Playback adapter for CD-ROM drives

Ever wanted to use an old CD-ROM drive as a CD player for audio playback? Now you can do it, with this nifty CD-ROM Playback Adapter. It can control one or two CD-ROM drives and has an infra-red remote control. A 16×2 line LCD screen provides track information and other data.

WE HAVE often been asked how to interface a hard drive or computer CD-ROM drive to a microcontroller. This is an interesting question, since there are countless old CD-ROM drives out there that are still in perfect functioning order, but which are 'obsolete'. Instead of letting them end up in landfill, you could do your bit and build this project.

Also, this project will be good experience for those readers who wish to learn more about the ATA interface and who want to use hard drives and CD-ROM drives in their own projects. The interface can be easily modified to suit any other micro, and only requires a few I/O ports and a reasonably fast processing core.

The main features of the Playback Adapter are listed below:

1) Can connect up to two ATAPI CD-ROM drives.
2) Auto detection of up to two connected drives.
3) Plays your favourite CDs.
4) Random play and repeat modes.
5) Controls volume (16 levels) and balance digitally.
6) Remote control with user-selectable key definitions.
7) Works with any RC5 remote control.
8) ISP (in-system programmable) if you wish to experiment with the firmware.
9) The CD is automatically locked when playing
10) LCD screen.

Part 1

By MAURO GRASSI

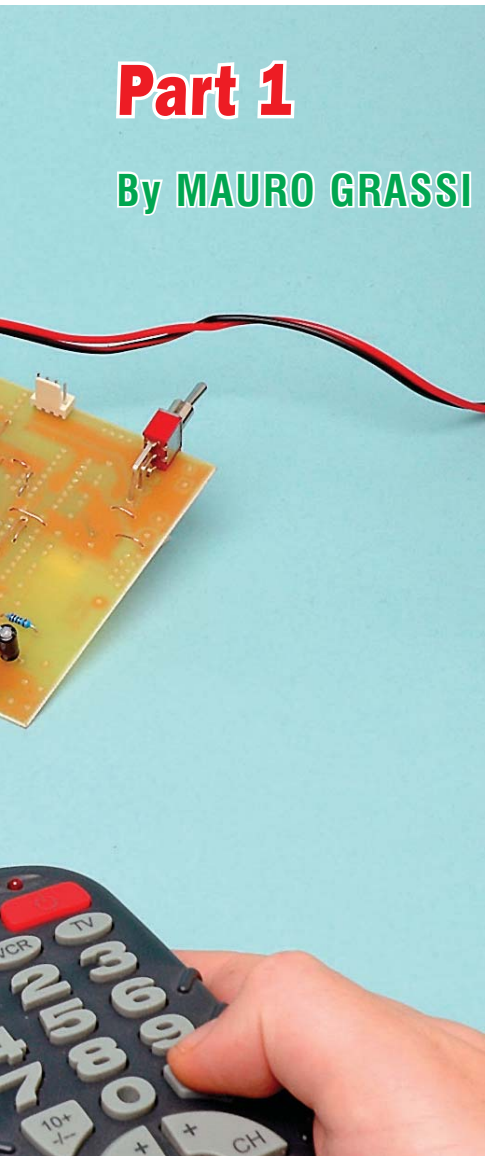


Table 1: the ATAPI register file. All ATA and ATAPI devices are controlled by reading and writing to these registers.

/CS1	/CS0	A2	A1	A0	/RD	/WR
1	0	0	0	0	Data	Data
1	0	0	0	1	Error	Features
1	0	0	1	0	Sector Count	Sector Count
1	0	0	1	1	Sector Number	Sector Number
1	0	1	0	0	Cylinder Low	Cylinder Low
1	0	1	0	1	Cylinder High	Cylinder High
1	0	1	1	0	Device/Head	Device/Head
1	0	1	1	1	Status	Command
0	1	1	1	0	Alt Status	Device Control

Accessing an ATAPI device

The CD-ROM Playback Adapter presented here lets you connect one or two drives and control each independently using a standard RC5 remote control.

CD-ROM drives that conform to the parallel ATA (AT attachment) standard can be used with the adapter, and most old drives fall into this category. In fact, most CD-ROMs will be ATAPI devices, which is a superset of ATA. It just means they support the packet interface, a feature that was added to the original ATA interface.

The resulting protocol was renamed ATAPI, with the 'PI' standing for packet interface. Most CD-ROMs, as well as DVD drives, are ATAPI devices, although others conform to different standards, like SCSI and SATA (serial ATA). These have a different connector and are not compatible with this project.

Interfacing to an ATAPI device is simple because most of the work is done inside the drive. In effect, it acts as a black box. It conforms to a standard and the internal implementation is left to the manufacturer. That is why the standard was originally called IDE (integrated drive/device electronics). It just means that a lot of the complexity of the interface is in the drive and the drive responds to a uniform set of commands.

It speaks well of the design that one of the easiest parts of a computer to get working is the hard drive or CD-ROM/DVD drive. Plug any drive into any motherboard and it will usually work first time.

Overview of the ATA interface

The ATA interface is register-based. There are essentially two banks of eight registers, although only one of the eight registers in the second bank is ever used.

The interface consists of two chip select lines, called /CS0 and /CS1 which are active low. There are three address lines designated A0, A1 and A2, as well as the read and write control lines. The latter are designated /RD and /WR respectively, and are also active low.

In order to access a register, one sets the register address given by [A2:A0] and then brings either /CS0 or /CS1 low (but not both). Then it is a matter of bringing either /RD or /WR low and reading or writing the data through data port D7:D0.

Note that the data bus width is actually 16 bits, but for accessing the ATA registers, only the lower eight bits are used. However, the full width of the bus is used for data transfers.

Note also, that the name of the register sometimes changes, depending on whether you are reading from or writing to the specific address. For example, at address 110b and with /CS1 low and /CS0 high, reading will give the Alternate Status register (a read only register), while writing will affect the Device Control Register (a write only register).

All commands to control the drive are sent through the register file (ie, the set of ATA registers). For example, the Command Register can be written with the opcode for a particular operation – eg, 'SLEEP' – and the drive will respond by going into power saving mode, barring any errors.

Note: the order in which you assert the control lines on the ATAPI/IDE bus is important. For example, you would think that you could assert /RD or /WR first, and then bring /CS1 or /CS0 low. However, this approach does NOT work on all drives.

The correct procedure is to assert /CS1 or /CS0 first, then to assert either /RD or /WR. Of course, because we are using a general-purpose micro, and these pins are on different ports, it is impossible to assert them simultaneously. This is not required however, but would be closer to a native IDE/ATAPI port interface.

Low-level drivers

It is relatively simple to write to an ATAPI device. As explained, you first prepare the data and the address, bring the chip select line low and then apply either the read or write signal.

