EMULOGIC

ECL-3211 PROM BURNER UTILITY PROGRAM

USER GUIDE

MAN-1001-00
ISSUED MAY 1982

TABLE OF CONTENTS

FIGURES

PREFACE

This manual describes the EMULOGIC ECL-3211 PROM Burner Utility Program and presents procedures for its use. The utility is employed during the course of microprocessor development. Since the PROM burner program is run on a Digital Equipment Coporation system, you should become familiar with both the system and the following documentation:

RT-11 System User's Guide
RT-11 Programmer's Reference Manual
RT-11 Software Support Manual
PDP-11 MACRO-11 Language Reference Manual

In addition, since the PROM burner requires an executable absolute load module (.LDA) created by the EMULOGIC ECL-3211 Microprocessor Development System, you should become familiar with the ECL-3211 Emulation System User's Guide.

INTRODUCTION

The EMULOGIC PROM/ROM support utility program permits user's to "burn" a developed microprogram into Read-Only-Memory (ROM). This program is designed to permit PROM burners manufactured by various companies to be interfaced to the ECL-3211 Microprocessor Emulation System. The PROM burner connects to the ECL-3211 system's RS-232-C Serial line printer port.

Microprograms are developed using a text editor process to create source code. An appropriate cross assembler then assembles the source code and creates an OBJECT (.OBJ) file. A LINKER program, that is compatible with the cross assembler, is then used to link the OBJECT file. The output of the LINKER is an executable absolute load module (.LDA) file. It is this .LDA file that is used by the ECL-3211 emulator programs for microprogram debugging.

As the debugging stage of the microprocessor development procedure nears completion it may be necessary to burn the program into read-only-memory (ROM). It is at this point where the PROM/ROM support utility would be used.

Two steps are involved in burning a ROM. First a ROM MAP file must be created from the final version of an .LDA file. Second the ROM MAP file must be formatted as required for a specific PROM burner and then the file is output to the PROM burner.

The PROM/ROM support utility program uses the .LDA file as input to create one to four ROM MAP files. The number of ROM MAP files depends on the object microprocessor's word length.

Using the ROM MAP file(s), the PROM/ROM support utility program reformats addresses and data, and creates checksums and record counts as required for specific PROM burner formats. This process is performed by several general purpose formatting subroutines. These formatting subroutines and their sequencing are controlled by a burner specification file. Several burner specification files (.SPC), covering most of the more common PROM burner formats, are included in the support utility package. In addition the PROM/ROM support utility program includes a facility to allow user's to create their own burner specification file. Most ASCII-HEX based burner formats are supported by this facility.

PROM/ROM SUPPORT UTILITY PROGRAM FACILITIES

Three facilities are contained in the PROM/ROM support utility. These facilities and their functions are:

1. Edit burner parameters - creates a burner specification file for a user defined burner format.

2. Create a ROM MAP - creates ROM MAP file(s) from an .LDA file.

3. Load ROM data into burner - formats ROM MAP file(s) to the specified PROM burner format and outputs the formatted data to the burner.

ROM MAP FILES

Up to four ROM MAP files are created when the "Create A ROM MAP" facility of the PROM/ROM support utility is run. The number of ROM MAP files depends on the object processor's word length as follows:

1. Eight-bit microprocessors require a single ROM MAP file.

2. Sixteen-bit microprocessors require two ROM MAP files - ROM MAP 0 (.RM0) for the least significant data bytes (8 bits), and ROM MAP 1 (.RM1) for the most significant data bytes.

3. 32-bit microprocessors require four ROM MAP files (one for each data byte). .RM0 contains the least significant data bytes. .RM3 contains the most significant data bytes.

ROM MAP file(s), once created, can be treated as any other text file. They can be displayed or printed. Using a text editor, you can change data contained in a ROM MAP file. When you change ROM data, care must be taken to ensure that the format of the data in the file is not altered. Each data byte must be represented by two hex digits and followed by an ASCII space.

Printed examples of ROM MAP files are provided in Appendix A.

Using the ROM MAP Files

ROM MAP files created by the "Create A ROM MAP" facility of the PROM/ROM support utility contain the following header information, inportant to the user, in lines 1 and 2·

1. .LDA filename from which the ROM MAP file was created.

2. Creation date of the ROM MAP file.

3. Microprocessor word length.

4. Microprocessor addressing mode.

5. ROM length in bytes.

6. Data bits contained in the ROM data columns.

```
****EXAM2.LDA       18-JUN-81    8 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS-----DATA BITS-  7 TO  0-----/   256 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000   0000  04 82 00 00 00 00 00 D5 AF B8 20 F0 62 55 25 FD
 0010   000   0010  37 1D 5D 5C 96 1A 89 08 04 1C 99 F7 0A D2 25 FE
 0020   000   0020  F2 2D 1E 04 53 FE F2 2A 37 AE CE 04 53 97 FE 13
 0030   000   0030  20 F6 37 FA C6 41 CA 97 7B AB E6 41 FA 37 C6 41
 0040   000   0040  1A BE 00 FA A8 97 A7 BC 00 04 4E FC 67 AC F8 F7
 0050   000   0050  A8 96 4B FF C5 93 FB 53 07 03 5C B3 64 67 6A 6D
 0060   000   0060  70 72 74 76 FC 3C 83 FC 3D 83 FC 3E 83 FC 3F 83
 0070   000   0070  0C 83 0D 83 0E 83 0F 83 89 40 FB 90 FD 3C 23 0F
 0080   000   0080  3C 83 35 23 B8 39 23 00 3D 23 04 3C 23 04 3F 14
 0090   000   0090  07 34 6C 24 02 00 00 00 00 00 00 00 00 00 00 00
 0100   001   0000  34 30 99 EF 89 10 0A B2 10 37 92 06 0A 37 B2 02
 0110   001   0010  99 5F 89 C0 80 99 BF 89 20 18 A0 F8 97 7A E6 02
 0120   001   0020  C6 00 F0 53 3C 77 77 03 32 A9 A3 07 A3 AA 24 1B
 0130   001   0030  F9 B3 46 46 74 82 4E 69 5C CE 93 A9 43 43 43 43
 0140   001   0040  43 43 FD 24 6C FD 09 53 F8 4B 39 24 6C FB 89 40
 0150   001   0050  FD 90 14 56 FC 47 AC 14 56 24 6C FC 14 56 FC 47
 0160   001   0060  AC 80 AD 14 56 FD 24 6B FD 14 56 90 BA FE B8 02
 0170   001   0070  99 BF 83 F9 89 40 99 F8 FC 53 07 3F B8 20 FD A0
 0180   001   0080  23 03 3C 23 93 90 23 00 3C FF 90 FE 90 23 07 3C
 0190   001   0090  24 6C FD 99 F8 23 08 8F 99 F8 89 03 FB 03 0A 53
 01A0   001   00A0  0D 3E BE 04 BA FD 19 24 B1 FF 3D 14 78 FF CF 96
 01B0   001   00B0  6E BF 0F 23 C8 6E A3 AD FE CE 96 6E 23 08 3E 23
 01C0   001   00C0  0F 3D 99 F8 23 07 9F 24 6C 07 0E 0D 0B FD 99 EF
 01D0   001   00D0  89 10 99 F8 23 04 3D 99 F8 89 03 23 02 3E 26 DE
 01E0   001   00E0  23 00 3E 99 F8 23 00 3D 99 F8 89 03 23 0F 3F 24
 01F0   001   00F0  6C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

Figure 1. Sample ROM MAP

Addresses

ROM MAPs created by the "Create A ROM MAP" facility of the PROM/ROM support utility include two address indicators. The first address. Program Address (PROGR ADRS), is used to cross reference a byte of ROM data to the program listing. The second address, ROM Address (ROM ADRS), may be used to cross reference the actual ROM location where a byte of data is stored. ROM addresses are useful for editing ROM data in an off-line situation.

When the length of the microprogram exceeds the length of the ROM specified microprogram data will be stored in multiple ROMs

within the same ROM MAP file. The existence of multiple ROMs can be identified by examining the ROM number column of a displayed ROM MAP file. Also notice as the ROM number is adjusted, ROM address resets to 0000 while the program address continues to reference addresses as they would appear in a microprogram listing.

ROM ID

Each line of ROM data also contains a three character ROM ID (ROM # column). The three ROM ID characters are the most significant bits of the program address adjusted downward, relative to the ROM length specified. The "Load ROM Data Into Burner" facility of the PROM/ROM support utlilty will ask for this ID in order to select data from the map for specific ROM loads.

Data

One byte (2 hex digits) of ROM data is stored in each ROM location. Displayed or typed, ROM MAP data is formatted into lines of sixteen columns. Each line represents the content of sixteen sequential bytes of program or ROM addresses. Each column represents one of those sixteen addresses designated by the least significant address digit 0 through f. address "0" is the leftmost address, address "f" the rightmost. Each column is appropriately annotated with *0, *1, etc. through *F in line four of the ROM MAP header.

Addressing Mode

The "Create A ROM MAP" facility of the PROM/ROM support utlilty will create ROM MAPs for word addressing or byte addressing oriented object microprocessors. ROM MAPs created for byte addressing microprocessors also provide the least significant two digits of the program address in header line 3, directly above each column of ROM data bytes (Figure 2). These addresses are annotated as even byte addresses 00 through 1E for .RM0 maps, and as odd byte addresses 01 through 1F for .RM1 maps. Therefor a specific byte may be located by adding the column header in line 3 to the program address.

ROM data is handled differently for each addressing mode. The two data bytes following program address 7 for a 16-bit word oriented microprocessor would have the first data byte placed in address 7 of .RM0 and the second byte placed in address 7 of .RM1. In byte oriented 16-bit microprocessors the first byte following program address 7 would be placed in address 3 of .RM1 (upper ROM) and the second data byte placed in address 4 of .RM0 (lower ROM).

