

# DONGLE SAFE

A dongle is a small plug-in device which must be attached to your computer's printer port to be able to run a certain program. Over the years, the dongle has been recognized as one of the most efficient ways of protecting software against illegal copying. Not surprisingly, a (personalized) dongle is supplied with many high-end CAD and DTP programs. These programs look constantly for the dongle, and often will not run at all if it is not fitted. Dongles are generally considered precious objects, but, unfortunately, not only by their rightful owners...

OVER the past few years, various techniques have been devised to combat illegal copying of computer software. In a constant effort to protect its products, the software industry has shown lots of activity in finding a copy protection system which gives a minimum of fuss to the licensed user. The

dongle, probably first used on Commodore-64 computers, has proven to be one of the most effective and least cumbersome protection systems available. And yet, the device has one important disadvantage. Simply because it is so difficult to 'hack' or 'clone', the dongle itself has become an interesting object to steal. Unfortunately, anyone who has a small

screwdriver can remove a dongle from a PC in a few seconds. What's more, because the dongle usually has a sticker on it proclaiming the name of the program it belongs to, it will have a high value on the black market.

The dongle safe presented in this article was designed as a simple, low-cost means of giving extra security and peace of mind to dongle owners. In effect, the safe makes stealing your precious dongle a lot harder, and makes casual theft almost impossible. The basic idea is simply to hide the dongle from sight. That is achieved with the aid of an insertion card into which a number of dongles can be plugged. Because the insertion card with the dongle(s) on it is fitted inside the computer, it is not immediately obvious to an outsider that a dongle is being used. Computer criminals on the hunt for a specific type of dongle will only be able to find it by opening the computer, which takes rather more time than simply pilfering the device from the parallel port at the back of the computer.

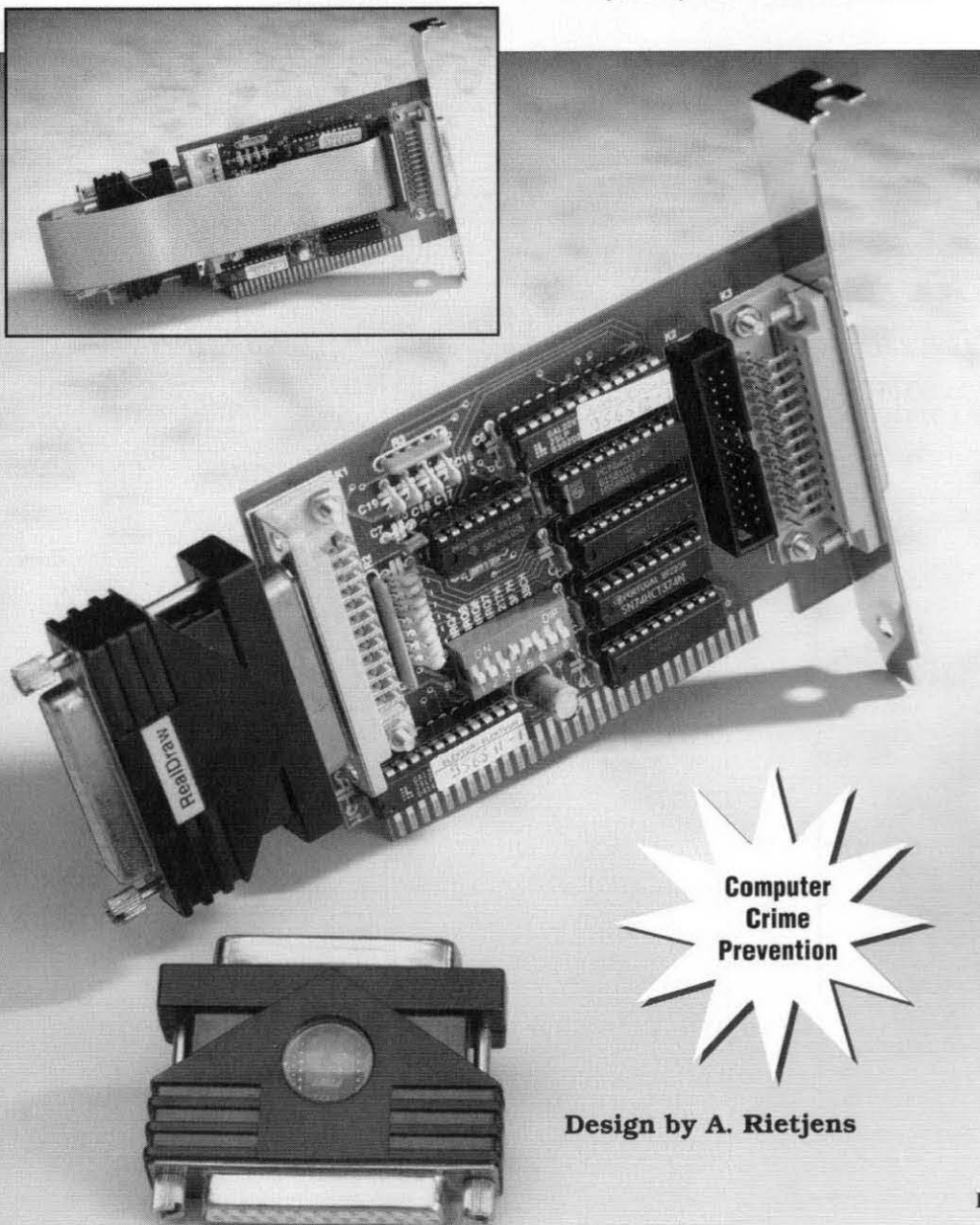
The dongle safe has two additional features. First, the printer port implemented on the insertion card may be connected to a socket on the card fixing bracket, via a short length of flat-cable connected to the last dongle. That is only possible, however, if the dongle is suitable for connecting between the Centronics port and the printer. Unfortunately, some dongles cause problems if that is done in combination with other dongles.