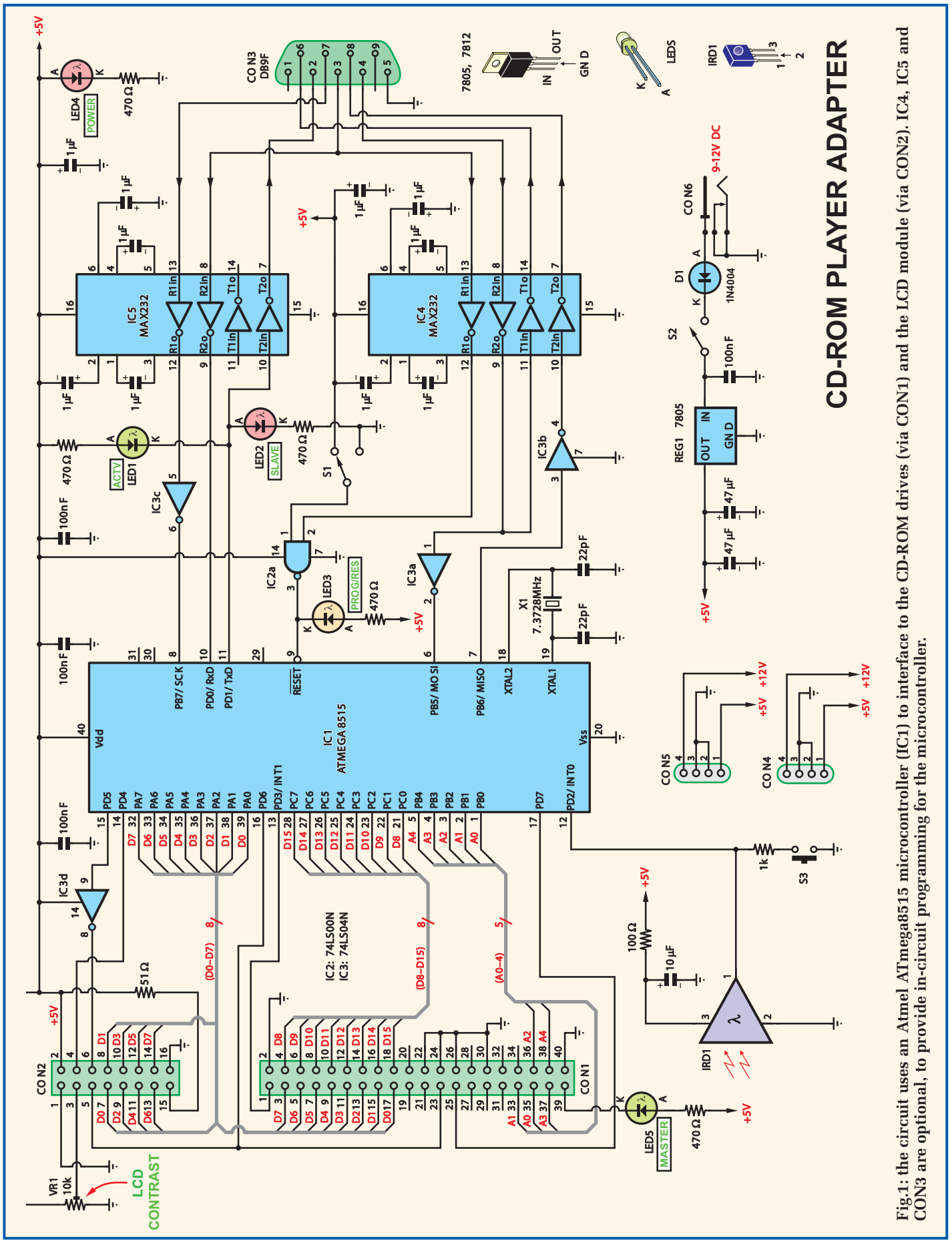


Fig.1: the circuit uses an Atmel ATmega8515 microcontroller (IC1) to interface to the CD-ROM drives (via CON1) and the LCD module (via CON2). IC4, IC5 and CON3 are optional, to provide in-circuit programming for the microcontroller.

Fig.2: the Error screen. The numbers give information about the state of the program and the drive when the error occurred.

This is the minimum you would need to interface to an ATA device like a hard drive. It would not be the fastest interface possible – you'd have to get involved in DMA (direct memory access) for that – but it would work.

With ATAPI devices like CD-ROM drives, most operations are initiated by writing packets rather than single byte commands. A packet is a string of 12 or sometimes 16 bytes that are sent to the drive in sequence.

In order to send packets, a more involved algorithm than just writing to the register file is needed. Here you have to worry about bus timings and whether the drive is busy or requesting data. There is a well-defined protocol for PIO (peripheral input output) access to an ATAPI device.

Feedback is provided by the bits BSY (bit 7) and DRQ (bit 3) in the Status register, which can be polled to determine the current state of the drive. When the drive shows BSY=1 it does not respond to commands, and reading any register except the Status register is undefined. In other words, the only valid information that can be read from the drive is bit 7 (BSY) of the Status or alternate status registers (when it is busy).

Opening command

As an example, the packet to open the tray of the drive is given by the 12-byte string: 0x1B, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00. To send this packet, you first send the PACKET command 0xA0 to the command register and then follow the packet protocol, as outlined in Flowchart 1.

The protocol begins with a packet being written to the drive. Optionally, there may follow a data read or write transfer, depending on the packet written to the drive.

The CoD (command/data) and IO (input/output) bits are in the Sector Count ATA register (also known as the Interrupt Reason Register in ATAPI devices). CoD is bit 0 and IO is bit 1. When CoD is 0, data is being transferred and when it is 1, a command (packet) is being transferred.

Table 2: this table shows the pinouts of the ATAPI interface. Note that this project leaves many pins unused, as they are unnecessary for PIO transfer.

Pin	Name	Description
1	/RESET	A low level on this pin resets all connected drives
2, 19, 22, 24, 26, 30, 40	GND	All these pins are connected to the common ground plane
3, 5, 7, 9, 11, 13, 15, 17	[D7:D0]	3=D7, 5=D6 . . . . . 15=D1, 17=D0 These are the eight least significant bits of the data bus
4, 6, 8, 10, 12, 14, 16, 18	[D15:D8]	4=D8, 6=D9 . . . . . 16=D14, 18=D15 These are the eight least significant bits of the data bus
20	KEY	This pin is not connected and is used to prevent the cable being connected the wrong way round
21	DDRQ	Data request pin – not used in this project
23	/WR	Write strobe, active low
25	/RD	Read strobe, active low
27	/IOREADY	Device ready pin, active low, not used in this project. Used to slow a controller if it is too fast for the drive
28	ALE	Address latch enable – not used in this project
31	IRQ	Interrupt request – not used in this project
32	IO16	Obsolete since ATA-3 – not used in this project
33, 35, 36	[A2:A0]	35=A0; 33=A1; 36=A2, address bus
37	/CS0	Chip select 0
38	/CS1	Chip select 1
39	/ACT	A low level on this pin indicates that the drive is working. It can be connected directly to an LED to show drive activity

The IO bit indicates the direction of transfer. When IO is 0, the host writes to the drive, and when it is 1, the host reads from the drive.

When the above packet has been successfully processed by the drive, it will respond by opening the tray. This packet does not require any extra data transfer, but other commands, such as reading the CD TOC (table of contents), do require reading from the drive. Other packet commands, such as setting the volume, require both reading and writing to the drive (refer to the ATAPI specification for the relevant packet codes).

The firmware

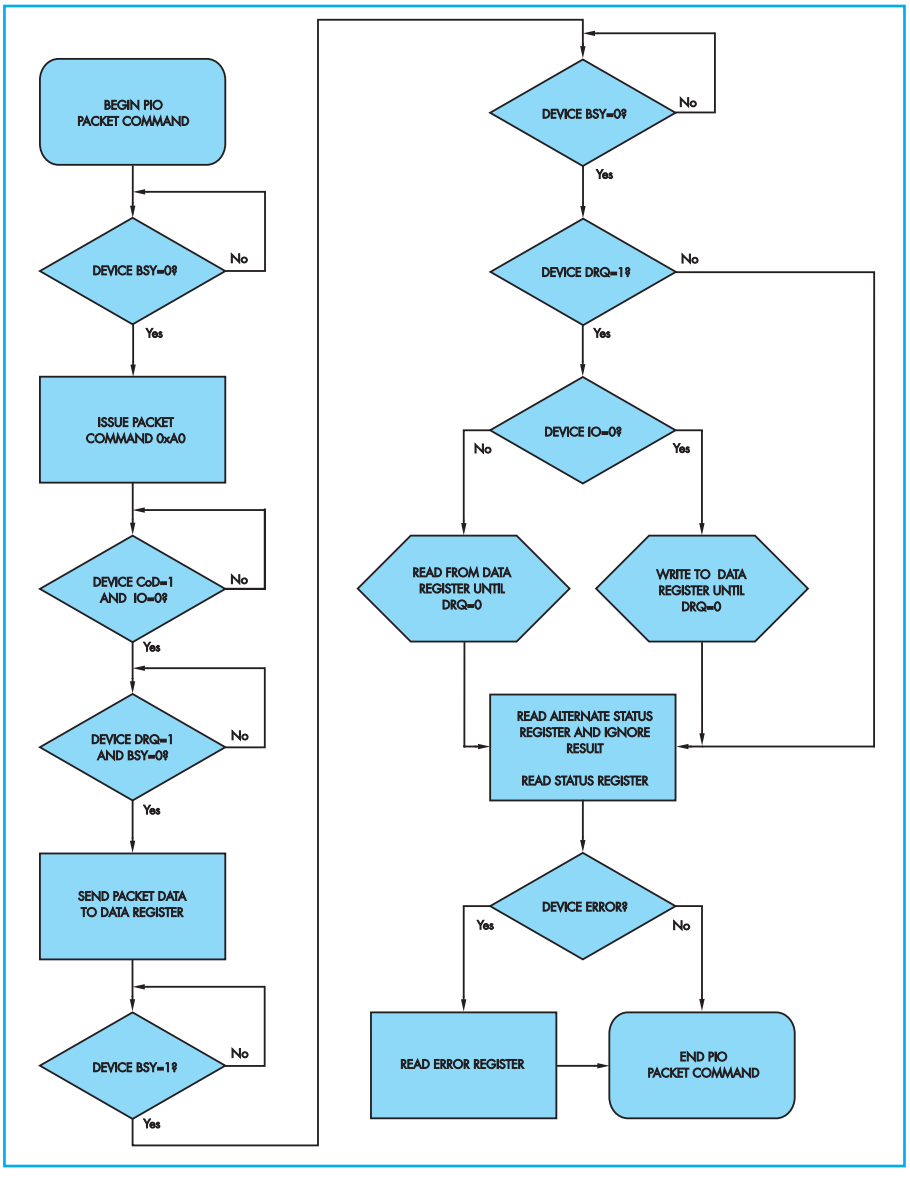
The main component of this project is the firmware, as the hardware is little more than an Atmel microcontroller. The firmware is responsible for interfacing to the drives, decoding the remote control signals, autodetecting the connected drives and controlling playback and volume, among other things.