```
****EXAM3.LDA        18-JUN-81  16 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                   /-----ROM CONTENTS----DATA BITS-  7 TO  0-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** #  **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000   0000  04 00 00 00 00 00 00 D5 B8 F0 55 FD 1D 5C 1A 08
 0010   000   0010  1C F7 D2 FE 2D 00 00 00 00 00 00 00 00 00 00 00
 0020   000   0020  00 00 00 04 FE 2A AE 04 97 13 F6 FA 00 00 00 00
 0030   000   0030  00 00 00 00 C6 CA 7B E6 FA C6 1A 00 A8 A7 00 4E
 0040   000   0040  67 F8 A8 00 00 00 00 00 00 00 00 00 00 00 00 00
 0050   000   0050  00 96 FF 93 53 03 B3 67 6D 72 76 3C FC 83 3E FC
 0060   000   0060  83 83 83 83 00 00 00 00 00 00 00 00 00 00 00 00
 0070   000   0070  00 00 00 00 00 00 00 83 40 90 3C 0F 83 23 39 00
 0080   000   0080  23 3C 04 14 34 24 00 00 00 00 00 00 00 00 00 00
 0100   000   0100  34 99 89 0A 10 92 0A B2 99 89 80 BF 20 A0 97 00
 0110   000   0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 E6 C6
 0120   000   0120  F0 3C 77 32 A3 A3 24 F9 46 74 4E 5C 93 43 43 43
 0130   000   0130  FD 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0140   000   0140  00 00 00 00 6C 09 F8 39 6C 89 FD 14 FC AC 56 6C
 0150   000   0150  00 00 00 00 00 00 00 00 00 00 00 00 14 FC AC AD
 0160   000   0160  56 24 FD 56 BA B8 99 83 89 99 FC 07 B8 FD 23 00
 0180   000   0180  00 00 3C 93 23 3C 90 90 07 24 FD F8 08 99 89 FB
 0190   000   0190  0A 0D BE BA 19 00 00 00 00 00 00 00 00 00 00 00
 01A0   000   01A0  00 00 00 00 00 00 00 00 B1 3D 78 CF 6E 0F C8 A3
 01B0   000   01B0  FE 96 23 3E 0F 99 23 9F 6C 0E 0B 00 00 00 00 00
 01C0   000   01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 99 89
 01D0   000   01D0  99 23 3D F8 03 02 26 23 3E F8 00 99 89 23 3F 6C
//////////////////////////-END OF ROM MAP-//////////////////////////
```

(a) EXAM3.RM0

Figure 2.
Sample ROM MAP, 16-Bit Microprocessor with Word Addressing
(Sheet 1 of 2)

```
****EXAM3.LDA       18-JUN-81  16 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 15 TO  8-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000  0000  82 00 00 00 00 00 00 AF 20 62 25 37 5D 96 89 04
 0010   000  0010  99 0A 25 F2 1E 00 00 00 00 00 00 00 00 00 00 00
 0020   000  0020  00 00 00 53 F2 37 CE 53 FE 20 37 00 00 00 00 00
 0030   000  0030  00 00 00 00 41 97 AB 41 37 41 BE FA 97 BC 04 FC
 0040   000  0040  AC F7 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0050   000  0050  00 4B C5 FB 07 5C 64 6A 70 74 FC 83 3D FC 83 3F
 0060   000  0060  0C 0D 0E 0F 00 00 00 00 00 00 00 00 00 00 00 00
 0070   000  0070  00 00 00 00 00 00 00 89 FB FD 23 3C 35 B8 23 3D
 0080   000  0080  04 23 3F 07 6C 02 00 00 00 00 00 00 00 00 00 00
 0100   000  0100  30 EF 10 B2 37 06 37 02 5F C0 99 89 18 F8 7A 00
 0110   000  0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00
 0120   000  0120  53 77 03 A9 07 AA 1B B3 46 82 69 CE A9 43 43 43
 0130   000  0130  24 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0140   000  0140  00 00 00 00 FD 53 4B 24 FB 40 90 56 47 14 24 FC
 0150   000  0150  00 00 00 00 00 00 00 00 00 00 00 00 56 47 80 14
 0160   000  0160  FD 6B 14 90 FE 02 BF F9 40 F8 53 3F 20 A0 03 00
 0180   000  0180  00 00 23 90 00 FF FE 23 3C 6C 99 23 8F F8 03 03
 0190   000  0190  53 3E 04 FD 24 00 00 00 00 00 00 00 00 00 00 00
 01A0   000  01A0  00 00 00 00 00 00 00 00 FF 14 FF 96 BF 23 6E AD
 01B0   000  01B0  CE 6E 08 23 3D F8 07 24 07 0D FD 00 00 00 00 00
 01C0   000  01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 EF 10
 01D0   000  01D0  F8 04 99 89 23 3E DE 00 99 23 3D F8 03 0F 24 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

(b) EXAM3.RM1

Figure 2.
Sample ROM MAP, 16-bit Microprocessor with Word Addressing
(Sheet 2 of 2)

```
****EXAM4.LDA       18-JUN-81  16 BIT WORDS ----BYTE ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS-  7 TO  0-----/ 1024 BYTES
*PROG**ROM** ROM* 00 02 04 06 08 0A 0C 0E 10 12 14 16 18 1A 1C 1E
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 00000 000  0000  04 00 00 00 AF 20 62 25 37 5D 96 89 04 99 0A 25
 00020 000  0010  F2 1E 53 F2 37 CE 53 FE 20 37 C6 CA 7B E6 FA C6
 00040 000  0020  1A 00 A8 A7 00 4E 67 F8 A8 4B C5 FB 07 5C 64 6A
 00060 000  0030  70 74 FC 83 3D FC 83 3F 0C 0D 0E 0F 89 FB FD 23
 00080 000  0040  3C 35 B8 23 3D 04 23 3F 07 6C 02 00 00 00 00 00
 00100 000  0080  34 99 89 0A 10 92 0A B2 99 89 80 BF 20 A0 97 E6
 00120 000  0090  C6 F0 3C 77 32 A3 A3 24 F9 46 74 4E 5C 93 43 43
 00140 000  00A0  43 FD 6C 09 F8 39 6C 89 FD 14 FC AC 56 6C 14 FC
 00160 000  00B0  AC AD 56 24 FD 56 BA B8 99 83 89 99 FC 07 B8 FD
 00180 000  00C0  23 3C 93 23 3C 90 90 07 24 FD F8 08 99 89 FB 0A
 001A0 000  00D0  0D BE BA 19 B1 3D 78 CF 6E 0F C8 A3 FE 96 23 3E
 001C0 000  00E0  0F 99 23 9F 6C 0E 0B 99 89 99 23 3D F8 03 02 26
 001E0 000  00F0  23 3E F8 00 99 89 23 3F 6C 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

(a) EXAM4.RM0

