Generating Algorithm with SUPERPRO

Algorithm Generator
Device Control Language
Library Generator

XELTEK

Copyrights

764 San Aleso Ave. Sunnyvale, CA 94086
TEL(408)745-7974 FAX(408)745-1401

CONTENTS

BOOK I Algorithm Generator

BOOK I

AG (Algorithm Generator)

This will provide the environment for generating files for algorithms. A file of DCL (Device Control Language) will be written and compiled in AG environment. Book II will explaine the Syntax of DCL.

BOOK II Device Control Language

BOOK III Library Generator

1 FILE

In this menu, there are seven submenus which will deal with file management and the interface with DOS.

Submenus

Load	Loads a file from disk
Save	Saves a file from the buffer of the editor onto disk.
Save Other Name	Saves the content of the buffer into a different name onto disk.
New	Clears the buffer and starts a new buffer.
Directory	Lists files in a specified directory.
DOS	Lets the user execute the DOS command without exiting the software of Superpro.
Quit	Exits the software.

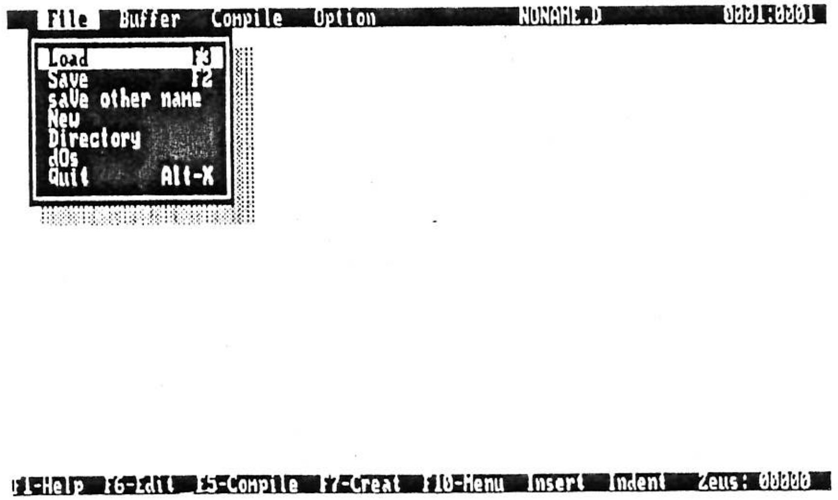


Figure 1.1 File

1.1 Load

This will load the text file from disk. This is not loading any data file for programming. The file name can be as long as 40 characters. The wildcards such as * and ? can be used, and in that case the files in that range will show up. An entry among those files can be highlighted and selected by pressing <return>.

After loading, the users can start editing right away. If users try to load another file when there's a file already in the buffer, a message asking whether the existing file to be saved or not will be displayed.

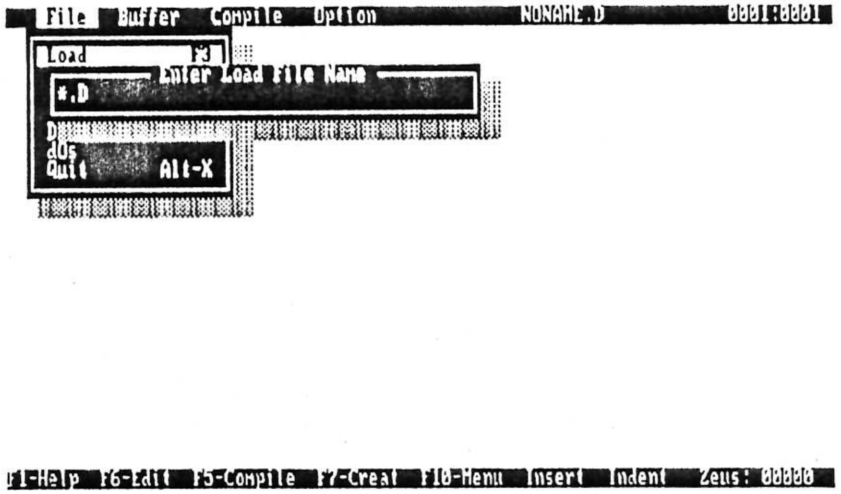

File Buffer Compile Option NONAME.D 0001:0001
Load F3
Enter Load File Name
*.D
D
dOs
Quit Alt-X
F1-Help F6-Edit F5-Compile F7-Creat F10-Menu Insert Indent Zeus: 00000

Figure 1.2 Loading Algorithm

1.2 Save

This will save the content of the buffer with the name it was called. If there isn't any name given "NONAME.D" will be given, and users will be given a chance to change it to an appropriate name. Please be cautious not to overwrite.

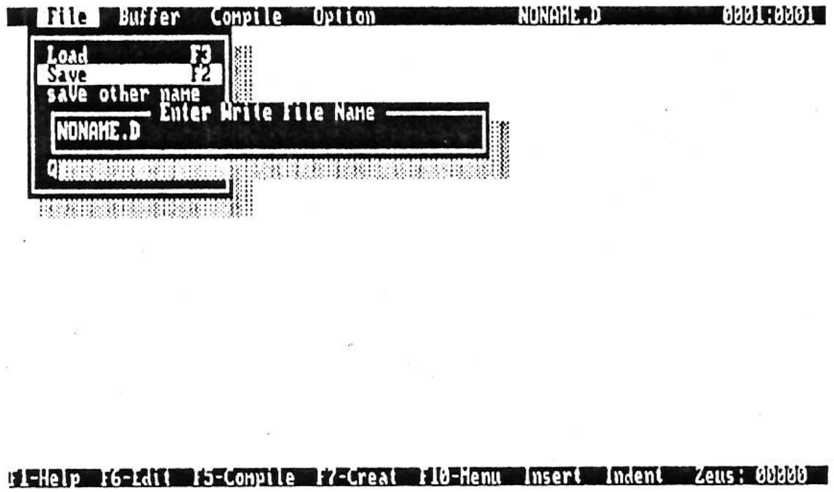

File Buffer Compile Option NONAME.D 0001:0001
Load F3
Save F2
save other name
Enter Write File Name
NONAME.D
Q
F1-Help F6-Edit F5-Compile F7-Creat F10-Menu Insert Indent Zeus: 00000

Figure 1.3 Save

1.3 Save other name

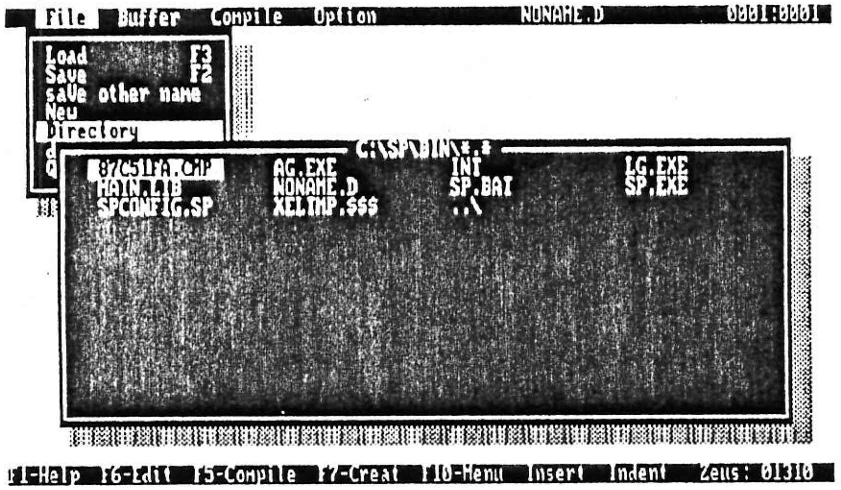
It saves a file in the buffer with a new or different name. Users will be asked for a name.

1.4 New

This deals with the buffer and prepares for the new file. When this menu is invoked if there is a file already in the buffer, the software will ask whether it should save the existing file or not. The default name for all unnamed files is " NONAME. D"

1.5 Directory

A path for a directory will be input and the files in that directory will be listed in the screen. If you don't type any character and press <return> the files in the current directory will be displayed.

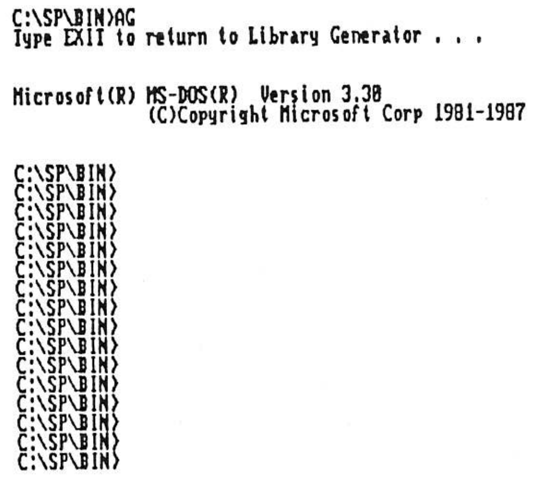


```
 File   Buffer   Compile   Option              NONAME.D              0001:0001
  Load            F3
  Save            F2
  saVe other name
  New
  Directory
                     C:\SP\BIN\*.*
   87C51FA.CHP     AG.EXE        INT              LG.EXE
   MAIN.LIB        NONAME.D      SP.BAT           SP.EXE
   SPCONFIG.SP     XELTMP.$$$    ..\

F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

Figure 1.4 Directory

1.6 DOS

Without exiting the software users can execute the DOS commands. To return to the software, please type "EXIT" on DOS.



```
C:\SP\BIN>AG
Type EXIT to return to Library Generator . . .

Microsoft(R) MS-DOS(R)  Version 3.30
             (C)Copyright Microsoft Corp 1981-1987

C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
C:\SP\BIN>
```

Figure 1.5 DOS

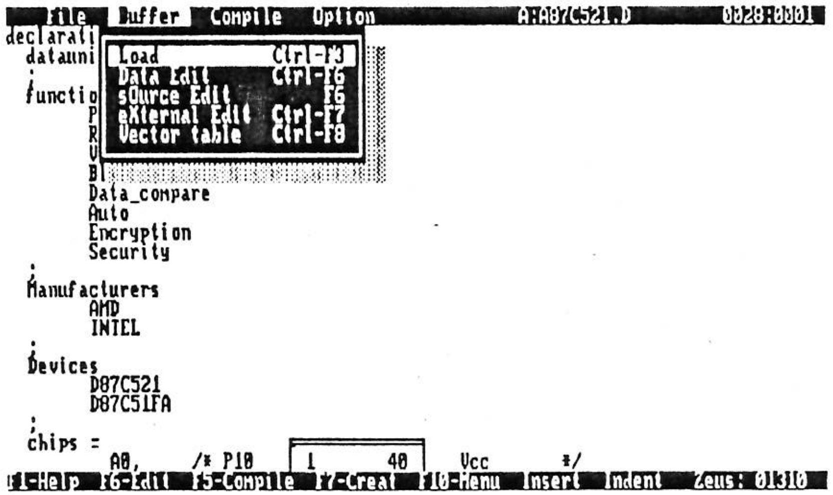
1.7 Quit

It exits the software and returns the software to DOS. When the current file isn't saved the software will ask whether the file should be saved or not. Before exiting, the software will save all the setups and selections in a file called "lopsconf. ag".

2 BUFFER

This menu will manage the all kinds of data such as JEDEC fuse maps, and HEX files.

Load	Loads the input file for programming
Data Edit	Data will be viewed and edited.
	Caution : If "compile_Default_form" is set "PLD" a screen for bit map will be displayed, and if "compile_Default_form" is set "Rom" then the buffer for HEX files will be displayed.
Source Edit	DCL (Device Control Language)
External Edit	This is used for GAL and Single chips. For GAL this is used for MES and UES read. For microcontrollers, it is used for encrypting tables.
Vector Table	This will read a vector file and edit and test the vectors for PLD's.



```
  File   Buffer   Compile   Option                 A:A87C521.D            0028:0001
declarati ┌──────────────────────────┐
  datauni │ Load            Ctrl-F3  │
   ;      │ Data Edit       Ctrl-F6  │
  functio │ sOurce Edit          F6  │
         P│ eXternal Edit   Ctrl-F7  │
         R│ Vector table    Ctrl-F8  │
         V└──────────────────────────┘
         B
         Data_compare
         Auto
         Encryption
         Security
  ;
  Manufacturers
         AMD
         INTEL
  ;
  Devices
         D87C521
         D87C51FA
  ;
  chips =
            A0,      /* P10  │ 1        40 │   Vcc        */
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

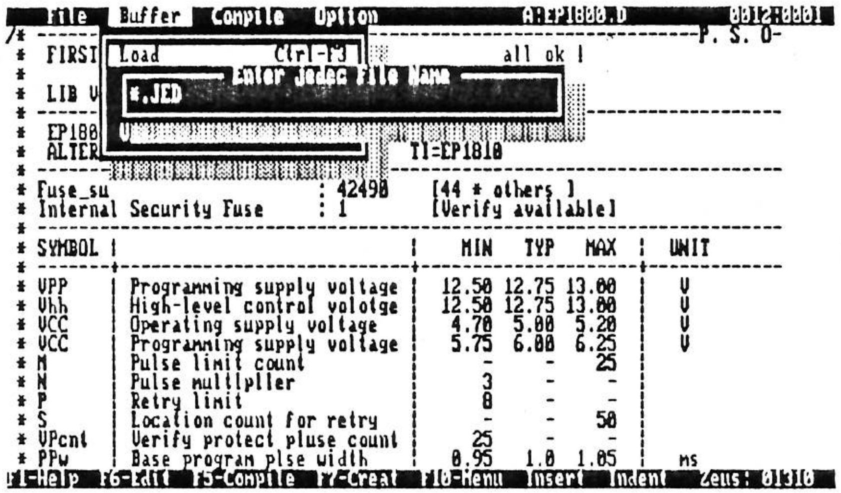
Figure 2.1 Buffer

2.1 Load

This will load binary files and JEDEC files. When this menu is selected a box for a file name will be provided. After a data file is loaded. The data will stay in the buffer unless other files are invoked, or a device is read.

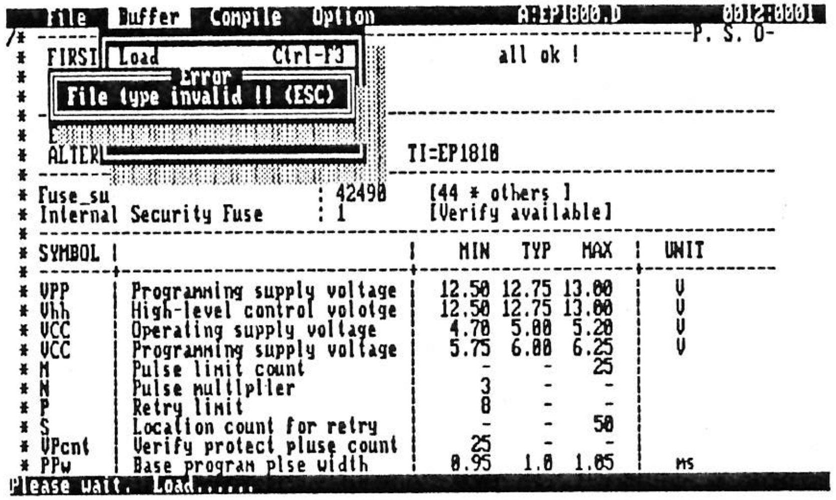
If "compile-Default_form" is set for "PLD" then files with the suffix of "JED" will be selected. If there is no file, an error message will be displayed. Users can type the exact name to select. Or users can press <return> to list all the files and highlight to select for loading.

