 40 of the December 2000 issue of Elektor Electronics.

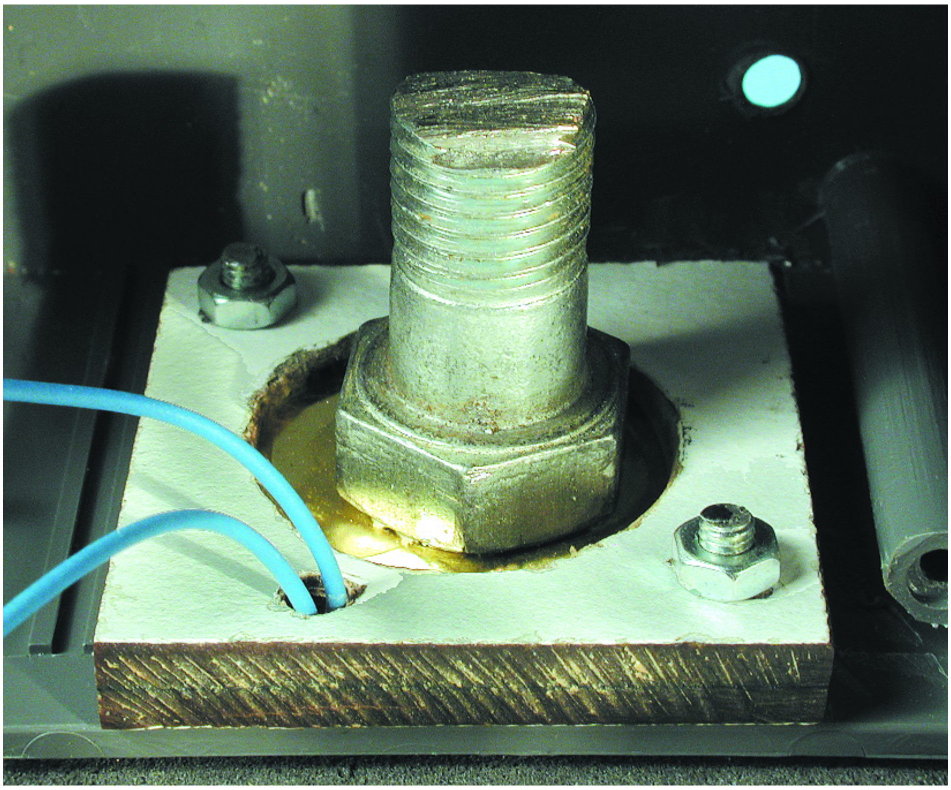


Figure 6. The piezo transducer is provided with the recommended counterweight, and then glued into a plastic frame.

Electronic Fuse

protection 'deluxe'

Design by U. Licht

In the hectic of 'just a quick test' the voltmeter probe slips off the voltage regulator pin, causes a short circuit and a handful of expensive semiconductors to give up the ghost. Although the short-circuit proof benchtop power supply will protect itself, it will briefly pump enough current into the load to cause extensive damage. The circuit described here was designed to act as a fuse, but one that's resettable, adjustable and fast-acting.

Note: this circuit has not been tested in the *Elektor Electronics* design laboratory. (*Ed.*)

Even a cursory look at the circuit diagram in **Figure 1** should convince most of you that this is not the umpteenth run of the mill electronic fuse circuit. The 'fuse' is a BTS432E2 from Infineon, whose datasheets may be found at www.infineon.com/cmc_upload/0/000/008/685/bts432e2.pdf)

This component from the PROFET series is basically an n-channel power FET with a lot of safety features built in. In normal operation, the FET is switched on and off via