The second additional feature of the dongle safe is its ability to act as an add-on parallel input port. Data applied to the inputs is then directly accessible to the CPU via the databus. This is a feature not offered by a conventional printer port.

## A simple approach

Dongles are generally connected to the PC's parallel printer ('Centronics') port. Three base addresses are reserved for this port: 3BCH, 278H and 378H. Printer port logic may be located at each of these addresses. The block diagram of a printer port is shown in **Fig. 1**. The function of the address decoder will be obvious: apart from fixing the base address, it also determines the addresses of the various printer port registers. The PC's databus is buffered via a bidirectional bus transceiver to prevent undue loading.

To be able to control a printer, you need a number of control signals such as strobe and init. The control levels



Design by A. Rietjens

for these lines are switched with the aid of a register. For the purpose of checking, the levels of these lines may be read back via another register.

While it is busy printing, the printer returns status information to the computer. The main signals used for this information exchange are Paper Empty (PE), Acknowledge (ACK) and Busy. The computer reads back this information via the status register.

The process of sending data to the printer operates as follows. First, the data to be transmitted is copied into the write register, whereupon a strobe pulse is generated. For checking purposes, the data can be read back from the write register. Usually, the write register is a buffered latch, so that data is continuously present at the output. This arrangement unfortunately does not allow an output to be used as an input. None the less, the I/O controller used here allows the data register to be switched to high-impedance mode. If this is done, the data register no longer affects signals which are being read. Those of you keen on experimenting may want to write their own software to give extra functionality to the port. Normally, however, you will not notice this option because after a reset the computer automatically arranges for all control signals to be given levels which select 'normal' printer port operation.

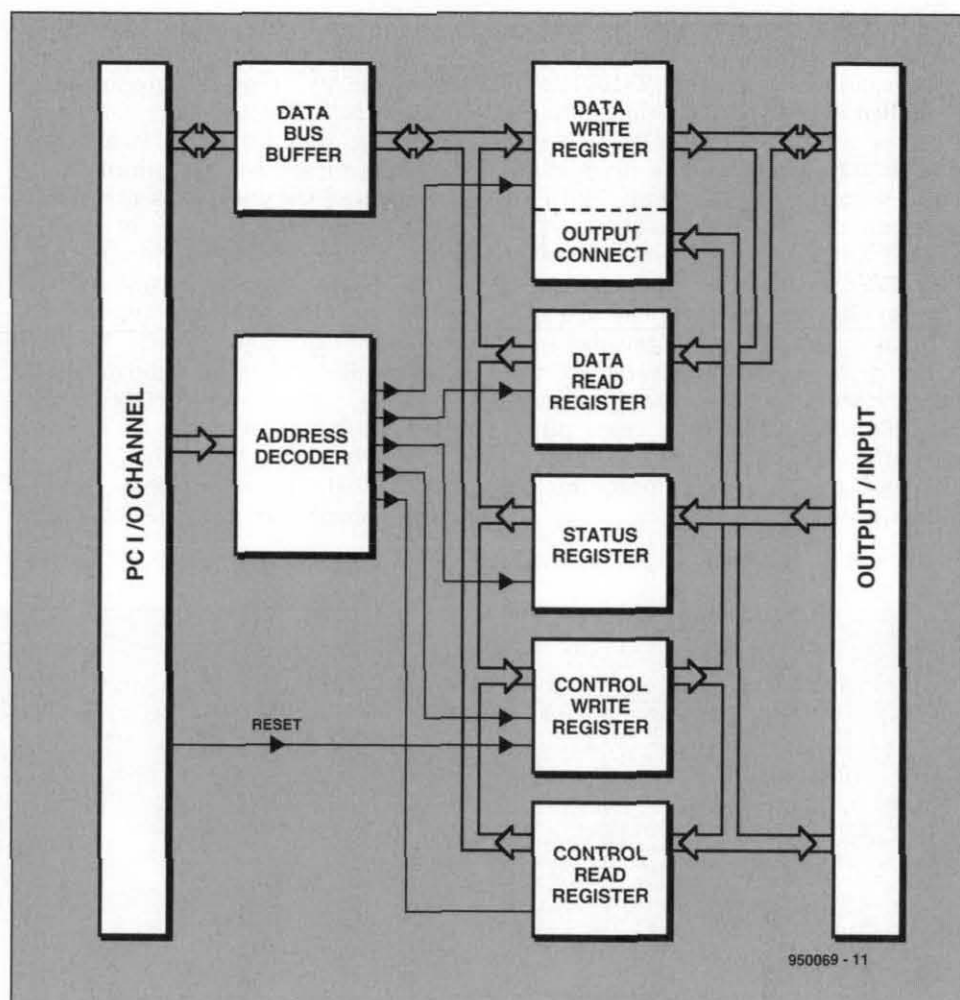


Fig. 1. This block diagram shows the architecture of a printer port in an MS-DOS PC.