If "compile->Default_form" is set for "ROM", then a box for HEX file name will be displayed. For loading HEX files the size for the buffer is 64K byte.



```
  File  Buffer  Compile  Option                 A:EP1800.D          0012:0001
/* -------                                                     --------P. S. O-
*  FIRST  Load          Ctrl-F3                  all ok !
*               Enter Jedec File Name
*  LIB V    *.JED
*  -------                                     ------------------------------
*  EP180  U
*  ALTER                               TI=EP1810
*  ------------------------------------------------------------------------
* Fuse_su                          : 42490     [44 * others ]
* Internal Security Fuse           : 1         [Verify available]
* ------------------------------------------------------------------------
* SYMBOL |                                |   MIN   TYP   MAX  |  UNIT
* -------+--------------------------------+--------------------+----------
* VPP    | Programming supply voltage     | 12.50 12.75 13.00  |   V
* Vhh    | High-level control volotge     | 12.50 12.75 13.00  |   V
* VCC    | Operating supply voltage       |  4.70  5.00  5.20  |   V
* VCC    | Programming supply voltage     |  5.75  6.00  6.25  |   V
* M      | Pulse limit count              |    -     -    25   |
* N      | Pulse multlplier               |    3     -     -   |
* P      | Retry limit                    |    8     -     -   |
* S      | Location count for retry       |    -     -    50   |
* VPcnt  | Verify protect pluse count     |   25     -     -   |
* PPw    | Base program plse width        |  0.95   1.0  1.05  |   ms
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

Figure 2.2 Bufler Load



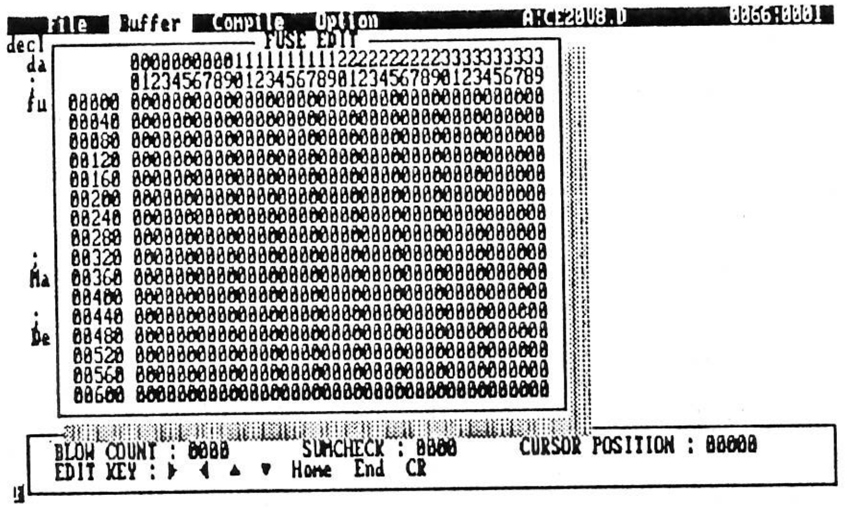
```
  File  Buffer  Compile  Option                 A:EP1800.D          0012:0001
/* -------                                                     --------P. S. O-
*  FIRST  Load          Ctrl-F3                  all ok !
*          Error
*   File type invalid !! (ESC)
*  -                                           ------------------------------
*  E
*  ALTER                               TI=EP1810
*  ------------------------------------------------------------------------
* Fuse_su                          : 42490     [44 * others ]
* Internal Security Fuse           : 1         [Verify available]
* ------------------------------------------------------------------------
* SYMBOL |                                |   MIN   TYP   MAX  |  UNIT
* -------+--------------------------------+--------------------+----------
* VPP    | Programming supply voltage     | 12.50 12.75 13.00  |   V
* Vhh    | High-level control volotge     | 12.50 12.75 13.00  |   V
* VCC    | Operating supply voltage       |  4.70  5.00  5.20  |   V
* VCC    | Programming supply voltage     |  5.75  6.00  6.25  |   V
* M      | Pulse limit count              |    -     -    25   |
* N      | Pulse multlplier               |    3     -     -   |
* P      | Retry limit                    |    8     -     -   |
* S      | Location count for retry       |    -     -    50   |
* VPcnt  | Verify protect pluse count     |   25     -     -   |
* PPw    | Base program plse width        |  0.95   1.0  1.05  |   ms
Please wait.  Load......
```

Figure 2.3 Error when loading JEDEC file

2.2 Data Edit

After loading a data file from disk or reading from a chip, data will be stored in the buffer. The data will be viewed and edited. Depending on the mode selected in "compile->Default_form" different buffers will show up.

If "ROM" is chosen, the buffer for the JEDEC fuse map file will be displayed. With "PLA" chosen, a screen as below will be displayed. There are three numbers displayed in the lower box. The first number is the number of fuse blown and the second is the checksum, and the last number is the position of the cursor in the buffer.



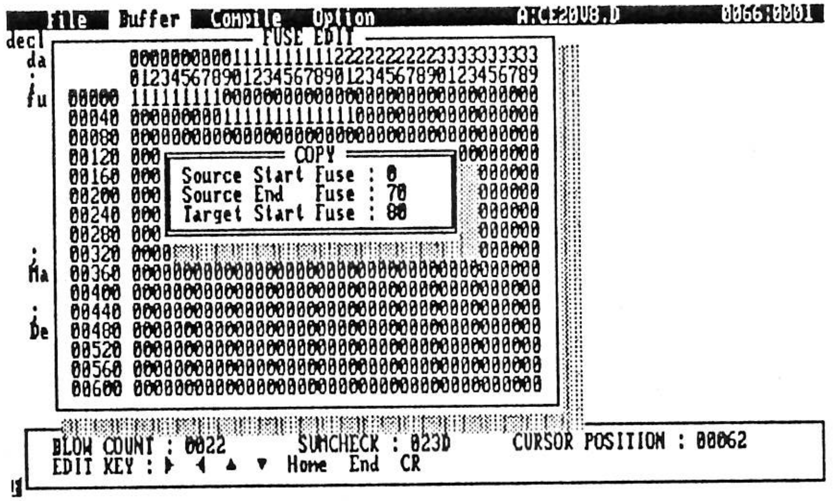
```
 file  Buffer  Compile  Option                    A:CE20U8.D            0066:0001
decl ─────────────────────────── FUSE EDIT ───────────────────────────
 da               0000000000111111111122222222223333333333
 :                0123456789012345678901234567890123456789
 fu    00000      0000000000000000000000000000000000000000
       00040      0000000000000000000000000000000000000000
       00080      0000000000000000000000000000000000000000
       00120      0000000000000000000000000000000000000000
       00160      0000000000000000000000000000000000000000
       00200      0000000000000000000000000000000000000000
       00240      0000000000000000000000000000000000000000
       00280      0000000000000000000000000000000000000000
 :     00320      0000000000000000000000000000000000000000
 Ma    00360      0000000000000000000000000000000000000000
       00400      0000000000000000000000000000000000000000
 :     00440      0000000000000000000000000000000000000000
 De    00480      0000000000000000000000000000000000000000
       00520      0000000000000000000000000000000000000000
       00560      0000000000000000000000000000000000000000
       00600      0000000000000000000000000000000000000000

   BLOW COUNT : 0000          SUMCHECK : 0000        CURSOR POSITION : 00000
   EDIT KEY : ▶  ◀  ▲  ▼  Home  End  CR
```

Figure 2.4 Bit Edit (PAL/GAL)

The keys for edition are as follows. (PAL,GAL)

Arrow keys	Moves up, down, left, and right
1	Input logic high
0	Input logic low
Home	To the left most position on the current line
End	To the right most position on the current line
PgUp	To the next page
PgDn	To the previous page
CR	To the first column of the next line
Ctrl-R	Copies a block and transfers it to a desired address
Ctrl-N	To the last page of the buffer allocated
Ctrl-PgUp	To the first page

Ctrl-Home	To the first column of the first row in the current screen
Ctrl-End	To the last column of the last row in the current screen
Esc	Exits from the editor screen



```
 file  Buffer  Compile  Option                    A:CE20U8.D            0066:0001
decl ─────────────────────────── FUSE EDIT ───────────────────────────
 da               0000000000111111111122222222223333333333
 :                0123456789012345678901234567890123456789
 fu    00000      1111111100000000000000000000000000000000
       00040      0000000001111111111110000000000000000000
       00080      0000000000000000000000000000000000000000
       00120      000 ═══════════ COPY ═══════════ 00000000
       00160      000  Source Start Fuse : 0         000000
       00200      000  Source End   Fuse : 70        000000
       00240      000  Target Start Fuse : 80        000000
       00280      000                                000000
 :     00320      0000                               000000
 Ma    00360      0000000000000000000000000000000000000000
       00400      0000000000000000000000000000000000000000
 :     00440      0000000000000000000000000000000000000000
 De    00480      0000000000000000000000000000000000000000
       00520      0000000000000000000000000000000000000000
       00560      0000000000000000000000000000000000000000
       00600      0000000000000000000000000000000000000000

   BLOW COUNT : 0022          SUMCHECK : 023D        CURSOR POSITION : 00062
   EDIT KEY : ▶  ◀  ▲  ▼  Home  End  CR
```

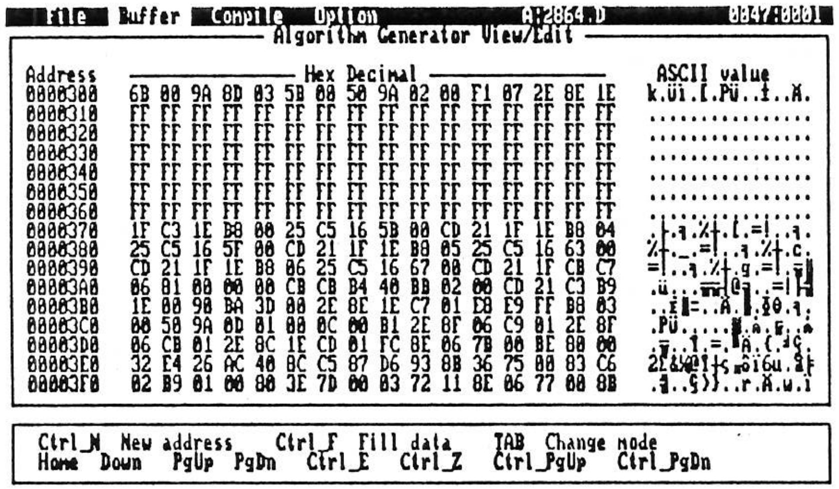
Figure 2.5 Ctrl-R key (PAL/GAL)

If "ROM" is chosen in "compile-Default_form" the data will be presented both as hexadecimal and ASCII values.

The keys for edition are as follows.

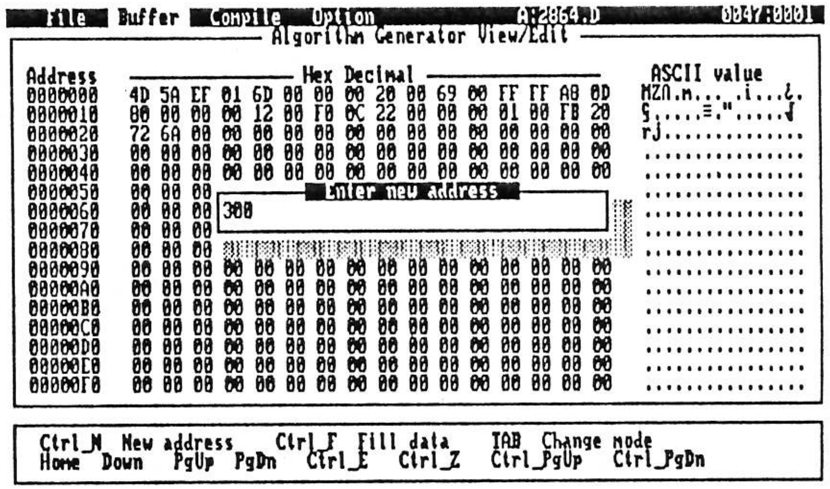
Arrow key	Moves to the left, right, up, and down
Ctrl-N	Go to the new address and display.
TAB	Toggles the cursor between Hexadecimal area and ASCII area.
Home	To the first column of the current line.
End	To the last column of the current line.

PgUp	To the previous page
PgDn	To the next page
Ctrl-F	Fills the data with the specified value
Ctrl-E	Scroll up one line
Ctrl-Z	Scroll down one l ne
Ctrl-PgDn	To the last page
Ctrl-PgUp	To the first page
Ctrl-Home	To the first colum.ı of the first row of the screen
Ctrl-End	To the last colum ı of the last line of the current screen.
ESC	Exits the data buf'er screen.



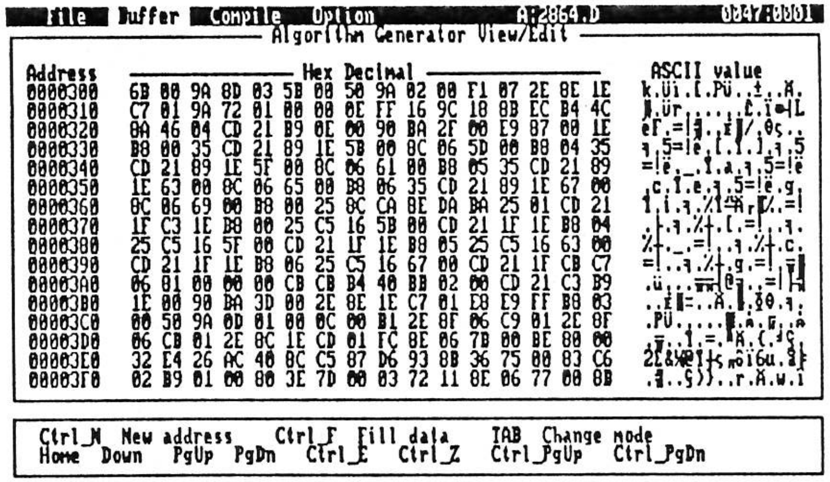
```
  File  Buffer  Compile  Option                    A:2864.D              0047:0001
 ─────────────────────── Algorithm Generator View/Edit ──────────────────────────
 Address   ─────────────── Hex Decimal ──────────────────        ASCII value
 0000300   6B 00 9A 8D 03 5B 00 50 9A 02 00 F1 07 2E 8E 1E      k.Üi.[.PÜ..±..Ä.
 0000310   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      ................
 0000320   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      ................
 0000330   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      ................
 0000340   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      ................
 0000350   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      ................
 0000360   FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      ................
 0000370   1F C3 1E B8 00 25 C5 16 5B 00 CD 21 1F 1E B8 04      .├.╕.%┼.[.=!..╕.
 0000380   25 C5 16 5F 00 CD 21 1F 1E B8 05 25 C5 16 63 00      %┼._.=!..╕.%┼.c.
 0000390   CD 21 1F 1E B8 06 25 C5 16 67 00 CD 21 1F CB C7      =!..╕.%┼.g.=!.╦╟
 00003A0   06 81 00 00 00 CB CB B4 40 BB 02 00 CD 21 C3 B9      .ü...╦╦┤@╗..=!├╣
 00003B0   1E 00 90 BA 3D 00 2E 8E 1E C7 01 E8 E9 FF B8 03      ..É║=..Ä.╟.Φθ.╕.
 00003C0   00 50 9A 0D 01 00 0C 00 B1 2E 8F 06 C9 01 2E 8F      .PÜ.....▒.Å.╔..Å
 00003D0   06 CB 01 2E 8C 1E CD 01 FC 8E 06 7B 00 BE 80 00      .╦..î.=.ⁿÄ.{.╛Ç.
 00003E0   32 E4 26 AC 40 8C C5 87 D6 93 8B 36 75 00 83 C6      2Σ&¼@î┼çôï6u.â╞
 00003F0   02 B9 01 00 80 3E 7D 00 03 72 11 8E 06 77 00 8B      .╣..Ç>}..r.Ä.w.ï

  Ctrl_N  New address    Ctrl_F  Fill data    TAB  Change mode
  Home  Down   PgUp  PgDn   Ctrl_E   Ctrl_Z   Ctrl_PgUp   Ctrl_PgDn
```

Figure 2.6 Edit Screen for ROM & Single



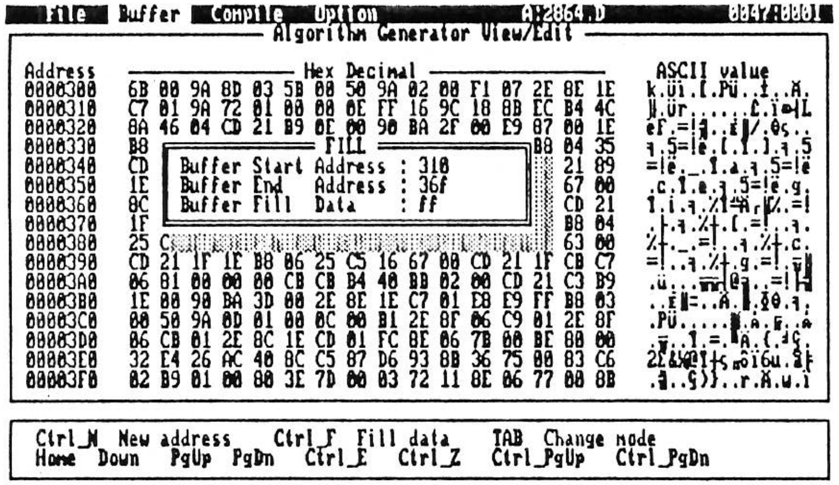
```
  File  Buffer  Compile  Option                    A:2864.D              0047:0001
 ─────────────────────── Algorithm Generator View/Edit ──────────────────────────
 Address   ─────────────── Hex Decimal ──────────────────        ASCII value
 0000000   4D 5A EF 01 6D 00 00 00 20 00 69 00 FF FF A8 0D      MZ∩.m... .i...¿.
 0000010   00 00 00 00 12 00 F0 0C 22 00 00 00 01 00 FB 20      §.....≡."....√ 
 0000020   72 6A 00 00 00 00 00 00 00 00 00 00 00 00 00 00      rj..............
 0000030   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 0000040   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 0000050   00 00 00 ┌──────── Enter new address ────────┐      ................
 0000060   00 00 00 │300                                 │      ................
 0000070   00 00 00 └────────────────────────────────────┘      ................
 0000080   00 00 00                                             ................
 0000090   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 00000A0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 00000B0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 00000C0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 00000D0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 00000E0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................
 00000F0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00      ................