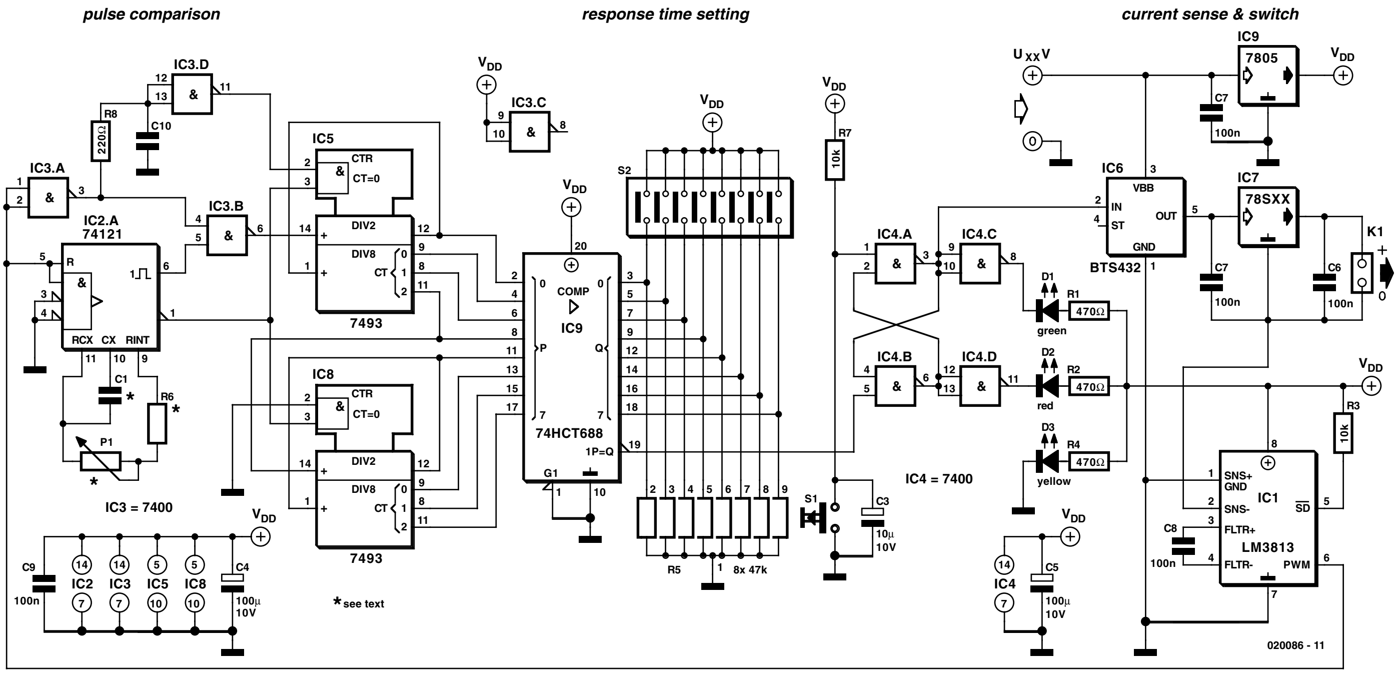


Figure 1. The circuit, which may be divided into sections referred to by 'pulse comparison', 'response time setting' and 'switch/current sensing'.

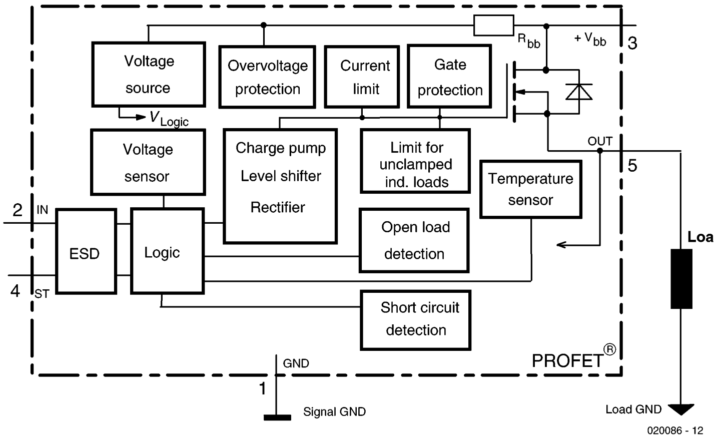


Figure 2. Internal workings of the PROFET.

the IN input. However, if either overvoltage or undervoltage condition arises, a short circuit to Vbb or ground, an open out output or a too high temperature, the BTS432ES takes its own measures against damage. A block diagram of the device is given in **Figure 2**.

The second essential component is the 'switching and current sensing' department is the LM1813 in position IC1. This precision current source monitors the current flow in the ground rail, introducing a minimum voltage drop itself. Its PWM output signal is a measure of the current flow through the Sense– and Sense+ pins. A positive current causes a pulse width ratio of 0.5 (at 0 A) to 0.955 (at 1 A). If a current is measured in the opposite direction, the pulse width ratio becomes 0.5 to 0.045 (at –1 A). The LM1813 is available in two versions: one for ±1 A and another for ±7 A. Its response time (which implies the delay between an overload condition and the IC signalling it) is about 50 ms. For an even faster electronic fuse, you may want to use the LM3813, which is marked by a response time of just 6 ms. The respective datasheets (with information on High-Side measurements) may be found on the National Semiconductor website at

www.national.com/ds/LM/LM3812.pdf
www.national.com/ds/LM/LM3814.pdf

IC9 is an ordinary fixed voltage regulator, supplying a regulated +5 V rail to the electronics in the fuse circuit. IC7 determines the actual output voltage.

