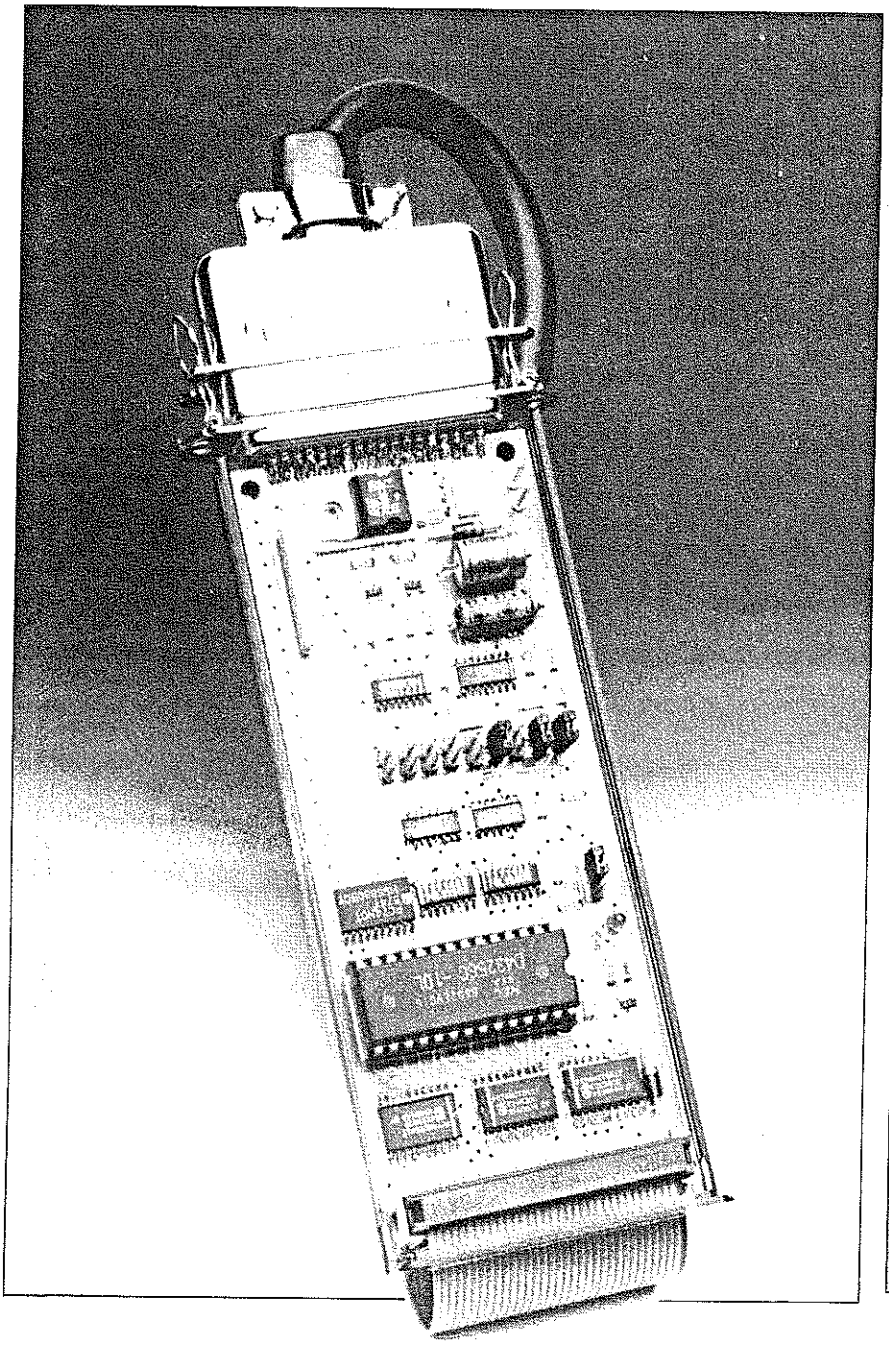
EPROM SIMULATOR

B.C. Zschocke

This circuit enables an EPROM-resident memory block in a microprocessor system to be worked on in a flexible and time-efficient manner, without having to remove, erase, and reprogram the EPROM every time its contents need to be changed. Ideal for the debugging stages of almost any circuit that uses an EPROM, this low-cost simulator works in conjunction with many types of personal computers. Special software for controlling the EPROM simulator is not required in most cases because the unit acts like a Centronics compatible printer.