All this is done with only 512 bytes of RAM! The firmware size is approximately 7.2KB and fits inside the micro's 8KB Flash memory.

ATA and ATAPI commands are either 'mandatory', 'optional' or not supported. To make sure that the CD-ROM Playback Adapter works with just about any ATAPI device, we've used only 'mandatory' commands as per the specification (rev 2.6 1996).

Note, however, that we cannot guarantee that it will work correctly with all ATAPI devices. Some are 'buggy' and the standard covers a period of many years. We've also come across drives that don't conform to the standard in every detail.

In our case, we tested the adapter with seven different ATAPI devices, including both CD-ROMs and DVD drives, and it worked correctly with six of these. The seventh drive had a problem in that it was not detected by the firmware and closer inspection and debugging revealed that the micro was unable to write to the drive's register



Flowchart 1: this is the packet writing routine used in the firmware. The interrupt signal INTRQ, intended for PCI buses on computers, is not used, and a method of polling for DRQ and BSY is used in its place.

file. Thus, it failed the first test of the autodetect subroutine, as explained below.

Basically, if a particular CD-ROM or DVD drive is not detected by the firmware on start-up, it will not be functional with the adapter. In that case, try using a different drive. Conversely, if the drive is correctly detected, there is a high chance that it will work correctly with this adapter.

How it works

Refer now to Fig.1 for the CD-ROM Player Adapter circuit details. The circuit is essentially just an Atmel ATmega8515 microcontroller (IC1) with its general IO pins configured to read and write to up to two drives. It also controls the LCD screen and reads the remote control sensor. Add in a power supply and a few support chips and that's about it.

IC4 and IC5 are MAX232 line drivers, and are used to interface the microcontroller to the serial port of a computer (RS-232). These devices are optional, and are only needed if you are planning to experiment by writing your own firmware.

Basically, they allow the board to be connected to a PC's serial port so that the microcontroller can be programmed in-circuit. The software to use for this job is called 'Pony Prog 2000' and is free for download from **www.lancos.com/ppwin95.html**.

IC2 and IC3 are simple logic gates, used here as 'glue logic' for the interface. These devices are 74LS00 and 74LS04 quad NAND gates and hex NOT gates respectively, but only one NAND gate and four NOT gates are used from these devices.

Infra-red receiver

IRD1 is an infra-red receiver module, containing a photodiode, amplifier, filter and demodulator – all in a compact package. It accepts a modulated infra-red signal on a 38kHz carrier and outputs a demodulated TTL level serial stream.

This stream is fed to pin 12 of IC1 and is decoded by the firmware in the microcontroller. Note that IRD1's output is usually high (around +5V) and varies as a square wave when an infra-red input is received.

Pushswitch S3 is used to select the remote control setup option at boot time. For normal operation it is open, and this allows the signal from IRD1 to pass to the microcontroller for decoding the remote control signals. Conversely, when S3 is pressed, it temporarily pulls this line low via a 1kΩ resistor to allow remote control set-up.

There are five indicator LEDs on the board. LED4 (red) is the power LED, while LED3 (orange) lights when the micro is being programmed or is in the reset state. This state can be entered using switch S1.

LED1 (green) shows the activity of the currently selected drive.

Finally, LED2 and LED5 make a pair. Only one will be lit at any one time. LED5 (green) indicates that the master device is being controlled, while LED2 (red) indicates that the slave device is being controlled. If you have two drives connected, you may toggle between them using the Line-In button on the remote.

Power supply

In order to power the drives, you will need a power supply capable of delivering +12V at 2A and +5V at 2A (eg, a computer power supply). By contrast, the board requires a +5V supply and draws just 200mA.

Basically, you've got two choices when it comes to the power supply. The first option is to power the PC board directly from a 9V to 12V plugpack supply and power the drives separately. In this case, the board supply is fed in via CON6 and is regulated to +5V using 3-terminal voltage regulator REG1. Diode D1 provides reverse

polarity protection, in case the supply is connected the wrong way around.

The second option is to plug a +12V/+5V supply into either CON4 or CON5 on the PC board. The board will then be directly powered from this supply, while the supply for the drives can then be taken from the unused connector. Note that you will need a Y-splitter cable if there are two drives.

In this case, you can use a surplus computer power supply to power both the boards and the drives. This will simply plug straight into either CON4 or CON5. Another option is to use a ready-made adapter like the Jentec JTA0202Y (from Taiwan). This unit supplies +12V and +5V at 2A each, which is enough to power two drives and the PC board. It also comes with the proper plug, so all you need then is a Y-splitter cable.

Setting up the drives

The two drives must be configured before being installed. Specifically, if you wish to connect only one drive, it can be configured as either a slave or master device.

Usually, this is accomplished by a jumper setting on the back of the drive. The drive will usually have a label indicating the appropriate position of the jumper.

If you wish to use two drives, however, make sure that one is configured as a master while the other is configured as a slave. It doesn't matter which is which, as long as they are not both slaves or both masters.

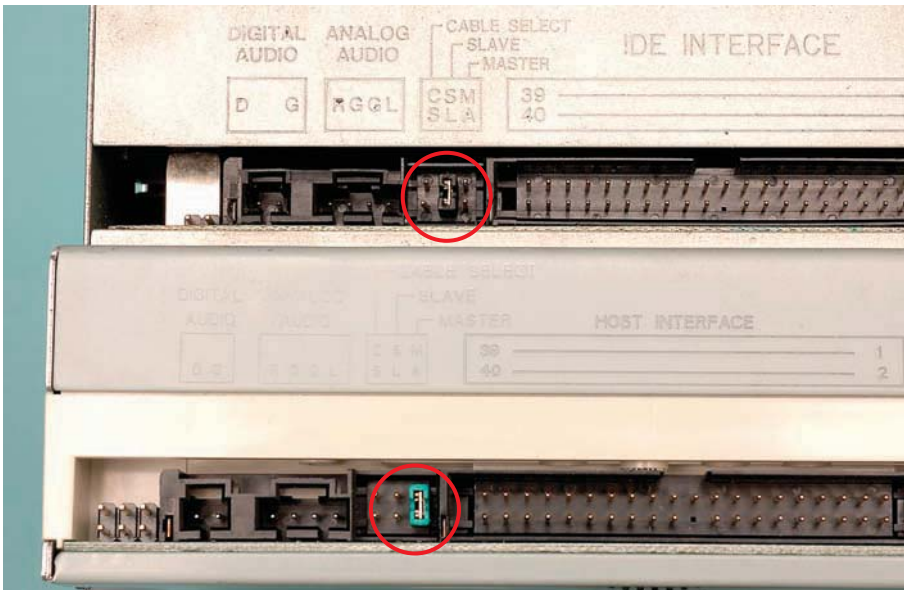
How auto-detection works

Let's now see how the micro detects any connected ATAPI devices at boot up.

First, a simple test is done. The micro writes a known value to an ATA register and then attempts to read that value. If the value read is the same as that written, the autodetect subroutine goes to the next stage.

Conversely, if the drive fails this test, it is assumed to be absent. Instead, 0xFF is returned as the value read due to internal pull-ups on all inputs (which incidentally, is not the value that is written).

The next stage of auto-detection involves searching for the signature that all ATAPI devices are required to have (according to the standard). In fact, all ATAPI devices have a unique signature of 0x14 and 0xEB (notice that 0x14 + 0xEB = 0xFF) in the Count Low and Count High registers on start up.



CD-ROM drives have three sets of jumper pins at the back to configure the drive. If you have just one drive, it can be configured as either a master (MA) or a slave (SL) using the jumper link. However, if you are using two drives, then one must be configured as a master and the other as a slave, as shown here.

The inquiry command of the ATA interface, while mandatory for ATA devices, is actually aborted by ATAPI devices. Instead, the effect of this ATA command on ATAPI devices is to put the ATAPI signature word in the Count Low and Count High registers. What is mandatory for ATAPI devices is to support the ATAPI inquiry command.

The algorithm for detecting the drives is as follows:

1) Perform a simple read-write test. Abort if this test fails, otherwise continue.
2) Select the drive by writing to the drive/head register.
3) Issue an ATA identify device command.
4) If the signature 0x14 0xEB in the Count Low and Count High registers is present, go to step 5.
5) If this signature is not present, the device is either absent or it is not an ATAPI device. Therefore, we may assume that no ATAPI device is present and terminate.
6) If the ATAPI signature is detected, we issue an ATAPI inquiry command to get further information about the drive and conclude that an ATAPI device is connected. The test is then terminated.

