SCRG

SOUTHERN CALIFORNIA RESEARCH GROUP

P.O. BOX 593
MOORPARK, CA 93021
(805) 529-2082

NEW PRODUCTS
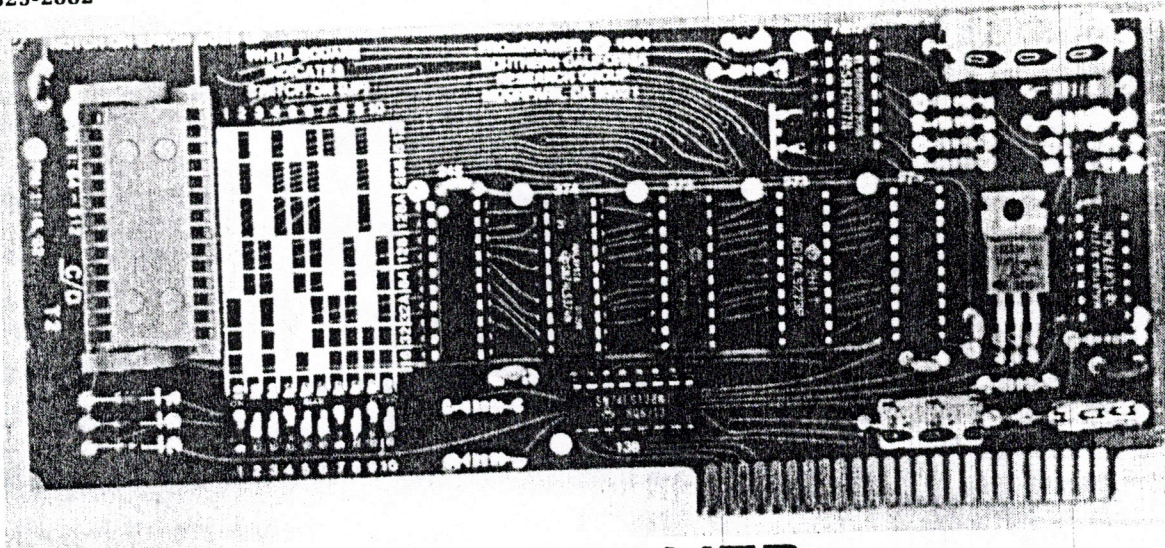
PROmGRAMER

The PROmGRAMER is an inexpensive EPROM (Erasable Programmable Read Only Memory) programmer for the APPLE][+ and //e computers.

FEATURES:

Programs all 5 volt 27XXX EPROMS, from the 2716 to the 27512.

Choice of standard or fast programming algorithms.

Disk-based software - easily personalized.

Source code included (S-C Assembler).

Complete! No personality modules needed.

On-board programming voltage generator.

Zero-insertion-force (ZIF) socket

Slot independent.

The PROmGRAMER, as all products from Southern California Research Group, comes with a six month warranty, and a 10 day trial period. The high-quality circuit board includes gold-plated connectors.

$149.50

DUAL-KEYBOARD ADAPTER

The DUAL-KEYBOARD ADAPTER (for the APPLE][and][+ only) allows the use of either of two keyboards that are plugged into the computer. There are no switches, the user simply starts typing on either the standard keyboard, or on the external one. The unit plugs into the keyboard socket of the computer, and both keyboards are then plugged into the adapter. **$39.50** Available MAY '85

These EPROMs are programmed one byte at a time, i.e. eight bits per programming pulse. The charge is as permanent as we wish to make it (Well, not absolutely permanent. Most EPROM manufactures will guarantee the charge retention for only(!) ten years). We can change this insulator back into a conductor, however, by shining a short-wavelength ultra-violet lamp through the little window on the EPROM. This causes the insulator to allow the charges to "leak" rapidly, thus erasing the EPROM. This usually takes about 20 minutes. Now, unless you know what you are doing, please read the rest of these instructions before you try programming your first EPROM.