  Ctrl_N  New address    Ctrl_F  Fill data    TAB  Change mode
  Home  Down   PgUp  PgDn   Ctrl_E   Ctrl_Z   Ctrl_PgUp   Ctrl_PgDn
```

Figure 2.7 Ctrl-N in Edit Screen



```
  File  Buffer  Compile  Option                    A:2864.D              0047:0001
 ─────────────────────── Algorithm Generator View/Edit ──────────────────────────
 Address   ─────────────── Hex Decimal ──────────────────        ASCII value
 0000300   6B 00 9A 8D 03 5B 00 50 9A 02 00 F1 07 2E 8E 1E      k.Üi.[.PÜ..±..Ä.
 0000310   C7 01 9A 72 01 00 00 0E FF 16 9C 18 8B EC B4 4C      ╟.Ür.......£.ï∞┤L
 0000320   8A 46 04 CD 21 B9 0E 00 90 BA 2F 00 E9 87 00 1E      èF.=!╣..É║/.Θç..
 0000330   B8 00 35 CD 21 89 1E 5B 00 8C 06 5D 00 B8 04 35      ╕.5=!ë.[.î.].╕.5
 0000340   CD 21 89 1E 5F 00 8C 06 61 00 B8 05 35 CD 21 89      =!ë._.î.a.╕.5=!ë
 0000350   1E 63 00 8C 06 65 00 B8 06 35 CD 21 89 1E 67 00      .c.î.e.╕.5=!ë.g.
 0000360   8C 06 69 00 B8 00 25 8C CA 8E DA BA 25 01 CD 21      î.i.╕.%î╩Ä┌║%.=!
 0000370   1F C3 1E B8 00 25 C5 16 5B 00 CD 21 1F 1E B8 04      .├.╕.%┼.[.=!..╕.
 0000380   25 C5 16 5F 00 CD 21 1F 1E B8 05 25 C5 16 63 00      %┼._.=!..╕.%┼.c.
 0000390   CD 21 1F 1E B8 06 25 C5 16 67 00 CD 21 1F CB C7      =!..╕.%┼.g.=!.╦╟
 00003A0   06 81 00 00 00 CB CB B4 40 BB 02 00 CD 21 C3 B9      .ü...╦╦┤@╗..=!├╣
 00003B0   1E 00 90 BA 3D 00 2E 8E 1E C7 01 E8 E9 FF B8 03      ..É║=..Ä.╟.Φθ.╕.
 00003C0   00 50 9A 0D 01 00 0C 00 B1 2E 8F 06 C9 01 2E 8F      .PÜ.....▒.Å.╔..Å
 00003D0   06 CB 01 2E 8C 1E CD 01 FC 8E 06 7B 00 BE 80 00      .╦..î.=.ⁿÄ.{.╛Ç.
 00003E0   32 E4 26 AC 40 8C C5 87 D6 93 8B 36 75 00 83 C6      2Σ&¼@î┼çôï6u.â╞
 00003F0   02 B9 01 00 80 3E 7D 00 03 72 11 8E 06 77 00 8B      .╣..Ç>}..r.Ä.w.ï

  Ctrl_N  New address    Ctrl_F  Fill data    TAB  Change mode
  Home  Down   PgUp  PgDn   Ctrl_E   Ctrl_Z   Ctrl_PgUp   Ctrl_PgDn
```

Figure 2.8 Address Shift by Ctrl-N

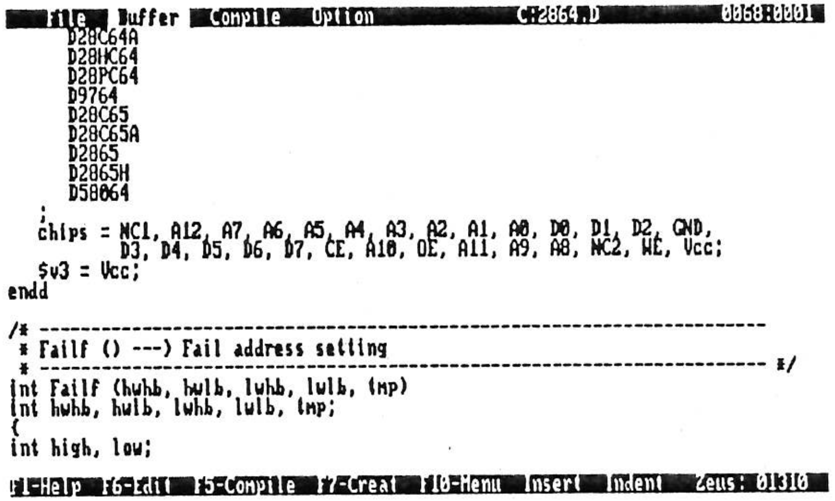


```
 File  Buffer  Compile  Option              A:2864.D              0047:0001
─────────────────── Algorithm Generator View/Edit ──────────────────────

 Address    ─────────────── Hex Decimal ──────────────       ASCII value
 0000300    6B 00 9A 8D 03 5B 00 50 9A 02 00 F1 07 2E 8E 1E   k.üi.[.Pü..±..Ä.
 0000310    C7 01 9A 72 01 00 00 0E FF 16 9C 18 8B EC B4 4C   ╟.ür......£.ï─┤L
 0000320    8A 46 04 CD 21 B9 0E 00 90 BA 2F 00 E9 87 00 1E   èF.=!╣..É║/.Θç..
 0000330    B8  ═══════════════ FILL ═════════════════ B8 04 35   ╕.5=!é.[.╕.].╕.5
 0000340    CD  │ Buffer Start Address : 310          │ 21 89   =!ë._.╕.a.╕.5=!ë
 0000350    1E  │ Buffer End   Address : 36f          │ 67 00   .c.╕.e.╕.5=!ë.g.
 0000360    8C  │ Buffer Fill  Data    : ff           │ CD 21   ╕.i.╕.%╕╪Ä.╓%.=!
 0000370    1F  └─────────────────────────────────────┘ B8 04   .├.╕.%┼.[.=!..╕.
 0000380    25 C                                        63 00   %┼._.=!..╕.%┼.c.
 0000390    CD 21 1F 1E B8 06 25 C5 16 67 00 CD 21 1F CB C7   =!..╕.%┼.g.=!.╤╟
 00003A0    06 81 00 00 00 CB CB B4 40 BB 02 00 CD 21 C3 B9   .ü...╤╤┤@╗..=!├╣
 00003B0    1E 00 98 BA 3D 00 2E 8E 1E C7 01 E8 E9 FF B8 03   ..ÿ║=..Ä.╟.ΦΘ.╕.
 00003C0    00 50 9A 0D 01 00 0C 00 B1 2E 8F 06 C9 01 2E 8F   .Pü.......▒.Å.╔..Å
 00003D0    06 CB 01 2E 8C 1E CD 01 FC 8E 06 7B 00 BE 00 00   .╤..î.=.ⁿÄ.{.╛..
 00003E0    32 E4 26 AC 40 8C C5 87 D6 93 8B 36 75 00 83 C6   2Σ&¼@î┼ç╓ôï6u.â╞
 00003F0    02 B9 01 00 00 3E 7D 00 03 72 11 8E 06 77 00 8B   .╣..>}..r.Ä.w.ï

 Ctrl_N  New address    Ctrl_F  Fill data     TAB  Change mode
 Home  Down   PgUp  PgDn   Ctrl_E    Ctrl_Z   Ctrl_PgUp    Ctrl_PgDn
```

Figure 2.9 Ctrl-F

2.3 Source Edit

This is used when a user writes an algorithm using DCL (Device Control Language). After the "Source Edit" is selected a screen such as below will be ready. The editor keys are explained below.



```
 File  Buffer  Compile  Option              C:2864.D              0068:0001
          D28C64A
          D28HC64
          D28PC64
          D9764
          D28C65
          D28C65A
          D2865
          D2865H
          D58064
      ;
      chips = NC1, A12, A7, A6, A5, A4, A3, A2, A1, A0, D0, D1, D2, GND,
              D3, D4, D5, D6, D7, CE, A10, OE, A11, A9, A8, NC2, WE, Vcc;
      $v3 = Vcc;
  endd

  /* ---------------------------------------------------------------
   * Failf () ---) Fail address setting
   * --------------------------------------------------------------- */
  int Failf (hwhb, hwlb, lwhb, lwlb, tmp)
  int hwhb, hwlb, lwhb, lwlb, tmp;
  {
  int high, low;
 F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

Figure 2.10 Source Edit

Moving Cursor

Arrow Keys	Move Up,Down, Left, ar.d Right
Home	To the first column of the current line
End	To the last column of the current line
PgUp	To the previous page
PgDn	To the next page
Ctrl-w	Scroll down one line
Ctrl-Z	Scroll up one line
Ctrl-F	Scroll one column to the left
Ctrl-G	Scroll one column to the right
Ctrl <-	Scroll one word to the left

Ctrl ->	Scroll one word to the right
Ctrl-PgUp	To the first page of the file
Ctrl-PgDn	To the first line of the screen
Ctrl-End	To the last line of the screen
Ctrl-QP	To the previous position of the cursor

Insertion & Deletion

Ins	Turn on / off the mode of insertion
Enter	Insert one line
Ctrl-N	One row will be created in front of and the back of the cursor
Ctrl-Y	Delete a row
Ctrl-H	Delete one character
Del	Delete one character
<-	Delete one character
Ctrl-T	Delete one word
Ctrl-QY	Delete a line after the cursor

Block Command

Ctrl-KB	Indicates the beginning of the block
Ctrl-KK	Indicates the end of the block
Ctrl-KC	Copies the block indicated
Ctrl-KV	Moves the block selected
Ctrl-KY	Deletes the block selected
Ctrl-KH	Deselects the block selected
Ctrl-KR	Reads the block
Ctrl-KW	Writes a file with the block selected

Searching and Replacing

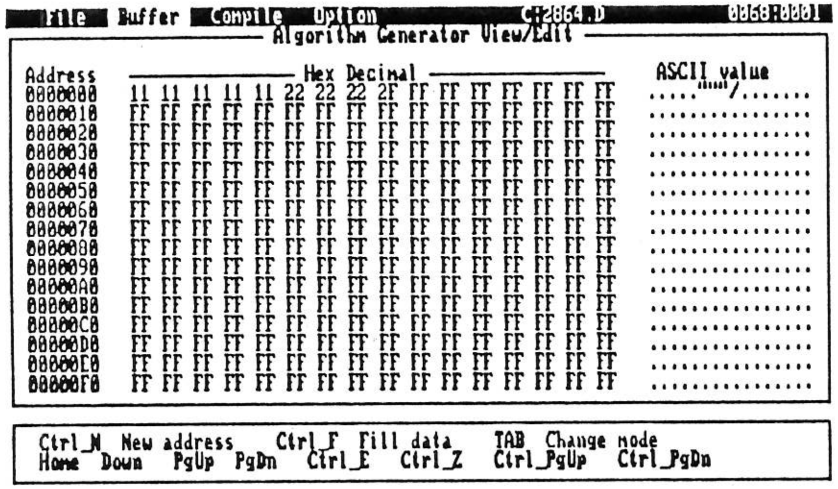
Ctrl-QF	Searches for one staring
Ctrl-QA	Searches for the string and replaces it with the correct one
Ctrl-L	Repeats the command

Option for Searching and Replacing

G	If this is selected, searching and replacing will be alone continuously
B	Searches and replaces backward
W	Searches only for the exact string
U	Case sensitive search
N	Replaces without asking
? or *	Wild cards

2.4 External Edit

This will manage the Encryption table for 87 series Microcontrollers and MES for GAL. The buffer for HEX value and ASCII field will show up. The editor keys are same with the keys used for Hex input file.



```
 File  Buffer  Compile  Option              C:2864.D                0068:0001
 ────────────────────── Algorithm Generator View/Edit ──────────────────────
 Address    ──────────── Hex Decimal ────────────────        ASCII value
 0000000    11 11 11 11 11 22 22 22 2F FF FF FF FF FF FF FF   .....""""/.......
 0000010    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000020    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000030    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000040    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000050    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000060    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000070    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000080    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 0000090    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 00000A0    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 00000B0    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 00000C0    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 00000D0    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 00000E0    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................
 00000F0    FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ................

  Ctrl_N  New address    Ctrl_F  Fill data    TAB  Change mode
  Home  Down   PgUp  PgDn   Ctrl_E    Ctrl_Z   Ctrl_PgUp   Ctrl_PgDn
```

Figure 2.11 External Edit

MES (Manufacturer's Electronic Signature) contains information about the device. Therefore, MES will not be edited in any case but viewed as necessary.

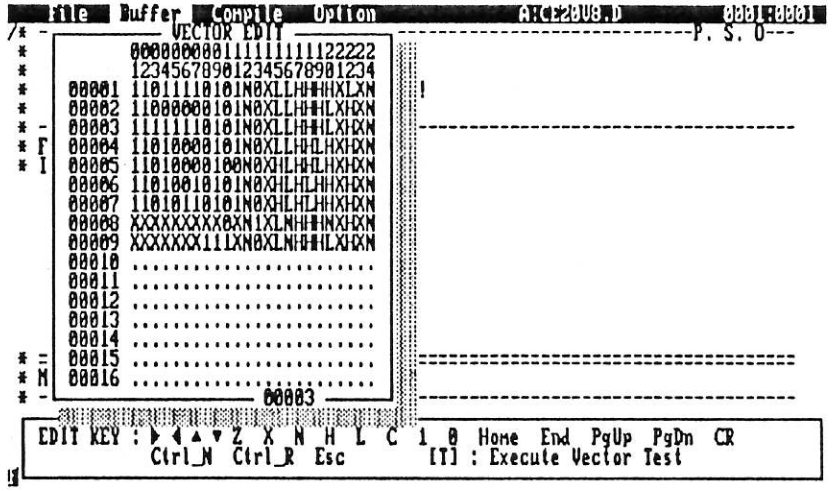
MES (for GAL) contains information as below.

Cycle counter	Number of programs done for a device
Algorithm revision	History of algorithm revision
Master Bit	1: master, 0: non-master If the device is a master device, the error message will be displayed when users try to program or erase the master chip.
Device code	8 bit device code (for example : RAL/8H4 [code=27])

When dealing with single chips, users can load the content of encryption buffer in the external buffer and download it into the device.

2.5 Vector table

This loads a vector table in JEDEC files for editing or testing. If this menu is selected, the buffer will be displayed as below. Pressing T will initiate the vector testing.