Firmware operation

Flowchart 2 shows the structure of the firmware. After initialisation, the program can optionally jump to a subroutine to set-up the remote control.

This should be done at least once, preferably the first time the program is run. Once the remote control has been successfully set up, the adapter is ready to be used.

The next stage in the firmware is the auto detection of the drives. Up to two drives can be connected, and they should be configured correctly as master or slave, as detailed previously.

The firmware then enters a 'finite state machine' by going to the neutral (or initial) state. It then listens for activity on the infra-red port and responds to the remote control commands.

There are three playing modes: 1) the default mode, 2) the repeat mode and 3) the random mode. In default mode, the adapter will play the current track and when that is done, will jump to the next track.

In repeat mode, the adapter plays the current track and then repeats it over and over. This mode is indicated by the digit '1' appearing as the last character of the first line of the display in playing mode.

Finally, in random mode, the adapter will play the current track and then select the next track randomly. This mode is indicated by the letter 'R' appearing as the last character of the first line of the display in playing mode.

In operation, the user can scroll between the default, repeat and random modes by pressing the 'Record'

What's a finite state machine?

A finite state machine (also known as a finite state automaton) is a set of states together with a transition table and a designated state that is the 'initial state'.

The transition table can be thought of as a table with three columns and a finite number of rows. The first column corresponds to the current state, the second column corresponds to the input,' and the third column corresponds to the next state. These triplets (X, I, Y) are interpreted as follows: if the machine is in state X and an input I is received, it moves to state Y. While there is no input, the machine stays in its current state.

For example, in our case, if the firmware is in the neutral state, and the user presses the Play key on the remote control, then the transition table dictates that the machine moves to the Playing state. This 'rule' would be written as the triplet ('Neutral', 'PLAY', 'Playing').

The user interface of this playback adapter is simply implemented as a finite state machine, meaning there are a number of rules that make up the transition table. The machine begins in the 'neutral' state, after a short initialisation.

Flowchart 2 shows the finite state machine implemented in the firmware. The transitions correspond to arrows, while the blue blocks are the possible states.

button on the remote control during play mode.

The volume is controlled by the Volume Up and Volume Down buttons on the remote. Up to 16 levels, ranging from muted (0) to full volume (15) can be selected.

The 'Mute' button has the usual effect of storing the current volume and then setting the volume to 0. If pressed again when the volume is 0, the original volume level is restored. The percentage balance of the right and left audio channels can be modified by the user, by pressing the Channel Up and Channel Down buttons on the remote. The percentages range from 0 to 100% in steps of 5%.

The volume for each channel is then calculated in terms of the balance using a simple formula:

1) *Volume (left) = (Balance left)/100 × Volume level*

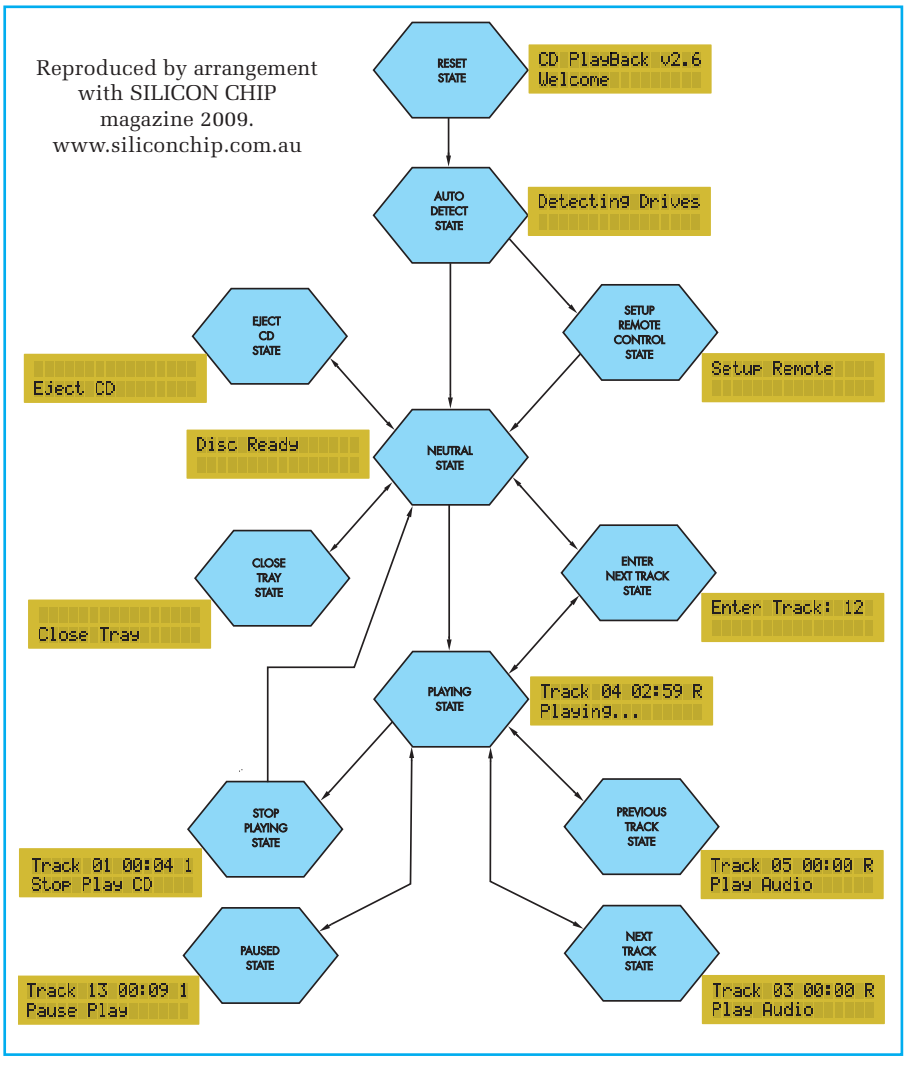
2) *Volume (right) = (Balance right)/100 × Volume level*

In playing mode, there are the usual control options, such as going to the next track (pressing the Fast Forward button) or to the previous track (pressing the Rewind button). You can also pause playing (by pressing Pause) or stop playing (by pressing Stop).

The 20+ button on the remote can be used to either eject the CD or close the tray (depending on whether the tray is already closed or open). The Line-In button is used to switch between master and slave devices, if two drives are connected and have been correctly detected by the firmware.

The 0 to 9 number buttons are used to select a particular track number to play. Simply press the correct number (which will be shown on the screen) and then press Play to play the selected track number.

As you can see, the user interface has been kept deliberately simple and intuitive. By the way, you can use virtually any RC5-compatible remote control since you can assign the buttons during the set-up procedure (more on this next month).



Flowchart 2: this flowchart shows the 'finite state machine' implemented by the firmware. After a short initialisation, which includes the automatic detection of connected drives, the firmware goes into the neutral state. From there, it starts accepting remote control commands that change the state of the machine. Typical display readouts corresponding to each state are also shown.

This page is intentionally left blank.

By MAURO GRASSI

Playback adapter for CD-ROM drives – Part 2

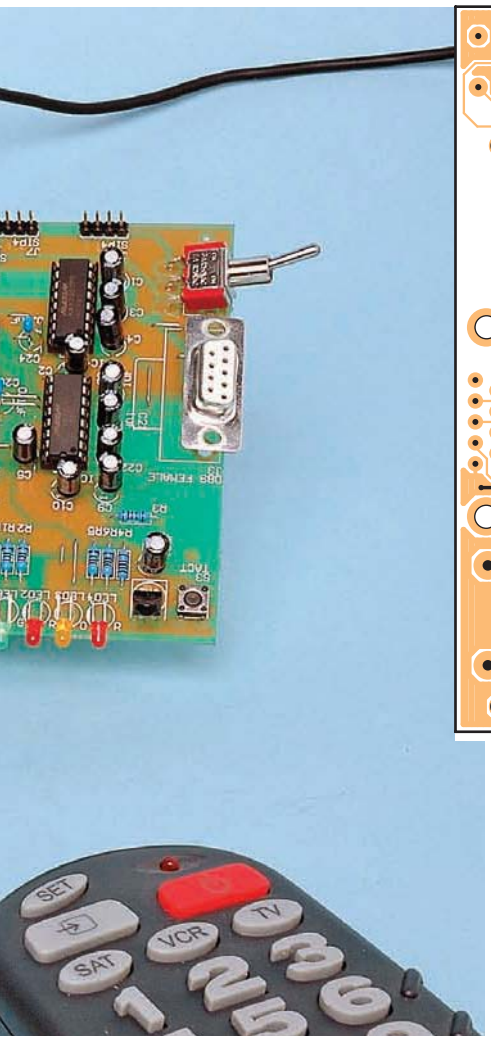
Last month, we published the circuit details of our new CD-ROM Player Adapter and described its operation. This month, we show you how to build it.

