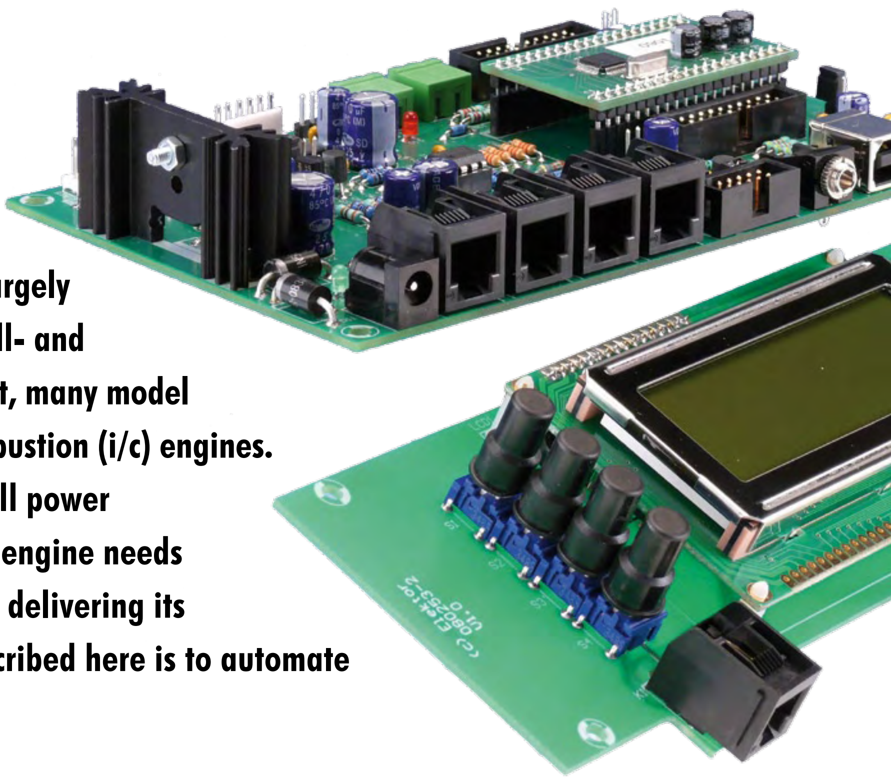
Automatic Running-

for internal combustion model engines

Part 1: the hardware

Michel Kuenemann (France)



Even though brushless electric motors have largely replaced internal combustion engines in small- and medium-sized radio-controlled model aircraft, many model enthusiasts are still attached to internal combustion (i/c) engines. But while an electric motor can be used at full power immediately it is brought into service, an i/c engine needs a period of running in before it is capable of delivering its maximum power. The idea of the project described here is to automate this important operation.

Technical specifications

- 32-bit ARM7 processor running at 59 MHz, 128 kB flash memory and 64 kB RAM.
- Throttle control by standard model servo. Configurable travel and direction of movement.
- Microcontroller-driven glow plug heating.
- Engine speed measurement from 0 to over 30,000 rpm.
- Engine temperature measurement from 0–160 °C.
- Ambient temperature measurement
- Mixture adjustment managed by the on-board software.
- Mobile pocket terminal with 4-line / 20 character alphanumeric LCD display, push buttons and encoder knob.
- USB link
- Direct Servo Control (DSC) interface
- Emergency stop push button
- Power supply: 7–15 Vdc.

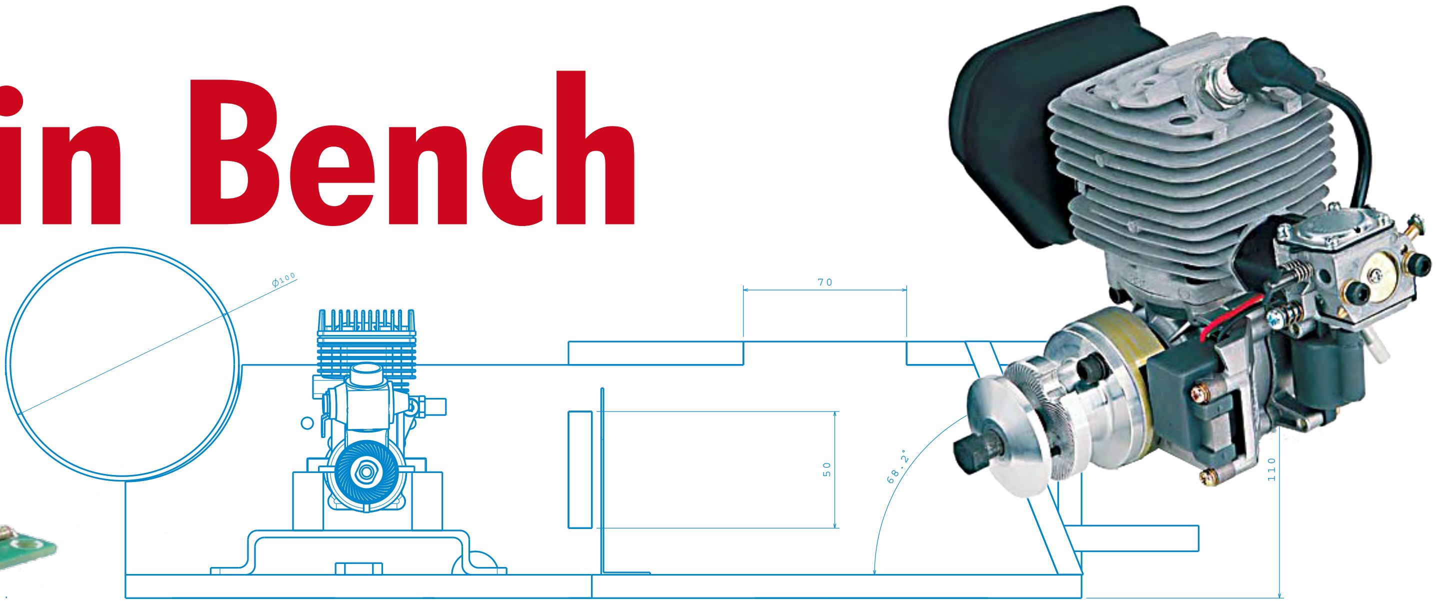
A miniature i/c engine can be run in either in the model it is destined for, or on a test bench dedicated to this purpose.
Running-in consists of running the engine, loaded with a propeller of suitable pitch and diameter, and putting it through controlled cycles of acceleration and deceleration. These alternating high and low speeds cause controlled 'wear' (particularly of the piston and cylinder wall) that enables an accurate fit to be achieved between these components. The way these cycles are achieved depends on the engine specifications, the manufacturer's recommendations, and individual habits. The key parameters to be managed are:

- Engine speed
- Engine temperature
- Richness of the air/fuel mixture

Traditionally, the speed is controlled by using the throttle, often operated by hand when running-in on a test bench. The engine speed is monitored using a hand-held rev counter, or just by ear. The engine temperature is often monitored by 'feel' and the richness of the mixture adjusted by hand. Under these conditions, the running-in operation is performed 100% manually with little objective feedback about how the process is progressing.
The idea of the board described here is to offer automation and repeatability in this phase, by managing the main parameters of the running-in

Figure 1. Running-in bench block diagram.