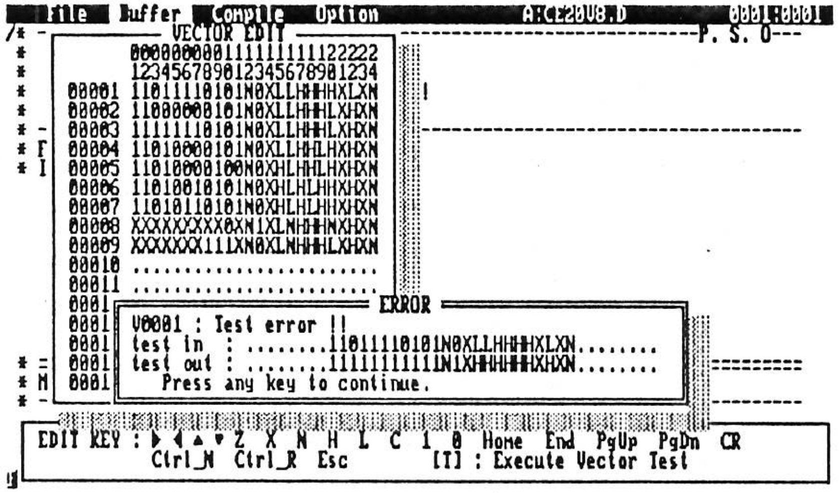
```
 File  Buffer  Compile  Option                     A:CE20V8.D          0001:0001
/* - ────── VECTOR EDIT ──────  ------------------------------P. S. O---
 *              000000000111111111122222
 *              123456789012345678901234
 *      00001   11011110101N0XLLHHHHXLXN      !
 *      00002   11000000101N0XLLHHHLXHXN
 * -    00003   11111110101N0XLLHHHLXHXN      ------------------------------------
 * F    00004   11010000101N0XLLHHLHXHXN
 * I    00005   11010000100N0XHLHHLHXHXN
        00006   11010010101N0XHLHLHHXHXN
        00007   11010110101N0XHLHLHHXHXN
        00008   XXXXXXXXX0XN1XLNHHHNXHXN
        00009   XXXXXXX111XN0XLNHHHLXHXN
        00010   ........................
        00011   ........................
        00012   ........................
        00013   ........................
        00014   ........................
 * =    00015   ........................      ====================================
 * M    00016   ........................
 * -  ──────────────── 00003 ──────────       ------------------------------------
  EDIT KEY : ► ◄ ▲ ▼ Z  X  N  H  L  C  1  0  Home  End  PgUp  PgDn  CR
                 Ctrl_N  Ctrl_R  Esc        [T] : Execute Vector Test
```

Figure 2.12 Screen for vector table

Editor keys

Arrow keys	Moves up, down, left and right
Z	Test for high impedance
X	Don't care term
N	Used for power pin (output will not be tested. Ex: VCC, GND)
H	Logic level high

L	Logic level low
C	Clock pin
1	Input logic high
0	Input logic low
Home	To the first row of the current line
End	To the last row of the current line
PgUp	To the previous screen
PgDn	To the next screen
Ctrl-R	Copies vector into the desired location
Ctrl-N	Jumps to the indicated address.
Ctrl-PgDn	To the last page of the file
Ctrl-PgUp	To the first page of the file
Ctrl-Home	To the first column of the first row of the screen
Ctrl-End	To the last column of the last line of the screen
ESC	Exits the screen for edition



```
  File  Buffer  Compile  Option               A:CE20V8.D            0001:0001
/* - ------ VECTOR EDIT -------  ----------------------------P. S. 0---
*           000000000111111111122222
*           123456789012345678901234
*     00001 110111101 01N0XLLHHHHXLXN    !
*     00002 11000000101N0XLLHHHLXHXN
* -   00003 11111110101N0XLLHHHLXHXN   -----------------------------------
* F   00004 11010000101N0XLLHHLHXHXN
* I   00005 11010000100N0XHLHHLHXHXN
      00006 11010010101N0XHLHLHHXHXN
      00007 11010110101N0XHLHLHHXHXN
      00008 XXXXXXXXX0XN1XLNHHHNXHXN
      00009 XXXXXXX111XN0XLNHHHLXHXN
      00010 ........................
      00011 ........................
      0001 ===================== ERROR ==============================
      0001  V0001 : Test error !!
      0001  test in  : ........110111101 01N0XLLHHHHXLXN........
* =   0001  test out : ........11111111111N1XHHHHHHXHXN........   =========
* M   0001     Press any key to continue.
* -                                                                 ---------
 EDIT KEY : ▶ ◀ ▲ ▼ Z  X  N  H  L  C  1  0  Home  End  PgUp  PgDn  CR
                Ctrl_N  Ctrl_R  Esc        [T] : Execute Vector Test
```

Figure 2.13 Vector Testing (Press T)

3 COMPILE

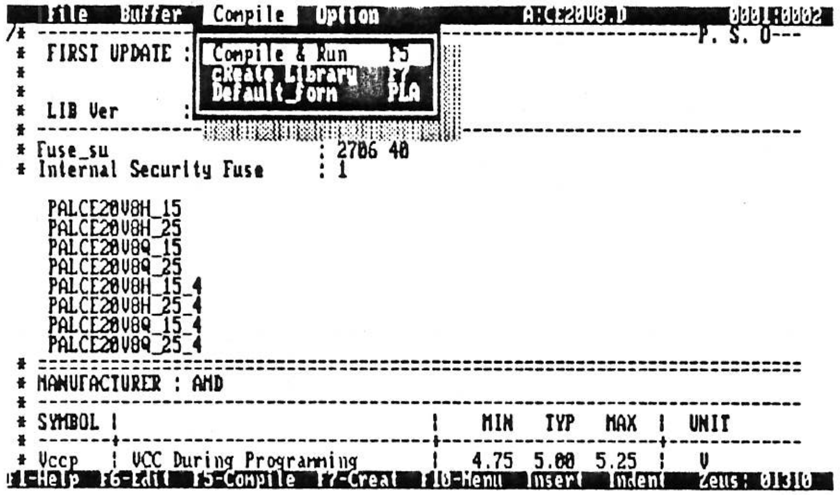
This big Menu has three submenus as below.

Compile & Run	Compiles the file of an algorithm written with DCL (Device control language)
Create Library	Produces the object file with the suffix of "lef" to be recognized by the user interface software "SP.EXE".
Default - form	Determines whether the chip is a PLD or a memory chip

3.1 Compile & Run

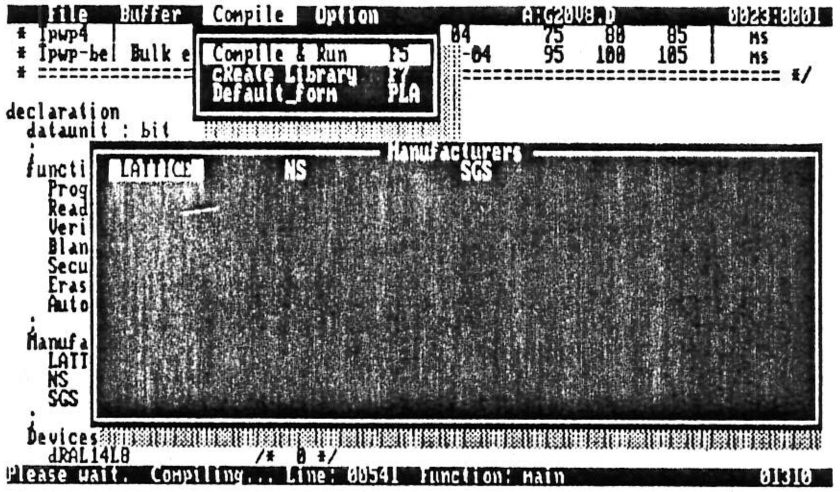
This compiles the DCL source file with the suffix of "d". The compilation is done with the file currently loaded in the source buffer. If there is an error, the error message will be displayed with the line number.

If the compilation is successful, the display of manufacturer will show up. Then a user can select the manufacturer of interest. Automatically the part list of the manufacturer will be displayed as shown in the figure below.



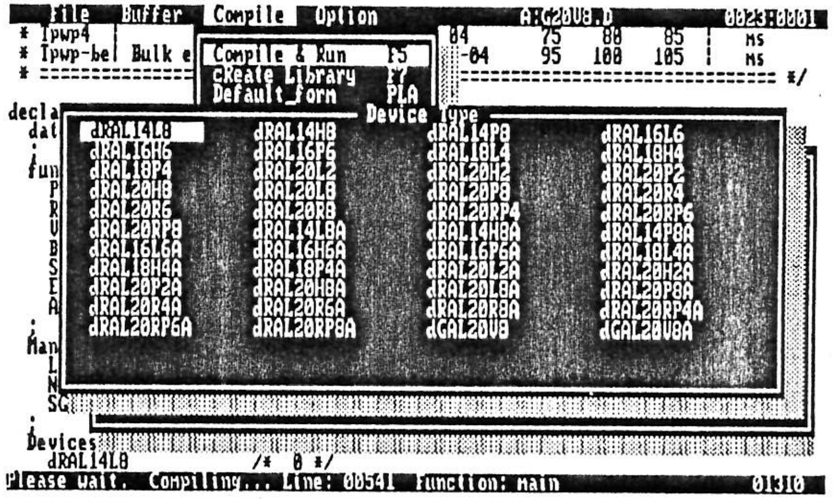
```
  File  Buffer  Compile  Option                    A:CE20V8.D          0001:0002
/* ---------------- ---------------------------------------------------P. S. 0---
*  FIRST UPDATE : Compile & Run    F5
*                 cReate Library   F7
*                 Default_form     PLA
*  LIB Ver      :
* ---------------------------------------------------------------------------
* Fuse_su                         : 2706 40
* Internal Security Fuse          : 1

   PALCE20V8H_15
   PALCE20V8H_25
   PALCE20V8Q_15
   PALCE20V8Q_25
   PALCE20V8H_15_4
   PALCE20V8H_25_4
   PALCE20V8Q_15_4
   PALCE20V8Q_25_4
* ===========================================================================
* MANUFACTURER : AMD
* ---------------------------------------------------------------------------
* SYMBOL |                              |   MIN   TYP   MAX  |  UNIT
* -------+------------------------------+--------------------+-----------
* Vccp   | VCC During Programming       |   4.75  5.00  5.25 |   V
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent   Zeus: 01310
```

Figure 3.1 Compilation & Run



```
  File  Buffer  Compile  Option                    A:G20V8.D           0023:0001
 * Ipwp4     |         Compile & Run    F5     04        75    80    85   |  MS
 * Ipwp-bel  Bulk e    cReate Library   F7     -04       95    100   105  |  MS
 * ============        Default_form     PLA    ============================ */

declaration
  dataunit : bit
  ;
 functi                     Manufacturers
   Prog    LATTICE            NS              SGS
   Read
   Veri
   Blan
   Secu
   Eras
   Auto
  ;
 Manufa
   LATT
   NS
   SGS
  ;
 Devices
   dRAL14L8              /*  0 */
Please wait.  Compiling... Line: 00541  Function: main                    01310
```

Figure 3.2 After a successful compilation

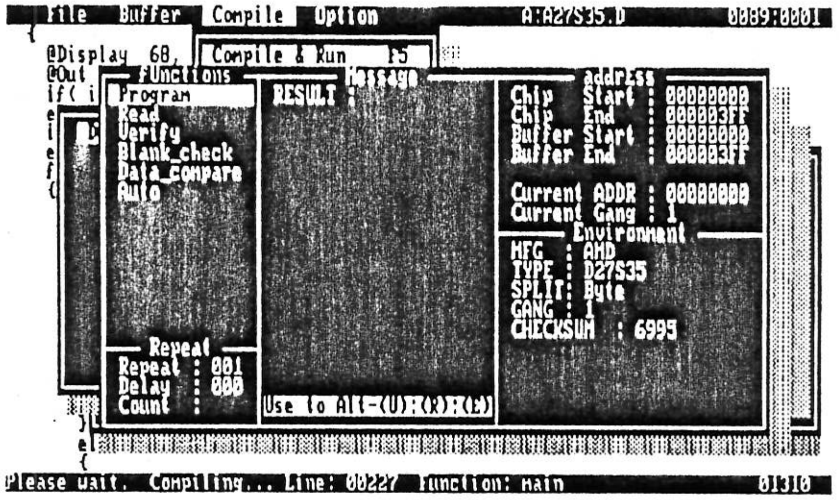


```
  File  Buffer  Compile  Option                    A:G20V8.D           0023:0001
 * Ipwp4     |         Compile & Run    F5     04        75    80    85   |  MS
 * Ipwp-bel  Bulk e    cReate Library   F7     -04       95    100   105  |  MS
 * ============        Default_form     PLA    ============================ */
 decla                           Device Type
  dat     dRAL14L8       dRAL14H8       dRAL14P8       dRAL16L6
  ;       dRAL16H6       dRAL16P6       dRAL18L4       dRAL18H4
 fun      dRAL18P4       dRAL20L2       dRAL20H2       dRAL20P2
   P      dRAL20H8       dRAL20L8       dRAL20P8       dRAL20R4
   R      dRAL20R6       dRAL20R8       dRAL20RP4      dRAL20RP6
   V      dRAL20RP8      dRAL14L8A      dRAL14H8A      dRAL14P8A
   B      dRAL16L6A      dRAL16H6A      dRAL16P6A      dRAL18L4A
   S      dRAL18H4A      dRAL18P4A      dRAL20L2A      dRAL20H2A
   E      dRAL20P2A      dRAL20H8A      dRAL20L8A      dRAL20P8A
   A      dRAL20R4A      dRAL20R6A      dRAL20R8A      dRAL20RP4A
  ;       dRAL20RP6A     dRAL20RP8A     dGAL20V8       dGAL20V8A
 Man
   L
   N
   SG
  ;
 Devices
   dRAL14L8              /*  0 */
Please wait.  Compiling... Line: 00541  Function: main                    01310
```

Figure 3.3 After the manufacturer is selected

After the device is selected, a big screen which contains menus for various fucntions and informations for down (up) loading will be displayed.

This menu will display a screen for all the pertinent information for programmings. There are five fields such as functions, message, address, repeat, and environment. Each field will be explained below.



```
A:A27S35.D                         0089:0001
@Display  68,
@Out
if( i

RESULT :

Chip   Start  : 00000000
Chip   End    : 000003FF
Buffer Start  : 00000000
Buffer End    : 000003FF

Current ADDR  : 00000000
Current Gang  : 1

MFG  : AMD
TYPE : D27S35
SPLIT: Byte
GANG : 1
CHECKSUM  : 6995

Repeat : 001
Delay  : 000
Count  :

Use to Alt-(U):(R):(E)

Please wait.  Compiling... Line: 00227  Function: main      01310
```

Figure 3.4 Function Select

Function field

Users can perform all the possible functions for implementing devices such as programming, reading, verifying, blank-check, data comparison, automatic programming, encryption and security programming.

Repeat field

Repeat, Delay, and Count will be recorded.

Message field

Error message or programming result will be displayed.

Address field

The informations about the chip and buffer address, current address, and the gang number selected will be displayed.

Environment field

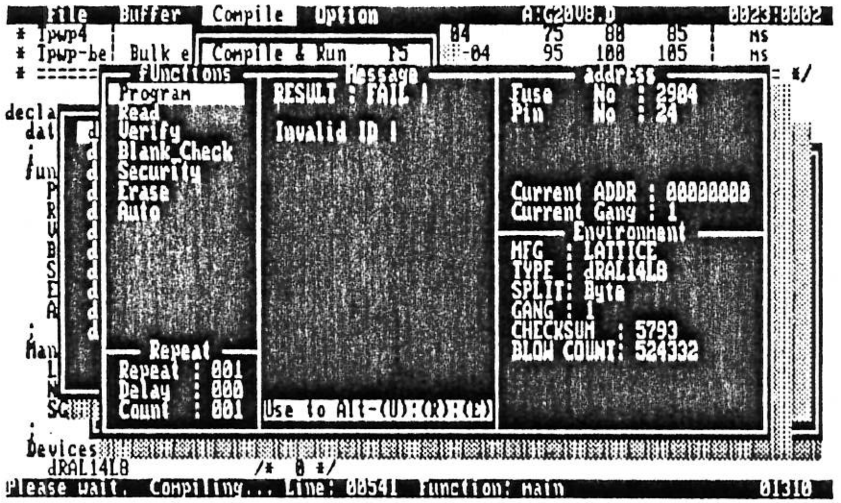
The information such as check sum, the number of gang selected, the way of byte splitting for programming, the name of a manufacturer and the part name of the device selected will be displayed.

Note: There are only three fields which are necessary to be accessed. They are fUnctions, Repeat, and addrEss. To go into the each section users need to press Alt and the capitalized character of the section title wanted. For example to change the start address of the buffer for programming users have to go into the address field and type over the start address field. To go into the address field users must press Alt-E.