TO KEEP costs down, we've designed a single-sided PC board for this project. This board is coded 740 and measures 136mm × 97mm. The complete board and the CD-ROM drives could optionally be encased in a plastic case or mini-tower computer case, along with the power supply.

Because it's single-sided, the PC board is somewhat larger than a double-sided board would be and there are quite a few wire links that have to be installed. Before installing any parts though, do inspect the PC board for hairline cracks in the copper tracks or shorts. Some of the tracks are very fine and quite close together, so check carefully.

Installing the wire links

Fig.2 shows the locations of the wire links and these should all be installed first. Because some of these links are quite close together, it's essential that they be perfectly straight so that they don't short together.



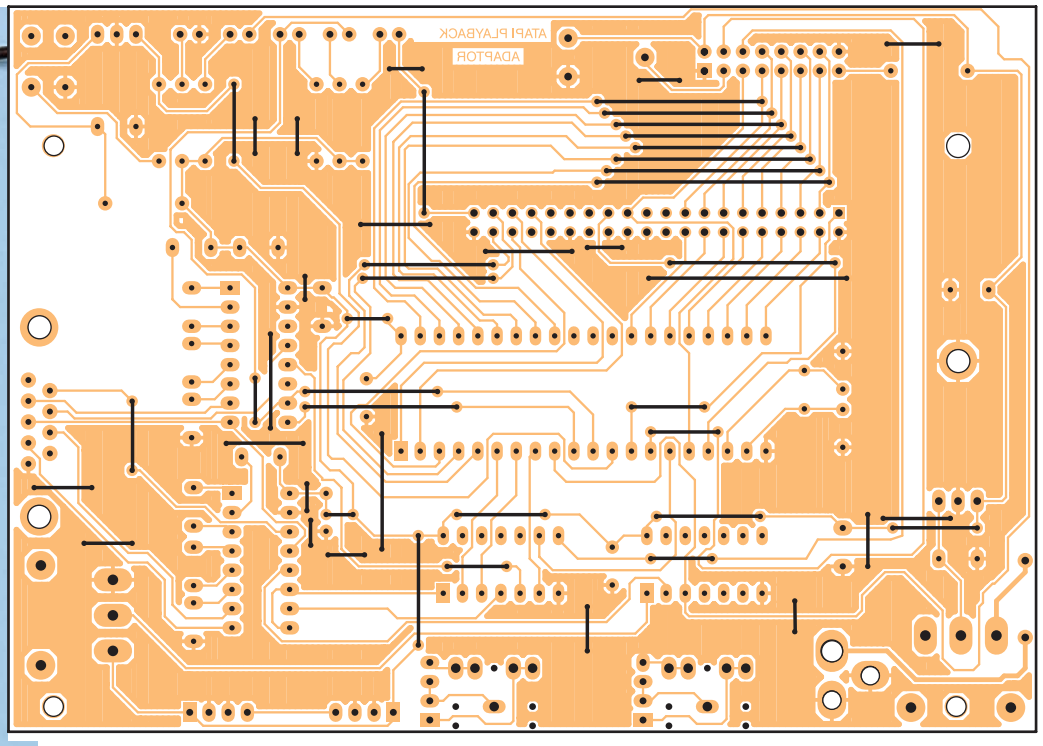


Fig.2: the first job in the assembly is to install all the wire links as shown here. Make sure that these links are straight, to prevent shorts – see text.

Programming The Microcontroller

IF you purchase a kit, then the microcontroller will be supplied pre-programmed. If not, then you will have to programme it yourself or purchase a pre-programmed chip from Magenta.

To programme it yourself, you will need to install both IC4 and IC5 (MAX232), as well as the other two logic ICs. You then load the hex file into Pony Prog 2000 and write to flash. If you don't already have this program, it is available as a free download from **www.lancos.com/ppwin95.html**.

You will need to first flick switch S1 and make sure the orange LED lights up. The micro is then ready to be programmed. We should also mention that if you are using Pony Prog 2000, you must change the setting under Setup -> Interface Setup and make sure that the only box that is ticked is the 'Invert Reset' box. Then select the correct device by going to Device -> AVR -> Atmega8515.

Prior to programming, Pony Prog 2000 needs to be calibrated for correct timing. To do this, simply go to Setup -> Calibration. This only needs to be done the first time you run Pony Prog 2000 on a new computer.

Now choose File -> Open Program (FLASH) File and select your hex file. Go to Command -> Program (FLASH) and Pony Prog 2000 should start programming your micro. Once programming is complete, you should flick switch S1 so that the orange LED goes out and then the firmware should start running.

The best way to straighten the link wire is to stretch it slightly by clamping one end in a vice and pulling on the other end using a pair of pliers. Each wire link can then be cut to length and its ends bent down at right-angles using needle-nose pliers before mounting it on the PC board.

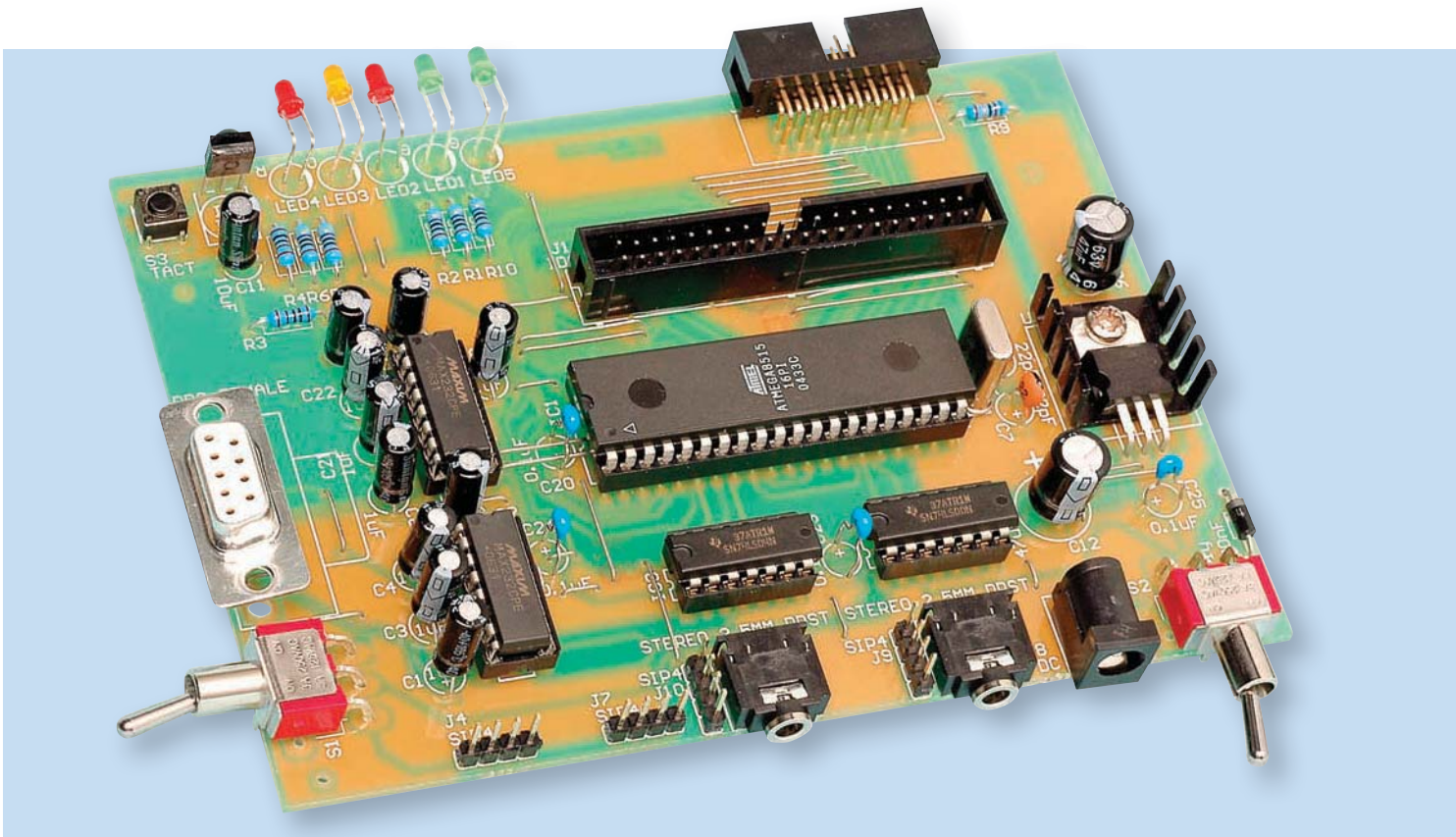
Once you've completed this task, you're ready to install the remaining parts. Fig.3 shows the parts layout on the board.