-in Bench



automatically.
The board also offers extended possibilities for testing and adjusting i/c engines (already run in) or electric motors for which we want to measure, estimate, or compare characteristics like static thrust, the power supplied, fuel curves, or torque and power curves. The board can also be helpful for adjusting the needle-valve level (acceleration).

The software and the functions of this project will be described in detail in Part 2 of our article to be published next month.

Block diagram

The block diagram of the running-in bench is given in **Figure 1.** At the heart of the system is a 32-bit microcontroller board which manages the engine and gathers the 'engine' parameters required for the running-in.
The throttle is operated by way of

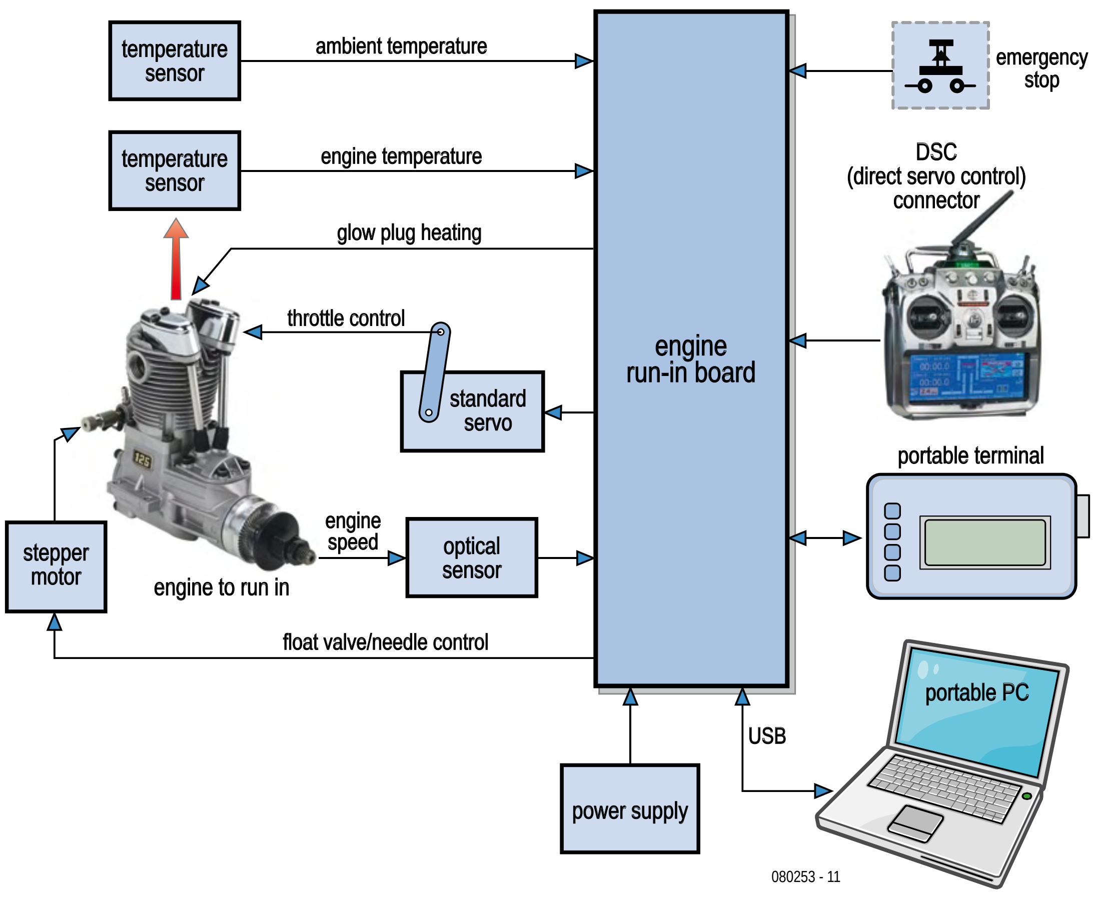


Table 1.
Microcontroller specifications and resources used for the application.

Resource	Specification	Notes
Central unit	ARM7-TDMI, 32-bit central unit.	RISC-type central unit, one instruction per clock pulse.
Clock	60 MHz	Clock frequency used in the application: 58.9824 MHz
RAM	64 kB	
Flash memory	128 kB	
UART0	16C551 compatible	Used for programming and communication with a PC
UART1	16C551 compatible	Available on expansion connector Multiplexed, with PWM generation
SPI		Available on expansion connector
I^2C #2	Up to 400 kbps	Available on expansion connector
‘bit bang’ I^2C	Up to 400 kbps	Used for the pocket terminal and temperature detector. Expandable. 3 connectors available
I/O port		3 ports available on expansion connector

a standard servo. The engine speed reached is measured using an optical detector. The board also manages the heating of the glow-plug and adjusts the mixture needle via a stepper motor. To complete the task, the board monitors the engine and ambient temperatures.

A pocket terminal comprising an LCD display, a coding button, a few push-buttons and a sounder, to let you control the running-in bench without needing a computer.

The USB link (full speed @ 12 Mbps), obligatory these days, lets you program the board, control it, and read off the recorded data.

The bench has a DSC (Direct Servo Control) interface, which lets you connect a remote-control transmitter and control the servo by means of the throttle control. This is also how you access the functions associated with optimising the fuel curve.

Provision has been made for an ‘emergency stop’ button in order to stop the engine quickly in the event of an critical problem.

With these facilities; the board lets you control the running-in of all types of 2- or 4-stroke, single- or multi-cylinder i/c engines, running on methanol or petrol, with glow or spark (electronic) ignition.

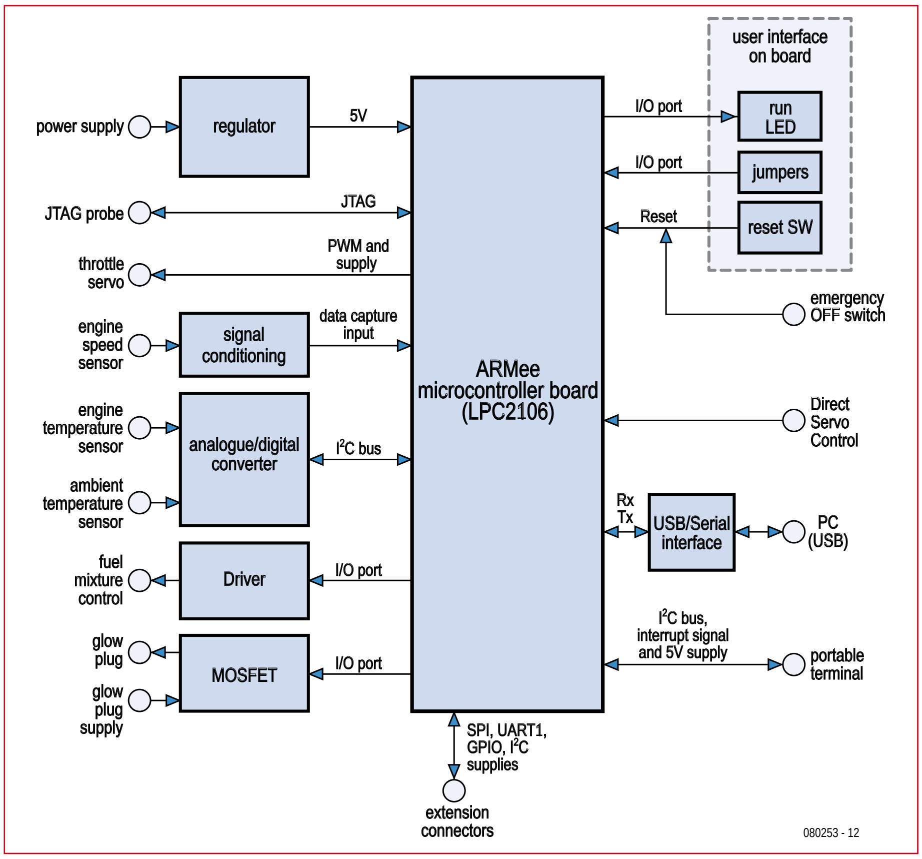


Figure 2. Running-in bench driver board block diagram.