With a development system far out of their financial reach, many microprocessor enthusiasts are forced to juggle with a number of EPROMs that contain debugged and tested parts of a larger program under development. The problems encountered during these and later programming stages are well-known: lost file documentation, incorrect address relocation, and missing variables during the linking stages. These and other difficulties invariably seem to accumulate to a level where the newly compiled program does not run at all while the previously written routines that make up the whole appear to work all right. Back to the subroutines and initialization routines, add one jump instruction, delete one call, re-assembly, erasing and re-programming of another EPROM. Another test, and another error

32-KBYTE EPROM SIMULATOR

- Simulation of EPROM types 2764, 27128 and 27256
- Centronics compatible
- Auto-reset for target system
- 8-, 16- or 32-bit configurations
- Data downloading uses resident printer port commands on external computer (Amiga, PC-XT/AT, CP/M, Atari)
- Control program available for PC-XT/AT (MSDOS) machines
  - port redirection LPT1:, LPT2: or LPT3:
  - interface test for data transfer
  - supports Tektronics, Intel-hex and Motorola file transfer standards
  - default: binary transfer
  - programmable address-offset
  - MSDOS PATH test
  - file length indication after transfer
  - 'quiet mode' to speed up file transfer
  - parameter transfer allowed via batch program

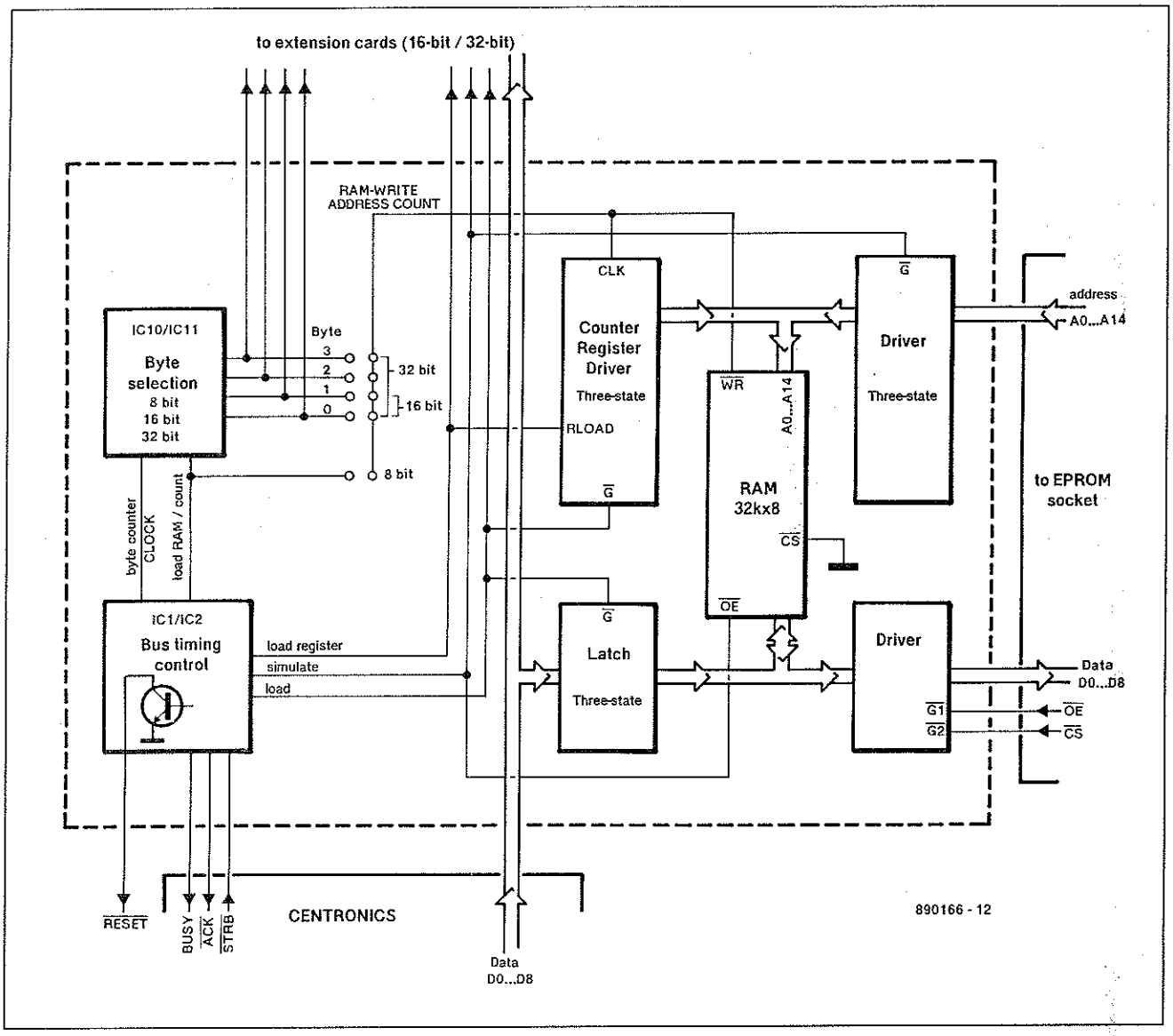


Fig. 1. Block diagram of the EPROM simulator. The 32 Kbyte RAM can be accessed from the host PC as well as from the target system.

is found.