Comparisons and values

The PWM signal — whose on/off ratio is a measure of the load current — is fed to monostable multivibrator circuit IC2. The LM3813 triggers the MMV type 741121 (IC2) every 50 ms. At the outputs we find a pulse (PMAX), whose length depends on the setting of pot P1.

The PWM signal is inverted by IC3a and then compared with PMX in IC3b. As long as the load current is within the allowable range, the output of IC3b remains High. If not, a Low pulse is produced as long as the PWM exceeds PMAX.

The Low pulses clock the cascaded 4-bit counters type 7493. The delay of $\overline{\text{PWM}}$ by the R-C network and IC3d ensures that the PWM pulse is actually recognised s a clock pulse before IC2 is able to supply a comparison pulse. In other words, the delay compensates the response time of the 74121.

The two binary counters are connected to an 8-bit magnitude comparator type 74HCT688. Its output, $\overline{\text{P=Q}}$, goes active (Low) when the counter value at the P inputs equals the (binary) value a the Q inputs. The latter value is determined by you, the user, by means of the DIP switch in the circuit. Using the 50-ms version of the current detector IC, the response time of the electronic can be set to 50 ms to 12.8 s in 50-ms increments. When the 6-ms version is used, the range is 6 ms to 1.5 s in 6 ms increments.

If the allowable current is exceeded before the preset counter value is reached (i.e., the load current drops below the maximum again), the counter is reset to 0 via the inverting output of the 74121. in other words, the fuse does not react to current surges which last shorter than the user-defined response time.

The $\overline{\text{P=Q}}$ pulse causes flip-flop IC4 to be set. In turn, the flip-flop switches off the supply voltage to the load via IC6. The flip-flop is reset by pressing the pushbutton.

LEDs D1, D2 and D3 indicate the fuse status: yellow = 'on'; green = 'current below trip level'; red = 'fuse activated, load disconnected'.

Practical use

The Electronic Fuse may be used in two basic circuit configurations. Either the unregulated voltage source is fused and the voltage regulator connected behind the fuse, or a regulated voltage supply is protected by the Electronic Fuse. This requires the positive output to be connected to IC6.

(020086-1)

IC1	Max. current	Response time	C1	P1	R6
LM3813M-1.0	±1 A	50 ms	1 μF	30-40 kΩ	2 kΩ
LM3813M-7.0	±7 A	50 ms	1 μF	30-40 kΩ	2 kΩ
LM3815M-1.0	±1 A	6 ms	100 nF	40 kΩ	wire
LM3815M-7.0	±7 A	6 ms	100 nF	40 kΩ	wire

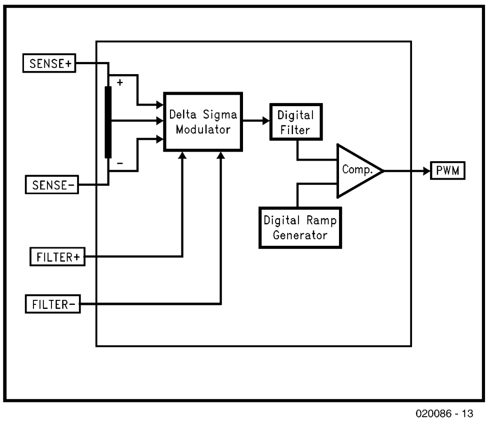


Figure 3. Curent measurement using LM1813/LM3813.