Block diagram of main board

The board, the block diagram for which is given in **Figure 2**, is designed around a microcontroller that may already be familiar to Elektor readers, the LPC2106 from NXP. This 32-bit processor using RISC ARM7 architecture has ideal characteristics for this project (see **Table 1**). Since the LPC2106 is only available in a 0.5 mm (0.02”) pitch SMD package, we thought it advisable to use a module that readers will be able to buy ‘ready-made’, in this case, the *ARMee* board described in our April and May 2005 issues [1][2].

On the left of **Figure 2** we find the ‘system’ interfaces and the interfaces with the engine to be run in.

The board works correctly with a power supply from 7–15 V. So the board can be powered from a mains adaptor, a car cigar lighter, or a 7-cell NiCd, NiMH or even 2S or 3S lithium polymer battery, which modelling enthusiasts will be familiar with.

The throttle servo is controlled quite conventionally by way of a PWM signal. Naturally, the board supplies the power for the servo, and the connector used is the same type as is found on all radio-control receivers. Hence the throttle control can use any ‘standard’ off-the-shelf model servo.

The engine speed detector comprises a phototransistor and an LED. The signal from the phototransistor is processed before being fed to one of the microcontroller’s data capture inputs.

As the microcontroller doesn't have any analogue inputs, it was necessary to make provision for an external analogue/digital convertor for inputting the temperature data. A type with an I²C interface was chosen.

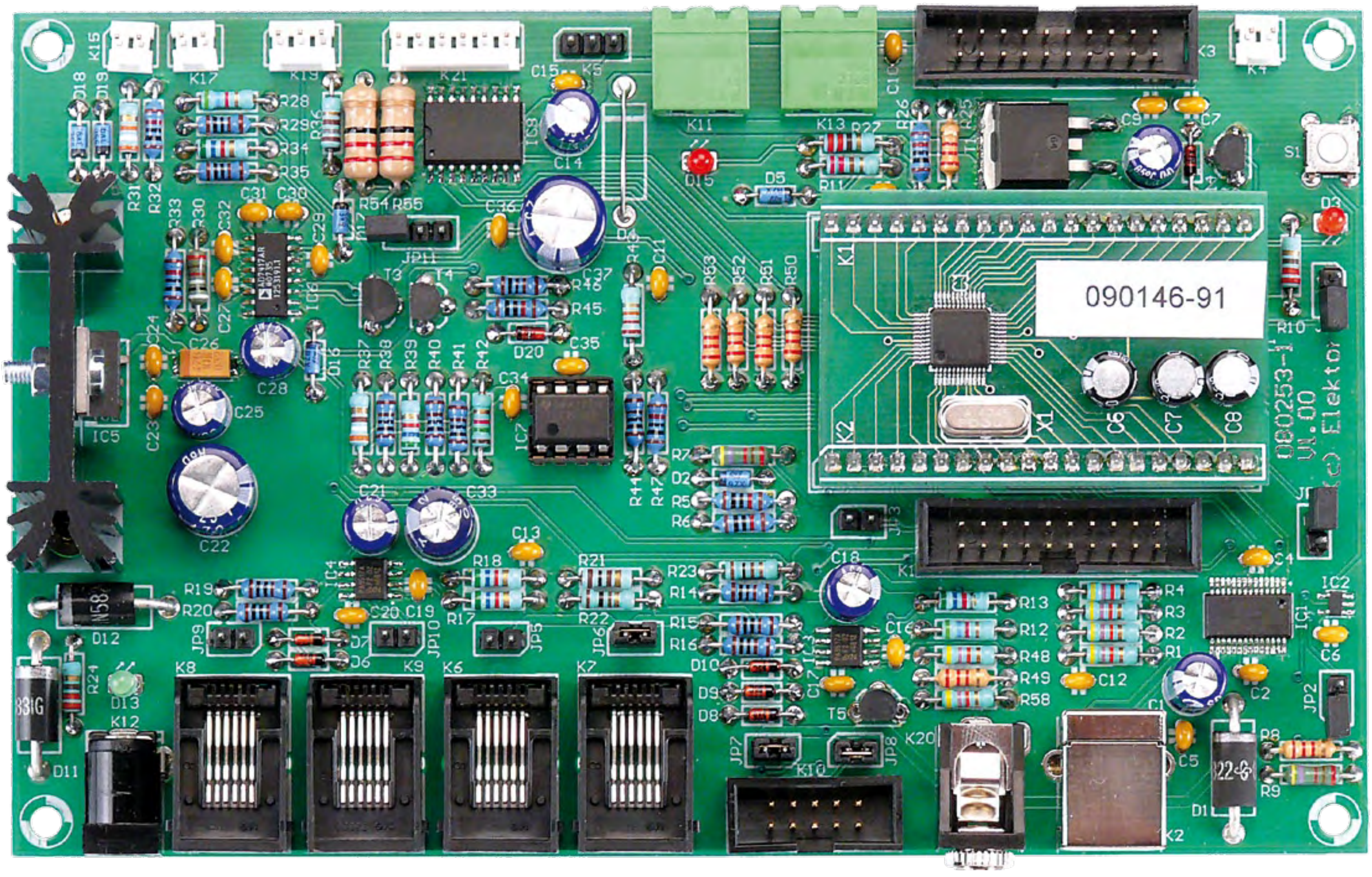
The single-pole stepper motor plus gearbox for adjusting the mixture is governed by an open-collector driver, controlled in turn by four of the microcontroller's I/O port lines.

Circuit diagram of main board

It's only a small step from the block diagram to the 'real' circuit diagram of the controller board (**Figure 3**). The large number of connectors and protection components make the circuit pretty impressive, but it's still relatively easy to pick out the elements of the block diagram. Pride of place right in the middle of the circuit goes to the ARMee module fitted with an

1.8 V for the microcontroller core), all the microcontroller's unused I/O pins (including an SPI bus, a UART, a PWM generation port and two I/O ports) and the #1 I²C bus with interrupt.

The #1 I²C bus is a 'bit bang' type, i.e. the pulse trains required by the I²C protocol are generated in software by the driver. This has the advantage of being able to convert any pair of the microcontroller's ports into an I²C bus.