A. fUntions

1. Program

This downloads the data in the buffer onto the chip. The size of the buffer is 64k bytes ranging from 0 ~ 0XFFFF. "Verify" function will be performed after programming. If there is an error, the error message will be displayed with the address where the error occurred. Any other result will be displayed in the message section.



```
 File  Buffer  Compile  Option               A:G20V8.D                0023:0002
 * Tpwp4   |                              04       75    80     85   |    ms
 * Tpwp-be| Bulk e| Compile & Run   F5 |  -04       95   100    105   |    ms
 * ======  fUnctions ──────── Message ──────────── addrEss ──────────── = */
 decla    Program        RESULT : FAIL !        Fuse   No : 2904
  dat   d Read                                  Pin    No : 24
  ;     d Verify         Invalid ID !
 fun    d Blank_Check
   P    d Security
   R    d Erase                                 Current ADDR : 00000000
   V    d Auto                                  Current Gang : 1
   B    d                                       ──── Environment ────
   S    d                                       MFG  : LATTICE
   E    d                                       TYPE : dRAL14L8
   A    d                                       SPLIT: Byte
  ;     d                                       GANG : 1
  Man     ── Repeat ──                          CHECKSUM   : 5793
   L      Repeat : 001                          BLOW COUNT: 524332
   N      Delay  : 000
   SC     Count  : 001   Use to Alt-(U):(R):(E)
 ;
 Devices
   dRAL14L8               /*  0 */
 Please wait.  Compiling... Line: 00541  Function: main                  01310
```

Figure 3.5 Program (PAL/GAL)

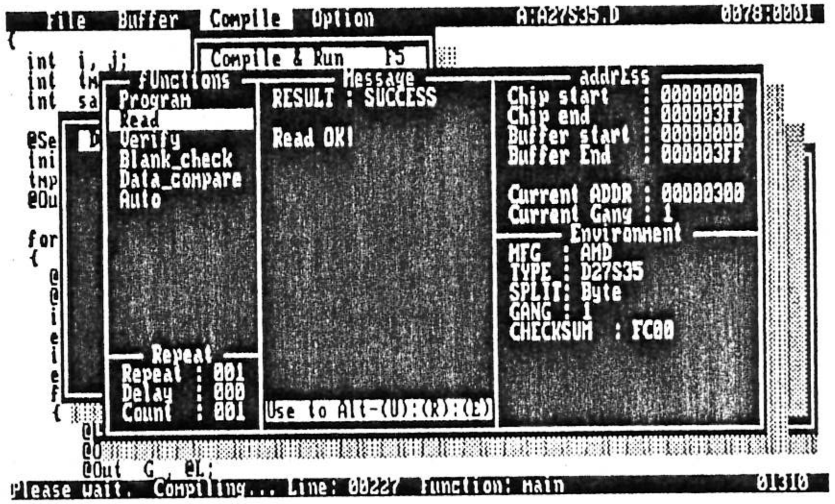
In the "addRess" section the address will be increased while the chip is being programmed or verified. The start address and the end address of the buffer can be corrected in the addrEss section.

2. Read

This reads the content of a chip into the buffer. After reading is done, in the address section, the checksum of the data will be displayed. If the chip is a PAL or a GAL, the blow count will be shown also.

When a GAL is programmed the device should match the manufacturer and the part name selected by the software. Otherwise, an error message will be displayed and the chip will not be programmed. If the security fuses are blown in a PAL or a GAL, the data read from the chip will be all 1's or 0's regardless of what the content is.

If the chip is ROM or Single chip, the data between the start address and the end address will be read into the buffer. The address being programmed will be displayed and the message will be displayed in the section of "message"

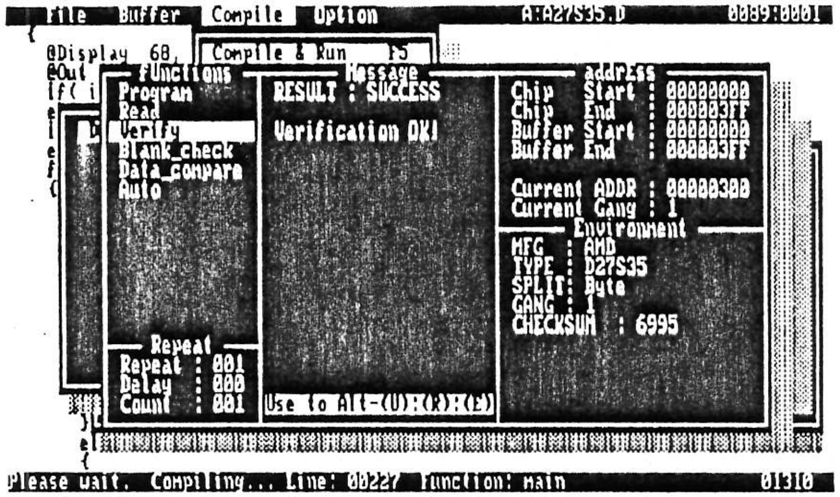


```
 File  Buffer  Compile  Option               A:A27S35.D              0078:0001
 {
  int  i, j;           | Compile & Run   F5 |
  int  tm fUnctions ──────── Message ──────────── addrEss ────────────
  int  sa  Program        RESULT : SUCCESS       Chip start    : 00000000
           Read                                  Chip end      : 000003FF
  @Se   D  Verify         Read OK!               Buffer start  : 00000000
  ini      Blank_check                           Buffer End    : 000003FF
  tmp      Data_compare
  @Ou      Auto                                  Current ADDR : 00000300
                                                 Current Gang : 1
  for                                            ──── Environment ────
  {                                              MFG  : AMD
    @                                            TYPE : D27S35
    @                                            SPLIT: Byte
    i                                            GANG : 1
    e                                            CHECKSUM   : FC00
    i      ── Repeat ──
    e      Repeat : 001
    f      Delay  : 000
  {        Count  : 001   Use to Alt-(U):(R):(E)
      @L
      @O
      @Out  G , @L;
 Please wait.  Compiling... Line: 00227  Function: main                  01310
```

Figure 3.6 Read (ROM / Single)

3. Verify

This compares the content of the buffer and the content of the chip. If an error occurs, it displays the error message and the address where it failed. If the chip is a ROM of Single chip, it verifies between the start address and the end address. The address being verified will be shown in the screen, being increased.

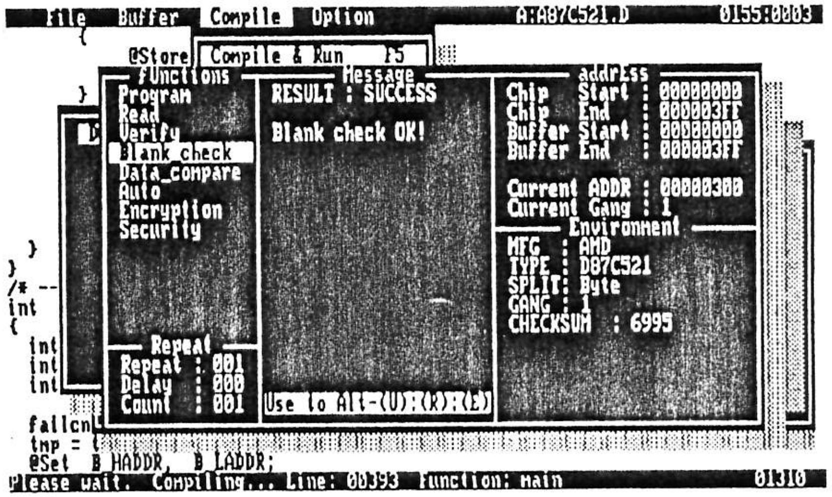


```
A:A27S35.D                                  0089:0001
Message
RESULT : SUCCESS

Verification OK!

Use to Alt-(U):(R):(E)

address
Chip   Start : 00000000
Chip   End   : 000003FF
Buffer Start : 00000000
Buffer End   : 000003FF

Current ADDR : 00000300
Current Gang : 1

Environment
MFG  : AMD
TYPE : D27S35
SPLIT: Byte
GANG : 1
CHECKSUM  : 6995

Repeat
Repeat : 001
Delay  : 000
Count  : 001

Please wait.  Compiling... Line: 00227  Function: main    01310
```

Figure 3.7 Verification

4. Blank Check

It reads the content of the data and compares it with the blank characters. if the chip is not blank it will display the discrepancy with the address. If the chip is a ROM or a Single, partial blank check is possible by indicating the start and end address.



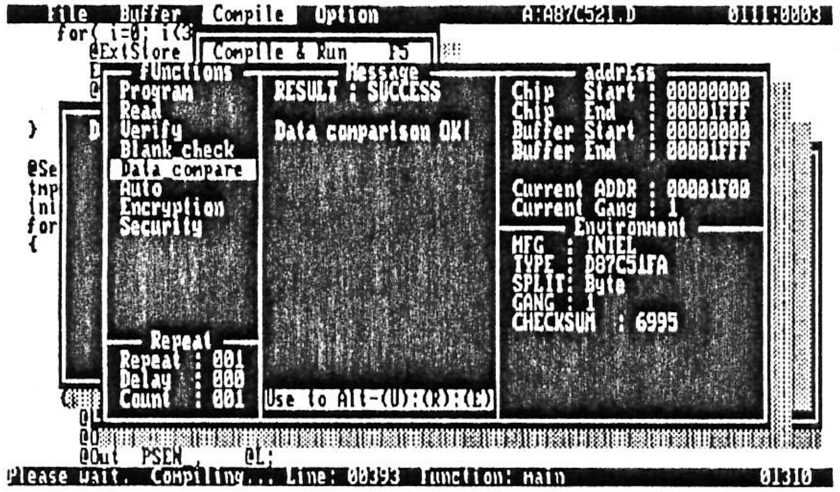
```
A:A87C521.D                                 0155:0003
Message
RESULT : SUCCESS

Blank check OK!

Use to Alt-(U):(R):(E)

address
Chip   Start : 00000000
Chip   End   : 000003FF
Buffer Start : 00000000
Buffer End   : 000003FF

Current ADDR : 00000300
Current Gang : 1

Environment
MFG  : AMD
TYPE : D87C521
SPLIT: Byte
GANG : 1
CHECKSUM  : 6995

Repeat
Repeat : 001
Delay  : 000
Count  : 001

@Set  B_HADDR,  B_LADDR;
Please wait.  Compiling... Line: 00393  Function: main    01310
```

Figure 3.8 Blank Check

5. Data Compare

This menu only applies to the ROM's and single micro-controllers. This is the same as "Verify" menu except that this will generate the file which will contain all the differences between the data of the chip and the buffer. After the execution of the menu of "Data Compare" the file name, which is the name of a device selected, with the suffix of "cmp" will be created in a current directory.

For example, if AMD 27256 has been selected in the softwre, the file created will be 27256.cmp. The file called 27267.cmp can be viewed in a regular editor and contain all the differences between the data of the chip and buffer. Not like the "verify" it will not stop in the first difference it encounters, but it will continue checking.



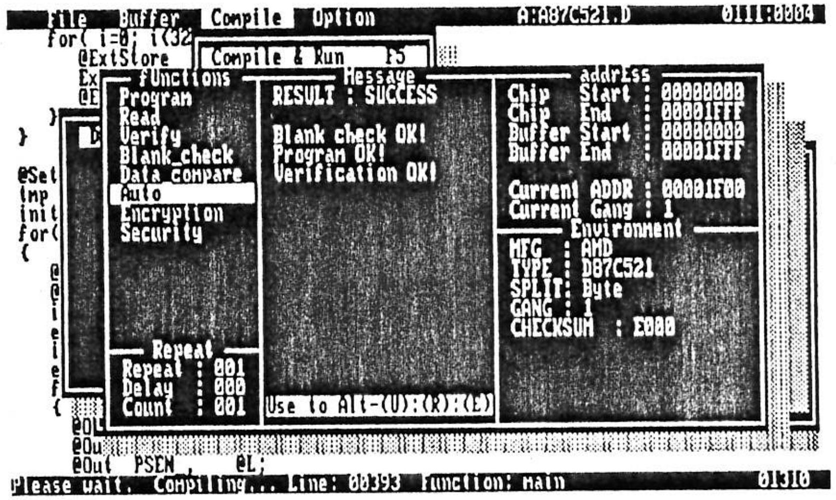
```
A:A87C521.D                    0111:0003
RESULT : SUCCESS

Data comparison OK!

Use to Alt-(U):(R):(E)

Chip   Start : 00000000
Chip   End   : 00001FFF
Buffer Start : 00000000
Buffer End   : 00001FFF

Current ADDR : 00001F00
Current Gang : 1

MFG  : INTEL
TYPE : D87C51FA
SPLIT: Byte
GANG : 1
CHECKSUM  : 6995

Repeat : 001
Delay  : 000
Count  : 001

Please wait.  Compiling... Line: 00393  Function: main    01310
```

Figure 3.9 Data Compare

6. Auto

This will execute many menus in a sequence. If the chip is a PAL or Gal this will execute Erase, blank check, program, verify and security.

If any of the menu is interrupted by an error, the next step will not be executed. If the chip is a ROM or Single chip this will execute Blank check, Program, and Verify. For the series of 87 Single micro-controllers encrypting is possible too.



```
A:A87C521.D                    0111:0004
RESULT : SUCCESS

Blank check OK!
Program OK!
Verification OK!

Use to Alt-(U):(R):(E)

Chip   Start : 00000000
Chip   End   : 00001FFF
Buffer Start : 00000000
Buffer End   : 00001FFF

Current ADDR : 00001F00
Current Gang : 1

MFG  : AMD
TYPE : D87C521
SPLIT: Byte
GANG : 1
CHECKSUM  : E000

Repeat : 001
Delay  : 000
Count  : 001

Please wait.  Compiling... Line: 00393  Function: main    01310
```

Figure 3.10 Auto

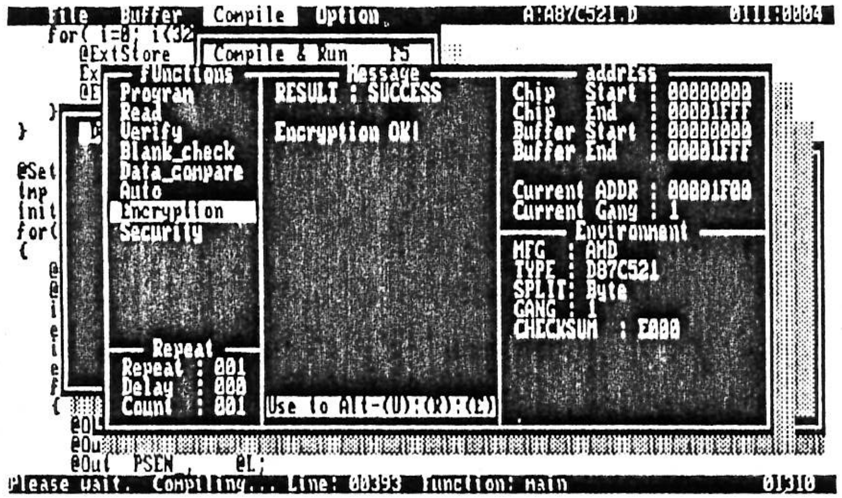
7. Security

If the security bit is programmed the chip will be recognized as a blank chip. Also, programming will not be possible. The security function will apply only to PAL's and GAL's.

8. Encryption Program

This only applies to single chips. This will program the content of the encryption table onto a chip. The content of the encryption table can be loaded, saved and edited.