```
****EXAM4.LDA       18-JUN-81  16 BIT WORDS ----BYTE ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 15 TO  8-----/ 1024 BYTES
*PROG**ROM** ROM* 01 03 05 07 09 0B 0D 0F 11 13 15 17 19 1B 1D 1F
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 00000 000  0000  82 00 00 D5 B8 F0 55 FD 1D 5C 1A 08 1C F7 D2 FE
 00020 000  0010  2D 04 FF 2A AE 04 97 13 F6 FA 41 97 AB 41 37 41
 00040 000  0020  BE FA 97 BC 04 FC AC F7 96 FF 93 53 03 B3 67 6D
 00060 000  0030  72 76 3C FC 83 3E FC 83 83 83 83 83 40 90 3C 0F
 00080 000  0040  83 23 39 00 23 3C 04 14 34 24 00 00 00 00 00 00
 00100 000  0080  30 EF 10 B2 37 06 37 02 5F C0 99 89 18 F8 7A 02
 00120 000  0090  00 53 77 03 A9 07 AA 1B B3 46 82 69 CE A9 43 43
 00140 000  00A0  43 24 FD 53 4B 24 FB 40 90 56 47 14 24 FC 56 47
 00160 000  00B0  80 14 FD 6B 14 90 FE 02 BF F9 40 F8 53 3F 20 A0
 00180 000  00C0  03 23 90 00 FF FE 23 3C 6C 99 23 8F F8 03 03 53
 001A0 000  00D0  3E 04 FD 24 FF 14 FF 96 BF 23 6E AD CE 6E 08 23
 001C0 000  00E0  3D F8 07 24 07 0D FD EF 10 F8 04 99 89 23 3E DE
 001E0 000  00F0  00 99 23 3D F8 03 0F 24 00 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

(b) EXAM4.RM1

Figure 3.
Sample ROM MAP, 16-Bit Microprocessor with Byte Addressing

BURNER SPECIFICATION FILES

Before running the "Load ROM Data Into Burner" facility of the PROM/ROM support utility program, a burner specification file must be established for the PROM burner you are inplementing. This is necessary to format the correct parameters to the burner. Burner specification files (.SPC) for the more common PROM/ROM burner formats are included in the PROM/ROM support utility package. Burner specification files can be displayed or printed the same as any other text file.

You can also create your own burner specification file using the "Edit Burner Parameters" facility of the PROM/ROM support utility program. The files you create can be filed under the .SPC filename extension or any other three character extension.

Burner Specification File Interpretation

A sample burner specification file display is presented in Figure 4. File content for predefined burner formats are presented in Appendix B.

```
H31HEBADC2HEBRS/
  1  50   2   1   2   1   2   1   3   0   1   1   0   2
  0    :        00
```

Figure 4. Sample Burner Specification (.SPC) File

Line 1 displays the content of the burner filespec. The burner filespec controls the sequencing of the general purpose formatting routines that output ROM MAP data bytes to a PROM burner when the "Load ROM Data Into Burner" facility is run.

Lines 2 and 3 display the content of the burner specification file. Figure 5 illustrates the positions of each parameter within a displayed or printed burner specification file.

********** Header Parameter Entry Positions **********

3 50 0 0 0 0 0 0 0 0 0 C 0 0

Header length in bytes

Header character Type:

1= Nulls 2= Deletes 3= Spaces

NOTE Trailer uses same spec but doesn't ask for definition. Header must be defined before Trailer can be used.

********** Start Code parameter entry positions **********

3 50 2 0 0 0 0 0 0 0 0 0 0 0
XX

Start of data file code type;

1 = 1 ASCII char. 2 = 2 ASCII characters
3 = STX code 4 = SOH code
5 = SOM code

Start of data file ASCII character(s)

********** Byte Count parameter entry position **********

3 50 2 7 0 0 0 0 0 0 0 0 0 0

BYTE COUNT MODIFIER +1
(Added to Byte Count as required by FORMAT spec.)

Figure 5. Burner Specification File Entries (sheet 1 of 3)

********** Record Count parameter entry position **********

```
3  50   2   7   2   0   0   0   0   0   0   0   0   0
```

RECORD COUNT;
1 = 2 bytes, 2 = 4 bytes

********** Checksum parameter entry positions **********

```
3  50   2   7   2   2   4   0   0   0   0   0   0   0
```

Checksum length in bytes

Checksum type;
1 = 2's complement
2 = Uncomplemented
3 = 1's complement
4 = 4-bit sum uncomplemented

********** Address parameter entry positions **********

```
3  50   2   7   2   2   4   3   2   0   0   0   0   0
           YY  ZZ
```

Address Trailer type;
1 = None
2 = 1 character
3 = 2 characters

Address Header type;
(Same as address trailer)

Address Field Trailer ASCII character(s)

Address Filed Header ASCII character(s)

Figure 5. Burner Specification File Entries (Sheet 2 of 3)

```
************** Data field parameter entry positions **************

   3  50    2    7    2    2    4    3    2    0    1    0    0    2
          %

                  Data Field Entered Flag

                  Zero Fill Indicator;
                    1 = Yes, 2 = No

                  Data Separator Character


************** End Of Data parameter entry positions *************

   3  50    2    7    2    2    4    3    2    0    1    1    0    2
         Q

                  End Of Data type;
                      1 = ASCII Character,  2 = ETX,  3 = ETX'

                  End Of Data ASCII character

************** End Of File parameter entry positions *************

   3  50    2    7    2    2    4    3    2    0    1    1    1    2
     F
                  End Of File type·
                      1 = An ASCII character
                      2 = Carriage return
                      3 = Carriage return and space

                  End Of File ASCII character
```

Figure 5. Burner Specification File Entries (sheet 3 of 3)

This portion of the manual presents detailed operating instructions for initiating and running the EMULOGIC ECL-3211 PROM Burner Utility Program. When you encounter the symbol <RET> in the operating instructions, it indicates you are to press the carriage return key.

STEPS TO BURNING A ROM

The normal steps to perform in burning a ROM are as follows:

1. Creation of a ROM MAP(s) from the latest version of an .LDA file.

2. Creation of a burner specification file (required only if a file does not exist for the burner format you are implementing).

3. Obtain the ROM number(s) listed in the ROM MAP file. This information is contained in the second column of the displayed ROM MAP file and is used when the "Load ROM Data Into Burner" facility asks for the ROM ID. If the ROM number is not known, it will be necessary for you to exit the PROM/ROM support utlilty and display the ROM MAP file. To display the ROM MAP file, enter TY xxxxxx.RMO<RET> following a system prompt character (where "xxxxxx" is the filename of the ROM MAP you created while running the "Create A ROM MAP" facility). Return to the PROM/ROM support utlilty.

4. Load the ROM data into the burner.

5. Burn the ROM.

NOTE

If there are multiple ROMs in the ROM MAP file the microprogram data bytes will be loaded into the burner in sections. Each section will contain the same number of bytes as the ROM byte length. The user loads a section, burns a ROM with the section just loaded, loads the next section, burns it, etc.. The PROM burner utility prompts the user for a new ROM ID before each load. This ID must be the appropriate ROM number, from the second column of the ROM MAP, for each section.

INITIATING THE PROM/ROM SUPPORT UTILITY

NOTE

Before initiating the EMULOGIC PROM/ROM Support Utility it is necessary to ensure that the operating system will pass unaltered control characters (i.e. ^C) to the PROM burner. To accomplish this enter

SET LP:CTRL <RET>

when the system prompt character (.) is displayed.