Glow-plug heating is taken care of by a power MOSFET, driven again from one of the microcontroller's I/O port lines.

On the right of the board block diagram we find a few LEDs that tell you the status of the board, a few jumpers, a reset button, the USB port and the DSC interface.

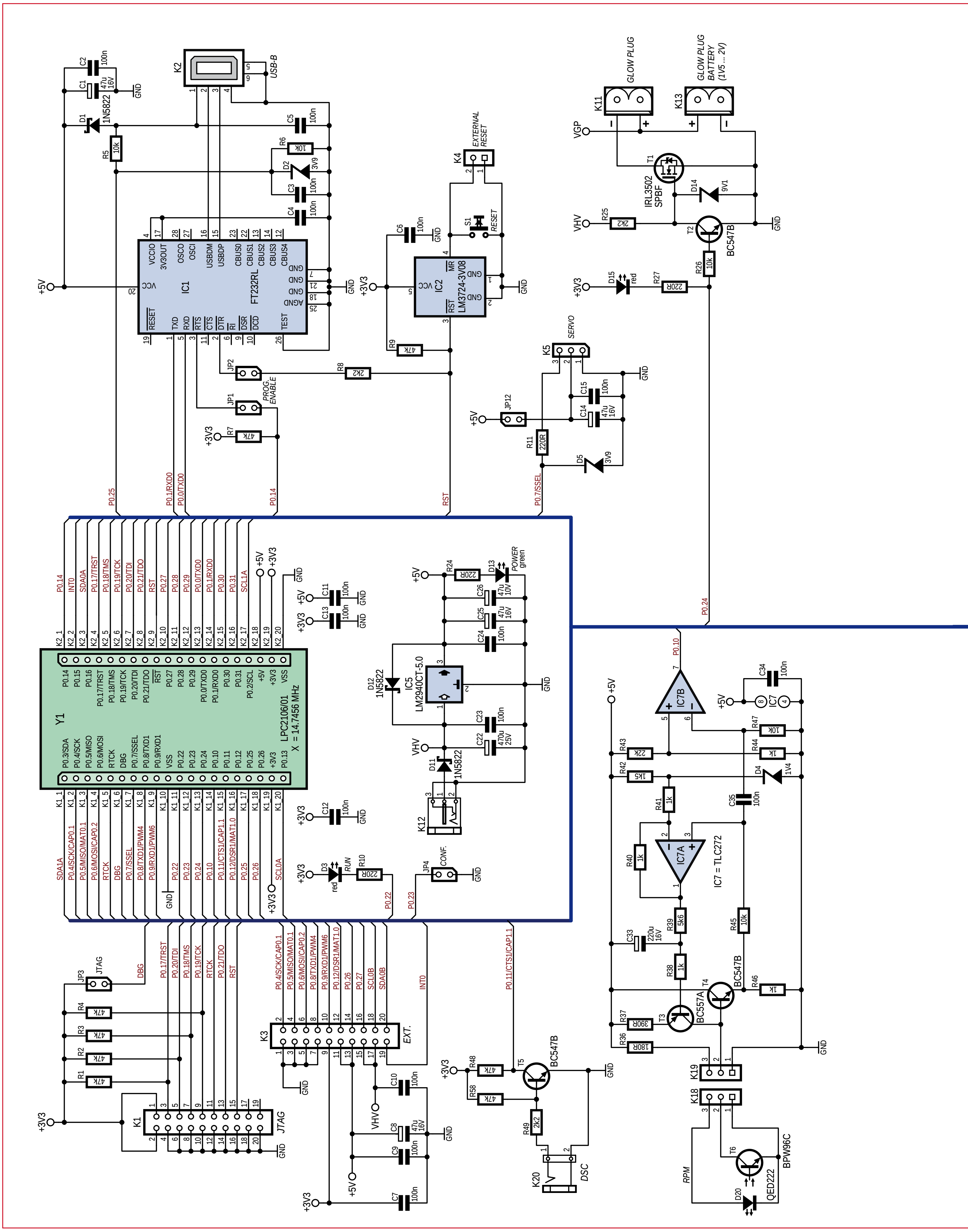
A pocket terminal...
...for controlling the board is connected to the main board via a 6-way cable with RJ11 connectors. This cable carries a 400 kbps I²C bus, an interrupt signal, and the power for the terminal (5 V).

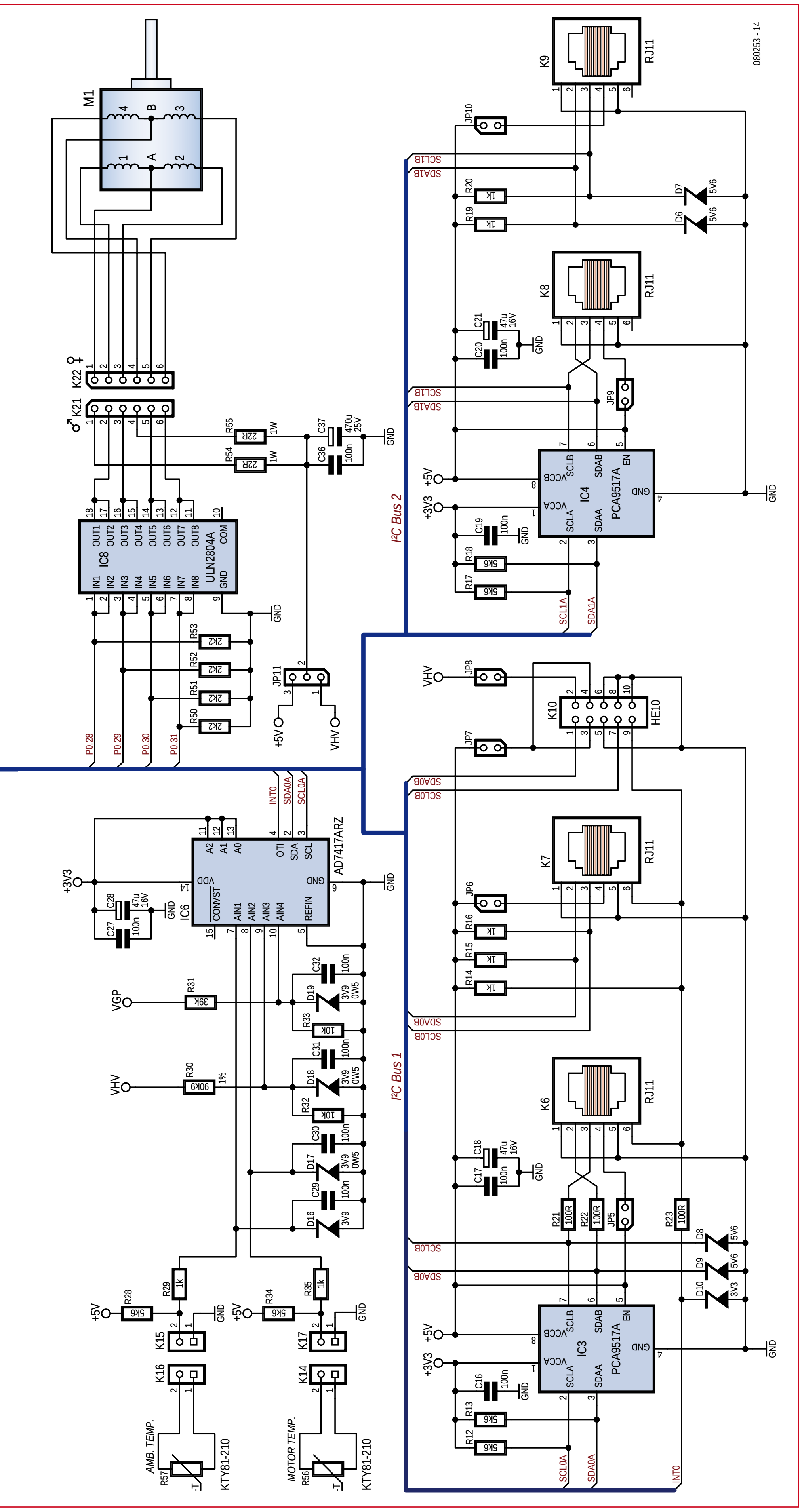
LPC2106/01 microcontroller and a 14.7456 MHz crystal. Do note that these components are different from the ones on the board described in 2005 [1][2]. If you want to use the 2005 board, all you have to do is change the original crystal. The ARMee module is powered from 5 V only, as the 1.8 V and 3.3 V rails required by the core and the inputs/outputs respectively are generated on the ARMee board itself. The 3.3 V supplied by the ARMee board is used (sparingly) by certain components on the main board.