Once the data of encryption is written the data in the main buffer will be Exclusive NORed with the data in the encryption table. If there is an error, an error message will be displayed.



```
A:A87C521.D                                  0111:0004
RESULT : SUCCESS

Encryption OK!

Chip   Start : 00000000
Chip   End   : 00001FFF
Buffer Start : 00000000
Buffer End   : 00001FFF

Current ADDR : 00001F00
Current Gang : 1

MFG   : AMD
TYPE  : D87C521
SPLIT : Byte
GANG  : 1
CHECKSUM  : E000

Repeat : 001
Delay  : 000
Count  : 001

Use to Alt-(U):(R):(E)

Please wait.  Compiling... Line: 00393  Function: main     01310
```

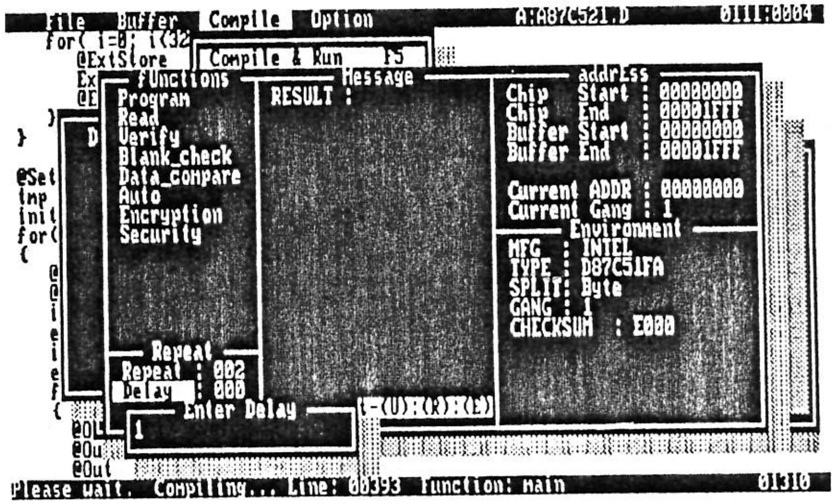
Figure 3.11 Encryption

9. MES Read

In GAL, Manufacturer's Electronic Signature contains the information for the chip. The data other than the main data will be assigned to the external buffer, and can be viewed through the menu, "Buffer -> External Edit".

B. Repeat

This determines the number of executions for a function to be repeated.



```
A:A87C521.D                                  0111:0004
RESULT :

Chip   Start : 00000000
Chip   End   : 00001FFF
Buffer Start : 00000000
Buffer End   : 00001FFF

Current ADDR : 00000000
Current Gang : 1

MFG   : INTEL
TYPE  : D87C51FA
SPLIT : Byte
GANG  : 1
CHECKSUM  : E000

Repeat : 002
Delay  : 000

Enter Delay
1

Please wait.  Compiling... Line: 00393  Function: main     01310
```

Figure 3.12 Repeat

1. Repeat

Users can determine the number of repetitions.

2. Delay

When users repeatedly use "Function Select" to program the same kind of devices many times this sets the time for pause between executions. In the pause a chip can be replaced.

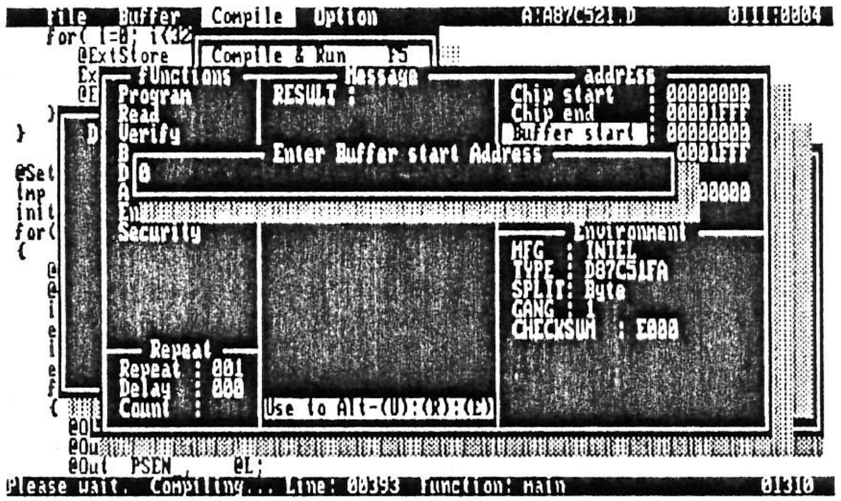
3. Count

The count of chips programmed will be displayed

C. Message

The message for success or failure will appear.

D. Address



```
 File  Buffer  Compile  Option              A:A87C521.D          0111:0004
   for( i=0; i<32
     @ExtStore    Compile & Run    F5
     Ex  Functions  Message                 address
     @E  Program    RESULT :               Chip start    : 00000000
         Read                              Chip end      : 00001FFF
 }    D  Verify                            Buffer start  : 00000000
         B  Enter Buffer start Address                     00001FFF
 @Set    D 0
 imp     A                                                 00000
 init    En
 for(    Security                          Environment
 {                                         MFG   : INTEL
                                           TYPE  : D87C51FA
                                           SPLIT : Byte
                                           GANG  : 1
                                           CHECKSUM  : E000
         Repeat
         Repeat : 001
         Delay  : 000
   {     Count  :     Use to Alt-(U):(R):(E)
     @O
     @Ou
     @Out  PSEN    @L;
Please wait.  Compiling... Line: 00393  Function: main            01310
```

Figure 3.13 Address

1. GAL

In programming GAL, this will show the number of fuses and pins of the chip in question, and it will show the current address which is being read, verified, compared, and etc. Also, the number of gangs selected will be shown.

2. ROM

In programming ROM's or Single microcontrollers, the entries in the address box will be displayed as explained below.

Chip Start

The address in the chip where programming will start will be designated.

Chip End

The end address where programming will stop will be designated. Hexa decimal values will be input. The address bigger than the last address of a chip will not be accepted.

Buffer Start Address

The Start address of the buffer to be programmed will be designated.

Buffer End Address

The last address of the buffer to be programmed will be designated.

The address increment and the number of gangs will be shown as in programming GAL's.

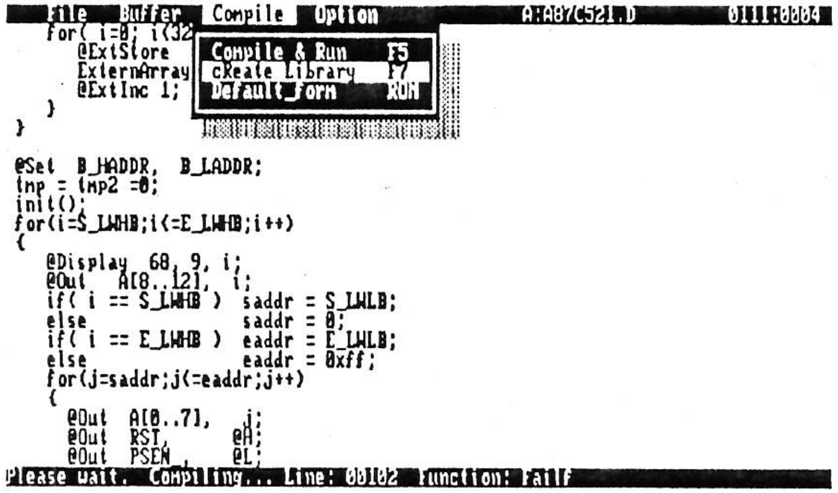
E. Environment

The information such as check sum, the number of gang selected, the way of byte splitting for programming, the name of a manufacturer and the part name of the device selected will be displayed.

3.2 Create Library

This will generate an object file with suffix of "lef" from a source file from the suffix of "d". The object file needs to be created to be recognized by the user interface software, SP.EXE. This will be possible when there is no error in compilation and testing is successful.

Therefore, this should be executed after the execution of "Compile & Run" menu. Users of SP need files with the suffix of "lef" and library files generated by the software called "LG. EXE".

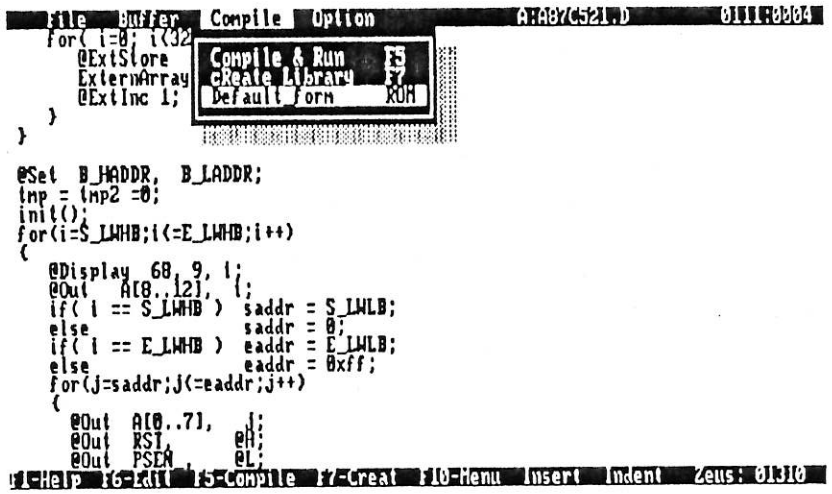


```
   for( i=0; i<32
      @ExtStore
      ExternArray
      @ExtInc 1;
   }
}

@Set  B_HADDR,  B_LADDR;
tmp = tmp2 =0;
init();
for(i=S_LWHB;i<=E_LWHB;i++)
{
   @Display  68, 9, i;
   @Out    A[8..12],  i;
   if( i == S_LWHB )  saddr = S_LWLB;
   else               saddr = 0;
   if( i == E_LWHB )  eaddr = E_LWLB;
   else               eaddr = 0xff;
   for(j=saddr;j<=eaddr;j++)
   {
     @Out  A[0..7],   j;
     @Out  RST,      @H;
     @Out  PSEN ,    @L;
Please wait.  Compiling... Line: 00102  Function: failf
```

Figure 3.14 Create Library

3.3 Default_Form

In this menu device type is determined. Depending on the menu selected different menus or buffers will be displayed. There are two choices: ROM and PLD.



```
   for( i=0; i<32
      @ExtStore
      ExternArray
      @ExtInc 1;
   }
}

@Set  B_HADDR,  B_LADDR;
tmp = tmp2 =0;
init();
for(i=S_LWHB;i<=E_LWHB;i++)
{
   @Display  68, 9, i;
   @Out    A[8..12],  i;
   if( i == S_LWHB )  saddr = S_LWLB;
   else               saddr = 0;
   if( i == E_LWHB )  eaddr = E_LWLB;
   else               eaddr = 0xff;
   for(j=saddr;j<=eaddr;j++)
   {
     @Out  A[0..7],   j;
     @Out  RST,      @H;
     @Out  PSEN ,    @L;
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

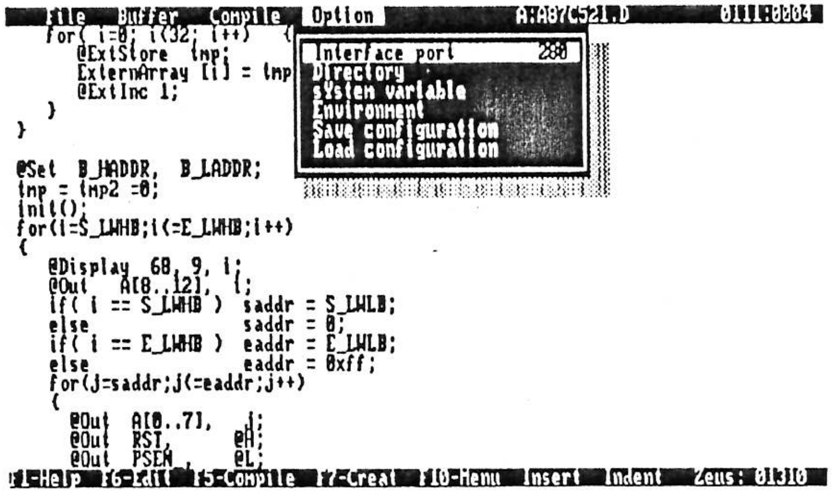
Figure 3.15 Default_Form

4 OPTION

This menu deals with subsidiary or environmental setups.

There are six submenus as below.

Interface port	Selects the appropriate port address
Directory	Specifies the directorys for the system and the files to be output.
System varlable	Sets the number of gangs, word format, and the external mode
Environment	When PLD's are dealt all the relevant options will be set.
Save configuration	Current setups will be stored for the later retrieval.
Coad configuration	Setup files stored will be retrieved.


File Buffer Compile Option A:A87C521.D 0111:0004
Interface port 280
Directory
sYstem variable
Environment
Save configuration
Load configuration
F1-Help F6-Edit F5-Compile F7-Creat F10-Menu Insert Indent Zeus: 01310

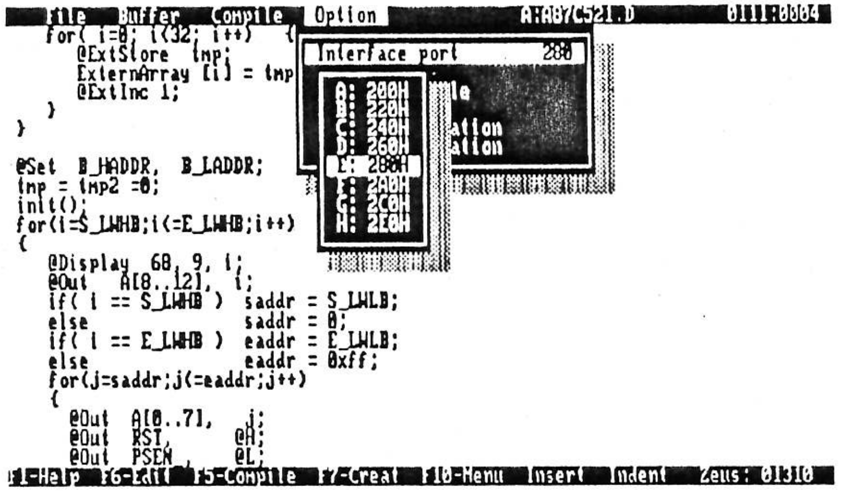
```
   for( i=0; i<32; i++)     {
      @ExtStore  tmp;
      ExternArray [i] = tmp
      @ExtInc 1;
   }
}

@Set  B_HADDR,  B_LADDR;
tmp = tmp2 =0;
init();
for(i=S_LWHB;i<=E_LWHB;i++)
{
   @Display  68, 9, i;
   @Out   A[8..12],  i;
   if( i == S_LWHB )  saddr = S_LWLB;
   else               saddr = 0;
   if( i == E_LWHB )  eaddr = E_LWLB;
   else               eaddr = 0xff;
   for(j=saddr;j<=eaddr;j++)
   {
     @Out  A[0..7],    j;
     @Out  RST,       @H;
     @Out  PSEN ,     @L;
```

Figure 4.1 Option

4.1 Interface Port

This should select the port address which matches the port in the interface card. There are 8 port addresses. When the port addresses between the software and the interface card do not match, an error message will be displayed.

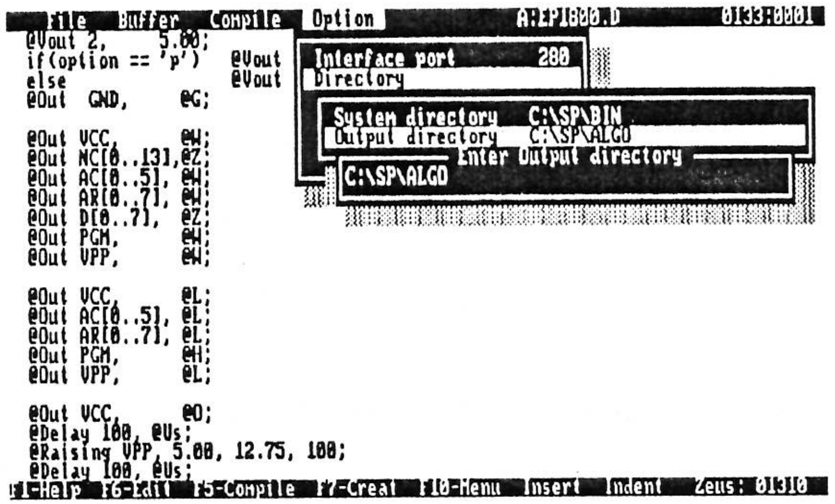
However, there are many factors causing the communication error. Therefore, users should check installation procedures and connections by referring to the chapters for introduction and installation in Superpro regular manual. To abandon the error message press the escape key.