## Hardware

A number of points which have been made in the previous description are immediately apparent from the circuit diagram in Fig. 3. Although the schematic may look complex at first because of the large number of connectors, it is really quite simple and easy to follow.

The data register is connected directly to the PC's databus, and is formed by IC<sub>1</sub>, a type 74HCT245. The data-write register, IC<sub>3</sub>, (a type 74HCT374) contains the data to be printed. The output signals of this IC

Base address	R/W action	Effect
+0	write	write data to printer
+0	read	read data from printer's databus
+1	read	read printer status
+2	write	write control word to printer
+2	read	read control word for printer
+3	write	not allowed
+3	read	not allowed

Table 1. Overview of reserved addresses around the base address of the dongle safe card.

```

C:\>debug
-d 0:400
0000:0400 F8 03 F8 02 00 00 00 00-78 02 BC 03 00 00 F1 4B .....x.....K
0000:0410 63 44 BF 80 02 00 00 00-00 00 30 00 30 00 0D 1C cD.....0.0...
0000:0420 64 20 20 39 30 0B 3A 27-34 05 30 0B 30 0B 0D 1C d 90.: '4.0.0...
0000:0430 73 1F 0D 1C 64 20 65 12-62 30 75 16 67 22 00 00 s...d e.b0u.g"...
0000:0440 6F 00 C0 00 00 00 00 00-00 03 50 00 00 10 00 00 o.....P.....
0000:0450 00 08 00 00 00 00 00 00-00 00 00 00 00 00 00 00 .....
0000:0460 07 06 00 D4 03 29 20 88-05 87 90 00 8E EC 0D 00 .....
0000:0470 00 00 00 00 00 01 00 00-14 14 14 14 01 01 01 01 .....
-q
C:\>

```

950069 - 14

Fig. 2. The DOS program DEBUG gives you an instant overview of the addresses in use for printer (and COM) ports.



go directly to the output ( $K_1$ ), and is also applied to the inputs of the data-read register,  $IC_4$  (a type 74HCT245). The outputs of  $IC_3$  are connected to an array of pull-up resistors, which ensure steady levels on the lines PD0-PD7 when  $IC_3$  is switched to high-impedance ('three-state'). The control register,  $IC_5$ , consists of a 74HCT273, and really acts as a kind of director in the circuit. The signals transmitted by the circuit (strobe, init, select and auto) are protected via buffer with open-drain outputs (74HC05).  $IC_5$  receives a reset pulse via buffer  $IC_{7e}$  when the computer is switched to the start configuration after a 'hard' reset.

The address decoder in the dongle safe is built around  $IC_2$ , a GAL type 20V8. The two read registers (status and control) also take the form of a GAL, only this time the larger type 22V10 is used. The programming descriptions of the two GALs are shown in the box opposite. Those of you not in possession of a GAL programmer will be pleased to know that the two GALs are also available ready-programmed through our Readers Services. The addresses determined by the GALs, and the associated functions, are listed in **Table 1**. The two address decoding functions are realized by GALs because these components reduce the component count

and thus the board space. In effect, they enable the circuit to remain as compact as possible.

The switches in DIL switch block  $S_1$  may be used to select the card address (278<sub>H</sub>, 378<sub>H</sub> or 3BC<sub>H</sub>), as well as the desired interrupt line (IRQ3, IRQ4, IRQ5 or IRQ7). The most obvious choice for the printer port is IRQ5 or IRQ7. The other two, IRQ3 and IRQ4, may also be used, provided they are not already in use by a communication (COM) port. Check to make sure!