A 20-pin expansion connector (K3), unused for the moment, carries all the board's supply voltages (except the

However, this type of bus represents a not inconsiderable load on the microcontroller if the bus is used intensively, and even more so if we want to use the bus in slave mode.
To get round these drawbacks, we operate the #1 I²C port in master mode only, and we've added an interrupt signal (INT0) to this bus so as to avoid scanning the push buttons and coder in the pocket terminal. This reduces the transactions on the I²C bus to the strict minimum.

To finish, it may be noted that this interface includes one active device (IC3), a PCA9517A . This device serves three functions:


K2
USB-B
C1 47u 16V
C2 100n
D1 1N5822
R5 10K
C5 100n
R6 10K
D2 3V9
C3 100n
C4 100n
+5V
GND
IC1
FT232RL
VCCIO
3V3OUT
OSCO
OSCI
USBDM
USBDP
CBUS0
CBUS1
CBUS2
CBUS3
CBUS4
VCC
RESET
TXD
RXD
RTS
CTS
DTR
RI
DSR
DCD
TEST
AGND
+3V3
C6 100n
K4
EXTERNAL RESET
S1
RESET
IC2
LM3724-3V08
MR
RST
R9 47K
JP2
R8 2K2
PROG. ENABLE
JP1
R7 47K
P0.25
P0.1/RXD0
P0.0/TXD0
P0.14
RST
P0.7/SSEL
VGP
K11
GLOW PLUG
K13
GLOW PLUG BATTERY (1V5 ... 2V)
T1
IRL3502 SPBF
D14 9V1
VHV
R25 2K2
T2
BC547B
R26 10K
D15 red
R27 220R
K5
SERVO
JP12
C15 100n
C14 47u 16V
R11 220R
D5 3V9
Y1
LPC2106/01
X = 14.7456 MHz
P0.24
P0.10
IC7B
IC7A
IC7 = TLC272
C34 100n
R43 22K
R47 10K
R44 1K
R42 1K5
D4 1V4
C35 100n
R41 1K
R40 1K
R45 10K
C33 220u 16V
R39 5K6
R38 1K
T4 BC547B
R46 1K
R37 390R
T3 BC557A
R36 180R
K19
K18
RPM
T6
BPW96C
D20
QED222
+5V
R24 220R
D13 POWER green
C26 470u 10V
C25 47u 16V
C24 100n
C11 100n
C13 100n
D12 1N5822
IC5
LM2940CT-5.0
C23 100n
C22 470u 25V
D11 1N5822
K12
C12 100n
D3 RUN red
R10 220R
JP4 CONF.
P0.22
P0.23
JP3 JTAG
DBG
R4 47K
R3 47K
R2 47K
R1 47K
K1 JTAG
K3 EXT.
INT0
P0.11/CTS1/CAP1.1
C10 100n
C8 47u 16V
C9 100n
C7 100n
VHVO
R48 47K
R58 47K
T5 BC547B
R49 2K2
K20
DSC



- adapting the voltage swing of the microcontroller (3.3 V) to the levels of the external bus (5 V);
- offering a protective barrier against 'onslaughts' from the outside world;
- buffering the signals from the micro-controller, thereby making it possible to get round the 400 pF limit specified for the I²C bus.

A number of 100 Ω series resistors, in association with 5.6 V zener diodes, round off the protection for this bus. The jumpers (JP5–JP8) let you power — or not — the peripherals connected to the three connector K6, K7, and K10.
The pocket terminal can be connected to either K6 or K7, it doesn't matter which.
K10 makes it possible to connect an expansion board using HE-10 connectors, far more practical than RJ11 connectors when the board is hand-wired on 2.54 mm (0.1") 'breadboard'. This connector supplies a 5 V rail, along with the unregulated board supply (via JP8).

The #2 I²C bus is connected to the microcontroller's 'official' I²C peripheral, with master and slave modes, and is capable of a maximum speed of 400 kbps. From a hardware point of view, it is just like the #1 I²C bus, except without the interrupt signal and HE-10 connector. Given the possibilities, this bus offers for expanding the system, we've opted to keep it free and make do with just the #1 I²C bus. Readers may leave out IC4 and its associated components.

The 12 Mbps full-speed USB interface is achieved using a device that may already be familiar to Elektor readers, the FT232RL from FTDI, connected directly to the microcontroller's UART0 interface. Diode D1 allows the board's power to be derived from the USB bus. This is particularly useful during programming or when recovering the data stored on the board when no other power source is available. In 'normal' operation, powering the board from the USB bus is not recommended, as this power source is not powerful enough. Port P0.25 makes it possible to

Figure 3. Circuit diagram of main board.