```
   for( i=0; i<32; i++)    {
      @ExtStore  tmp;
      ExternArray [i] = tmp
      @ExtInc 1;
   }
}

@Set  B_HADDR,  B_LADDR;
tmp = tmp2 =0;
init();
for(i=S_LWHB;i<=E_LWHB;i++)
{
   @Display  68, 9, i;
   @Out   A[8..12],  i;
   if( i == S_LWHB )  saddr = S_LWLB;
   else               saddr = 0;
   if( i == E_LWHB )  eaddr = E_LWLB;
   else               eaddr = 0xff;
   for(j=saddr;j<=eaddr;j++)
   {
     @Out  A[0..7],   j;
     @Out  RST,      @H;
     @Out  PSEN ,    @L;
```

Figure 4.2 Interface port

4.2 Directory



```
 @Vout 2,     5.00;
 if(option == 'p')    @Vout
 else                 @Vout
 @Out  GND,      @G;

 @Out VCC,       @H;
 @Out NC[0..13],@Z;
 @Out AC[0..5], @H;
 @Out AR[0..7], @H;
 @Out D[0..7],  @Z;
 @Out PGM,      @H;
 @Out VPP,      @H;

 @Out VCC,      @L;
 @Out AC[0..5], @L;
 @Out AR[0..7], @L;
 @Out PGM,      @H;
 @Out VPP,      @L;

 @Out VCC,      @O;
 @Delay 100, @Us;
 @Raising VPP, 5.00, 12.75, 100;
 @Delay 100, @Us;
```

File
Buffer
Compile
Option
A:EP1800.D
0133:0001
Interface port 280
Directory
System directory C:\SP\BIN
Output directory C:\SP\ALGO
Enter Output directory
C:\SP\ALGO
F1-Help F6-Edit F5-Compile F7-Creat F10-Menu Insert Indent Zeus: 01310

Figure 4.3 Directory

A. System directory

The directory where the executable (AG.EXE) is specified

B. Output directory

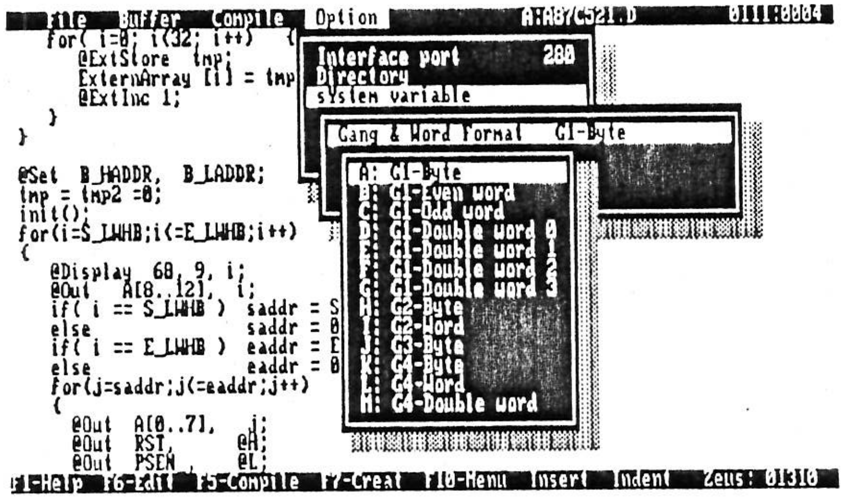
The object file with the suffix of "lef", which is generated after the execution of "Compile->Create Library", will be stored in the directory specified here.

4.3 System Variable

This deals with the number of gangs, word format, limit of program, blank character and extern mode

A. Gang & Word Format

This menu determines the number of gangs and the type of word format. The word format command will configure the way of retrieving the data in the current buffer. The number following the capital G in the menu is the number of gangs selected. There is an optional four socket adapter for purchase. This is based on the assumption that users use the four socket adapter. The four socket adapter will program each gang serially.



```
   for( i=0; i<32; i++)    {
      @ExtStore  tmp;
      ExternArray [i] = tmp
      @ExtInc 1;
   }
}

@Set  B_HADDR,  B_LADDR;
tmp = tmp2 =0;
init();
for(i=S_LWHB;i<=E_LWHB;i++)
{
   @Display  68, 9, i;
   @Out    A[8..12],  i;
   if( i == S_LWHB )  saddr = S
   else               saddr = 0
   if( i == E_LWHB )  eaddr = E
   else               eaddr = 0
   for(j=saddr;j<=eaddr;j++)
   {
     @Out  A[0..7],    j;
     @Out  RST,       @H;
     @Out  PSEN ,     @L;
```

Interface port 280

Gang & Word Format G1-Byte

Figure 4.4 Gang & Word Format

1. G1-Byte

This is for one gang and programs a byte (8 bits) at a time.

2. G1-Even Word

It is used for one gang and processes two bytes (16 bits) at a time. Since "Even" is indicated it will program the even bytes. The definition of the even byte will be explained as below.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD

Result after programming :

Address of Chip	Data of Chip
00	01
01	45
02	89
03	CD
.	.
.	.
.	.

3. G1-Odd word

It is used for one gang and process 2 bytes (16 bits) at a time. Since "Odd" is selected, it will program the odd bytes. The example is as follows.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF

Result:

Address of Chip	Data of Chip
00	23
01	67
02	AB
03	EF
.	.
.	.
.	.

4. G1-Double word 0

This is used for one gang and processes 4 bytes (32 bits) at a time. This will program the data of the buffer in the addresses of 0th, 4th, 8th, ... and so on.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF
08	FE
09	DC
0A	BA
0B	98
0C	76

Result:

Address of Chip	Data of Chip
00	01
01	89
02	FE
03	76
.	.

5. G1-Double word 1

This is used for one gang and processes 4 bytes (32 bits) at a time. This will program the data of the buffer in the addresses of 1st, 5th, 9th, ... and so on.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF
08	FE
09	DC
0A	BA
0B	98
0C	76

Result:

Address of Chip	Data of Chip
00	23
01	AB
02	DC
.	.
.	.

6. G1- Double word 2

This is used for one gang and processes 4 bytes (32 bits) at a time. This will program the data of the buffer in the addresses of 2nd, 6th, 10th, ... and so on.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF
08	FE
09	DC
0A	BA
0B	98
0C	76

Result:

Address of Chip	Data of Chip
00	45
01	CD
02	BA
.	.
.	.

7. G1-Double word 3

This is used for one gang and processes 4 bytes (32 bits) at a time. This will program the data of the buffer in the addresses of 3rd, 7th, 11th, ... and so on.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF
08	FE
09	DC
0A	BA
0B	98
0C	76

Result:

Address of Chip	Data of Chip
00	67
01	EF
02	98
.	.

8. G2-Byte

This is used for two gangs and processes one byte (8 bits) at a time. The same data from the buffer will be programmed for the two gangs. When users read in this mode only the first gang will be read and the data read will be loaded onto the buffer. But every other functions will be executed in both gangs.

9. G2-Word

This is for two gangs and processes two bytes (16 bits) at a time. The even data will be programmed in the first gang, and the odd data will be programmed into the second gang.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF

Result:

Address of Chip	Data of Gang 1	Gang 2
00	01	23
01	45	67
02	89	AB
03	CD	EF

10. G3-Byte

This is for three gangs and processes one byte (8 bits) at a time. All the functions except "read" will be applied to each gang. When users read in this mode only the chip in the first gang will be read and loaded into the buffer. All the three gangs will be programmed with the same data.

11. G4-Byte

This is for four gangs and processes one byte (8 bits) at a time. All the functions except "read" will be applied to each gang. When users read in this mode only the chip in the first gang will be read and loaded into the buffer. All the four gangs will be programmed with the same data.

12. G4-Word

This is used for four gangs and processes two bytes (16 bits) at a time. The even bytes will be programmed onto the first and the third gangs, and the odd bytes will be programmed onto the second and fourth gangs.

All the functions except "read" will be applied to each gang. When users read in this mode only one chip in the first gang out of four gangs will be read and loaded into the buffer. As the result of programing the first and the third gang will receive the same data, and the second and the fourth gang will receive the same data.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF

Result:

Address	Data of Chips In			
of Chip	Gang 1	Gang 2	Gang 3	Gang 4
00	01	23	01	23
01	45	67	45	67
02	89	AB	89	AB
03	CD	EF	CD	EF

13. G4-Double word

This is used for four gangs and processes 4 bytes (32 bits) at a time. The first byte will be programmed in the first gang, second byte in the second gang, third byte in the third gang, and fourth in the fourth.

(Example)

Given:

Address of Buffer	Data of Buffer
00	01
01	23
02	45
03	67
04	89
05	AB
06	CD
07	EF
08	FE
09	DC
0A	BA
0B	98
0C	76
0D	54

Result:

Address of Chip	Data of Chips in Gang1	Gang2	Gang3	Gang4
00	01	23	45	67
01	89	AB	CD	EF
02	FE	DC	BA	98
03	76	54		

B. Limit of Program

When users program chips, Superpro verifies each byte after programming. If there is an error programming will be repeated until programming is successful within the limit of repetitions.

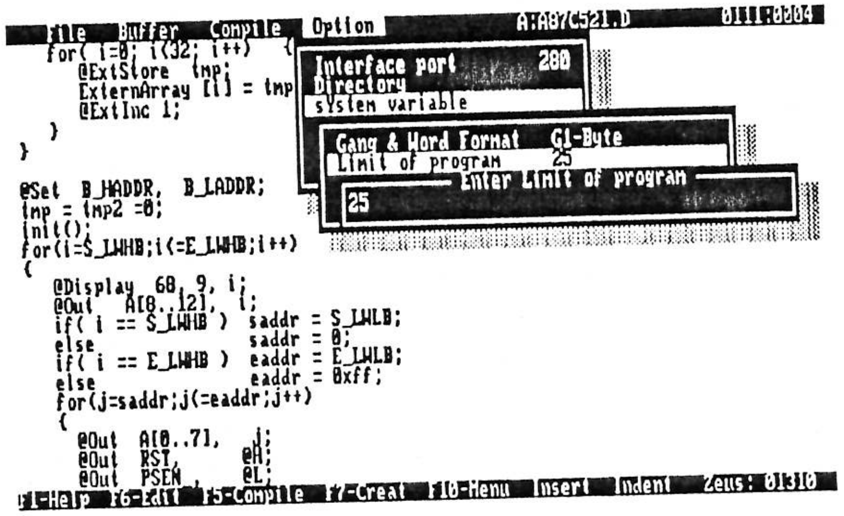

```
  File   Buffer   Compile   Option                A:A87C521.D        0111:0004
   for( i=0; i<32; i++)     {   Interface port      280
      @ExtStore  tmp;           Directory
      ExternArray [i] = tmp     system variable
      @ExtInc 1;
   }                               Gang & Word Format    G1-Byte
}                                  Limit of program      25
                                   ───── Enter Limit of program ─────
@Set  B_HADDR,  B_LADDR;          25
tmp = tmp2 =0;
init();
for(i=S_LWHB;i<=E_LWHB;i++)
{
   @Display  68, 9, i;
   @Out   A[8..12],  i;
   if( i == S_LWHB )  saddr = S_LWLB;
   else               saddr = 0;
   if( i == E_LWHB )  eaddr = E_LWLB;
   else               eaddr = 0xff;
   for(j=saddr;j<=eaddr;j++)
   {
      @Out  A[0..7],   j;
      @Out  RST,       @H;
      @Out  PSEN ,     @L;
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent   Zeus: 01310
```

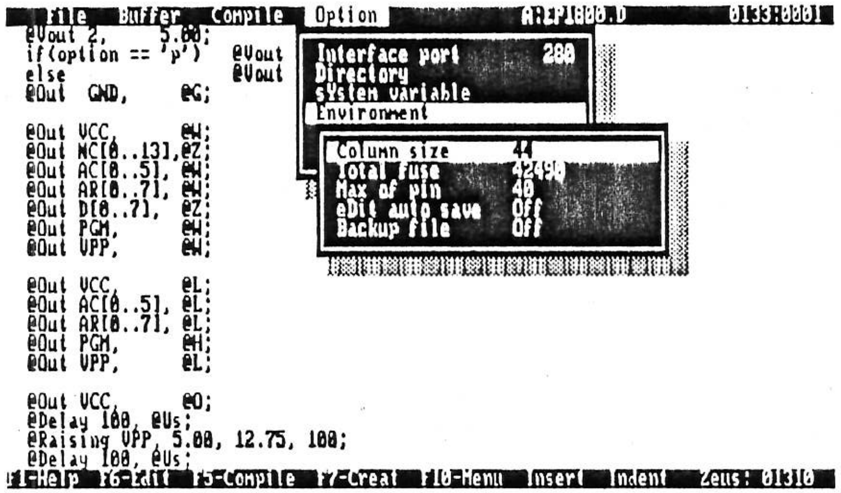
Figure 4.5 Limit of Program

C. Extern Mode

This mode should be turned off except reading 87 series with encryption table. When the mode is on and users read the device, the data exclusive-NORed between the data of the chip and the data of encryption table will be displayed in the buffer.

4.4 Environment

This has informations about column size, number of fuses, max number of pins, edit auto save, and back up files.



```
 File   Buffer   Compile   Option          A:EP1800.D          0133:0001
 @Vout 2,        5.00;
 if(option == 'p')    @Vout
 else                 @Vout
 @Out  GND,       @G;

 @Out  VCC,       @H;
 @Out  NC[0..13],@Z;
 @Out  AC[0..5], @H;
 @Out  AR[0..7], @H;
 @Out  D[0..7],  @Z;
 @Out  PGM,      @H;
 @Out  VPP,      @H;

 @Out  VCC,      @L;
 @Out  AC[0..5], @L;
 @Out  AR[0..7], @L;
 @Out  PGM,      @H;
 @Out  VPP,      @L;

 @Out  VCC,      @O;
 @Delay 100, @Us;
 @Raising VPP, 5.00, 12.75, 100;
 @Delay 100, @Us;
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

```
Interface port    280
Directory
sYstem variable
Environment

Column size       44
Total fuse        42490
Max of pin        40
eDit auto save    Off
Backup file       Off
```

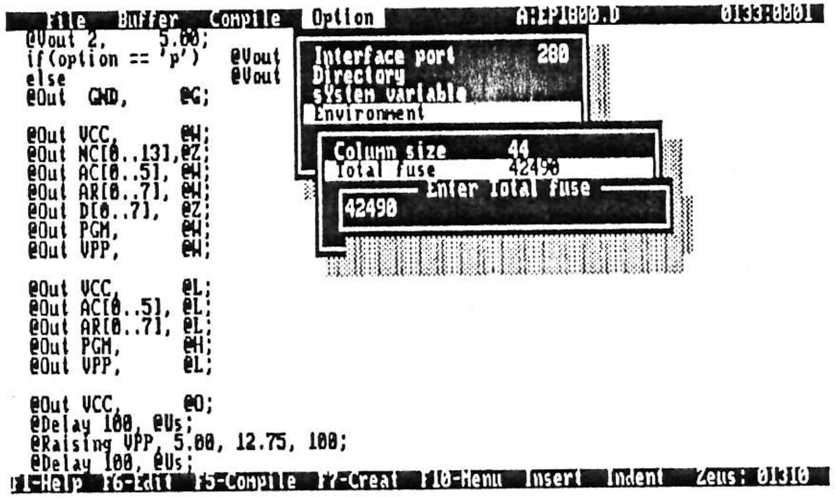
Figure 4.6   Environment

A. Column Size

When users editing a fuse map, this menu determines the number of input lines. The maximum number for this is 64 in decimal. If the number exceeds 64 an error message will be displayed. If the number of input lines is greater than 64, then users can divide the number of input lines by 2 and enter it. This will reduce the number of columns by half but double the number of rows.

B. Total Fuse

When users edit a fuse map this menu determines the number of fuses. A decimal number will be entered. The number of rows in the buffer of a fuse map will be calculated by dividing the number of the total fuses by the number of columns entered in the menu, "Option -> Environment -> Column Size".



```
 File   Buffer   Compile   Option          A:EP1800.D          0133:0001
 @Vout 2,        5.00;
 if(option == 'p')    @Vout
 else                 @Vout
 @Out  GND,       @G;

 @Out  VCC,       @H;
 @Out  NC[0..13],@Z;
 @Out  AC[0..5], @H;
 @Out  AR[0..7], @H;
 @Out  D[0..7],  @Z;
 @Out  PGM,      @H;
 @Out  VPP,      @H;

 @Out  VCC,      @L;
 @Out  AC[0..5], @L;
 @Out  AR[0..7], @L;
 @Out  PGM,      @H;
 @Out  VPP,      @L;

 @Out  VCC,      @O;
 @Delay 100, @Us;
 @Raising VPP, 5.00, 12.75, 100;
 @Delay 100, @Us;
F1-Help  F6-Edit  F5-Compile  F7-Creat  F10-Menu  Insert  Indent  Zeus: 01310
```

```
Interface port    280
Directory
sYstem variable
Environment

Column size       44
Total fuse        42490

── Enter Total fuse ──
42490
```

Figure 4.7   Total Fuse

C. Max of Pin

This specifies the number of pins in the device. The buffer of vector table will be generated according to the number of pins entered here. The max number of pins is 40.