Battle of the DVD Standards

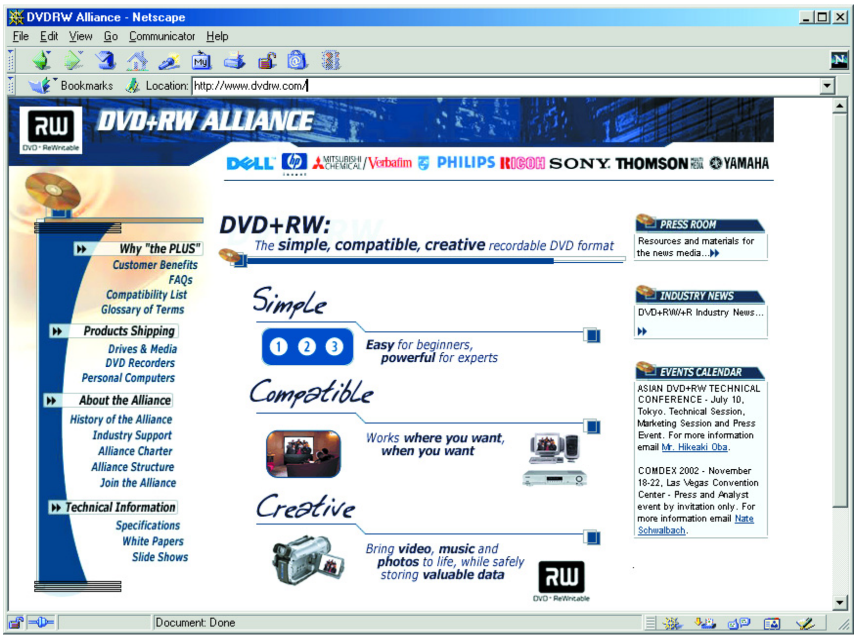
DVD-RW versus DVD+RW

By Harry Baggen

DVD burners are here to stay and we can reasonably expect them to supersede CD-burners in PCs before long. After all, a DVD can hold 4.7 GBytes yes that's seven times the amount of data you can cram on a CD-ROM. About 10 manufacturers are shipping models already, with prices coming down quickly from the £300 mark. There's one niggling problem, though: some bickering has arisen over a common standard to be adopted.

During the past two years or so, the DVD has become popular as a replacement for the pre-recorded VHS tape. A glut of popular films and concerts is currently available on DVD from the high street stores offering superb image and (surround-) sound quality. This has become possible thanks to the DVD's huge storage capacity of 4.7 Gigabytes per layer.

With the DVD a resounding success and embraced by the public, the next logical step is, of course, the recordable DVD. In 1997, the first DVD recorders were introduced, at then astronomical prices. Since then, prices have come down to consumer (= more acceptable) levels and an internal DVD burner can now be bought for just under 300 pounds. Prices of media (i.e., blank DVD disks) are between 3 and 9 pounds each. So what, you may ask, deters us from buying one of these mass storage devices, just for data or perhaps even to record the family's holiday videos in MPEG2 format? Nothing, or…?



DVD+RW: The simple, compatible, creative recordable DVD format

Easy for beginners, powerful for experts

Works where you want, when you want

Bring video, music and photos to life, while safely storing valuable data

Resources and materials for the news media…

DVD+RW/+R Industry News…

ASIAN DVD+RW TECHNICAL CONFERENCE - July 10, Tokyo. Technical Session, Marketing Session and Press Event. For more information email Mr. Hikeaki Oba.

COMDEX 2002 - November 18-22, Las Vegas Convention Center - Press and Analyst event by invitation only. For more information email Nate Schwalbach.

Things are never as easy as they seem at first glance. Just as various standards marked the introduction of video cassette tapes for the consumer and semi-professional market, there are now two competing DVD camps. One consists of a couple of manufacturers complying with the only official standard released for the recordable DVD medium. This standard is available at the **DVD Forum** [1] which sets all the important DVD formats, including DVD-RW and DVD-R, where the suffix 'RW' indicates read/write (i.e., erasable), and 'R' read only (i.e., write once), just as with CD-ROMs. Pioneer and Matsushita (Panasonic) are the main names in this group and they follow suit by producing DVD burners to this standard.

Last year, however, a number of large manufacturers launched a new DVD standard marked b a plus sign: DVD+RW and DVD+R. The plus indicates certain advantages over the 'minus' format, including (claimed) better compatibility with existing DVD players and drives, and suitability for video as well as data. Besides, a few additional, attractive, features have been included. For example, this standard allows the user to choose between constant linear velocity (CLV as with ordinary CD-ROMs) and constant angular velocity (CAV) when burning a DVD, CAV enabling faster access times to be achieved for data use. Other features include 'defect management' for higher reliability and 'quick formatting' to save time if you want to use a blank disc straight away. For video use, 'lossless linking' is available to make the system compatible with variable bit rates.