To iniate the PROM/ROM support utility:

1. Enter R EMPSUP <RET> following the system prompt character(.).

2. System responds by displaying a list of the facilities (hereafter referred to as main selection level) contained in this utility and requests you to select a option (facility) as shown below.

```
EMULOGIC PROM/ROM SUPPORT UTILITY REV-
     0. EDIT BURNER PARAMETERS
     1. CREATE A ROM MAP
     2. LOAD ROM DATA INTO BURNER
     3. EXIT ROM SUPPORT


SELECT OPTION-
```

3. Select the facility of the support utility you desire to run by entering the option number followed by <RET>.

4. System responds by displaying the title of the facility you selected.

5. Proceed to the operating instructions for the facility you have selected.

EXITING PROM/ROM SUPPORT

If you selected option 3, the system terminates the PROM/ROM Support Utility and displays the system prompt character.

CREATING A ROM MAP

If you selected option 1:

1. System responds by requesting you to enter the name of the executable absolute load module (.LDA) that contains the microprogram for the ROM MAP you want to create.

2. Enter the .LDA filename followed by <RET>. (You must enter the entire filename including the .LDA extension - i.e. xxxxxx.LDA.)

3. System responds by requesting you to enter the object microprocessor's word length.

4. Enter the object microprocessor's word length followed by <RET>. (Object microprocessor word length may be 8, 16, or 32 bits.)

5. System responds by opening ROM MAP output file(s) and displaying ROM MAP output filename(s) after each file has successfully been "opened" by the system.

NOTE

ROM MAP output file(s) are created with the filename(s) xxxxxx.RM0 through xxxxxx.RM3 (where xxxxxx is the name of the .LDA file specified in Step 2). Any ROM MAP files already existing under the same name will be deleted from the system.

6. If you entered an object microprocessor word length of 8, proceed to step 8. If you entered an object processor word length of 16 or 32, the system requests you to select the addressing mode of the object microprocessor (byte or word).

7. Select the addressing mode of the object microprocessor by entering the correct option number (1 or 2) followed by <RET>.

8. System requests you to enter the object microprocessor's ROM length in bytes.

9. Enter the ROM length in bytes followed by <RET>. ROM length must be specified in powers of 2, and can be any length between 32 and 8192 (32, 256, 512, etc.).

10. System inputs addresses and byte data from the .LDA file, reformats the byte data into ASCII-HEX digits and places the reformatted data into the appropriate ROM MAP file(s). As each line of sixteen data bytes is filled, or when the .LDA file specifies a new address that is outside of the current sixteen bytes of data, the lines of HEX formatted data are output to the ROM MAP file(s). After all byte data from the .LDA file has been formatted into HEX ROM MAP data, an "end of ROM MAP" line is output to each ROM MAP file, all files are "closed" and the support utility returns to the main selection level for further user selection. (Initiating the PROM/ROM Support Utility, Step 2.)

LOADING ROM DATA INTO BURNER

If you selected option 2:

1. System responds by displaying a list of the current ROM burner format options and requests you to select a format as shown below.

```
CURRENT ROM BURNER OPTIONS
     0. DATA I/O TYPE 50
     1. INTEL INTELLIC FORMAT  (DATAIO 83)
     2. MOS TECHNOLOGY FORMAT  (DATAIO 81)
     3. MOTOROLA DATA FILES    (DATAIO 82)
     4. TEKTRONIX HEX FORMAT   (DATAIO 86)
     5. RCA COSMAC FORMAT      (DATAIO 70)
     6. USER DEFINED BURNER FORMAT
     7. EXIT ROM SUPPORT

SELECT OPTION-
```

2. Select the appropriate ROM burner format for the current job by entering the specified option number followed by <RET>.

3. If you selected one of the predefined ROM burner formats in Step 2, proceed to Step 6.

4. If you selected "User Defined Burner Format" in Step 2, the system responds by displaying the option title and requests you to enter the burner spec filename.

5. Enter the burner specification filename followed by <RET>. (You must enter the entire filename, including the filename extension you assigned when the burner specification file was created during the "Edit Burner Parameters" facility.)

6. System responds by displaying the content of the burner specification file to be used and requests you to enter the ROM MAP filename for the ROM you are going to burn or to exit the routine.

7. At this point you can exit the routine and return to the main selection level of the PROM/ROM Support Utility Program by entering 0<RET>. (Refer to "Initiating PROM/ROM Support Utility", Step 2.)

   Otherwise enter the ROM MAP filename followed by <RET>. (you must enter the entire filename including the .RMn extension - i.e. xxxxxx.RMn - where xxxxxx is the ROM MAP name you specified when you created the ROM MAP and n is the ROM MAP number you are going to burn.)

8. System requests you to enter a three character ROM ID or to exit this file. The ROM ID is found in the second column of the ROM MAP you specified in Step 7.

9. At this point you can initiate an exit of the "Load ROM Data Into Burner" facility by entering N<RET> and returning to Step 7.

   Otherwise at the point you must ensure the PROM burner is ready to receive data.

   If you are using a DATA I/O PROM burner, before you enter the ROM ID you must perform the following procedures at the PROM burner

   a. Press SELECT.

   b. Enter the translation-format select code. This select code is the two digit number defined (DATAIO xx) included in the current burner option selection menu displayed in Step 1.

   c. Press START.

   d. Press SELECT.

   e. Enter the input operation select code as defined in the DATA I/O instruction manual. This select code is normally D1 (selects "Initiates the input to RAM via the serial port") but can be any other input option.

   f. Press START.

10. Enter the ROM ID followed by <RET>.

11. System responds by loading the PROM burner. When loading is complete the system responds as follows:

    a. If there are multiple ROMs specified in the ROM MAP file, the system requests you to enter the next three character ROM ID or to exit this file.

       At this point you must burn the outputted section of data into the ROM. If you are going to burn a quantity of ROMs with the outputted data you can exit the routine and release the system for other use. Return to Step 9.

    b. If there are no multiple ROMs specified in the ROM MAP file, the system asks you to enter another ROM MAP filename or to exit the routine. Return to Step 7.

EDITING BURNER PARAMETERS

If you selected option 0:

1. System responds by outputting a list of burner filespec parameter code characters as shown below.

```
  INITIAL INPUT OF BURNER FILESPEC SKELETON
H=HEADER       S=START CODE      A=ADRESS FIELD
B=BYTE COUNT   C=CHECKSUM        D=DATA FIELD
E=END DATA     F=END OF FILE     X=DATA SEPARATOR
T=TRAILER      /=END OF SPEC.    ,=OPTIONAL FIELD SEPARATOR
R=RECORD COUNT
     1=START OF DATA LOOP      2=END OF DATA LOOP(OUTPUTS)
     3=(OUTPUT TO BURNER       4=(OUTPUT TO BURNER
        CONTINUE IN SPEC)         RETURN TO ADDRESS FIELD)
          BURNER MAY NOT REQUIRE ALL FIELDS
```

2. System requests you to enter the burner filespec parameters.

NOTE

Filespec parameters should be entered in a string. When entering parameters, field separators (,) can be used or omitted. A string of parameters should be terminated using the end of spec code character (/).

3. Enter the burner filespec parameters code letters followed by <RET>. When inputting the burner filespec parameters you must specify all of the required parameters in the order they are required for the specific PROM burner format you are going to inplement. Use the parameter codes specified when you enter the burner filespec. Use the numerics (1 to 4) to specify subroutines, loops and outputs to the burner.

4. System:

   a. Displays one line containing the code character for the parameters already entered and the parameter to be entered at this time.

   b. Displays selection information for the parameter to be entered at this time.

   c. Asks one or more questions about the parameter.

Figure 6 presents the selection information and questions asked for each parameter. Remember, the selection information and questions will be presented in the order which you entered them into the burner filespec request.

5. Enter your correct response for each question requested. Follow your entry by <RET>.

6. Steps 4 and 5 are repeated until the end of spec code is reached. When the end of spec code is reached, proceed to Step 7.