## Construction

The artwork for the dongle safe insertion card is shown in **Fig. 4**. The card

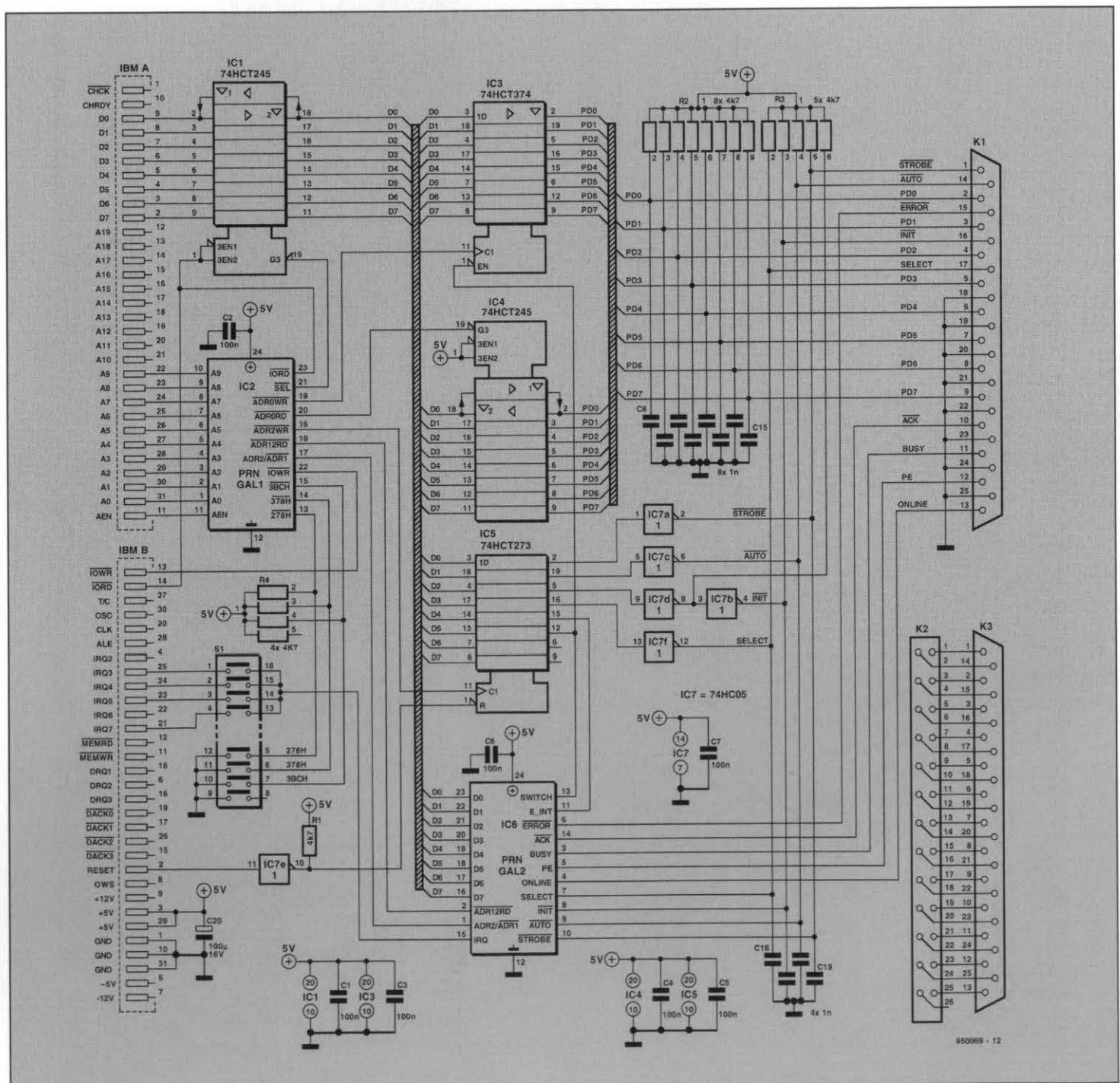


Fig. 3. Circuit diagram of the dongle safe card. The card allows dongles to be housed inside the PC, where they are much safer from theft.

is double-sided and through-plated, and the one supplied through the Readers Services comes with gold-plated PC extension bus connections. Those of you with access to a GAL programming system may burn their own GALs for the project with the aid of the listings on the previous page. Do pay attention to the inverting marks in the equations. Fortunately, these two ICs are also available ready-programmed through our Readers Services.

Boxheader  $K_2$  and connector  $K_3$  may be omitted if you do not wish to use the option of connecting the output of the last dongle to an extra port. If you do want the extra port, that can be created by making a short cable, consisting of a short length of 25-way flatcable. One end is fitted with a 25-way IDC style (press-on) sub-D plug, the other, with a 26-way boxheader (also IDC style). Pin 26 of this header is not used. The connection via the cable is laid out so that it is not necessary to twist the cable at one of the ends. Pin 1 of  $K_1$  is connected to pin 1 of  $K_2$ .

## Setting up and testing

Before you install the completed card into your computer, run a check on the addresses already in use for printer ports. Most of today's PCs (i.e.,

## COMPONENTS LIST

### Resistors:

- $R_1 = 4k\Omega$
- $R_2 = 4k\Omega$  8-way SIL array
- $R_3 = 4k\Omega$  5-way SIL array
- $R_4 = 4k\Omega$  4-way SIL array

### Capacitors:

- $C_1 - C_7 = 100 \text{ nF}$
- $C_8 - C_{19} = 1 \text{ nF}$
- $C_{20} = 100\mu\text{F}$  16 V radial

### Semiconductors:

- $IC_1, IC_4 = 74\text{HCT}245$
- $IC_2 = \text{GAL}20\text{V}8$  (order code 956511-1)
- $IC_3 = 74\text{HCT}374$
- $IC_5 = 74\text{HCT}273$
- $IC_6 = \text{GAL}22\text{V}10$  (order code 956512-1)
- $IC_7 = 74\text{HC}05$

### Miscellaneous:

- $K_1, K_3 = 25\text{-way}$  angled D socket, PCB mount.
- $K_2 = 26\text{-way}$  boxheader.
- $S_1 = 8\text{-way}$  DIP switch.
- PC card fixing bracket with hole for D25 connector.
- PCB and GALs: set order code 950069-C (see page 70).