Not a few of such programming sessions take hours of painstaking work that can be alleviated by this EPROM simulator. The unit is used in conjunction with an external PC to write, arrange, and then download experimental software into the target system until this runs as required. The EPROM eraser and programmer are not called upon until the system has been debugged completely. At this stage, the working EPROM code is available as a binary file on the external computer.

RAM instead of EPROM

The EPROM simulator essentially replaces an EPROM (or a ROM) by a random-access memory (RAM) which is read by the target system and written to by an external computer. The EPROM data may be supplied by an assembler or compiler running on the external computer. In most cases, such programs are capable of writing a binary object code file to the Intel-hex, Tektronics, or Motorola standard. The object code file is usually sent to an intelligent EPROM programmer which uses a special program that allows binary data in either one or more of the above file standards to be read via a serial port, recognized and blown into an EPROM.

The EPROM simulator described here does not require a special control program. Rather, it is designed to make use of standard system commands and utilities available to control the Centronics (8-bit parallel printer) port of the external PC. The advantages of this approach are mainly fast data transfer to the simulator, a relatively simple interface circuit in the EPROM simulator, and ease of data control via familiar programs and resident commands on the PC.

Circuit description

The EPROM simulator is capable of replacing EPROMs of 8 Kbyte (2764) to 32 Kbyte (27256). A maximum of four EPROM simulators may be connected to work on software for a 32-bit microprocessor system.

EPROM data may be supplied by any computer having a Centronics compatible 8-bit printer port. The dataflow is controlled by the STROBE pulses, which signal to the EPROM simulator that a dataword is stable and valid. As indicated in the block diagram of Fig. 1, the pulses on the STROBE line serve to clock and enable three-state counters IC3 and IC4. The outputs of these counters address a 32 Kbyte RAM. Data received from the computer is stored direct in the RAM at the address location selected by the counters. After loading the last byte, the counters are switched to the high-impedance mode. At this stage, the RAM contents may be read by the target system, which takes over the addressing. The address inputs and the data outputs of the RAM are buffered.

The circuit diagram of the basic version of the EPROM simulator is given in Fig. 2. The control circuits around the

dual-ported RAM, IC9, consist of three blocks.

Timing controller IC1-IC2 ensures the timing of the internal signals as well as those on the Centronics port. It also supplies a $\overline{\text{RESET}}$ signal for the target system. Only one timing controller is required, irrespective of whether the simulator is used in an 8-, 16- or 32-bit configuration.

Byte selector IC10-IC11, if used, ensures the correct distribution of 8-bit datawords received on the Centronics port to the 16- and 32-bit extensions.

RAM address and load address counter IC3-IC4, together with data latch IC5 and drivers IC6, IC7 and IC8, arranges the addressing of the RAM, as well as read and write operations, either by the external computer (download mode) or the target system (simulate mode).

The RAM chip used, a 43256-10, is a static type with a memory capacity of 32 Kbyte, allowing EPROMs up to and including the 256-kbit Type 27256 to be simulated in the target system. If smaller EPROMs are used, the non-used address lines of the EPROM simulator must be made low (A14 for the 27128, and A13-A14 for the 2764).

Timing, pulse levels and the Centronics interface

To ensure correct operation of peripherals loading on the negative as well as on the positive edge of the $\overline{\text{STROBE}}$ pulse, data on a Centronics compatible computer port must be stable before and after the $\overline{\text{STROBE}}$ line goes low. Most 8-bit parallel printers load data on the negative pulse edge, as specified in the Centronics standard. The EPROM simulator uses both the positive and the negative pulse edge.

Components R8-C3 reset the circuit at power-on. Bistables FF1 and FF2 are set, and monostables MMV1 and MMV2 are reset. Bistable FF2 resets all counters and switches the circuit to EPROM simulation mode.

The timing diagram of Fig. 3 refers to operation of the 8-bit version of the EPROM simulator. The negative edge of $\overline{\text{STROBE}}$ triggers MMV1 and resets FF1 and FF2. The latter bistable switches the circuit to the load mode, and actuates the $\overline{\text{RESET}}$ line. The other bistable, FF1, ensures that the BUSY line on the Centronics interface is actuated. The pulse transition supplied by FF1 causes the counter value to be loaded into the counter register, and the dataword on the Centronics port to be latched. The duration of the $\overline{\text{STROBE}}$ pulse allows the digital levels that form the dataword and the address for the RAM to settle. The positive edge of the $\overline{\text{STROBE}}$ signal triggers MMV1, whose monostable period is used to actuate the $\overline{\text{WRITE}}$ signal for the RAM and the $\overline{\text{ACK}}$ (acknowledge) handshake signal on the Centronics interface. After the monostable period has lapsed, FF1 is set so that