SETTING THE SWITCHES

On the PROmGRAMER is a set of ten switches. All switches should be in the down position (OFF) except for those shown by the white square above the switch. On the right edge of the white squares are some numbers representing the different types of EPROMs. This unit will program the single voltage supply type of EPROM from the 2708 to the 27512. The number 27 does not appear on the guide. For instance, to read or program a 27128, find the number 128 to the right of the guide. You will notice that the squares above switch numbers 1, 4, 7, and 9 are highlighted. These switches should be turned on (UP). NOTE: It is especially important that switches 7 and 8 be in the correct position for the type of EPROM used. These switches control the level of programming voltage, and failure to have them in the correct position may result in catastrophic failure of the EPROM.

DIFFERENT EPROMS

There are several EPROM types other than those listed on the guide. It is important to remember that the PROmGRAMER can only program the single voltage-type EPROMS. Some of the earlier EPROM types need three voltages to operate. These types should be considered obsolete, and the PROmGRAMER will not program them. In particular, TI (Texas Instruments) had an EPROM called a 2716 which uses three voltages. Their 2516 is the equivalent of everyone elses 2716. The Intel type 2716, which everyone else makes, can be programmed by the PROmGRAMER.

Due to space limitations, we did not list all EPROM types that you can use. If you have a single voltage 2708, use the settings for the 2716. If you have a 2764A, use the settings for the 2764, except switch 8 should be up, and switch 9 down.

PLUGGING IN THE EPROM

When you have the switches in the correct position for the type of EPROM you will be using, you may plug the EPROM into the ZIF (Zero Insertion Force) socket. This is the large blue socket to the left of the guide. The lever above (and below) the socket will securely clamp the leads of the EPROM when it is moved to the right. First, make sure that the lever is to the left. This will allow you to easily insert the EPROM into the socket. If it is a new EPROM, the leads may be bent out a bit, and may have to be bent in slightly. ANOTHER IMPORTANT NOTE: If the EPROM is inserted into the socket incorrectly, you may cause (you guessed it!) catastrophic failure of the EPROM. Pin #1 of the EPROM is on the same edge as a notch in the middle of one edge, as shown in figure one. Pin #1 must be toward the top of the ZIF socket. There is a note at the top of the card reminding you of this, and two dots on the left side of the socket showing the location of pin #1. Note that if you have a 28 pin device (2764, 27128, 27256, or 27512), pin #1 is on the upper left side of the ZIF socket. If you have a 24 pin device (2708, 2716, or 2732), pin #1 is the third pin down from the top on the left of the ZIF socket. Thus, if you have a 24 pin device in the socket, there will be two holes visible above the EPROM on each side of the ZIF socket.

INSTALLATION

With power off, plug the PROmGRAMER into any slot (except zero) of your APPLE][,][+, or //e. Turn on the power, boot the disk, and the "HELLO" program on the disk will load the driver program (B.PROMGRAMER). NOTE: If you have not yet done so, make a copy of the disk, and put the original in a safe place. You may use any standard copy program.

PROGRAMMING THE EPROM

Programming the EPROM basically consists of taking an area of memory, and transferring it to the EPROM. The PROmGRAMER takes care of the mechanics, and it is up to you to make sure to get the correct data into the EPROM. We will show this by example, but first, a word about the driver program (the one that makes the PROmGRAMER work).

The program loads into memory at $803. (The dollar sign signifies that the number following is in hexadecimal. The length of the program is $753, bringing you to $F55. The rest of memory (up to $9600 where DOS begins) is available to you as the working array. A memory map is shown in figure 2.

The program will ask you which slot the PROmGRAMER is plugged into. You will then be asked to enter the type of EPROM. (Type in a number between 1 and 8). You will then be presented with a menu of choices. Anthropomorphically speaking, you have to tell the computer which area of memory you are using (called the WORKING ARRAY). You do this by specifing the starting address in memory (symbolized by SSSS), and the ending address (EEEE). You also have to tell the computer where you want to start programming the EPROM. You do this by giving a relative address (PPPP), where the start of the EPROM is 0000. All these addresses are in hexadecimal.

To program an EPROM, you should have a binary file ready on disk with the necessary information. The length of the data you can fit in depends on the specific EPROM you are using. TABLE ONE, on the back cover, shows EPROM types, the length of the file that can fit in, and programming times.