In the 'plus' camp we find almost all big guns like Philips, Sony, HP, Yamaha, Ricoh and Dell. They are united under the name **DVD+RW Alliance** [2]. An equally interesting, though unofficial, site is called **DVD-plusRW** [3], it contains lots of news and information from the 'plus camp. As a remarkable detail, most of these manufacturers are also a member of the DVD Forum, but have failed so far to push this format to the status of 'standard'.

The upshot is that two different types of DVD burners are currently found on the market, each using their own media and refusing to take that of the competition as far as writing is concerned.

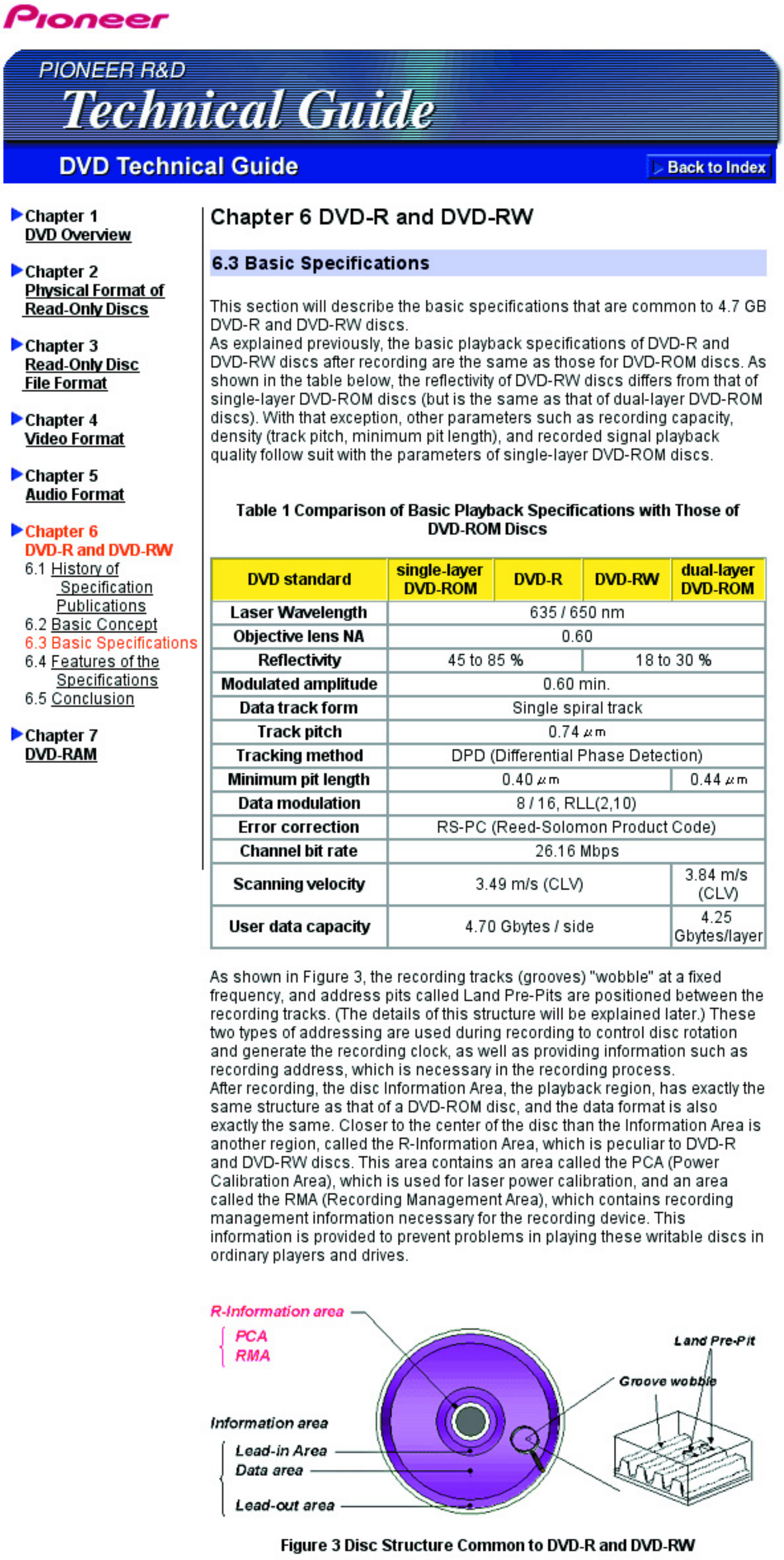
If you want to know more about the technical aspects of the two rivalling formats, have a look at the **White Papers** of the DVD Alliance [4] and Pioneer's **DVD Technical Guide** [5] (on DVD RW of course).