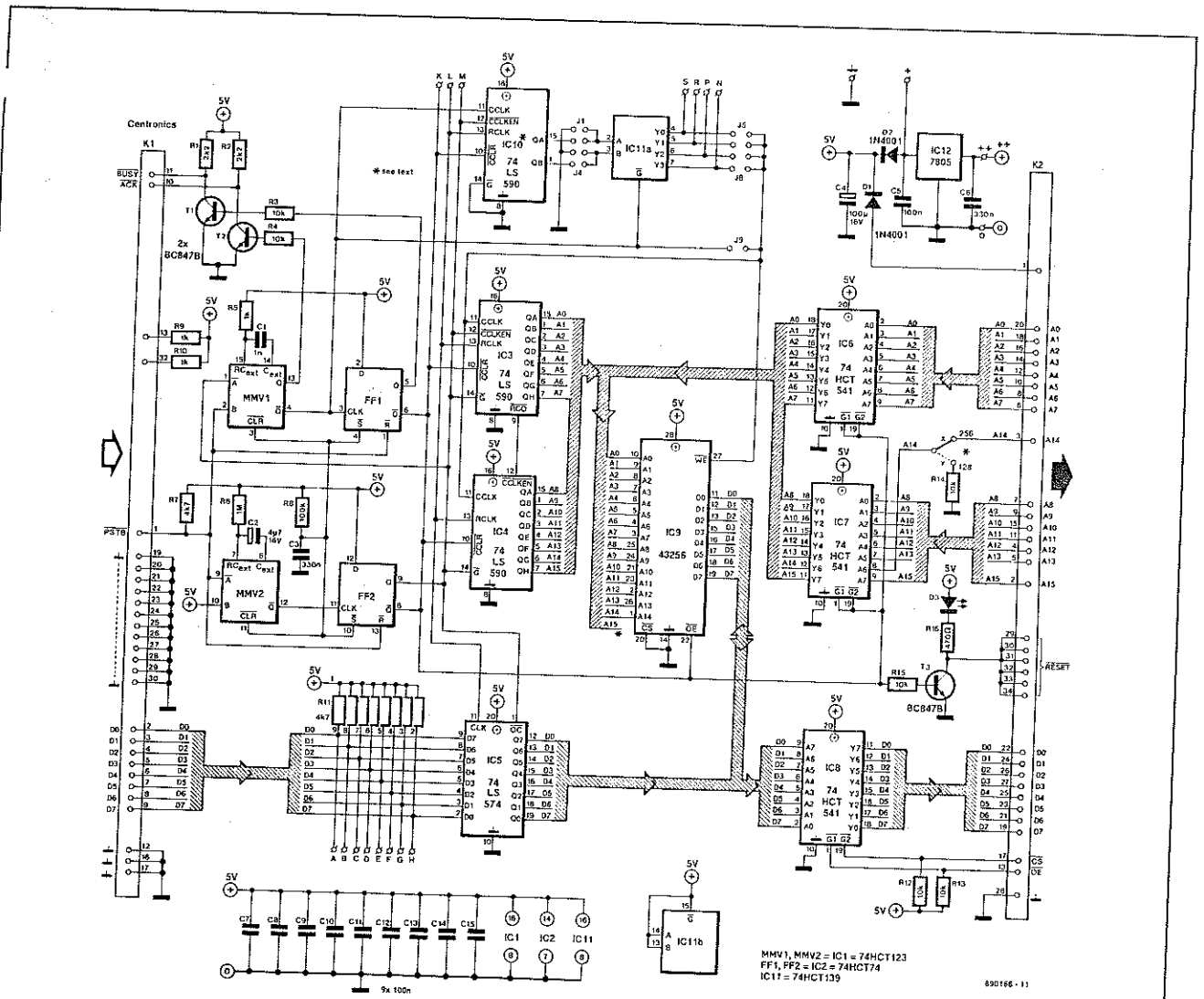


Fig. 2. Circuit diagram of the basic (8-bit) version of the EPROM simulator. The circuit may be extended to work on 16- and 32-bit systems.