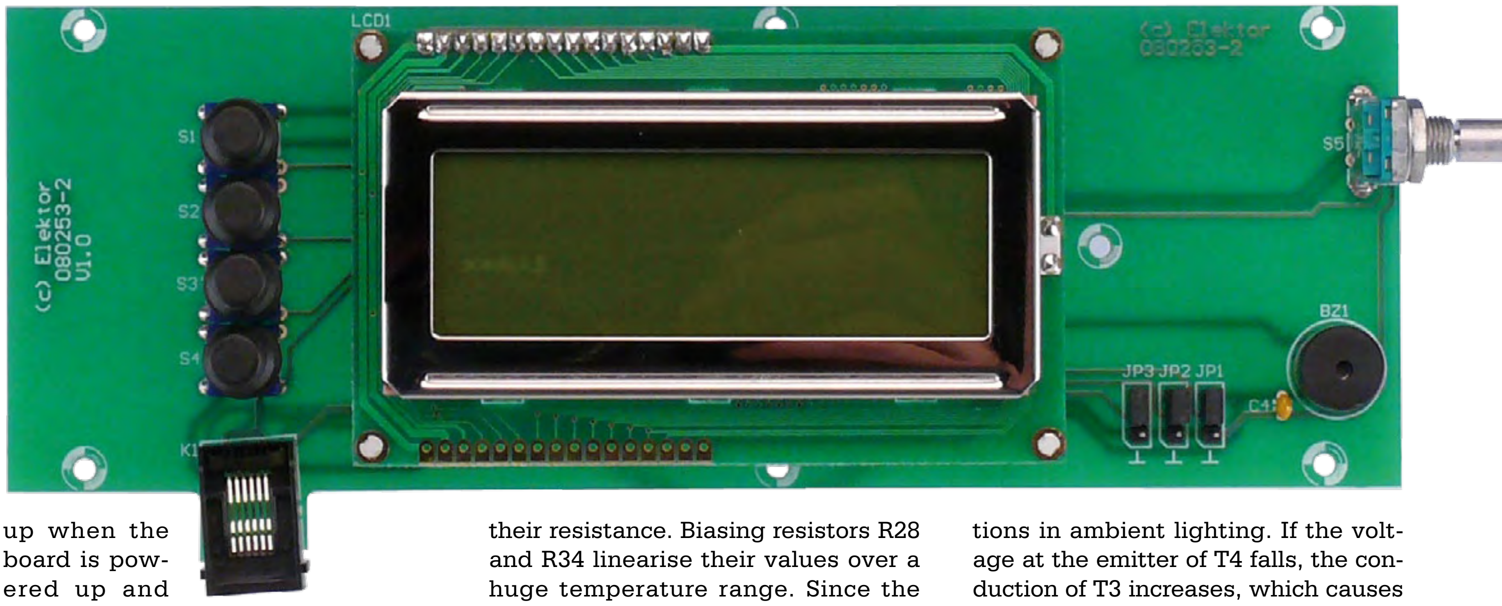
detect if the USB bus is connected and active. Jumpers JP1 and JP2 work in tandem: if they are fitted, the 'programming' mode is active. In this mode, it is possible to easily and quickly load new software into the microcontroller through the intermediary of the (free) flashing by NXP [3] (see box). Without these jumpers, the USB link operates as a simplified conventional serial link, but with a markedly higher transmission speed (3 Mbps maximum).

Management of the microcontroller's reset is entrusted to a specialized device, the LM3724 from National Semiconductor. This ensures correct start-up when the board is powered up and allows additional reset push-buttons to be added; a feature that we have made use of, since the board has two reset buttons (one on the board, the other external one connected via K4).

The throttle servo is controlled by the microcontroller's P0.7/SSEL/PWM2 output. R11 and D5 protect this pin in the event of an external voltage being injected onto the control line. The servo is powered from the board's 5 V rail or via a jumper (JP12). This jumper lets you choose not to power the servo socket, so as to avoid an external voltage being injected. This can occur if the user connects an electric motor speed controller with a BEC (Battery Eliminator Circuit) function to this output. In this way, either a servo or a speed controller with BEC can be connected to this output.
Powering of the board is entrusted to a 'low voltage drop' linear regulator. Thus the board works correctly from as little as 7 V, which means the board can be powered from a battery of dual-element lithium polymer accumulators, delivering a nominal voltage of 7.4 V. Diode D11 protects the circuit against possible reverse polarity. The VHV voltage is tapped off at the regulator input and used to power the stepper motor (see below).

The 'engine' and 'external ambient' temperature measurements are made using KTY81-210 linear two-terminal detectors in TO-92 packages. These detectors, whose active element is made of silicon, have the advantage of exhibiting a virtually linear variation in their resistance. Biasing resistors R28 and R34 linearise their values over a huge temperature range. Since the microcontroller doesn't have any analogue inputs, it was necessary to resort to an external convertor. The convertor chosen for this task is an AD7417 from Analog Devices (IC6). This convertor, with four 10-bit inputs, and connected to the #1 I²C bus, has an internal 2.5 V reference. The device has an internal temperature detector that provides the device temperature — which is also the prevailing ambient temperature around the board. This convertor is kind enough to return this temperature directly in degrees Celsius, without needing any scaling or calibration. The interrupt line to which the convertor is connected makes it possible to warn of any possible overheating. The remaining two inputs are used to monitor the supply voltages to the board (VHV) and the glow plug (VGP).

The engine speed detector is one of the key elements of the circuit. The signal processing circuit for this detector was designed by Paul Goossens and was the subject of the project 'Rev counter for models' [4]. The optical detector, a phototransistor, works by reflection and should be positioned a few centimetres from the propeller. The LED built into the detector provides a little local light source. Depending on the ambient lighting, the conduction of this device varies considerably, and it is impossible to detect the movement of the propeller unless the processing stage allows for these variations. To do this, operational amplifier IC7A holds the voltage at the emitter of T4 at an average value of 1.4 V, which sets the phototransistor's operating point and compensates for the variations in ambient lighting. If the voltage at the emitter of T4 falls, the conduction of T3 increases, which causes the voltage on the detector terminals to increase, and hence likewise on the emitter of T4, as it is connected as a follower. The low-pass filter formed by R39 and C33 slows down this control loop to avoid the short pulses associated with the movement of the propeller in front of the detector moving the operating point. These pulses, present at the emitter of T4, are picked off by a high-pass filter C35/R47 which eliminates the 1.4 V DC component and any slow voltage variations. IC7B, wired as a comparator, looks after shaping these analogue pulses to make them 'microcontroller-compatible'.



A logic-controlled power MOS transistor has the job of controlling the heating of the glow plug. The transistor gate is biased with the board input voltage in order to take advantage of its 'high' value. However, zener diode D14 prevents this voltage reaching 10 V, the maximum gate voltage. When heating of the glow plug is activated, a red LED lights to warn the user. The power source for the glow plug can be either a NiMH cell (1.2 V), a lead-acid cell (2 V), or the glow plug heating output of a model 'power panel'.

The Direct Servo Control (DSC) input is particularly simple, as a single NPN transistor is all it takes to interface with the microcontroller. The resistor values in the circuit have been tested using a Graupner MX16s transmitter. If your transmitter is a different type, you may need to adjust some values.

The stepper motor driver output allows you to drive a stepper motor with gearbox with a rated voltage of 5 V or 12 V. The values of resistors R54 and R55 may need adjusting, depending on the motor you're using. If you're not planning to use a stepper motor in your application, these four open-collector outputs can be put to other uses, such as driving lamps, LEDs, DC motors, or relays.

Pocket terminal

The electrical circuit of the pocket terminal (**Figure 4**) is very simple, thanks to the high level of integration in the devices used. The heart of this board is an MCP23017 port expander with I²C bus from Microchip, which provides no fewer than 16 I/Os, ideal for producing a handy user interface.
The 6-way RJ11 input connector provides the MCP23017 with power, the I²C bus, and an interrupt signal. The IC is protected from electrostatic discharge and over-voltage by three zener diodes (D1–D3).

The MCP23017's three address select lines have been connected to the same number of jumpers, which means you can select the terminal's bus address. The alphanumeric display is interfaced in 4-bit mode and it takes up the whole of the MCP23017's port B.