For our first example, let's try programming a 27128 with a unique pattern that tests every possible combination. Looking at the above table, we can see that the maximum length of the file is $4000 bytes. On the disk that we sent you is a program called "TEST PATTERN". This just consists of a sequence of numbers from 0 to $FF, repeating as necessary to fill the $4000 byte range. First, we set the switches, plug in the EPROM, and plug into a

disk, and type the slot number.

Since we are using a 27128, type the number 5, which is the type number for a 27128.

DOS commands can be typed within the PROmGRAMER driving program, so we type;

BLOAD TEST PATTERN

This pattern will load starting at location $1000.

We will now "burn" the EPROM with the memory range from $1000 to $4FFF, starting at EPROM location 0, so we type:

B 1000 4FFF 0000

Note that it is necessary to put in the blank spaces, and four digits per number. (Actually, as we shall explain later, it is only necessary to type the letter "B", then return, as these parameters are the default numbers for the 27128).

Now, make yourself a cup of tea, and find something to pass the time. Each byte takes 50 milliseconds to program (0.05 seconds), so we have a wait of about .05 seconds times 16384 = 819.2 seconds, or 13 minutes 39.2 seconds. You can follow the progress of the programming, as the current EPROM address is printed on the lower right side of the screen.

After programming is complete, the PROmGRAMER will check if the job was done correctly. It will compare each byte of memory with the corresponding location in the EPROM, and verify that both are the same. If so, there will be a four note audio signal, and a visual message. If not, the computer will give you a six note audio signal, and start printing the addresses of the offending byte(s). You may abort the listing at any time by hitting the ESC or RETURN key.

While the PROmGRAMER is programming, there is a high voltage on one of the pins. As damage may occur if the wrong voltage is removed first, DO NOT insert or remove EPROMs while the PROmGRAMER is in the programming state. Since the programming voltage is removed in the read or idle state, there should be no problem inserting or removing EPROMs, even with power on. However, you MUST turn off power before removing the card, or dire consequences will follow.

READING EPROMS

Turn off the computer to assure yourself that the test pattern in memory has disappeared. Turn the computer back on, and initialize the program. Type 5 (the EPROM type), and;

L

This command will load memory locations $1000 through $4FFF with the information in the EPROM starting at EPROM location zero. To display the memory, type D. You should see the test pattern appear. (You may pause the listing by pressing any key (except

ESC or RETURN), and continue it with another keypress). The information you are looking at was stored inside the EPROM. At this point, you can be sure that the PROmGRAMER and your technique are working correctly.

NORMAL OR FAST BURN?

You have a choice of two programming algorithms (an algorithm is the program that the computer uses.) The first, which we call "BURN", gives a 50 ms pulse for every address. This is the pulse length recommended by most EPROM manufacturers. Most EPROMs, however, will program sucessfully with a much shorter pulse. The "FAST BURN" algorithm will give a 2 ms. pulse, and read the byte back. If it does not read back correctly, it will get another 2 ms. pulse, as so forth until the byte reads back correctly. The PROmGRAMER will then give an "overprogramming" pulse lasting 4 ms. As most bytes will program with the first pulse, programming time is drastically reduced. For most EPROMs, however, this method is not guarenteed by the manufacturer (although off-the-record, most of the engineers say this method works fine.) It's up to you which method to use. Our recommendation is to use the FAST algorithm until you get the "bugs" out of the program. Then use the BURN algorithm to program your final EPROM. Incidentally, you will find that in almost all cases, the programming time will be shorter than that listed in the table, because both algorithms will skip any byte of $FF, as that is the same as an unprogrammed EPROM byte.

DEFAULTS

(Quick definition; a DEFAULT is a parameter chosen by the program, but can be overridden by the user.) Most of us will program EPROMs at the same address. Since it is a pain to have to enter a string of numbers each time we want to program an EPROM, the software has built-in defaults. These numbers are chosen to give the most utility to the most people. But we all know what happens when a manufacturer decides what's good for you. They are usually wrong. There will be times that you will not want to use our standard defaults. That's fine with us. The default parameters assume that you want to program the entire EPROM, from a file loaded at $1000. If you press "B", and return, the default parameters will be printed, just as if you had typed them in. If you wish to use different parameters, just type them in after the command letter. Note that the default parameter for the 27512 is for the lower half of the EPROM. To program the upper half, you will have to type in the parameters.