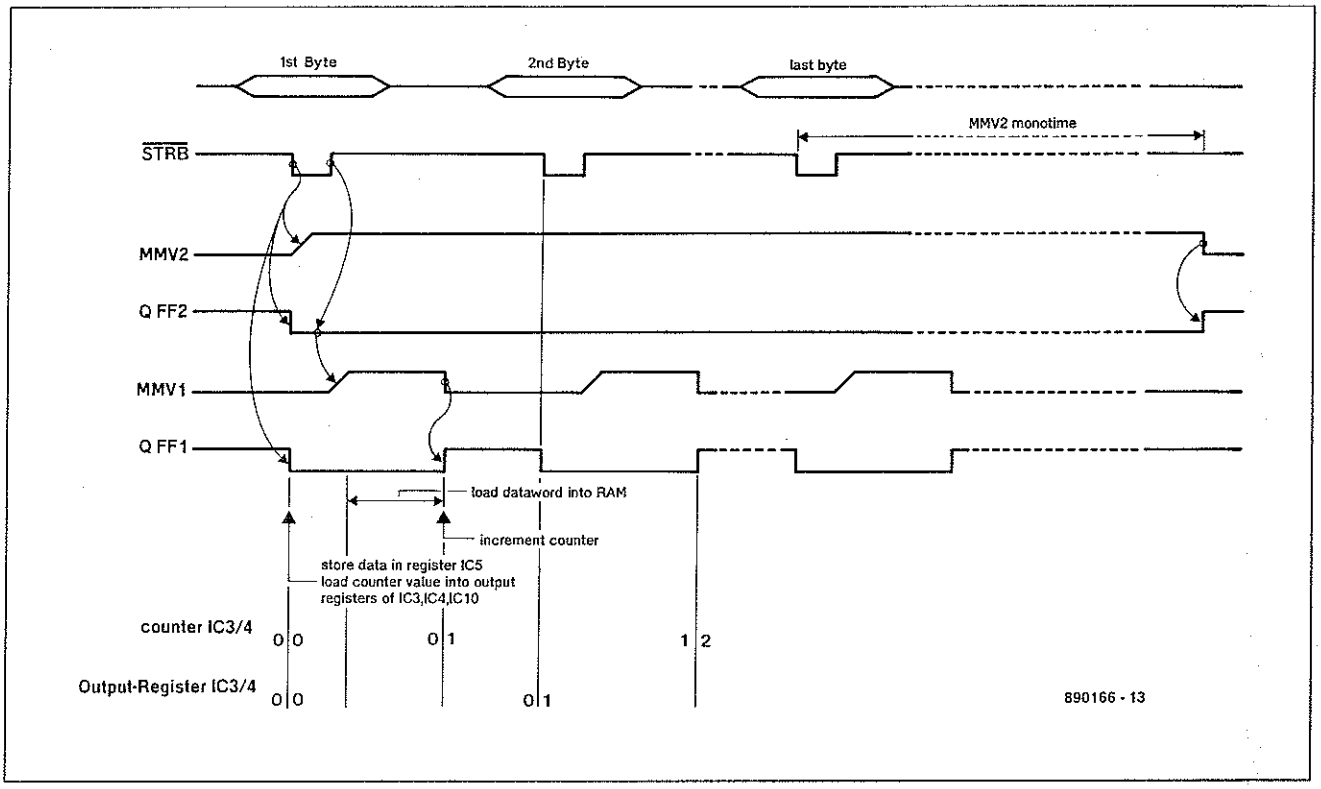


Fig. 3. Timing diagrams to illustrate the operation of the 8-bit version of the EPROM simulator.

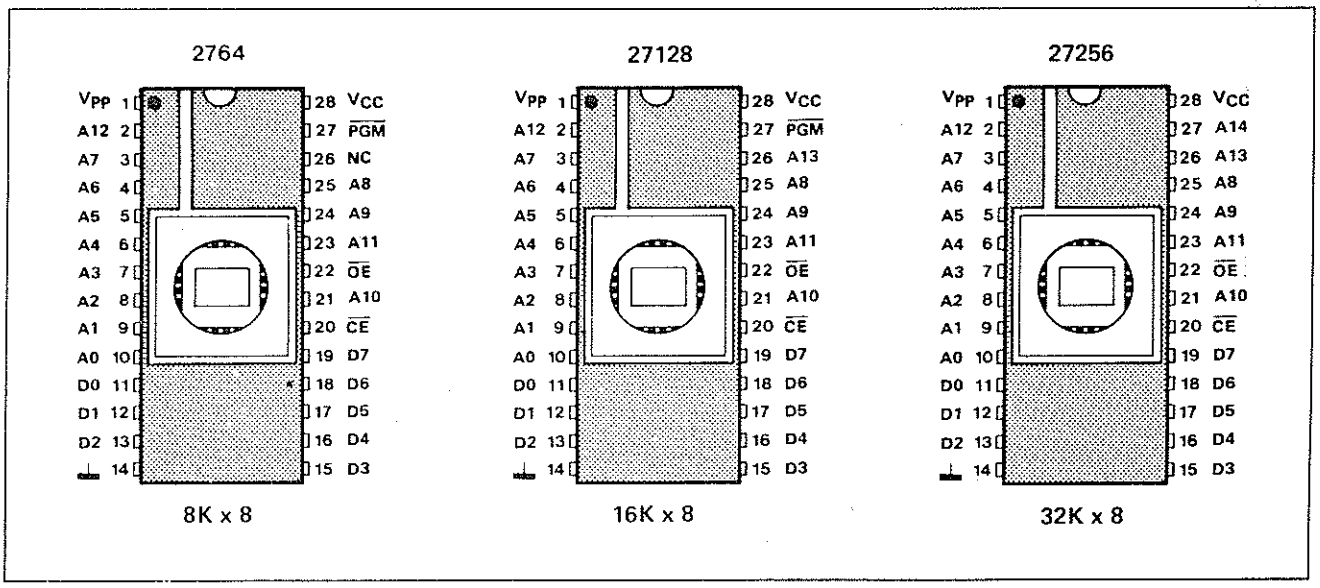
BUSY is made low. At the same time, the counter is advanced by one address location. A this stage, the first byte has been loaded into RAM, and the circuit is ready to accept the next dataword.