As far as compatibility is concerned, the two formats are near equal performers. Although DVD+ claims better compatibility, several tests run by computer magazines indicate about equal performance of 'plus' and 'minus'. DVD-R and DVD+R discs seem to be accepted by nearly all modern drives and players. Worse results are generally obtained from rewritable DVDs. An extensive compatibility overview is available on the web at **VCDHelp** [6].

Who will win the market for DVD burners? Since the 'plus' format is supported by all the big names in the industry, it seems to stand a fair chance of success. The 'minus' supporters are a minority and that may turn out to be a significant disadvantage (reminiscences of Sony's lonely battle for its Betamax system). Microsoft has shown an interest in the DVD+ format and intends to offer support for it in future versions of Windows. That could well be the decisive factor.

Until then, consumers will have to decide: is it going to be a 'plus' or a 'minus' DVD burner?

(025060-1)



Pioneer

PIONEER R&D

Technical Guide

DVD Technical Guide

Chapter 1
DVD Overview

Chapter 2
Physical Format of Read-Only Discs

Chapter 3
Read-Only Disc File Format

Chapter 4
Video Format

Chapter 5
Audio Format

Chapter 6
DVD-R and DVD-RW
6.1 History of Specification Publications
6.2 Basic Concept
6.3 Basic Specifications
6.4 Features of the Specifications
6.5 Conclusion

Chapter 7
DVD-RAM

Chapter 6 DVD-R and DVD-RW

6.3 Basic Specifications

This section will describe the basic specifications that are common to 4.7 GB DVD-R and DVD-RW discs.
As explained previously, the basic playback specifications of DVD-R and DVD-RW discs after recording are the same as those for DVD-ROM discs. As shown in the table below, the reflectivity of DVD-RW discs differs from that of single-layer DVD-ROM discs (but is the same as that of dual-layer DVD-ROM discs). With that exception, other parameters such as recording capacity, density (track pitch, minimum pit length), and recorded signal playback quality follow suit with the parameters of single-layer DVD-ROM discs.

Table 1 Comparison of Basic Playback Specifications with Those of DVD-ROM Discs

DVD standard	single-layer DVD-ROM	DVD-R	DVD-RW	dual-layer DVD-ROM
Laser Wavelength	635 / 650 nm			
Objective lens NA	0.60			
Reflectivity	45 to 85 %		18 to 30 %	
Modulated amplitude	0.60 min.			
Data track form	Single spiral track			
Track pitch	0.74 µm			
Tracking method	DPD (Differential Phase Detection)			
Minimum pit length	0.40 µm			0.44 µm
Data modulation	8 / 16, RLL(2,10)			
Error correction	RS-PC (Reed-Solomon Product Code)			
Channel bit rate	26.16 Mbps			
Scanning velocity	3.49 m/s (CLV)			3.84 m/s (CLV)
User data capacity	4.70 Gbytes / side			4.25 Gbytes/layer

As shown in Figure 3, the recording tracks (grooves) "wobble" at a fixed frequency, and address pits called Land Pre-Pits are positioned between the recording tracks. (The details of this structure will be explained later.) These two types of addressing are used during recording to control disc rotation and generate the recording clock, as well as providing information such as recording address, which is necessary in the recording process.
After recording, the disc Information Area, the playback region, has exactly the same structure as that of a DVD-ROM disc, and the data format is also exactly the same. Closer to the center of the disc than the Information Area is another region, called the R-Information Area, which is peculiar to DVD-R and DVD-RW discs. This area contains an area called the PCA (Power Calibration Area), which is used for laser power calibration, and an area called the RMA (Recording Management Area), which contains recording management information necessary for the recording device. This information is provided to prevent problems in playing these writable discs in ordinary players and drives.

Figure 3 Disc Structure Common to DVD-R and DVD-RW

Internet addressses

[1] www.dvdforum.com/forum.shtml
[2] www.dvdrw.com/
[3] www.dvdplusrw.org/
[4] www.dvdrw.com/whitepapers.html
[5] www.pioneer.co.jp/crdl/tech/dvd/6-3-e.html
[6] www.vcdhelp.com/dvdplayers.php