Start with the resistors, taking care to ensure that the correct value is used at each location. Table 1 shows the resistor colour codes, but it's also a good idea to check each one using a DMM (digital multimeter) before soldering it on to the PC board.

Next, solder in protection diode D1, making sure that it is oriented correctly, then install the small tactile switch (S3). The latter only fits correctly if it is the right way around.

The next step is to solder in the 40-pin IC socket for the microcontroller, plus the two 14-pin and two 16-pin DIP sockets for the other ICs. Note that only IC1, IC2 and IC3 are required for normal operation, while IC4 and IC5 are required only if you are planning to program the micro via this board. Make sure that the sockets are all oriented correctly – ie, with their notched ends arranged as shown on Fig.3.

The TO-220 regulator (REG1) is next on the list. As shown, this part is fitted with a small heatsink and is mounted horizontally on the PC board.



This view shows the fully assembled prototype PC board. Note that the two MAX232 ICs and the DB9 socket (CON3) are only necessary if you intend programming the microcontroller on the board. Note also that trimpot VR1 (contrast) and several wire links were added to the board after this photo was taken.

The correct procedure here is to first bend the regulator's leads down though 90°, exactly 5mm from its body. That done, the device and its heatsink are fastened to the PC board using an M3 × 10mm screw and nut. The leads are then soldered.

Don't solder the leads before bolting the device to the PC board. If you do, you could stress and break the PC tracks as the device is tightened down on the board.

Trimpot VR1 can go in next, followed by the 2.5mm DC socket (CON6) and the electrolytic capacitors. The latter are polarised, so make sure they go in the right way around.

Now solder in the 100nF bypass capacitors. Take particular care with the 100nF capacitor immediately to the left of IC1. It straddles a couple of wire links and should be mounted proud of the board so that its leads don't short against these links. The other 100nF capacitors can be pushed all the way down onto the board.

LED mounting

The five LEDs (LEDs 1 to 5) and the infrared receiver module (IRD1) can now be installed. As shown in the photos, the LEDs all go in with their leads bent at right angles and are mounted about 5mm proud of the PC board. A cardboard spacer cut to 5mm makes a handy gauge when it comes to bending the LED leads – spacing them evenly off the board, so that they all line up.

Take care to ensure that the LEDs all go in the right way around. Just remember that the anode (A) lead is always the longer of the two, and that the cathode (K) lead usually has a 'flat' on the package body next to it.

IRD1 can be mounted so that its lens lines up with the centres of the LEDs. It must be oriented so that its lens faces out from the PC board.

Installing the headers

The next job is to solder in the 16-pin and 40-pin IDC headers. Pin 1 of each of these is indicated by an arrow on the side of the header and this corresponds to the square pad on the PC board. Be sure to get them the right way around.

Table 1: Resistor Colour Codes

❑	No.	Value	4-Band Code (1%)	5-Band Code (1%)
❑	1	1kΩ	brown black red brown	brown black black brown brown
❑	5	470Ω	yellow violet brown brown	yellow violet black black brown
❑	1	100Ω	brown black brown brown	brown black black black brown
❑	1	51Ω	green brown black brown	green brown black gold brown

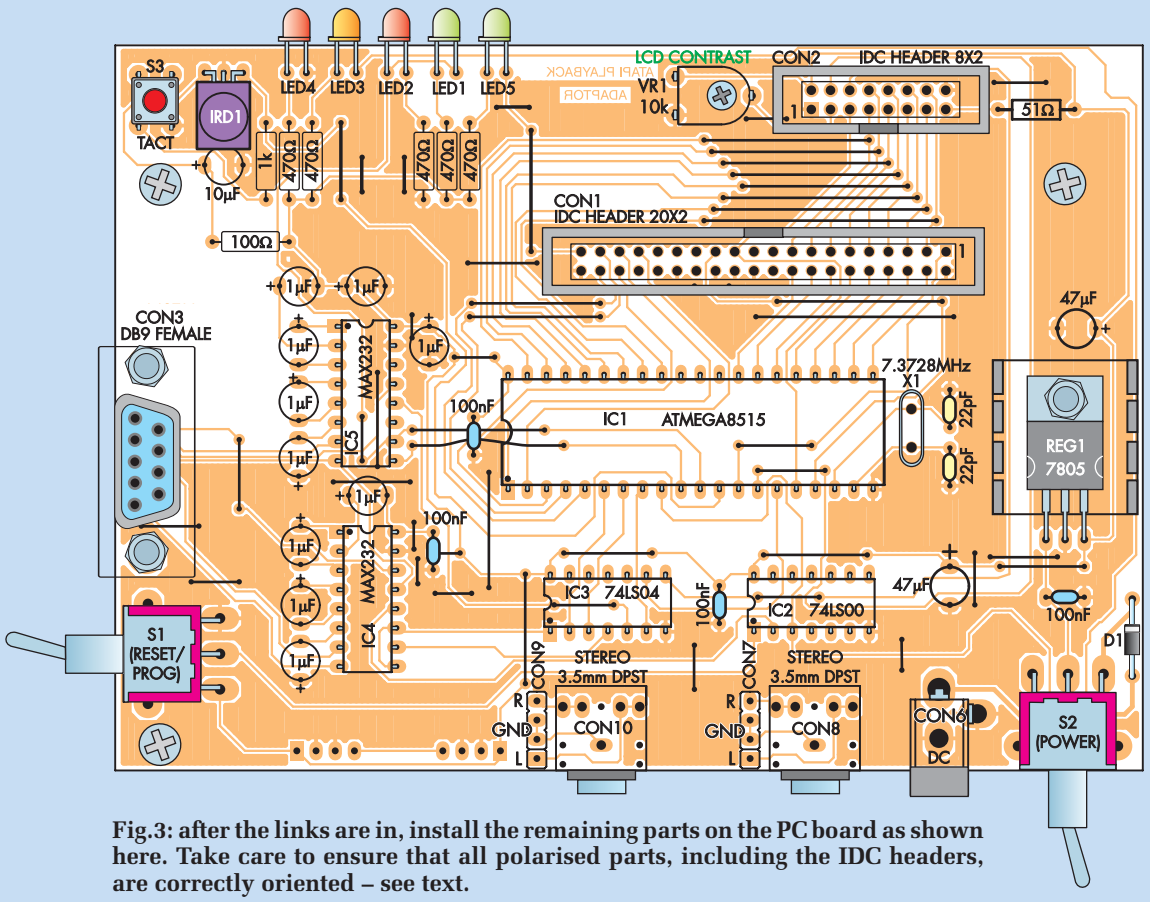


Fig.3: after the links are in, install the remaining parts on the PC board as shown here. Take care to ensure that all polarised parts, including the IDC headers, are correctly oriented – see text.

In each case, it's a good idea to initially solder just two pins of the header and then check that it is sitting flat against the PC board. After that, it's a routine job to solder the rest of the pins.

Finally, complete the PC board assembly by installing the 7.3728MHz crystal (it can go in either way), the two 22pF capacitors, the DB9 serial port connector (CON3), the two stereo jack sockets (CON8 and CON10), the two 4-way SIL pin headers (CON7 and CON9) and the two toggle switches.

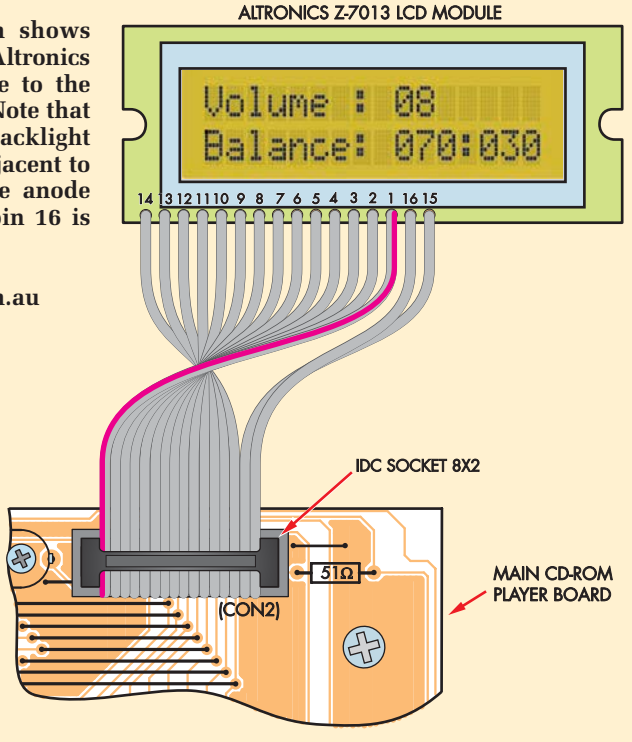
Connecting the LCD module

The LCD module to use must conform to the Hitachi HD44780 industry standard. This has an interface consisting of 16 or 14 lines, depending on how the LED backlight is connected.