7. System responds by outputting a burner filespec input completed message and requesting you to enter a filename to save the burner spec file.

8. Enter the filename for the burner spec file in the format xxxxxx.yyy<RET> (where xxx can be any 1 to 6 characters in length and yyy is any 3 character filename extension.)

9. System creates a burner specification file and returns to the main selection level (Refer to "Initiating the PROM/ROM Support Utility", Step 2).

```
******************** HEADER PARAMETERS ***********************

       HEADER TYPE INCLUDES 1=NULLS 2=DELETE CODES 3=SPACES

        INPUT TYPE-
        INPUT LENGTH IN BYTES-


******************** BYTE COUNT PARAMETER ********************

       BYTE COUNT IS ASSUMED TO BE EXPRESSED BY 2 HEX DIGITS
       IT MAY INCLUDE A MODIFIER FOR ADDRESS AND CHECKSUM

        INPUT BYTE COUNT MODIFIER -


******************** END DATA PARAMETER **********************

       END OF DATA CHARACTER CAN BE:
       1. ASCII CHARACTER   2. ETX    3. ETX'

        INPUT TYPE-
        INPUT CHARACTERS-


******************** RECORD COUNT PARAMETER ******************

       RECORD COUNT IS EITHER 1. 2 BYTES   2. 4 BYTES
       IT IS ASSUMED TO BE IN HEX FORMAT.

        INPUT TYPE-


******************** START CODE PARAMETER ********************

       START OF DATA FILE CAN BE INDICATED BY:
                           1. 1 ASCII CHAR.    2. 2 ASCII CHAR.
                           3. STX CODE         4. SOH CODE
                           5. SOM CODE

        INPUT TYPE-
        INPUT CHARACTERS-
```

Figure 6.
Burner Filespec Selection Information and Entry Requests
(Sheet 1 of 2)

```
******************** CHECKSUM PARAMETERS **********************

       CHECKSUM IS ASSUMED TO BE THE SUM OF DATA,ADDRESS,BC
       XOR CHECKSUMS ARE NOT SUPPORTED. SELECT ONE OF THE FOLLOWING:
         1. 2'S COMPLEMENT   2. UNCOMPLEMENTED  3. 1'S COMPLEMENT
         4. 4 BIT SUM UNCOMPLEMENTED(OTHERS ARE BYTE DATA SUMS.

        INPUT TYPE-
        INPUT LENGTH IN BYTES-


******************** END OF FILE PARAMETERS ******************

       END OF DATA FILE CAN BE EXPRESSED BY
       1. AN ASCII CHARACTER  2. CAR. RET.    3. CAR. RET./SPACE

        INPUT TYPE-


******************** ADDRESS FILED PARAMETERS ****************

       ADDRESS FIELD IS 4 HEX CHAR.'S PRECEDED BY:
               1. NONE   2. 1 CHAR.   3.  2 CHAR.'S

        INPUT TYPE-
        ADDRESS FIELD TRAILER?
               1. NONE   2. 1 CHAR.   3.  2 CHAR.'S

        INPUT TYPE-


******************** DATA FIELD PARAMETERS *******************

       BURNER MAY REQUIRE ZERO DATA TO BE FILLED
        ENTER    1. IF YES     2. IF IT DOES NOT  -

        DATA SEPARATOR IS AN ASCII CHAR.
        INPUT CHARACTERS-



******************** SAVE SPEC FILE ***************************

       INPUT OF BURNER FILESPEC COMPLETE
       PARAMETERS MUST BE SAVED FOR USE IN BURNING ROMS

       INPUT FILENAME TO SAVE BURNER SPEC-
```

Figure 6.
Burner Filespec Selection Information and Entry Requests
(Sheet 2 of 2)

The following ROM MAP was created for use by an 8-bit CPU and would be stored in a single ROM, 1024 bytes in length. ROM MAPs for 8-bit CPUs are always created in word addressing mode as in this case. Note that header information also contains the creation date of the ROM MAP and the ROM length specified. In this case the program address (PROG ADRS) and the ROM address (ROM ADRS) are equivalent because all program bytes are stored in one ROM (ROM # 000).

NOTES

1. The "Load a ROM Burner" option will request this ROM I.D. (ROM # column) to be input before outputting data to the PROM burner.

2. In this example, as with all of the following examples, the assumption is made that DATA BIT 0 is the least significant bit of data.

****EXAM1.LDA 18-JUN-81 8 BIT WORDS ----WORD ADDRESSES ROM LENGTH
* /-----ROM CONTENTS----DATA BITS- 7 TO 0-----/ 1024 BYTES

PROG	*ROM*	ROM*														
ADRS	#	*ADRS*	*0	*1	*2	*3	*4	*5	*6	*7	*8	*9	*A	*B	*C	*D	*E	*F
0000	000	0000	04	82	00	00	00	00	00	D5	AF	B8	20	F0	62	55	25	FD
0010	000	0010	37	1D	5D	5C	96	1A	89	08	04	1C	99	F7	0A	D2	25	FE
0020	000	0020	F2	2D	1E	04	53	FE	F2	2A	37	AE	CE	04	53	97	FE	13
0030	000	0030	20	F6	37	FA	C6	41	CA	97	7B	AB	E6	41	FA	37	C6	41
0040	000	0040	1A	BE	00	FA	A8	97	A7	BC	00	04	4E	FC	67	AC	F8	F7
0050	000	0050	A8	96	4B	FF	C5	93	FB	53	07	03	5C	B3	64	67	6A	6D
0060	000	0060	70	72	74	76	FC	3C	83	FC	3D	83	FC	3E	83	FC	3F	83
0070	000	0070	0C	83	0D	83	0E	83	0F	83	89	40	FB	90	FD	3C	23	0F
0080	000	0080	3C	83	35	23	B8	39	23	00	3D	23	04	3C	23	04	3F	14
0090	000	0090	07	34	6C	24	02	00	00	00	00	00	00	00	00	00	00	00
0100	000	0100	34	30	99	EF	89	10	0A	B2	10	37	92	06	0A	37	B2	02
0110	000	0110	99	5F	89	C0	80	99	BF	89	20	18	A0	F8	97	7A	E6	02
0120	000	0120	C6	00	F0	53	3C	77	77	03	32	A9	A3	07	A3	AA	24	1B
0130	000	0130	F9	B3	46	46	74	82	4E	69	5C	CE	93	A9	43	43	43	43
0140	000	0140	43	43	FD	24	6C	FD	09	53	F8	4B	39	24	6C	FB	89	40
0150	000	0150	FD	90	14	56	FC	47	AC	14	56	24	6C	FC	14	56	FC	47
0160	000	0160	AC	80	AD	14	56	FD	24	6B	FD	14	56	90	BA	FE	B8	02
0170	000	0170	99	BF	83	F9	89	40	99	F8	FC	53	07	3F	B8	20	FD	A0
0180	000	0180	23	03	3C	23	93	90	23	00	3C	FF	90	FE	90	23	07	3C
0190	000	0190	24	6C	FD	99	F8	23	08	8F	99	F8	89	03	FB	03	0A	53
01A0	000	01A0	0D	3E	BE	04	BA	FD	19	24	B1	FF	3D	14	78	FF	CF	96
01B0	000	01B0	6E	BF	0F	23	C8	6E	A3	AD	FE	CE	96	6E	23	08	3E	23
01C0	000	01C0	0F	3D	99	F8	23	07	9F	24	6C	07	0E	0D	0B	FD	99	EF
01D0	000	01D0	89	10	99	F8	23	04	3D	99	F8	89	03	23	02	3E	26	DE
01E0	000	01E0	23	00	3E	99	F8	23	00	3D	99	F8	89	03	23	0F	3F	24
01F0	000	01F0	6C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

////////////////////////-END OF ROM MAP-////////////////////////

ROM MAP EXAMPLE #2

As with ROM MAP EXAMPLE #1, the following ROM MAP was created for use on an 8 bit CPU, however ROM data would be stored in multiple ROM's. In this example, the ROM's are 256 bytes in length. The length ofthe program exceeds the length of a single ROM. ROM I.D. # 000 contains the program bytes 0000 to 00FF. ROM I.D. # 001 contains program bytes 0100 to 01FF.(The ROM # is simply the high order bits of the program address adjusted downward in relation to ROM length.)