If you will be exclusively using a particular slot and/or EPROM, you may save a copy of the program with these as defaults, so you will not have to select the slot or EPROM type each time you start the program. The byte at $806 usually holds $00, which signifies that a choice of slot has to be made. You may put the slot number times 16 in this location. The EPROM type is in location $807, where $00 signifies that no choice has been made, or a value from $B1 to $B8. You make save these parameters by running the program, making your slot and EPROM choices, then type "BSAVE (name), A$803, L$753. BRUNning this program will bypass the choices for slot and EPROM. If you wish to change the EPROM type, you may press "Z", which will restart the program.

OTHER NOTES OF INTEREST

If you EXIT the program to APPLESOFT, you may restart it by typing "&", and RETURN. If you go to the MONITOR, you may restart by typing 803G, or by typing CONTROL Y, then RETURN.

At any time, you may abort listings or programing by typing ESC or RETURN. You may pause listings or programing by pressing any other key, and pressing a key again to continue.

You may ERASE CHECK a portion of an EPROM. For instance, typing E 400 7FF will check the indicated bytes.

You may issue a PR#1 command, if you want to turn on a printer.

The source code for the driving program is included on the disk (S.PROMGRAMER). It is written for the S-C Assembler (available from S-C Software, Post Office 280200, Dallas, TX 75228. Telephone (214) 324-2050). If you wish, you may experiment with, and modify, the program.

While in the FAST BURN Mode, a defective EPROM will cause the program to halt with a "CANNOT BURN" message.

Although the source code and driving programs are copyrighted, the disk is not copy-protected. You are urged to make back-up copies, for your own use, of course.

While DOS commands can be used from within the program, you will have to remember that RUNning a program will destroy the driving program, as will BLOADing or BRUNning a program between $803 and $F55.

To maintain compatibility with earlier versions of this program, V (for verify) will do the same as C, K = E, and R = L

PROGRAMMING THE 27256

For some reason, most people who get into programming EPROMs start their files at $2000. That is a nice round number, it won't interfere with the PROmGRAMER driving program, as it does seem to be a standard, so why not continue to use it? Well, we'll tell you why. If you start at $2000, the file for a 27256 will end at $9FFF. We all remember (don't we?) that DOS resides starting at $9600, so we have a conflict. In this type of conflict, no one wins. We must avoid overwriting DOS. If we start the file at $1000, it will end at $8FFF neatly avoiding problems, so let's set a standard: All files will start at $1000 (naturally, you may start at another location if it is necessary for a particular job).

One other thing; under DOS 3.3 the maximum file length you can save to disk is $7FFF bytes, one byte less than that needed for a 27256 (amazing how often things like that seem to happen). We can do one of two things, save one byte less than we need, or change DOS. If you type POKE -22172,255, DOS will now accept files up to 64 Kbytes long. We suggest that you do not INIT any disks after making this change, because your DOS is no longer standard (until you re-boot).

PROGRAMMING THE 27512

The 27512 can hold 64K of memory, which is the entire memory of the standard APPLE ll. Obviously, most of that is something we would not want to program, so we must do it in two steps. Assuming the files are saved as "FIRST HALF" and "SECOND HALF", we would proceed somewhat as follows:

```
[initialize program]
BLOAD FIRST HALF, A$1000
8 [eprom type]
B 1000 8FFF 0000
   [after 28 minutes]
BLOAD SECOND HALF, A$1000
8
B 1000 8FFF 8000
```

CREDITS

The hardware design of the PROmGRAMER is by BOB BRICE. The Software was written by BOB SANDER-CEDERLOF. DOS 3.3 is a copyrighted program by APPLE COMPUTER. PROmGRAMER is a trademark of SOUTHERN CALIFORNIA RESEARCH GROUP.

A MESSAGE TO quikLoader OWNERS

We're sure that you would like the convenience of having the driver program on the quikLoader, available for instant loading without the bother of having to find the correct disk. Would we let you down? On the program diskette is a file called "QUIKLOADER". It is all set up to load in location $1000, and is complete with overhead files to store in a 2716. To store it in a larger chip, bload it at the correct address (i.e. 2732 - A$1800, 2764 - A$2800, etc.)