Incremental encoder S5 gives the user the impression of an analogue control, arguably much more ergonomic than a pair of +/− buttons when you're driving a servo. The MCP23017 has an 'interrupt-on-change' mode, which means that a change of state on any of its 16 pins generates an interrupt. Hence the incremental coder doesn't have to be scanned: the software will only launch a read cycle on the I²C bus after receiving such an interrupt, which reduces the load on the bus to a strict minimum.

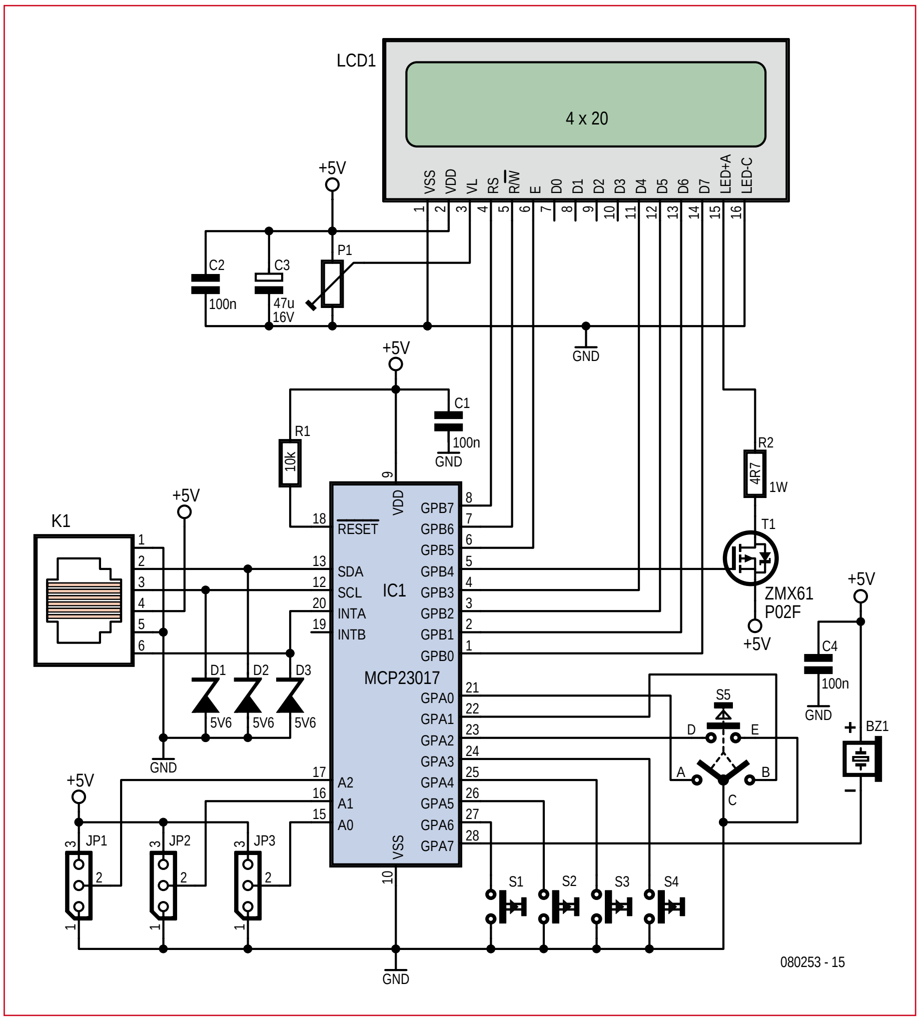


Figure 4. Circuit diagram of pocket terminal.

Pushbuttons S1–S4, along with the pushbutton on S5, employ the same type of event-driven processing. No pull-up resistors are needed, as they are built into the MCP23017.
And lastly, the terminal's sounder is controlled by a P-channel MOS transistor.

Construction

To make building the boards easier, we've chosen traditional through-hole components wherever possible. Start by soldering the SMD devices IC1–IC6, IC8, and T1. Take care not to overheat them, and to remove any shorts between pins using desoldering braid once soldering is finished. Then fit the through-hole components and end with the connectors. Check the orientation carefully for all the polarised components like the ICs, electrolytic and tantalum capacitors, and diodes. The ARMee module is fitted with the help of the board's component overlay.

Building the pocket terminal board is quick and easy and doesn't call for any special remarks. Depending on the type of display you've chosen, it may be necessary to adjust the value of R2 to suit the backlight current for it. At the end of building, fit the three jumpers JP1–JP3 in the '5 check' position.

Testing the boards

Testing takes place in four steps:

• Power up for the first time to check supply rails;
• Fit the ARMee module and flash-in the test firmware;
• Operating test of the main controller board ('CBRM');
• Operating test of the pocket terminal ('GMMI')

Powering up for the first time

Do not connect any peripherals to the board connectors, remove all the jumpers and the ARMee module, then

Firmware flashing procedure

First install the free LPC2000 Flash Utility from NXP [3] on your computer.

Power up the controller board and connect it to the PC via a USB cable. Check that the operating system has correctly recognised the new USB serial port. If the number assigned to the port is higher than COM5, change it.

Start LPC2000 Flash Utility In 'Connected To Port', select the COM port to be used and select a speed of 115,200 baud. Check the 'Use DTR/RTS for Reset and Boot Loader Selection' box.

In the box 'Device:', select the LPC2106 and enter the value 14745 in the 'XTAL Freq. [kHz]:' box.

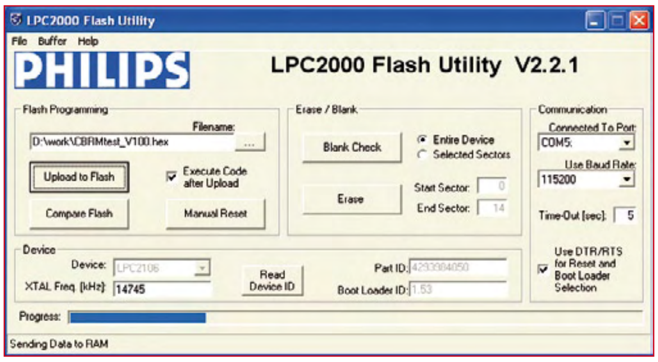
On the controller board, fit jumpers JP1 and JP2 and remove jumper JP3.

Click the 'Read Device ID' button. The 'Part ID' and 'Boot Loader ID' should get filled in. If not, go back through the procedure step by step – it is vital to get through this stage successfully, otherwise it won't be possible to program the controller.

Use the button alongside the 'Filename' box to select the .hex file to load into the controller. Click the 'Upload to Flash' button and wait for the operation to finish.

Exit the tool to free up the serial port and remove jumpers JP1 and JP2.

All this seems very long-winded, but as the software saves the selected options, flashing is very quick after the first time.


LPC2000 Flash Utility V2.2.1

Filename: D:\work\CBRMtest_V100.hex
Execute Code after Upload: ✓
Entire Device: ●
Start Sector: 0
End Sector: 14
Connected To Port: COM5:
Use Baud Rate: 115200
Time-Out [sec]: 5
Use DTR/RTS for Reset and Boot Loader Selection: ✓
Device: LPC2106
XTAL Freq. [kHz]: 14745
Part ID: 4293984050
Boot Loader ID: 1.53
Sending Data to RAM

Figure 5. How to configure the LPC2000 Flash Utility programming utility.