NOTE

When ROM I.D. # 001 begins, ROM address (ROM ADRS) resets to 0000. while the program address (PROG ADRS) continues to reference addresses as they would appear in a program listing file.

****EXAM2.LDA 18-JUN-81 8 BIT WORDS ----WORD ADDRESSES ROM LENGTH
* /-----ROM CONTENTS----DATA BITS- 7 TO 0-----/ 256 BYTES

PROG	*ROM*	ROM*														
ADRS	#	*ADRS*	*0	*1	*2	*3	*4	*5	*6	*7	*8	*9	*A	*B	*C	*D	*E	*F
0000	000	0000	04	82	00	00	00	00	00	D5	AF	B8	20	F0	62	55	25	FD
0010	000	0010	37	1D	5D	5C	96	1A	89	08	04	1C	99	F7	0A	D2	25	FE
0020	000	0020	F2	2D	1E	04	53	FE	F2	2A	37	AE	CE	04	53	97	FE	13
0030	000	0030	20	F6	37	FA	C6	41	CA	97	7B	AB	E6	41	FA	37	C6	41
0040	000	0040	1A	BE	00	FA	A8	97	A7	BC	00	04	4E	FC	67	AC	F8	F7
0050	000	0050	A8	96	4B	FF	C5	93	FB	53	07	03	5C	B3	64	67	6A	6D
0060	000	0060	70	72	74	76	FC	3C	83	FC	3D	83	FC	3E	83	FC	3F	83
0070	000	0070	0C	83	0D	83	0E	83	0F	83	89	40	FB	90	FD	3C	23	0F
0080	000	0080	3C	83	35	23	B8	39	23	00	3D	23	04	3C	23	04	3F	14
0090	000	0090	07	34	6C	24	02	00	00	00	00	00	00	00	00	00	00	00
0100	001	0000	34	30	99	EF	89	10	0A	B2	10	37	92	06	0A	37	B2	02
0110	001	0010	99	5F	89	C0	80	99	BF	89	20	18	A0	F8	97	7A	E6	02
0120	001	0020	C6	00	F0	53	3C	77	77	03	32	A9	A3	07	A3	AA	24	1B
0130	001	0030	F9	B3	46	46	74	82	4E	69	5C	CE	93	A9	43	43	43	43
0140	001	0040	43	43	FD	24	6C	FD	09	53	F8	4B	39	24	6C	FB	89	40
0150	001	0050	FD	90	14	56	FC	47	AC	14	56	24	6C	FC	14	56	FC	47
0160	001	0060	AC	80	AD	14	56	FD	24	6B	FD	14	56	90	BA	FE	B8	02
0170	001	0070	99	BF	83	F9	89	40	99	F8	FC	53	07	3F	B8	20	FD	A0
0180	001	0080	23	03	3C	23	93	90	23	00	3C	FF	90	FE	90	23	07	3C
0190	001	0090	24	6C	FD	99	F8	23	08	8F	99	F8	89	03	FB	03	0A	53
01A0	001	00A0	0D	3E	BE	04	BA	FD	19	24	B1	FF	3D	14	78	FF	CF	96
01B0	001	00B0	6E	BF	0F	23	C8	6E	A3	AD	FE	CE	96	6E	23	08	3E	23
01C0	001	00C0	0F	3D	99	F8	23	07	9F	24	6C	07	0E	0D	0B	FD	99	EF
01D0	001	00D0	89	10	99	F8	23	04	3D	99	F8	89	03	23	02	3E	26	DE
01E0	001	00E0	23	00	3E	99	F8	23	00	3D	99	F8	89	03	23	0F	3F	24
01F0	001	00F0	6C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

////////////////////////-END OF ROM MAP-////////////////////////

ROM MAP EXAMPLE #3 (EXAM3.RM0)

The following two ROM MAP's were created for use by a 16 bit CPU utilizing word addressing mode. The ROM data bytes are to be stored in two ROM's, each of which is 1024 bytes in length. The example below stores the least significant data bytes in ROM # 000. Note in this case, the program address (PROG ADRS) and the ROM address (ROM ADRS) are equivalent.

NOTE

A 16 bit CPU requires the creation of 2 ROM MAP files: RM0,which will store the least significant byte (shown below) and RM1, which will store the most significant byte (refer ROM MAP EXAM3.RM1 for DATA BITS 15 TO 8).

The following ROM MAP is the contents of EXAM3.RM0:

```
****EXAM3.LDA      18-JUN-81  16 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                 /-----ROM CONTENTS----DATA BITS-  7 TO  0-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000  0000  04 00 00 00 00 00 00 D5 B8 F0 55 FD 1D 5C 1A 08
 0010   000  0010  1C F7 D2 FE 2D 00 00 00 00 00 00 00 00 00 00 00
 0020   000  0020  00 00 00 04 FE 2A AF 04 97 13 F6 FA 00 00 00 00
 0030   000  0030  00 00 00 00 C6 CA 7B E6 FA C6 1A 00 A8 A7 00 4E
 0040   000  0040  67 F8 A8 00 00 00 00 00 00 00 00 00 00 00 00 00
 0050   000  0050  00 96 FF 93 53 03 B3 67 6D 72 76 3C FC 83 3E FC
 0060   000  0060  83 83 83 83 00 00 00 00 00 00 00 00 00 00 00 00
 0070   000  0070  00 00 00 00 00 00 00 83 40 90 3C 0F 83 23 39 00
 0080   000  0080  23 3C 04 14 34 24 00 00 00 00 00 00 00 00 00 00
 0100   000  0100  34 99 89 0A 10 92 0A B2 99 89 80 BF 20 A0 97 00
 0110   000  0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 E6 C6
 0120   000  0120  F0 3C 77 32 A3 A3 24 F9 46 74 4E 5C 93 43 43 43
 0130   000  0130  FD 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0140   000  0140  00 00 00 00 6C 09 F8 39 6C 89 FD 14 FC AC 56 6C
 0150   000  0150  00 00 00 00 00 00 00 00 00 00 00 00 14 FC AC AD
 0160   000  0160  56 24 FD 56 BA B8 99 83 89 99 FC 07 B8 FD 23 00
 0180   000  0180  00 00 3C 93 23 3C 90 90 07 24 FD F8 08 99 89 FB
 0190   000  0190  0A 0D BE BA 19 00 00 00 00 00 00 00 00 00 00 00
 01A0   000  01A0  00 00 00 00 00 00 00 00 B1 3D 78 CF 6E 0F C8 A3
 01B0   000  01B0  FE 96 23 3E 0F 99 23 9F 6C 0E 0B 00 00 00 00 00
 01C0   000  01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 99 89
 01D0   000  01D0  99 23 3D F8 03 02 26 23 3E F8 00 99 89 23 3F 6C
//////////////////////////-END OF ROM MAP-//////////////////////////
```