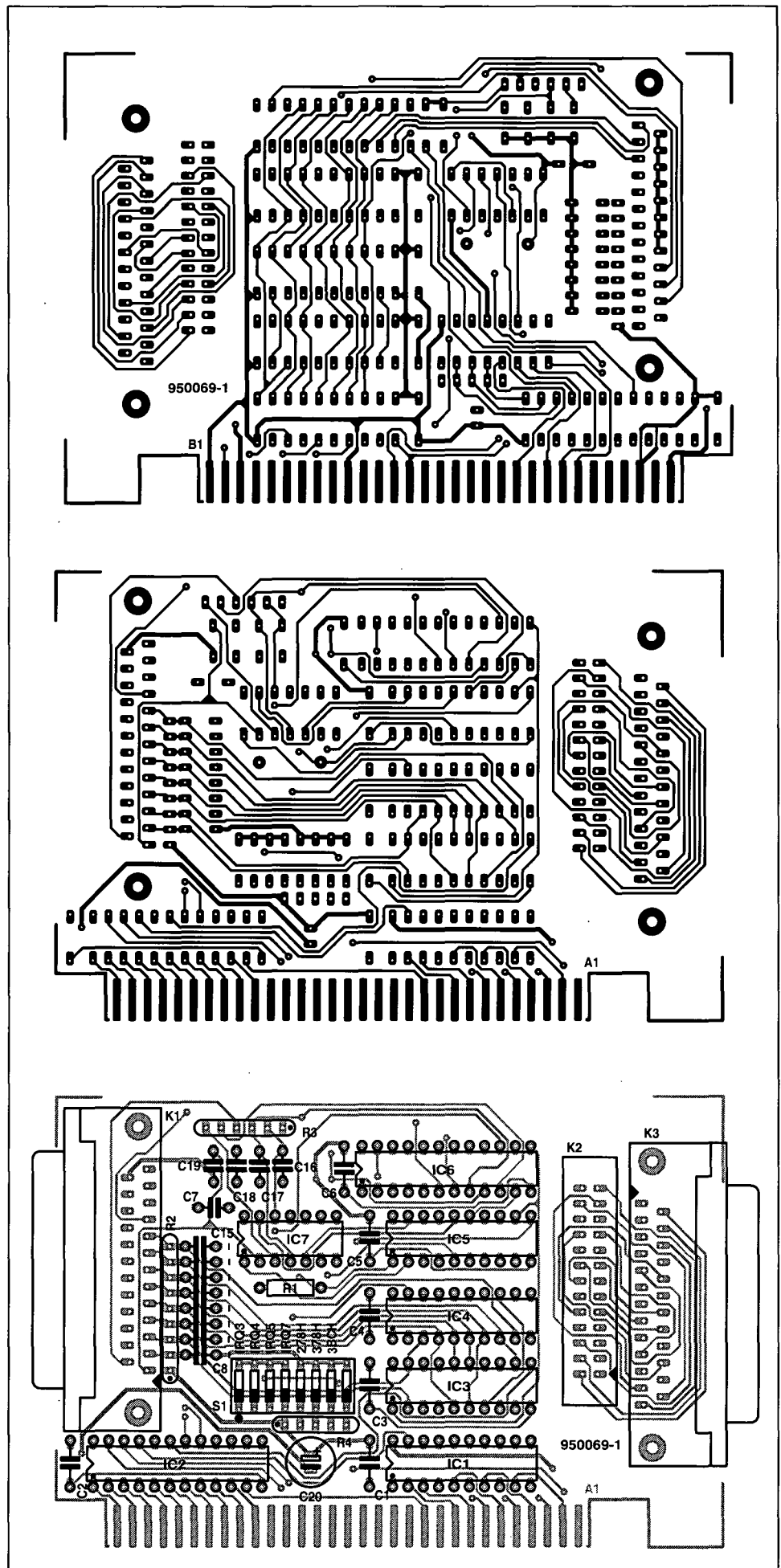


Fig. 4. Copper layouts and component mounting plan of the double-sided through-plated printed circuit board designed for the dongle safe card (board available ready-made through the Readers Services, see page 70).

## The two GALs on the dongle safe card

The first GAL to be examined in greater detail here is address decoder IC<sub>2</sub>.

The structure of the programmed GAL is straightforward, as illustrated below. The three possible port addresses, 3BC<sub>H</sub>, 278<sub>H</sub> and 378<sub>H</sub> are decoded by three AND gates, and combined into a single selection signal by an OR gate. Next, this selection signal is combined with the lowest address bits (A0 and A1) and the read and write signals (IOR and IOWR). These signals are used to give access to all registers. Because the status and control registers are combined in hardware, a single control signal is created for these functions. The ADR2/ADR1 signal does the final selection between these two registers.