A 16-way (or 14-way) ribbon cable is used to make the connection to the LCD module, and this is terminated at the other end in a 16-way IDC line socket, with the red stripe on the cable going to pin 1. This end then plugs directly into the 16-way IDC header on the PC board.

Fig.4: this diagram shows how to connect the *Altronics Z-7013 LCD module to the 16-pin IDC socket. Note that pins 15 and 16 (the backlight connections) are adjacent to pin 1. Pin 15 is the anode connection, while pin 16 is the cathode.

***www.altronics.com.au**



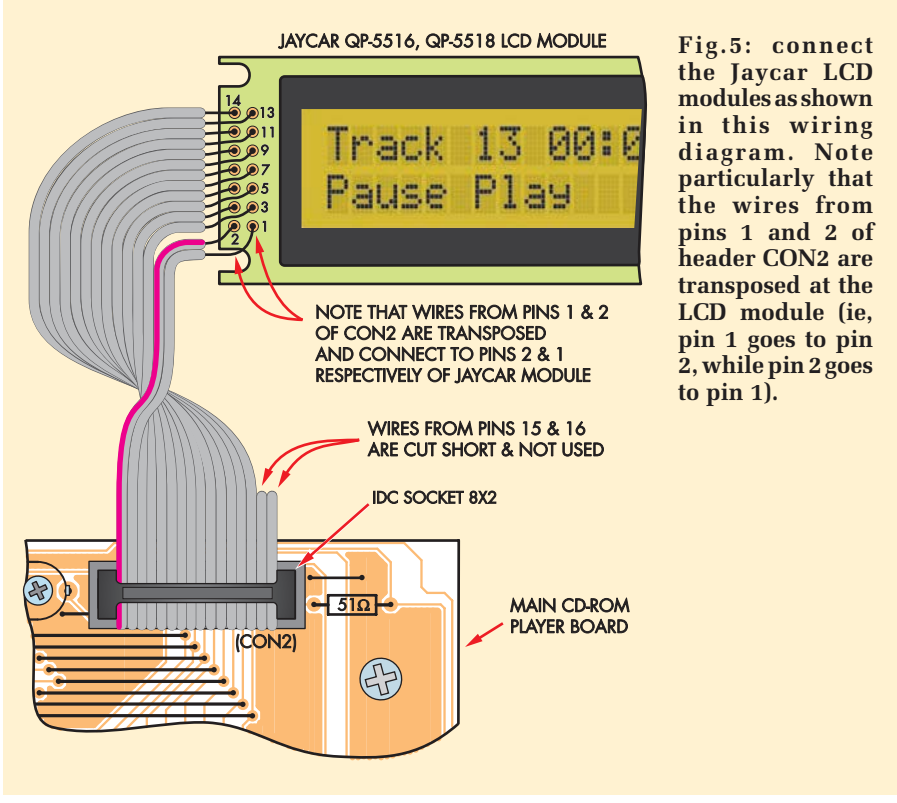
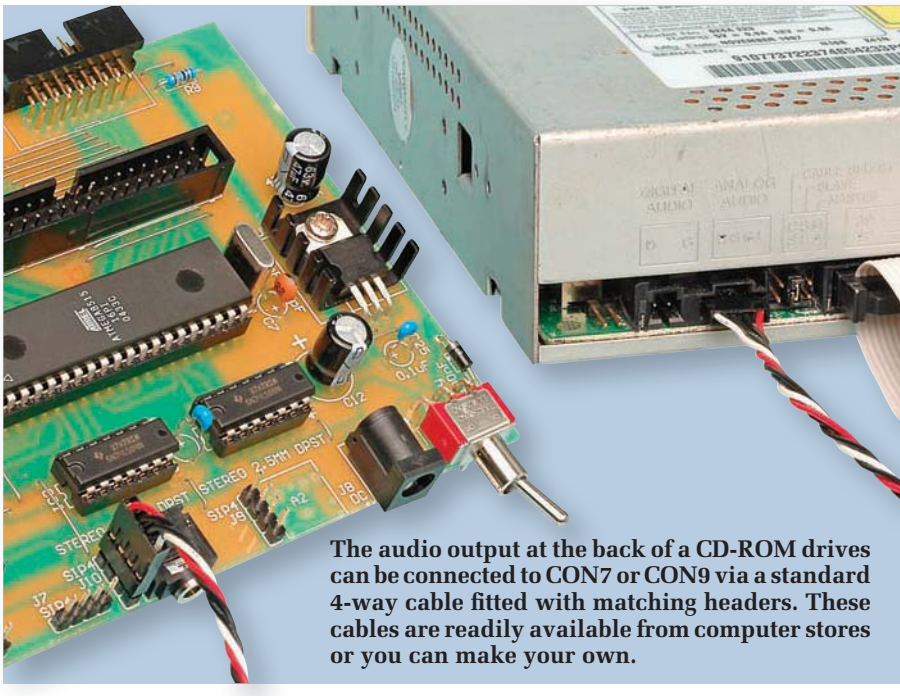


Fig.5: connect the Jaycar LCD modules as shown in this wiring diagram. Note particularly that the wires from pins 1 and 2 of header CON2 are transposed at the LCD module (ie, pin 1 goes to pin 2, while pin 2 goes to pin 1).

Fig.4 shows the connections to the Altronics Z-7013 LCD module. This device has 16 pins all in one line along the bottom edge of the board (although pins 15 and 16 are adjacent to pin 1).

Alternatively, the Jaycar QP-5516 and QP-5518 LCD modules each have a 2 × 7-pin arrangement at one end; ie, there are only 14 connections. The backlight connections are made on the module itself, so pins 15 and 16 of CON2 are not connected in this case. Fig.5 shows the wiring connections for the specified Jaycar modules.

In particular, note that pin 1 on the Jaycar modules is the +5V connection, while pin 2 is the 0V connection. It's the other way around on the Altronics module, where pin 1 is 0V and pin 2 is +5V.



The audio output at the back of a CD-ROM drives can be connected to CON7 or CON9 via a standard 4-way cable fitted with matching headers. These cables are readily available from computer stores or you can make your own.

Fig.6: assigning the buttons on the remote for the various functions is easy – just follow the prompts on the LCD readout. This is the prompt for assigning the 'Volume Up' button.

Software

The software files will be available via the *EPE* Library site, accessed via **www.epemag.com**. Pre-programmed Atmel micros will also be available from Magenta Electronics – see their advert in this issue for contact details.

Testing and troubleshooting

Great care has been taken to ensure that the firmware is free from bugs, but we cannot guarantee that it will work with every CD-ROM drive. We did test the board with six different CD-ROM drives and it worked well.

The only problem was that two of the drives did not respond to the volume change command. However, we are not sure that these two drives were actually functioning correctly all of the time, as they appeared to have intermittent faults.

Whichever drive you want to use for this project, make sure it is an ATAPI device (check that the IDC connector on the back of the drive has 40 pins, as opposed to 50 pins for a SCSI connector). Note also that the adapter will not work with some smaller form factor CD-ROM drives which have 44-pin connectors (akin to the 2.5-inch notebook hard drives).

Before plugging in the micro (IC1), the first thing to do is to check the

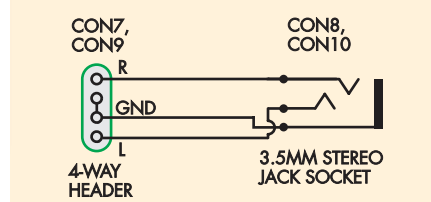


Fig.7: 4-way headers CON7 and CON9 are connected to the two 3.5mm stereo jack sockets. This makes it easy to connect to CD-ROM drive audio outputs via a standard stereo jack plug.

Table 3: CON2 Pin Assignments

Pin	Pin	Description
1	V_{SS}	Supply rail for module; typically GND
2	V_{DD}	Supply rail for module; typically +5V
3	V0	Set LCD contrast
4	RS	RS = 0 selects instruction; RS = 1 selects data
5	R/W	R/W = 0 selects write; R/W = 1 selects read
6	E	E = 1 selects the LCD module
7	D0	Data bus bit 0
8	D1	Data bus bit 1
9	D2	Data bus bit 2
10	D3	Data bus bit 3
11	D4	Data bus bit 4
12	D5	Data bus bit 5
13	D6	Data bus bit 6
14	D7	Data bus bit 7
15	A	LED backlight anode
16	K	LED backlight cathode

power supply rails. To do this, first connect a 9V or 12V DC plugpack to the DC socket (CON6) and switch on. That done, check that the OUT terminal of REG1 is at +5V with respect to ground. Similarly, you should be able to measure +5V on pin 40 of the 40-pin socket, while pin 20 should be at 0V.