The following ROM MAP details the contents of EXAM3.RM1. It contains the most significant byte of data from EXAM3.LDA:

```
****EXAM3.LDA       18-JUN-81  16 BIT WORDS ----WORD ADDRESSES     ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 15 TO  8-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000  0000  82 00 00 00 00 00 00 AF 20 62 25 37 5D 96 89 04
 0010   000  0010  99 0A 25 F2 1E 00 00 00 00 00 00 00 00 00 00 00
 0020   000  0020  00 00 00 53 F2 37 CE 53 FE 20 37 00 00 00 00 00
 0030   000  0030  00 00 00 00 41 97 AB 41 37 41 BE FA 97 BC 04 FC
 0040   000  0040  AC F7 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0050   000  0050  00 4B C5 FB 07 5C 64 6A 70 74 FC 83 3D FC 83 3F
 0060   000  0060  0C 0D 0E 0F 00 00 00 00 00 00 00 00 00 00 00 00
 0070   000  0070  00 00 00 00 00 00 00 89 FB FD 23 3C 35 B8 23 3D
 0080   000  0080  04 23 3F 07 6C 02 00 00 00 00 00 00 00 00 00 00
 0100   000  0100  30 EF 10 B2 37 06 37 02 5F C0 99 89 18 F8 7A 00
 0110   000  0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00
 0120   000  0120  53 77 03 A9 07 AA 1B B3 46 82 69 CE A9 43 43 43
 0130   000  0130  24 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0140   000  0140  00 00 00 00 FD 53 4B 24 FB 40 90 56 47 14 24 FC
 0150   000  0150  00 00 00 00 00 00 00 00 00 00 00 00 56 47 80 14
 0160   000  0160  FD 6B 14 90 FE 02 BF F9 40 F8 53 3F 20 A0 03 00
 0180   000  0180  00 00 23 90 00 FF FE 23 3C 6C 99 23 8F F8 03 03
 0190   000  0190  53 3E 04 FD 24 00 00 00 00 00 00 00 00 00 00 00
 01A0   000  01A0  00 00 00 00 00 00 00 00 FF 14 FF 96 BF 23 6E AD
 01B0   000  01B0  CE 6E 08 23 3D F8 07 24 07 0D FD 00 00 00 00 00
 01C0   000  01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 EF 10
 01D0   000  01D0  F8 04 99 89 23 3E DE 00 99 23 3D F8 03 0F 24 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

ROM MAP EXAMPLE #4 (EXAM4.RM0)

Again, the following 2 ROM MAP's were created for use on a 16 bit CPU, and would be stored in 2 ROM's, each being 1024 bytes in length. The example below stores the least significant bytes of data in ROM # 000. The most significant bytes (DATA BITS 15 TO 8) are stored in ROM # 000 of EXAM4.RM1.

NOTE

ROM MAPS for 16 bit CPU's can be created in two addressing modes as follows:

1) Word addressing mode (shown in Example #3)
2) Byte addressing mode (shown in this example)

Byte oriented processors place the most significant data bytes data (odd byte addresses) in RM1 and the least significant data bytes (even addresses) in RM0.

In byte addressing mode, the lower digits of the program address (PROG ADRS) are provided in a line directly above the ROM column addresses. Therefore, a specific data byte may be located by adding this column header to the program address. (i.e. It can be seen that Program Byte 78 is actually stored in ROM # 000 address 3C.)

The following ROM MAP details the contents of EXAM4.RM0, containing the least significant byte of data (at even program addresses in EXAM4.LDA.

```
****EXAM4.LDA       18-JUN-81  16 BIT WORDS ----BYTE ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS-  7 TO  0-----/ 1024 BYTES
*PROG**ROM** ROM* 00 02 04 06 08 0A 0C 0E 10 12 14 16 18 1A 1C 1E
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 00000 000  0000  04 00 00 00 AF 20 62 25 37 5D 96 89 04 99 0A 25
 00020 000  0010  F2 1E 53 F2 37 CE 53 FE 20 37 C6 CA 7B E6 FA C6
 00040 000  0020  1A 00 A8 A7 00 4E 67 F8 A8 4B C5 FB 07 5C 64 6A
 00060 000  0030  70 74 FC 83 3D FC 83 3F 0C 0D 0E 0F 89 FB FD 23
 00080 000  0040  3C 35 B8 23 3D 04 23 3F 07 6C 02 00 00 00 00 00
 00100 000  0080  34 99 89 0A 10 92 0A B2 99 89 80 BF 20 A0 97 E6
 00120 000  0090  C6 F0 3C 77 32 A3 A3 24 F9 46 74 4E 5C 93 43 43
 00140 000  00A0  43 FD 6C 09 F8 39 6C 89 FD 14 FC AC 56 6C 14 FC
 00160 000  00B0  AC AD 56 24 FD 56 BA B8 99 83 89 99 FC 07 B8 FD
 00180 000  00C0  23 3C 93 23 3C 90 90 07 24 FD F8 08 99 89 FB 0A
 001A0 000  00D0  0D BE BA 19 B1 3D 78 CF 6E 0F C8 A3 FE 96 23 3E
 001C0 000  00E0  0F 99 23 9F 6C 0E 0B 99 89 99 23 3D F8 03 02 26
 001E0 000  00F0  23 3E F8 00 99 89 23 3F 6C 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

The following ROM MAP details the contents of EXAM4.RM1, containing the most significant byte of data (at odd program addresses) from EXAM4.LDA:

```
****EXAM4.LDA       18-JUN-81  16 BIT WORDS ----BYTE ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 15 TO  8-----/ 1024 BYTES
*PROG**ROM** ROM* 01 03 05 07 09 0B 0D 0F 11 13 15 17 19 1B 1D 1F
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 00000 000   0000  82 00 00 D5 B8 F0 55 FD 1D 5C 1A 08 1C F7 D2 FE
 00020 000   0010  2D 04 FE 2A AE 04 97 13 F6 FA 41 97 AB 41 37 41
 00040 000   0020  BE FA 97 BC 04 FC AC F7 96 FF 93 53 03 B3 67 6D
 00060 000   0030  72 76 3C FC 83 3E FC 83 83 83 83 83 40 90 3C 0F
 00080 000   0040  83 23 39 00 23 3C 04 14 34 24 00 00 00 00 00 00
 00100 000   0080  30 EF 10 B2 37 06 37 02 5F C0 99 89 18 F8 7A 02
 00120 000   0090  00 53 77 03 A9 07 AA 1B B3 46 82 69 CE A9 43 43
 00140 000   00A0  43 24 FD 53 4B 24 FB 40 90 56 47 14 24 FC 56 47
 00160 000   00B0  80 14 FD 6B 14 90 FE 02 BF F9 40 F8 53 3F 20 A0
 00180 000   00C0  03 23 90 00 FF FE 23 3C 6C 99 23 8F F8 03 03 53
 001A0 000   00D0  3E 04 FD 24 FF 14 FF 96 BF 23 6E AD CE 6E 08 23
 001C0 000   00E0  3D F8 07 24 07 0D FD EF 10 F8 04 99 89 23 3E DE
 001E0 000   00F0  00 99 23 3D F8 03 0F 24 00 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

ROM MAP EXAMPLE #5 (EXAM5.RM0)

The following 4 ROM MAPS's were created for use on a 32 bit CPU. The ROM data bytes would be stored in four ROM's, each of which is 1024 bytes in length. The example below stores the least significant data bytes in ROM # 000. Again, notice the program address (PROG ADRS) and the ROM address (ROM ADRS) are equivalent.

NOTE

A 32-bit CPU requires the creation of 4 ROM MAP files: RM0, which contains the least significant byte of data; RM1, contains the most significant byte of the low order 16-bits of the 32 bits; RM2, contains the least significant byte of the high order 16-bits; and finally, RM3, which contains the most significant byte of the high order 16 bits.

The following ROM MAP details the contents of EXAM5.RM0. It contains data bits 7 to 0 (the least significant byte) of data from EXAM5.LDA.

****EXAM5.LDA 18-JUN-81 32 BIT WORDS ----WORD ADDRESSES ROM LENGTH
* /-----ROM CONTENTS----DATA BITS- 7 TO 0-----/ 1024 BYTES

*PROG**	ROM**	ROM*														
*ADRS**	#	**ADRS*	*0	*1	*2	*3	*4	*5	*6	*7	*8	*9	*A	*B	*C	*D	*E	*F
0000	000	0000	04	00	00	00	00	00	00	D5	F0	FD	5C	08	F7	FE	00	00
0020	000	0020	00	00	00	04	2A	04	13	FA	00	00	00	00	00	00	00	00
0030	000	0030	00	00	00	00	C6	7B	FA	1A	A8	00	67	A8	00	00	00	00
0050	000	0050	00	96	93	03	67	72	3C	83	FC	83	83	00	00	00	00	00
0070	000	0070	00	00	00	00	00	00	00	83	90	0F	23	00	3C	14	24	00
0100	000	0100	34	89	10	0A	99	80	20	97	00	00	00	00	00	00	00	00
0110	000	0110	00	00	00	00	00	00	00	00	00	00	00	00	00	00	E6	F0
0120	000	0120	77	A3	24	46	4E	93	43	FD	00	00	00	00	00	00	00	00
0140	000	0140	00	00	00	00	6C	F8	6C	FD	FC	56	00	00	00	00	00	00
0150	000	0150	00	00	00	00	00	00	00	00	00	00	00	00	14	AC	56	FD
0160	000	0160	BA	99	89	FC	B8	23	00	00	00	00	00	00	00	00	00	00
0180	000	0180	00	00	3C	23	90	07	FD	08	89	0A	BE	19	00	00	00	00
01A0	000	01A0	00	00	00	00	00	00	00	00	B1	78	6E	C8	FE	23	0F	23
01B0	000	01B0	6C	0B	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01C0	000	01C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	99	99
01D0	000	01D0	3D	03	26	3E	00	89	3F	00	00	00	00	00	00	00	00	00

////////////////////////-END OF ROM MAP-////////////////////////

The following ROM MAP details the contents of EXAM5.RM1. It contains the most significant byte (data bits 15 to 8) of the low order 16-bits of the 32 bits from EXAM5.LDA:

```
****EXAM5.LDA       18-JUN-81  32 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 15 TO  8-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000  0000  82 00 00 00 00 00 00 AF 62 37 96 04 0A F2 00 00
 0020   000  0020  00 00 00 53 37 53 20 00 00 00 00 00 00 00 00 00
 0030   000  0030  00 00 00 00 41 AB 37 BE 97 04 AC 00 00 00 00 00
 0050   000  0050  00 4B FB 5C 6A 74 83 FC 3F 0D 0F 00 00 00 00 00
 0070   000  0070  00 00 00 00 00 00 00 89 FD 3C B8 3D 23 07 02 00
 0100   000  0100  30 10 37 37 5F 99 18 7A 00 00 00 00 00 00 00 00
 0110   000  0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 53
 0120   000  0120  03 07 1B 46 69 A9 43 24 00 00 00 00 00 00 00 00
 0140   000  0140  00 00 00 00 FD 4B FB 90 47 24 00 00 00 00 00 00
 0150   000  0150  00 00 00 00 00 00 00 00 00 00 00 00 56 80 FD 14
 0160   000  0160  FE BF 40 53 20 03 00 00 00 00 00 00 00 00 00 00
 0180   000  0180  00 00 23 00 FE 3C 99 8F 03 53 04 24 00 00 00 00
 01A0   000  01A0  00 00 00 00 00 00 00 00 FF FF BF 6E CE 08 3D 07
 01B0   000  01B0  07 FD 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 01C0   000  01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 EF F8
 01D0   000  01D0  99 23 DE 99 3D 03 24 00 00 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

The following ROM MAP details the contents of EXAM5.RM2. It contains the least significant byte (data bits 23 to 16) of the high order 16 bits of the 32 bits from EXAM5.LDA:

```
****EXAM5.LDA       18-JUN-81  32 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 23 TO 16-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000  0000  00 00 00 00 00 00 00 B8 55 1D 1A 1C D2 2D 00 00
 0020   000  0020  00 00 00 FE AE 97 F6 00 00 00 00 00 00 00 00 00
 0030   000  0030  00 00 00 00 CA E6 C6 00 A7 4E F8 00 00 00 00 00
 0050   000  0050  00 FF 53 B3 6D 76 FC 3E 83 83 00 00 00 00 00 00
 0070   000  0070  00 00 00 00 00 00 00 40 3C 83 39 23 04 34 00 00
 0100   000  0100  99 0A 92 B2 89 BF A0 00 00 00 00 00 00 00 00 00
 0110   000  0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 C6 3C
 0120   000  0120  32 A3 F9 74 5C 43 43 00 00 00 00 00 00 00 00 00
 0140   000  0140  00 00 00 00 09 39 89 14 AC 6C 00 00 00 00 00 00
 0150   000  0150  00 00 00 00 00 00 00 00 00 00 00 00 FC AD 24 56
 0160   000  0160  B8 83 99 07 FD 00 00 00 00 00 00 00 00 00 00 00
 0180   000  0180  00 00 93 3C 90 24 F8 99 FB 0D BA 00 00 00 00 00
 01A0   000  01A0  00 00 00 00 00 00 00 00 3D CF 0F A3 96 3E 99 9F
 01B0   000  01B0  0E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 01C0   000  01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 89 23
 01D0   000  01D0  F8 02 23 F8 99 23 6C 00 00 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

The following ROM MAP details the contents of EXAM5.RM3. It contains the most significant byte (data bits 31 to 24) of the high order 16-bits of the 32 bits from EXAM5.LDA:

```
****EXAM5.LDA       18-JUN-81  32 BIT WORDS ----WORD ADDRESSES      ROM LENGTH
*                  /-----ROM CONTENTS----DATA BITS- 31 TO 24-----/ 1024 BYTES
*PROG**ROM** ROM*
*ADRS** # **ADRS* *0 *1 *2 *3 *4 *5 *6 *7 *8 *9 *A *B *C *D *E *F
 0000   000  0000  00 00 00 00 00 00 00 20 25 5D 89 99 25 1E 00 00
 0020   000  0020  00 00 00 F2 CE FE 37 00 00 00 00 00 00 00 00 00
 0030   000  0030  00 00 00 00 97 41 41 FA BC FC F7 00 00 00 00 00
 0050   000  0050  00 C5 07 64 70 FC 3D 83 0C 0E 00 00 00 00 00 00
 0070   000  0070  00 00 00 00 00 00 00 FB 23 35 23 04 3F 6C 00 00
 0100   000  0100  EF B2 06 02 C0 89 F8 00 00 00 00 00 00 00 00 00
 0110   000  0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 77
 0120   000  0120  A9 AA B3 82 CE 43 43 00 00 00 00 00 00 00 00 00
 0140   000  0140  00 00 00 00 53 24 40 56 14 FC 00 00 00 00 00 00
 0150   000  0150  00 00 00 00 00 00 00 00 00 00 00 00 47 14 6B 90
 0160   000  0160  02 F9 F8 3F A0 00 00 00 00 00 00 00 00 00 00 00
 0180   000  0180  00 00 90 FF 23 6C 23 F8 03 3E FD 00 00 00 00 00
 01A0   000  01A0  00 00 00 00 00 00 00 00 14 96 23 AD 6E 23 F8 24
 01B0   000  01B0  0D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 01C0   000  01C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 10 04
 01D0   000  01D0  89 3E 00 23 F8 0F 00 00 00 00 00 00 00 00 00 00
//////////////////////////-END OF ROM MAP-//////////////////////////
```

```
********** DATA I/O TYPE 50 **********

        S1A3DX2EH/
          3  20   3   0   0   0   0   3   2   0   1   2   0   2
                   $A ,
        COPYRIGHT (C) EMULOGIC INC. 1981

********** INTEL INTELLIC FORMAT **********

        H31HEBADC2HEBRS/
          1  50   2   1   2   1   2   1   3   0   1   1   0   2
         0     :         00
        COPYRIGHT (C) EMULOGIC INC. 1981

********** MOS TECHNOLOGY FORMAT **********

        1HSBADC2HSBRC/
          1  20   1   1   2   2   4   1   1   0   1   0   0   2
         ;
        COPYRIGHT (C) EMULOGIC INC. 1981

********** MOTOROLA DATA FILES **********

        1SBADC2EFBRC/
          0   0   2   4   2   3   2   1   1   0   1   1   1   2
         S1 9  S
        COPYRIGHT (C) EMULOGIC INC. 1981

********** TEKTRONIX HEX FORMAT **********

        1SABCDC2SRBC/
          0   0   1   1   2   4   2   1   1   0   1   0   0   2
         /
        COPYRIGHT (C) EMULOGIC INC. 1981

********** RCA COSMAC FORMAT **********

        1SADE2SRF/
          0   0   2   0   2   0   0   1   2   0   1   1   1   2
         !M    ;      !M
        COPYRIGHT (C) EMULOGIC INC 1981
```