The above structure results in a fairly simple file for the GAL programmer. The command and file format is compatible with the Opal Junior software which is supplied with the *Elektor Electronics* GAL programmer. The listings allow anyone with access to this programming system to burn his/her own GALs for the project.

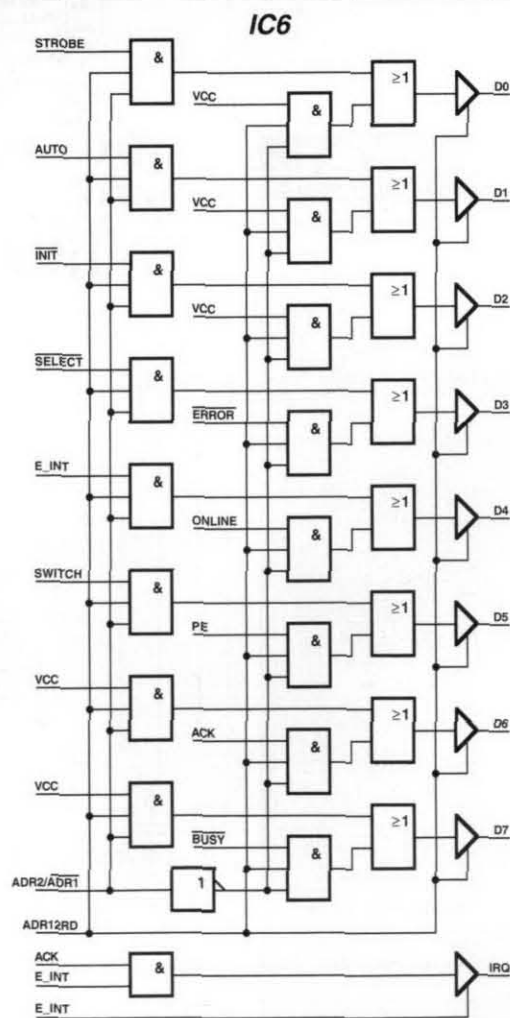
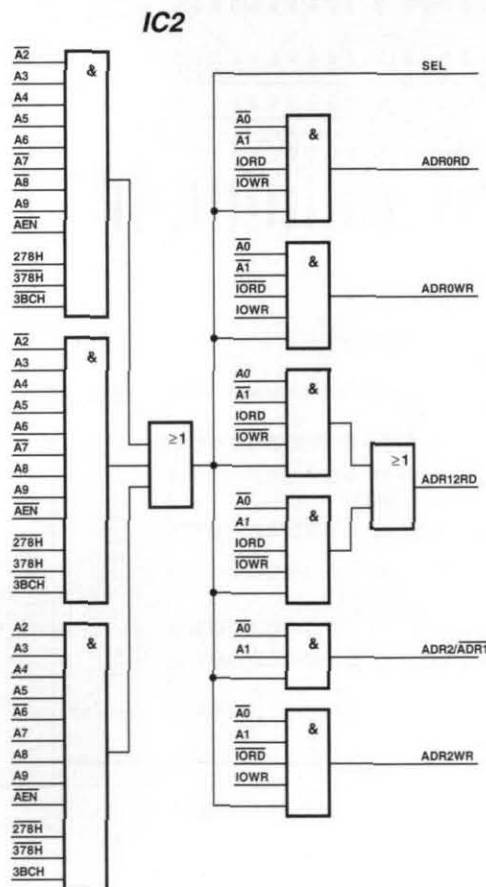
The first few lines of the programming file provide some information on the GAL to be produced. The command CHIP is used to indicate the GAL type to be used, in this case, a 20V8. Next, a logic name is assigned to each pin of the GAL, observing the active level (low or high) of the relevant pin. This makes writing the Boolean equations a lot easier. Because the 'define' instructions described in the next few lines may be inserted in the equations, they too help to make the equations easy to follow.

The mechanism so programmed in the GAL defines the three base addresses as well as the read and write conditions, and makes setting up the final equations very easy indeed.

### Status and control registers

The contents of IC<sub>6</sub>, the second GAL in this project, is also shown in the box below. This GAL simulates the status and control register. The selection logic is used to choose between one of two AND gate banks. The level of the ADR2/ADR1 line determines which set of AND gates is used. A high level enables the 'control' signals to be passed, a low level, the 'status' signals. As soon as the ADR12RD read signal becomes active, the information appears. In addition to the two previously mentioned registers, the GAL also contains the interrupt enable (E\_INT) decoding logic.

The programming file for this GAL is largely similar to that for IC<sub>2</sub>. However, a 22V10 is used in lieu of a 20V8 because nine outputs are required. New in the programming file are the \*\*.OE instructions which determine when the outputs may be switched through. Like the file for IC<sub>2</sub>, this programming instruction is compatible with the software supplied with the *Elektor Electronics* GAL programmer.