Monostable MMV_2 is triggered by a new dataword if this is applied within its monotime. The loading sequence is as described for the first STROBE pulse. If no dataword is received within the monotime of MMV_2, the circuit switches to simulation mode, resets the counters, and ends the reset condition of the target system.

In the 16 and 32-bit versions of the simulator, IC_{10} and IC_{11} arrange the distribution of the counter values (addresses) and the internal WRITE signal for the RAM. Depending on the jumper configuration, either the first, second, third or fourth byte is loaded, and the number of clock pulses to the counters is reduced accordingly.

Although the outputs of MMV_2 and FF_2 appear to behave identically as far as their timing is concerned, there is good reason to use the additional bistable. The timing diagram of Fig. 3 shows that the time between triggering and actuation of the monostable, MMV_2, is not short enough, particularly at relatively long monotimes. Hence, bistable FF_2 prevents a possible timing error because it is actuated by the trigger signal of MMV_2, and de-actuated by the negative edge of the monostable signal.

Transistor T_3 turns on LED D_3 when the computer loads data into the EPROM simulator. The 'active low' collector volt-



For easy reference: pinning of the EPROM types that can be handled by the EPROM simulator.

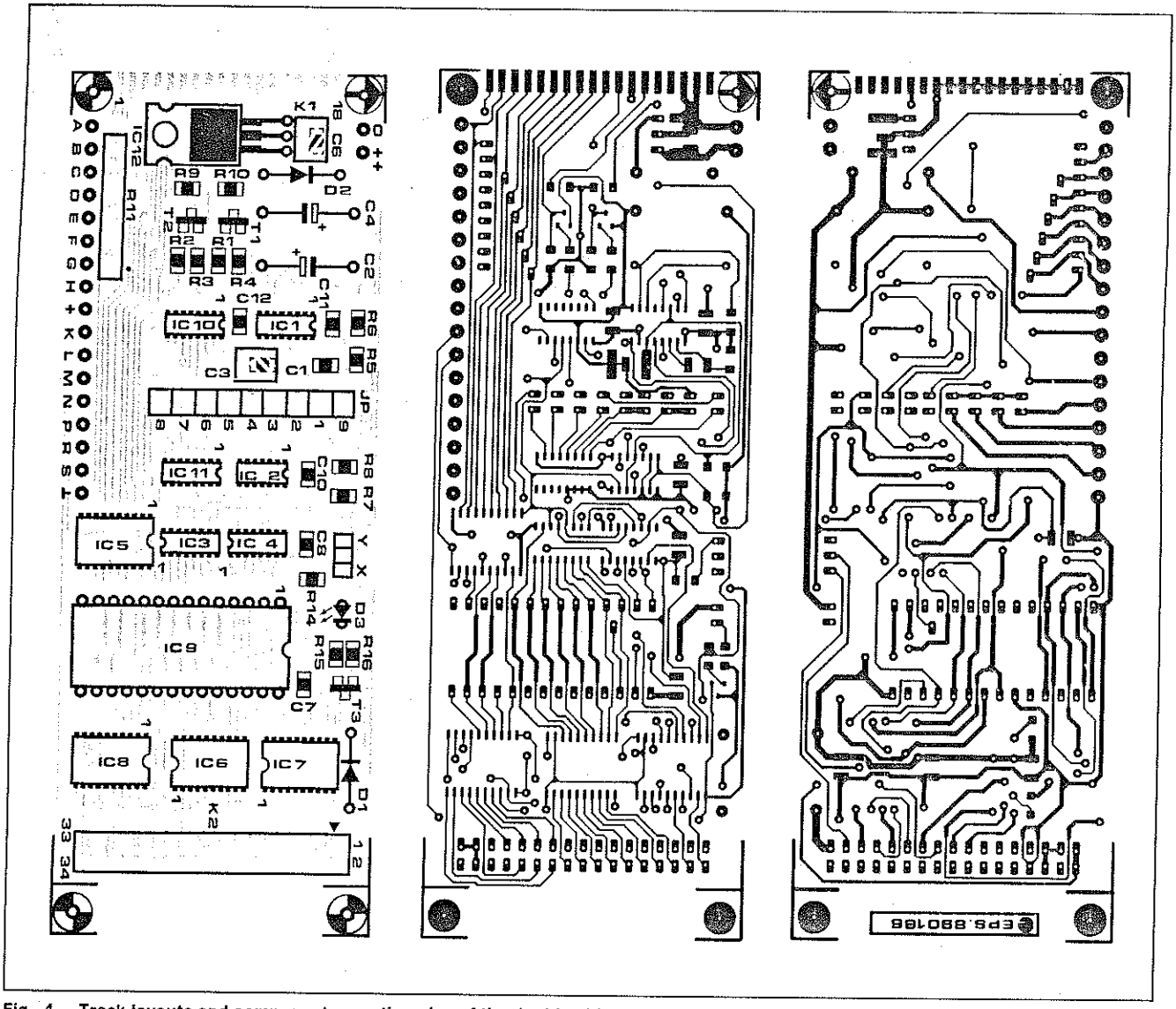


Fig. 4. Track layouts and component mounting plan of the double-sided, through-plated printed-circuit board for the EPROM simulator.

age of T_3 is taken to connector K_2 to keep the target system reset (or disabled) while data is being downloaded into the simulator. On completion of this process, the target system is automatically started, and runs the new software in the simulator RAM.

16- and 32-bit systems

Circuits IC_{10} and IC_{11} must be fitted on the main board if a 16- or 32-bit system EPROM is simulated. The chips enable the respective RAMs to be addressed sequentially. The 16- or 32-bit extension of the EPROM simulator is composed of up to three additional cards, which hold only the components stated in the parts list for the extension card. All cards are interconnected via terminals PC1 – PC17. Jumpers JP_1 – JP_4 define the number of cards (or RAM areas), while jumpers JP_5 – JP_8 define the order of the RAMs during the loading procedure. Table 1 lists the various jumper configurations.