JTAG

If you have a JTAG probe, you'll be able to program the microcontroller after connecting your probe to K1 (Keil Ulink compatible connector) and fitting JP3. Don't forget to remove JP3 afterwards. If JP3 is fitted while the JTAG probe is disconnected, the program will work, but ports P0.22 and P0.31 will remain in Embedded Trace Macrocell (ETM) mode and so won't be accessible to the program.

This will mean that the RUN LED, the 'user' jumper, the glow plug drive, USB status reading, and the stepper motor driver won't work.

power the board up from a bench supply set to 8 V / 200 mA. The current consumption should not exceed 70 mA. The green power LED should light, and you should check with a multimeter the value of the 5 V supply to pin 3 of IC5, which must be between 4.9 V and 5.1 V. If all is well at this stage, move on to the next; if not, check once again for solder bridges or reversed components.

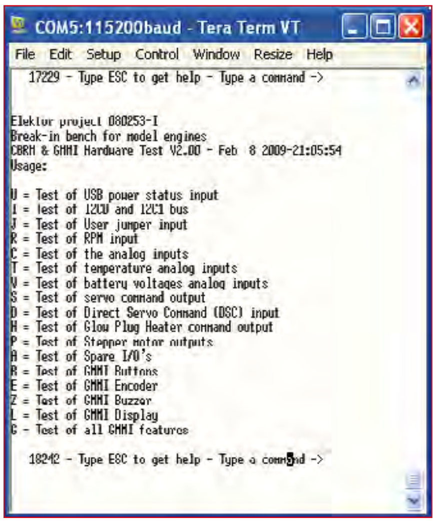
Now let's add the ARMee board

The next step is going to consist of powering down the board and fitting the ARMee module, taking care not to get it the wrong way round. Check that it is correctly fitted with a 14.7456 MHz crystal. If not, it is vital to change it before proceeding. Power the board up again and check the presence of the 3.3 V rail on pin 1 of IC3. The current consumption should remain under 70 mA if the microcontroller has never been programmed or while the reset button is pressed. That's the hardest part over!

Now we need to test that all the stages on the board are working correctly and check that the microcontroller is able to communicate with the outside world. To do this, the microcontroller needs to be 'flashed' with the *CBRMtest.hex* software, available by download. Refer to the box for this operation. Once the board has been programmed, unplug the USB cable and check that there are no jumpers fitted. Power the board up again. The current consumption should now settle at around 100 mA. Pressing the reset button causes the current to drop to around 60 mA. The red 'RUN' LED should be flashing regularly. Now disconnect the power.

Testing the main CRBM board

If not already there, install the Tera-


COM5:115200baud - Tera Term VT
File Edit Setup Control Window Resize Help

```
  17229 - Type ESC to get help - Type a command ->


Elektor project 080253-I
Break-in bench for model engines
CBRM & GMMI Hardware Test V2.00 - Feb  8 2009-21:05:54
Usage:

U = Test of USB power status input
I = Test of I2C0 and I2C1 bus
J = Test of User jumper input
R = Test of RPM input
C = Test of the analog inputs
T = Test of temperature analog inputs
V = Test of battery voltages analog inputs
S = Test of servo command output
D = Test of Direct Servo Command (DSC) input
H = Test of Glow Plug Heater command output
P = Test of Stepper motor outputs
A = Test of Spare I/O's
B = Test of GMMI Buttons
E = Test of GMMI Encoder
Z = Test of GMMI Buzzer
L = Test of GMMI Display
G - Test of all GMMI features

  18242 - Type ESC to get help - Type a command ->
```

Opening screen for CBRMtst_v200.hex.

Term Pro [5] freeware on your PC. Connect the board to one of the USB ports on your PC. The green power LED should light and the red 'RUN' LED should flash. Run TeraTerm and in the Setup -> Serial port menu, configure the port to which the board is connected as follows:

Baud rate: 115200
Data: 8 bits
Parity : none
Stop: 1 bit
Flow control: none

Close the configuration window and press the Escape key on the PC keyboard in order to get the screen in **Figure 6**.

Select the parts to be tested in turn by pressing the corresponding letter on the PC keyboard. The software is self-documented and explains what should be happening with the hardware as each element is tested. An oscilloscope and multimeter are required.

For the time being, don't activate the tests for the GMMI board (pocket terminal).

Testing the pocket terminal (GMMI)

Unplug the USB cable and power the CBRM board up again using the bench supply set to 8 V / 500 mA. Fit jumper JP8 and connect a ribbon cable fitted with 6/6 RJ11 connectors to K7 and the GMMI board. The current consumption should not increase significantly. Adjust the contrast pot P1 until little dark rectangles appear on lines 1 and 3 of the display. Check that the three jumpers JP1–JP3 are in the '5 V' position. Connect the board to the PC again under TeraTerm and now run the tests devoted to the GMMI board and follow the instructions. It will probably be necessary to adjust the contrast and possibly R2 which sets the backlight current.

The rest

You now have a powerful 32-bit ARM7 microcontroller board and a handy data entry terminal. In Part 2 of this article, we'll be looking in detail at how to connect the board to its detector and actuators, together with the application software for this project. In the meantime, good luck with the construction!

(080253-I)

A few words about the author

A graduate from the National Institute of Applied Sciences at Lyon, France, Michel Kuenemann has been an independent electronics consultant for about 20 years. Michel currently works on electrical supply systems for a large transport aircraft and enjoys building much smaller ones in his spare time.

References and Internet Links

[1] LPC210x 'ARMee' Development Board (1), Elektor Electronics March 2005. Online: www.elektor.com/080444-1.

[2] LPC210x 'ARMee' Development Board (2), Elektor Electronics April 2005. Online: www.elektor.com/080444-2.

[3] LPC2000 Flash Utility: www.nxp.com/products/microcontrollers/support/software_download/lpc2000/

[4] Rev counter for R/C Models, Elektor Electronics November 2003. Online: www.elektor.com/024111.

[5] TeraTerm: ttssh2.sourceforge.jp/

[6] www.elektor.com/080253

Note: in view of the length of the components list, this is being offered as a free download from the website for this article [6]. That way, you can download it at the same time as the software you'll need to make the board work.

Advertisement

See your project in print!

Elektor magazine is looking for

Technical Authors/Design Engineers

If you have

- ✔ *an innovative or original project you'd like to share with Elektor's 140 k+ readership and the electronics community*
- ✔ *above average skills in designing electronic circuits*
- ✔ *experience in writing electronics-related software*
- ✔ *basic skills in complementing your hardware or software with explanatory text*
- ✔ *a PC, email and Internet access for efficient communications with Elektor's centrally located team of editors and technicians*

then don't hesitate to contact us for exciting opportunities to get your project or feature article published.
Our Author Guidelines are at: www.elektor.com/authors.

Elektor
Jan Buiting MA, Editor
Regus Brentford, 1000 Great West Road, Brentford TW8 9HH, United Kingdom
Email: editor@elektor.com