950069 - 13

486s and up) have a BIOS which displays this information as the system starts up. If you find that all three addresses are in use already, one of these ports will have to be disabled. If you don't, bus conflicts are sure to occur if you insert the dongle safe card.

If you have a computer with an older BIOS, the printer port address information may be obtained with the aid of the DOS command DEBUG. **Fig. 2** shows a screendump which illustrates how the required addresses may be read from the system's internal software.

Once a port address is selected, set the card address correspondingly with the aid of the DIP switches. Insert the card into the PC, and switch on the PC. The basic check on the dongle safe is to see if the BIOS (or DEBUG) reports a card present at the selected address. In principle, this test is sufficient. For an extensive test of all available registers, however, use the BASIC program given in **Fig. 6**. Look carefully at lines 40, 50 and the sub-routine starting at line 1000. These lines have been added to freeze the contents of the registers, and return them to their original state when the program is ended. This is necessary because the BIOS uses shadow registers to keep track of the LPT registers. Because BASIC bypasses the BIOS to

#### Programming file for GAL IC2

Address decoder for printer port (IC2)  
National Semiconductor OPAL Junior Example

CHIP address\_printerport GAL20V8

;define pin layout for the GAL

A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AEN GND  
/H278 /H378 /H3BC /ADR2WR ADR2\_ADR1  
/ADR12RD /ADR0WR /ADR0RD /SEL /IOWR /IORD VCC

;define product terms for each printer port address

;define offsets

```
@define ADR0  "/A1*/A0"
@define ADR1  "/A1* A0"
@define ADR2  " A1*/A0"
```

;define address selection for LPT1...LPT3

```
@define LPT_A  " A9* A8*/A7* A6* A5* A4* A3*/A2*/AEN* H378*/H278*/H3BC"
@define LPT_B  " A9*/A8*/A7* A6* A5* A4* A3*/A2*/AEN* H278*/H378*/H3BC"
@define LPT_C  " A9* A8* A7*/A6* A5* A4* A3* A2*/AEN* H3BC*/H378*/H278"
@define READ   " IORD*/IOWR"
@define WRITE  "/IORD* IOWR"
```

EQUATIONS

;addresses reading address offsets 1 and 2 are combined to allow output  
;enabling for the second GAL with ADR12RD signal  
;ADR2\_ADR1 selects if the read is directed to address offset 1 or 2

```
SEL      = LPT_A + LPT_B + LPT_C
ADR0RD   = LPT_A * ADR0* READ + LPT_B * ADR0* READ + LPT_C * ADR0* READ
ADR0WR   = LPT_A * ADR0* WRITE+ LPT_B * ADR0* WRITE+ LPT_C * ADR0* WRITE
ADR12RD  = LPT_A * ADR1* READ + LPT_B * ADR1* READ + LPT_C * ADR1* READ +
          LPT_A * ADR2* READ + LPT_B * ADR2* READ + LPT_C * ADR2* READ
ADR2_ADR1 = ADR2
ADR2WR   = LPT_A * ADR2* WRITE+ LPT_B * ADR2* WRITE+ LPT_C * ADR2* WRITE
```

#### Programming file for IC6

Register decoder for printer port (IC6)  
National Semiconductor OPAL Junior Example

CHIP register\_printerport GAL22V10

;define pin layout for the GAL

ADR2\_ADR1 /ADR12RD BUSY ONLINE PE /ERROR  
SELECT /INIT /AUTO /STROBE E\_INT GND  
SWITCH /ACK IRQ D7 D6 D5 D4 D3 D2 D1 D0 VCC

EQUATIONS

;The register output is enabled when a read of offset address 1 or 2 occurs  
;thus enabling bits D0...D7

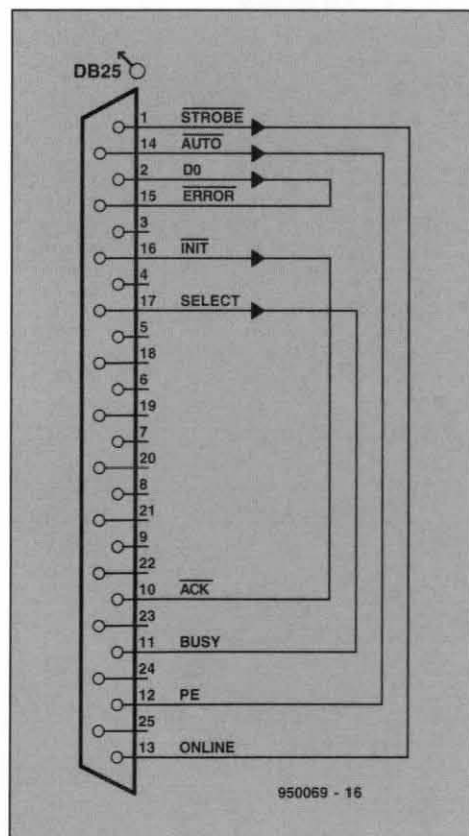
```
D0.OE = ADR12RD
D1.OE = ADR12RD
D2.OE = ADR12RD
D3.OE = ADR12RD
D4.OE = ADR12RD
D5.OE = ADR12RD
D6.OE = ADR12RD
D7.OE = ADR12RD
```