Power supply

Diodes D_1 and D_2 allow the EPROM simulator to be powered either by the target system or by the on-board regulator, which takes its unregulated input voltage from a mains adapter. In most cases, the supply capacity of the target system will determine whether or not an external power supply is needed. The basic version of the simulator (without IC_{10} and IC_{11}) draws about 80 mA, mostly on account of the two 74LS590s. The current consumption of more than one simulator (16-bit or 32-bit systems) may be estimated by multiplying 40 mA by the number of 74LS590s used. Multi-simulator systems require only a single voltage regulator, because the supply voltage is bused via connector pins PC16 and PC17.

8/16/32-kByte selection

Wire link X-Y selects between simulation of a 27128 and a 27256 EPROM. In most microprocessor systems, pin 27 of a 27128 (PGM input; see Fig. 5) is made logic high. The resultant logic high level at pin 3 of K_2 must, however, not be passed to address line A14 of the the simulator RAM, requiring jumper Y to be installed. Similarly, a 2764 requires both A14 and A13 of the RAM to be held low. If, for any reason, a logic high level exists at pin 26 of the 2764 in the target system, the RAM must first be loaded with an 8 Kbyte block. Contrary to what is indicated in many application

Jumper	8-bit 1×32k	16-bit 2×32k	32-bit 4×32k	RAM range	
JP1	x	–	–	JP5	1
JP2	–	x	x	JP6	2
JP3	x	x	–	JP7	3
JP4	–	–	x	JP8	4
	for 8-bit version only:			JP9	1

Table 1. Jumper configurations.

Parts list

MAIN BOARD

non-SMA components

Resistors:
R_{11} = 8-way SIL resistor network 4k7

Capacitors:
C_2 = 4µ7; 16 V; axial
C_4 = 100µ; 16 V; axial

Semiconductors:
D_1;D_2 = 1N4001
D_3 = LED (3 mm)
IC_9 = 43256-10
IC_{12} = 7805

Miscellaneous:
K_1 = 36-way Centronics socket with straight pins.
K_2 = 34-way pin header (double row).
K_3 = 34-way IDC socket.
K_4 = 28-way IDC DIP header.
PCB Type 890166 (see Readers Services page).
Approx. 50 cm 34-way flat-ribbon cable.
Enclosure: Heddic Type 222.