YOU'RE ON YOUR OWN, NOW

Well, not quite, we are available to give you any necessary assistance. Call us at (805) 529-2082. While we cannot be expected to teach you to program, we will be happy to help you with any PROmGRAMER questions.

WARRANTY

The PROmGRAMER is warrented for six months for defects in parts or workmanship. If you have any problem within this time, return it postpaid to us, and we will repair or replace it. Our address is:

SOUTHERN CALIFORNIA RESEARCH GROUP
Post Office Box 593
Moorpark, CA 93021

Telephone (805) 529-2082

PROmGRAMER
INSTRUCTIONS

INTRODUCTION

If you understand EPROMS and the mechanics of programming, set the switches, boot the disk, and ignore the rest of these instructions. If you are in a rush to use the PROmGRAMER, or are not interested in the technology behind this product, you may skip the next section, and GOTO "SETTING THE SWITCHES".

WHAT IS AN EPROM

Taking our APPLE as an example, we know that some of its memory is volatile, that is, when we turn off the power, the computer "forgets" what it had in it. This type of memory is called RAM (Random Access Memory), or, more correctly, R/W (Read-Write) memory, since you can write information into it, and read it out again. However, there is some memory that is not volatile. We know that as soon as we turn on the computer, some programs are available to us, even without a disk. Examples of these programs include APPLESOFT and the system monitor. These programs are stored in a non-volatile memory called ROM (Read Only Memory). (we leave it as an exercise to the reader to figure out what a WOM (Write Only Memory) is good for).

A major problem with ROMs is that they are manufactured with the program in place, and it is impossible for the user to change the memory. The cost to make just one is in the vicinity of $3,000 (U.S.). Obviously, this is too high to be practical, unless you make many of them, since each additional one is only a few dollars.

Along came the PROM (Programmable Read Only Memory). This device is manufactured with many tiny fuses in it. Where a fuse exists, a logical ONE would be read. To change it to a logical ZERO, a high pulse of current is sent through the fuse, melting it. The obvious problem with this is that you can program the device once. If you make a mistake, buy another one, and try again.

Which brings us to the EPROM (Erasable Read Only Memory). This device is also manufactured with each "cell" reading a logical ONE. If you want a logical ZERO, you have to program it in. A rigorous explanation is beyond the scope of these instructions, but, in short, this is done by "pumping" a charge into an insulator. This is the job of the PROmGRAMER. The wanted address and data are set up and a relatively high voltage is applied to the chip. This causes an insulator to temporarily change into a conductor. A charge of electrons is injected into the middle of the insulator, and, when the high voltage is removed, these electrons now find that they are trapped, surrounded by an almost perfect insulator. This "floating" charge affects circuitry which "sees" a logical ZERO where such charge exists.

	0 - $802	SYSTEM
	$803 - $F55	DRIVER
	$F56 - $95FF	WORKING ARRAY
	$9600 - $FFFF	DOS & SYSTEM

FIGURE 2

28 PIN DEVICE

PIN 1

FIGURE 1

24 PIN DEVICE
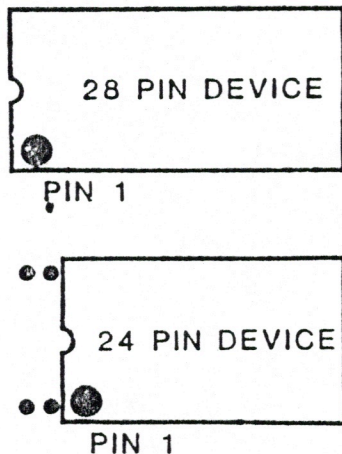
PIN 1

EPROM TYPE	LENGTH (in hex)	LENGTH (in dec.)	NORMAL PROGRAM (min. & sec.)	FAST BURN (min. & sec.)
2716	800	2048	1:42	0:13
2732	1000	4096	3:24	0:25
2764	2000	8192	6:50	0:50
27128	4000	16384	13:40	1:40
27256	8000	32768	27:18	3:20
27512	10000	65536	54:36	6:40

TABLE ONE