;The contents of of the bits depends on which offset address is selected  
;which is determined by the level the ADR2\_ADR1 input: zero selects offset  
;address 1 and one selects offset address 2

```
D0 = ADR12RD* /ADR2_ADR1* VCC      +ADR12RD* ADR2_ADR1* STROBE
D1 = ADR12RD* /ADR2_ADR1* VCC      +ADR12RD* ADR2_ADR1* AUTO
D2 = ADR12RD* /ADR2_ADR1* VCC      +ADR12RD* ADR2_ADR1* /INIT
D3 = ADR12RD* /ADR2_ADR1* VCC      +ADR12RD* ADR2_ADR1* /ERROR
D4 = ADR12RD* /ADR2_ADR1* ONLINE  +ADR12RD* ADR2_ADR1* E_INT
D5 = ADR12RD* /ADR2_ADR1* PE       +ADR12RD* ADR2_ADR1* SWITCH
D6 = ADR12RD* /ADR2_ADR1* /ACK     +ADR12RD* ADR2_ADR1* VCC
D7 = ADR12RD* /ADR2_ADR1* /BUSY   +ADR12RD* ADR2_ADR1* VCC
```

; The IRQ output is tristate when interrupts are not allowed  
; An interrupt can be generated if an acknowledge signal is received

```
IRQ.OE =E_INT
IRQ = E_INT*ACK
```

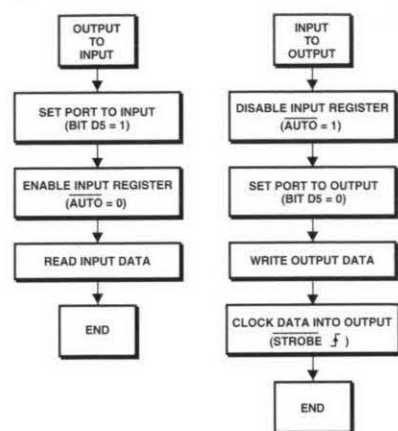
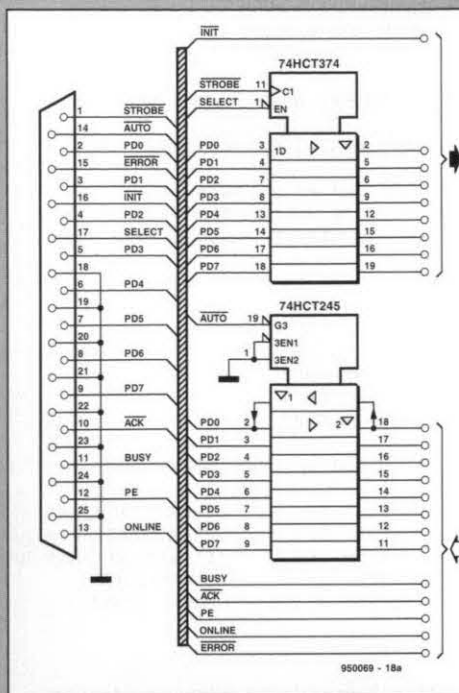


**Fig. 5.** The BASIC test program requires this simple loopback connector. The plug is compatible with the one that comes with CheckIt.



As described earlier in this article, the dongle safe card may also be used as an advanced I/O port. The circuit diagram shown here indicates how the extra hardware may be connected. An essential point to look after is the prevention of two outputs short-circuiting each other. Hence, the outputs of the 74HCT245 are switched to high impedance mode ('three-state') after a system reset (output Auto=high). Because Auto goes high during the hardware reset, fault conditions (outputs A0-A7 on outputs PD0-PD7) can not arise as a result of a software reset.

The two flow charts alongside the circuit diagram show the required software structure. The left-hand flowchart shows how to use the output as an input. First, the port is set up as an input (bit 5 of IC<sub>5</sub> is made logic high). Next, output Auto is pulled low (Q8 of IC<sub>5</sub>). Finally, information is requested via buffer IC<sub>4</sub>. The right-hand making Auto logic high (Q8 of IC<sub>5</sub>). Next data to IC<sub>3</sub>, and generate a strobe pulse this I/O function, errors can be avoided



950059-18H

change the register contents, problems may arise when a print command is given after running a test program as described here. Such problems may be prevented by returning the registers to their original values.

The BASIC program can only complete its extensive testing if a special 'loopback' test plug is connected to K<sub>1</sub>. The wiring diagram of the test plug, which connects five outputs to an equal number of inputs, is shown in **Fig. 5**. As far as the layout is concerned, this wiring corresponds to that of the test plug supplied with the widely known PC test program 'CheckIt'.

If everything functions properly, connect the dongle you wish to use to the connector on the dongle safe card, and start the relevant program. The program should start immediately. If desired, create the extra printer port by attaching the flatcable. Finally, close the PC case. (950069)

ELEKTOR ELECTRONICS SEPTEMBER 1995