SMA components:

Resistors:
R_1;R_2 = 2k2
R_3;R_4;R_{12} – R_{15} = 10k
R_5;R_9;R_{10} = 1k0
R_6 = 1M0
R_7 = 4k7
R_8 = 100k
R_{16} = 470Ω

Capacitors:
C_1 = 1n0
C_3 = 330n
C_5;C_7 – C_{15} = 100n
C_6 = 330n

Semiconductors:
T_1;T_2;T_3 = BC847B
IC_1 = 74HCT123
IC_2 = 74HCT74
IC_3;IC_4 = 74LS590
IC_5 = 74HCT574
IC_6;IC_7;IC_8 = 74HCT541
IC_{10} = 74LS590 (see text)
IC_{11} = 74HCT139 (see text)

circuits of the 2764, pin 26 must never be left unconnected. If necessary, fit a 10 kΩ resistor to ground to ensure a permanent logic low level.

Hardware

The population of the double-sided, through-plated printed-circuit board for the EPROM simulator (Fig. 4) depends on whether an 8-, 16- or 32-bit system is to be debugged. For an 8-bit system, the card functions as the main board. For 16-bit and 32-bit applications, however, it functions as an extension card. Circuits IC_{10}

Parts list

EXTENSION BOARD

non-SMA components

Capacitors:
C_4 = 100µ; 16 V; axial

Semiconductors:
IC_9 = 43256
D_1;D_2 = 1N4001

Miscellaneous:
K_2 = 34-way pin header (double row).
K_3 = 34-way IDC socket.
K_4 = 28-way IDC DIP header.
Approx. 50 cm 34-way flat-ribbon cable.
PCB Type 890166 (see Readers Services page).

SMA components

Resistors:
R_{12};R_{13};R_{14} = 10k

Capacitors:
C_7;C_8;C_9;C_{13};C_{14};C_{15} = 100n

Semiconductors:
IC_3;IC_4 = 74LS590
IC_5 = 74HCT574
IC_6;IC_7;IC_8 = 74HCT541

and IC_{11} may be omitted only if future upgrades from 8-bit to 16-bit or 32-bit are not foreseen.

For an 8-bit application, only the main board is required. A 16-bit application requires one main board with IC_{10} and IC_{11} fitted, and one extension board. A 32-bit application requires one main board with IC_{10} and IC_{11} fitted, and three extension boards.

Fit Centronics connector K_1 direct on the board by pushing its straight pins over the relevant copper islands. The orientation of the connector is indicated by the '1' on the component overlay. Solder the connector pins to the islands, then check for short-circuits between pins. Place the connector and the PCB on the bottom half of the Heddic enclosure, and determine the size of the clearance in the top half to allow for the Centronics connector.

A home-made cable is required to connect the EPROM simulator to the EPROM socket in the target system. The units are interconnected via pin header K_2, a mating IDC (insulation displacement) connector, a length of 34-way flat ribbon cable, and a 28-way IDC DIP header for mating with the EPROM socket in the target system. The six wires connected to pins 28 – 34 of connector K_2 carry the RESET signal for the target system. These wires may be joined at the DIP header side of the flatcable and connected to a flying lead with a small crocodile clip. The length of the 34-way flatcable must not exceed 50 cm.

The EPROM simulator is built mainly with surface-mount assembly (SMA) components, which must be handled and soldered with great care and precision. Pay attention to the orientation of each and every SMA IC before fitting it!

The use of the transparent, smoke-coloured Heddic Type 222 enclosure stated in the Parts List requires the PCB to be shortened by cutting off the section with the two corner holes near K_2. These holes are only required if boards are stacked in 16- or 32-bit applications, for which a higher enclosure is required.

Cut a slot in the short side of the top half of the enclosure to enable the 34-way flatcable to pass. Insert a few mica washers or a ceramic insulation plate between the metal tab of the 7805 and the board to prevent a short-circuit with the tracks running underneath. Cut the jumper blocks from a larger double-row pin header. Mount LED D_3 direct on to the board.

Software

A system utility for the parallel printer port is used to download data into the EPROM simulator. The only requirement for this utility is a capability to send a file as 8-bit binary data to the Centronics port. Some examples of system utilities are listed below.

PC/MSDOS

COPY <filename> LPT1: /B

The /B switch causes the file to be sent in binary form (see your DOS manual).

For more advanced applications, a special control program, EPROMSIM, is available through the Readers Services. The program disk may be ordered as item ESS 129 (360 kB, 5¼-inch, MSDOS format). The main features of this versatile program are listed in the technical specifications box on the first page of this article.

CP/M systems

PIP LST:= <filename> [O]

The [O] switch causes the file to be sent in binary form.

Amiga

COPY <filename> PAR:

Be sure to use PAR:, not PRT:

Atari TOS

On Atari ST computers, click twice on the filename in the desktop menu, then send it to the printer port. Note that the operating system, TOS, closes the file with a CR-LF (carriage return & line feed) sequence, so that the last two bytes of a 32 KByte EPROM can not be used. A simple printer program that does allow the last two bytes to be used should not be too difficult to write in Pascal, C or BASIC.