If these checks are OK, switch off and plug in the micro. Make sure that this device is oriented correctly and that all its pins go into the socket. In particular, take care to ensure that none of the pins are folded back under the device.

That done, set trimpot VR1 to mid-range and switch on again. Check that the LCD module initialises correctly, then adjust VR1 for optimum display contrast.

Remote control functions

The firmware has an option that allows you to use any RC5 protocol remote control. That means that you can use virtually any universal remote control, plus most of the remotes that are commonly used with TV sets, VCRs and DVD players.

Parts List – CD-ROM Controller

1 PC board, code 740, available from the *EPE PCB Service*, 136 mm × 97mm
1 16×2 backlit LCD module (Jaycar QP-5516 or QP-5518, Altronics Z-7013)
1 PC-mount 40-pin IDC header (CON1)
1 PC-mount 90° 16-pin IDC header (CON2)
1 PC-mount DB9 female RS-232 socket (CON3) (optional for programming)
1 28-pin or 40-pin SIL header strip
1 16-way IDC ribbon cable (to connect LCD module, length to suit)
1 16-way IDC line socket
1 40-way IDE HDD cable (to connect CD-ROM drives)
2 3.5mm stereo sockets, PC-mount (CON8,10)
1 2.5mm DC power socket, PC-mount (CON6)
1 PC-mount micro tactile switch (S3)
2 SPDT 90° PC-mount toggle switches (S1, S2)
2 16-pin IC sockets (optional for programming)
1 40-pin IC socket
2 14-pin IC sockets
1 TO-220 mini finned heatsink
1 7.3728MHz crystal (X1)
1 10kΩ horizontal trimpot (VR1)
1.5m tinned copper wire for links
1 M3 × 10mm machine screw
1 M3 nut

Semiconductors
1 ATMega 8515 microcontroller, programmed with CDATA.hex (IC1)
1 74LS00 quad NAND gate (IC2)
1 74LS04 hex inverter (IC3)
2 MAX232 RS-232 transceivers (IC4,IC5) (optional – see text)
1 infrared receiver module (IRD1) (Jaycar ZD-1952)
1 7805 3-terminal regulator (REG1)
1 1N4004 400V 1A rect. diode (D1)
2 3mm green LEDs (LED1,LED5)
2 3mm red LEDs (LED2,LED4)
1 3mm orange LED (LED3)

Capacitors
2 47μF 16V electrolytic
1 10μF 16V electrolytic
10 1μF 63V electrolytic (optional for programming)
4 100nF monolithic
2 22pF ceramic

Resistors (0.25W, 1%)
1 1kΩ 1 100Ω
5 470Ω 1 51Ω

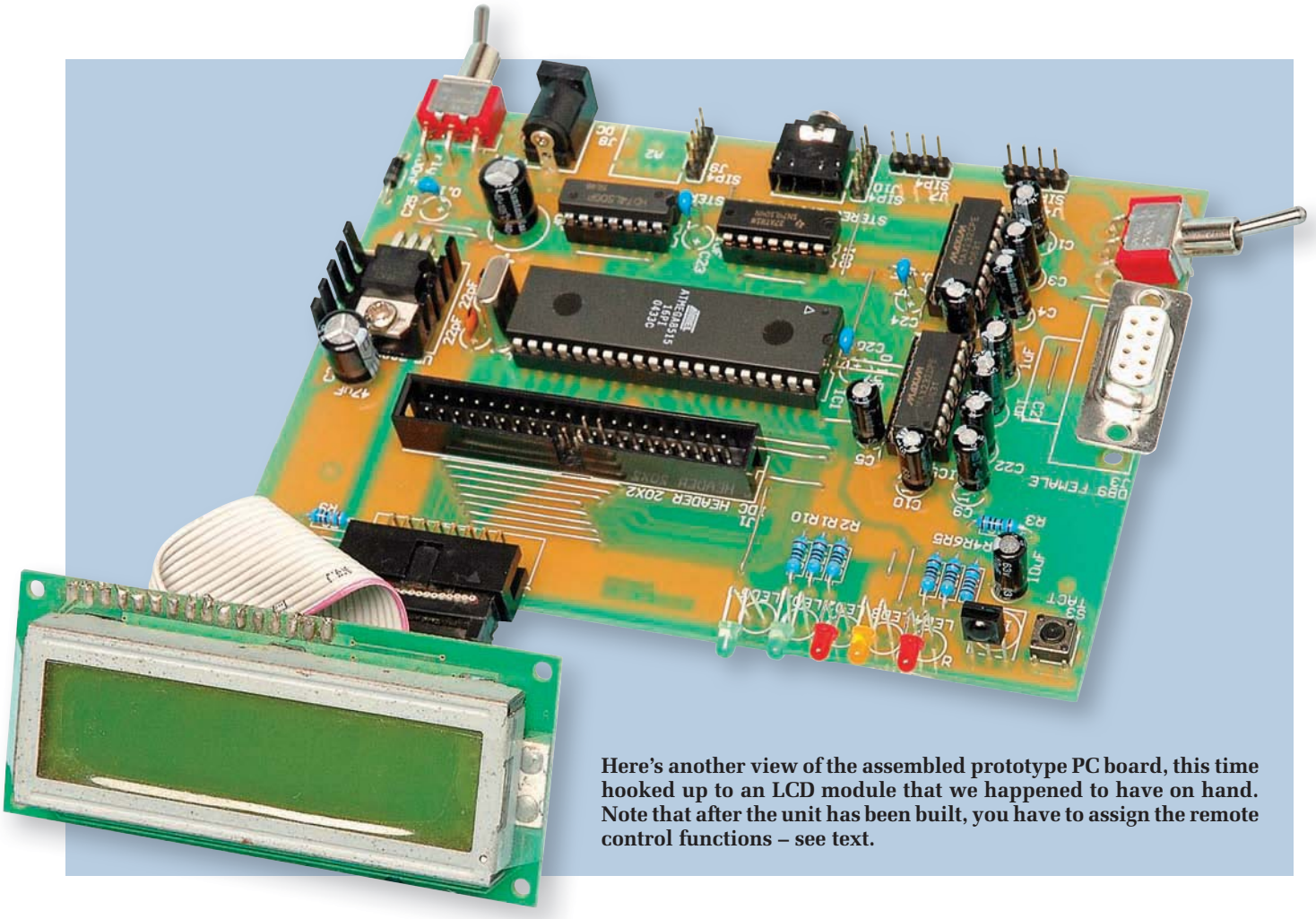
Power Supply Options

Last month, we stated that one of the supply options for the board was to plug a computer power supply into either CON4 or CON5. We have since decided to scrap that option and now recommend that you stick to powering the board from a 9V or 12V DC plugpack.

The disk drives can be powered directly from a computer power supply. Alternatively, if you don't want the fan noise of a computer power supply, you can use a mains adaptor like the Jentec JTA0202Y. This unit supplies +12V and +5V rails at 2A each, which is enough to power two drives and comes with the correct plug (you'll need a Y-splitter cable to power two drives).

You purchase this adaptor via eBay.

The first step is to assign the buttons that will control the various functions. To do this, you first need to press and hold the 'Remote Program' button (S3) while the device resets. To get the device to reset, you toggle switch S1 so that the orange LED lights and then toggle it again to turn the LED off (ie,



Here's another view of the assembled prototype PC board, this time hooked up to an LCD module that we happened to have on hand. Note that after the unit has been built, you have to assign the remote control functions – see text.

you hold S3 down while you toggle S1 twice).

This resets the micro and takes you to the 'Setup Remote' screen. Here you program the keys used for the project. The device will guide you through the set-up, and the keys that you define will be stored in EEPROM for later use.

For example, when the screen displays 'Press Vol Up' (see Fig.6), you simply press the 'Volume Up' button on your remote. It's just a matter of cycling through all the options until the button assignment has been completed.

This means that you can use any spare RC5 remote and define the keys as you see fit. The 'Power' button is deliberately unused for this project, and this lets you control the device with your TV remote control, for example.

In other words, because the 'Power' button is unused, you can have your TV off and use its remote to control the CD-ROM Player Adapter. Then, when you are finished with the adaptor, you can switch it off and use the remote to control your TV again.

Of course, you won't be able to play a CD and watch your TV simultaneously using the same remote, but this feature can keep costs down. It means that you don't have to purchase a separate universal remote control, although you can if you wish.

Operation

The user interface has been kept quite simple. Occasionally, issuing a command will result in an error screen. This is perfectly OK as the firmware has been designed to be quite tolerant of errors. If it happens, simply try the command again, but if the problem persists, it may indicate an incompatibility or fault with your drive.

The 'Error' screen typically looks like that shown in Fig.2 last month, but may have different numbers that are used for debugging purposes. The hexadecimal numbers give an indication of the state of the ATA registers and the state of the machine when the error occurred.

If the errors consistently re-occur, this information will help to diagnose the problem.