D. Edit Auto save

When a user edits a source file with the suffix of "D", there are occasions when he exits the software or calls another file. In that cases, the software will ask whether the existing file in the buffer should be saved or not. However, if "Edit auto save" is on the software will not ask the question, but saves it automatically in the name it was called for.

E. Backup File

If this menu is on the back up file will be made when a source file is edited.

4.5 Save Configuration

Any setups in the option menu such as the value of interface port and others will be saved in the file called "LOPSCONF.AG" to be restored. If there is no file called "LOPSCONF.AG" the system will create one.

4.6 Load Configuration

This will load a file which has the informations about the setups in the option menus and reconfigure the system. If users don't specify the name of a setup file the default file, "LOPSCONF.AG", will be loaded for reconfiguration.

BOOK II

DCL (Device Control Language)

DCL is written with C and an assembler. This has many subroutines and functions to control voltage, pulse width, and etc. Thus, updating devices is extremely easy compared with updating in C language.

1 INTRODUCTION

DCL (Device Control Language) provides numerous subroutines and reserved words to control pins to generate algorithms for the chips. This will provide the utmost flexbility for generating algorithm than any other programmer in the market (most of the chips will be implenented).

DCL has all the necessary opertors and statements which is enough to control all the pins. To make an algorithm the steps below should be followed.

1. Write an algorithm with DCL source code using AG.EXE.
2. Compile the source code using a menu in AG.EXE and test to verify the algorithm.
3. Generate an object file with the extension of LEF using AG.EXE.
4. Register the device using LG.EXE.
5. Run SP.EXE (user interface software).

Details for the procedures will be explained later.

1.1 Characteristics of DCL

DCL is very similiar with C and has many pin control functions. So far we used C to generate algorithms for chips. That envolved lots of mappings and calculations between the microprocessor, pripherial interface adapter, and the socket pins of the programmer. DCL, which is higher than C language, removed all the troubles of mappings and calcultions. Thus the speed of generating algorithms with DCL is too fast to be compared with that of C. Also, in extreme case such as EPROM's one algorithm could inplement 30 or 40 devices due to the utilization of pin control functions. The characteristics will be explained below.

A. The structure of DCL program is consisted of the declaration section and the function section. The declaration section will define the manufacturers, the part names, the name of functions, the name of each pin of the device, and etc. The function section will list the functions for programming, reading, verifying , and etc.

B. DCL doesn't have pointers.

C. Only one dimensional array is available. Normally the address pins and the data pins will be represented by one dimensional arrays. Indexes of arrays can be written with or without brackets. But to be used without brackets the index must be a constant.

Example:

1. Address pins : A0 is same as A [0]
2. Data pins : D0 is same as D [0]
3. A[i] is not valid since the index is not a constant. If it is not a pin array the index should accompany a pair of brackets.

```
ex)  int i[4]
     i0 = 0;          ; not valid
     i[0] = 0;        ; valid
```

D. The name of an array will not be passed as a parameter to other function.

E. The keywords for DCL will be case insensitive.
ex) DECLARATION or declaration

F. Parameters and variables will be case sensitive.
ex) Vpp is not same as VPP

G. Functions can be defined as integer type only.

H. Only the pin array can use two dots to indicate the range of index. No variable will be used for rainging.

```
ex) Pin A [0..7]        ; Valid
    Char i[0..3]        ; Not valid
    Pin  A[0..K]        ; Not valid
```

I. All the data type will be unsigned.

2 DCL TOKENS

DCL has identifier, keyword, constant, arithmatic operators, lgical operators, and etc.

2.1 Identifier

There are three rules for the identifiers

A. Variables will be consisted of letters, numbers and underscore. Max 30 characters are allowed.

```
ex)  _key, key1, key_123, P2716     : valid
     P2764/L, i8, P2816*            : not valid
```

B. Variable names are case sensitive. Vpp and VpP are different.

C. There are two kinds of variables: local variables and global variables. A local variable will only be effective within the function which declares the variable. The global variable will be effective in the entire program. However, if a global variable is declared again as a local variable in a function the local variable will take the higher priority in the fuction.

2.2 Keyword

There are two kinds of keywords. One kind is for controlling pins, and the others will be all the keywords which are irrelevant to the keywords for controlling pins. The pin keywords will start with @ to be destinguished with the regular keywords. All the keywords are case-insensitive.

ex) @Out is same as @OUT.

A. Pin Keywords

Pin keywords can be catagorized into five kinds according to their functions. This will be explained in details in Ch 5.

1. Pin Keywords for reading and writing data.
   @OUT, @IN, @W, @Z
2. Pin keywords for assigning values to the pins of a chip.
   @G, @C, @L, @H
3. Pin keywords related to the voltages
   @VOUT, @O, @F, @RAISING, @FAILING
4. Pin keywords related to the buffer
   @SET, @LOAD, @STORE, @INC, @DEC
5. Pin keywords related to delaying time intervals
   @ DELAY, @PINDELAY, @US, @MS

B. General Keywords

BIT, BREAK, BYTE, CHAR, CHIPS, CONTINUE, DATAUNIT, DECLARATION, DEVICES, ELSE, ENDD, FOR, FUNCTIONS, IF, INT, MANUFACTURES, PIN, VOLTAGE, WHILE

2.3 Three Kinds of Constants

A. Numbers

Binary Integer	: Prefix 0b	ex) 0b1010
Decimal Integer	: No prefix	ex) 120
Hexadecimal Integer	: Prefix 0x	ex) 0xFFA3

B. Pin Contrl Constants

@L	: Logic low
@H	: Logic high
@O	: Turns on the voltage sources (V1, V2, and V3)
@F	: Turns off the voltage sources (V1, V2, and V3)
@C	: Applies to the clock pin
@Z	: Sets the ports for reading data from the chip

@G	: Grounding
@W	: Sets the ports for writing data to the chip

C. Voltage constants

$V1	: Voltage source (0 - 25V)
$V2	: Voltage source (0 -25V)
$V3	: Voltage source normally used as VCC (0 - 12V)

Note: the voltages will be expressed with two digits for the integer part and two digits after the decimal point. ex) 12.34

3 DATA TYPE

Being similiar to C language, DCL has necessary data types.

3.1 Int

This data type is two bytes and, and only unsigned interger is defined.

3.2 Char

This has one byte. ASCII characters are standard. A character will be put between single quotation marks, and a string of characters will be put between double quotation marks.

```
ex)     'A', 'B', 'I', 'O',
        "DCL", "LOPS", "IDE"
```

3.3 Pin

This has one byte. If a variable is defined as this data type it will be taken by the pin control constants.

```
ex)     Chips = Vpp, A13, A7, AG, DO, D1,D2, GND, D3, D4, D5,
        D6, D7, CE, ..... VCC
        ;
        Function A( )
        {
        Pin i, j;
        @Out Vpp, i;
        }
```

3.4 Array

Only one demensional array will be accepted.

A. Indexes of arrays can be either a variable or a constant.

ex) I[4], I[k]

B. In the declaration statement "CHIPS=....", brackets are not allowed for the indexes.

```
ex)  CHIPS   = Vpp, A7, AG, A5, A4....         ; Valid
     CHIPS   = Vpp, A[7], A[6], A[5], A[4]...  ; Not valid
```

C. Index rainging is only possible for the data type "chips". Also rainging with a variable is not allowed.

```
ex)  CHIPS   = Vpp, A7, A6, A5, A4;
     Int i0, i1, i2, i3
     @Out A[4..7]; @W          : Valid
     @Out I[0..4]              : not valid
     @Out A[I..J], @W          : not valid
```

D. Only the globlal variables can be itialized, but the local variables cannot be initialized. Initial values will be enclosed by a pair of brackets and will be seperated by commas.

```
ex)  Declarations
     DATAUNIT: byte;
     .
     .
     endd
     PIN I[3] = {@L, @H, @L}      ; Valid (global)
     FUNC( )
     {
     PIN J[3] = {@G, @L, @Z}      ; Not valid (local)
     }
```

Note: If the number of initial values is less than the number of elements in an array to be initialized then unassigned values will be assigned with zeros.

```
ex)  PIN I[5] = {@L, @H, @L};
     I[0] = @L; I[1] = @H; I[2] = @L;
     I[3] = 0; I[4] = 0;
```

E. The name of an array cannot be passed as a parameter to other functions.

4 EXPRESSION

There are arithmetic operators, relational operators, logical operators, bit wise operators, bitwise logical opertors, and etc.

All the operators will be listed from the lowest priority toward the highest priority.

=	Assigns
+=	Adds the right value to the left. The result will be assigned to the left variable.
-=	Subrtacts the right value from the left. The result will be assigned to the left variable.
*=	Multiplies the right value to the left. The result will be assigned to the left variable.
/=	Devides the left with the right and the result will be assigned to the left.
%=	Devides the left values by the right. The remainder will be assigned to the left.
&=	ANDed and assigned
|=	ORed and assigned
^=	Exclusively ORed and assigned
<<=	One bit is shifted to the left and assigned
>>=	One bit is shifted to the right and assigned
|	Bitwise OR
^	Bitwise exclusive OR
&	Bitwise AND
==	Equal
!=	Not equal

<	Less than		
>	Greater than		
>=	Greater than or equal to		
<=	Less than or equal to		
&&	Logical AND		
!!	Logical OR		
<<	One bit is shifted to the left		
>>	One bit is shifted to the right		
!&	NAND		
!|	NOR		
!^	NXOR		
+	Plus		
-	Minus		
*	Multiply		
%	Divides for remainder		
/	Divides for quotient		
++	1) ++ variable	;	Increase and assign
	2) variable ++	;	Assign and increase
--	1) -- variable	;	Decrease and assign
	2) variable --	;	Assign and decrease
!	Not		
~	Complement		

5 DCL PROGRAM

This chapter will illustrate the structure and mechanism with the examples.

5.1 Structure

A program can be divided into two major parts. First declaration section should come first, and the definition section for functions will come.

```
Program:
        Declaration
                Definelist
        Endd
        Definitions
```

A. Declaration

```
Declaration
        Dataunit : byte;
        Functions
                Program
                Read
                .
                .
                .
                Auto
                :
        Manufacturers
                INTEL
                AMD
                :
```

```
        Devices
                D27c128
                D27128A
                :
        Chips = Vpp, Q12, A7, A6, A5, A4, A3, A2, A1, AO,
                DO, D1, D2, GND, D3, D4, D5, D6, D&, CE,
                A10, OE, A11, A9, A8, A13, PGM, Vcc
                  :
        @v2 = Vpp;
        @v3 = Vcc;
```

As in the example above this section contains the data format, manufacturers, devices, pin assignments, and voltage source assignments.

The main structure is as below.

```
    Declaration
        Dataunit: bit or byte ;
        Functions
                function1
                function2
                .
                .
                .
                :
        Manufacturers
                company1
                company2
        Devices
                part_name1
                part_name2
        Chips
                The pins of a chip will be assigned.
        Voltage connection
                ( The three voltage sources will be assigned to
                the pins which require voltage sources.)
        endd
                (The end mark statement for the declaration
                section.)
```

1. Declaration

This should be written in the beginning of the declaration section.

2. Dataunit

For ROM's and Single microcontroller, write "byte", and for PLD's "bit". This is the unit of data processing.

```
ex)     Dataunit : byte;
```

3. Functions

The functions such as program, read, and any functions which may be defined later in the section of functions. Later these names will appear in the menu. And any of the menus can be highlighted for execution.

```
ex)     Function
                Program
                Read
                Verify
                Blank-Check
                ;
```

4. Manufacturers

The name of manufactures will be listed to be shown in the menu later on. Since whatever users type will be shown in the menu, they should mind the case sensitivity.

```
ex)     Manufacturers
                Intel
                AMD
                Microchip
                NEC
                TI
                CYPRESS
                ;
```

5.Device

The name of the devices will be listed to be shown in the menu later on. the device names should start with D to be recognized by the compiler.

```
ex)   Devices
              D27128A
              D27C128
              :
```

6. Chips

All the pins in the device will be named and declared.

```
Chips=    Vpp, AF, A7, A5, A4, A3, A2, A1, A0, D0, D1, D2, GWD,
          D3, D4, D5, D6, D7, CE, A10, OE, A11, A9, A8, A13,
          PGM, Vcc
;
```

7. Voltage connections

In Superpro there are three voltage sources. The first and second voltage sources can produce voltages up to 25 volts, and the third voltage source can produce up to 12.5 volts (normally used as VCC). The name of the voltage sources will start with a dollar sign followed by V and one number among 1,2, or 3. A voltage source will be assigned to a pin as below.

```
Chips  =    Vpp, A12... PGM_OE;
$V2    =    Vpp;
$V3    =    Vcc;
```

8. Conclusion

To intergrate the information about the section of declaration we will generate the declaration section with the example of Intel 27C128.

First the data type is byte, and the functions needed are program, read, virify, blank-check, data compare, and auto for automatic

sequential executions. The name of the manufacturer is Intel and the device name is 27C128. Since variable names cannot start with a number put D in front of the device name. Pin assignments can be done in the area for "chips" and the voltage sources should be connected to the appropriate pins. Now we are ready to synthesize the description above into an algorithm with DCL.

B. Function Section

This will be consisted of the algorithms which will perform the data transfer. In the declaration section the name of the functions will be listed, and in the function section the functions listed will be implemented.
As an example let's write up an algorithm for read flow of 27C128.

ex) According to the algorithm Vpp is 5 volts and Vcc is 5 volts for reading. The pins which should be controlled are CE, OE, PGM. When the chip is read the CE pin is low and the OE pin is low, but PGM is pulled high. This will be written with DCL.

```
Declaration
      .
      .
      .
      $V2 = Vpp;
      $V3 = Vcc;
      endd
      int read (  )
      {
      @ Vout 2, 5.0;       /* Send out 5 volts to Vpp */
      @ Vout 3, 5.0;       /* Send out 5 volts to Vcc */
      @ Out PGM, @H;       /*  PGM is set to logic high */
      @ Out CE, @L;        /*  CE is set to Logic Low */
      @ Out OE, @L;        /*  OE is set to Logic Low */
      for (;;)
      {
      int          lwlb;
      .
      .
      for (lwlb=0; lwlb<=0xFF; lwlb++)
      {
      @Out A [0..7], lwlb;
```

```
@In  D [0..7], tmp; /* Reads from the data pins. */
@Out CE, @L;
@Out CE, @L;
```

5.2 Device Control Statements

This controls the pins of the device in questions or data transfer. These statement is always preceded with @.

A. Data Transfer Statement

1. @IN

Format : @IN pin_variable, variable;
Function : Reads the data from a chip and pass the data into a variable.
ex) @In D[0..7], tmp;

The data from the 8 data pins will be read into a variable named tmp.

2. @OUT

Format	: @Out_pin name, expr;
pin_name	: a pin name or a variable
expr	: pin contrl constrant as esplained below.

@ L(logic low)
@ H (logic high)
@ O (Turns on a voltage source)
@ F (turns off a voltage source)
@ W (Sets ports for programming)
@ Z (Sets ports for reading)

Function: Programs data from a variable into a chip.

ex1) @OUT A[0..7], @L; Logic low will be applied to the address pins.

ex2)
```
PIN tmp;
tmp = @H;
@OUT CE, tmp;  Logic Low to CE pin
```

ex3) @OUT D[0...7], tmp + 2; Writes the value of tmp + 2 into the data pins.

B. Buffer Statement

1. @LOAD

format:	@Load expr;
expr:	a constant, a variable and an expression
function:	Loads a vlaue from a constant, an expression or a variable into the current address of the data buffer.

ex)
```
@Load tmp:
@Load temp *3;
@Load 4;
```

2. @STORE

format:	@STORE symbol;
symbol:	can only be a variable
function;	A value from the current address of the data buffer will be stored into a variable.

ex) @store tmp; A data from the current address of the data buffer is saved into a variable named tmp.

3. @EXTLOAD

format:	@EXTLOAD expr;
expr:	a constant, a variable or an expression
function:	Loads a value from a constant, an expression, or a constant into the current address of the external buffer.

ex)
```
@EXTLOAD tmp;
@EXTLOAD i/8;
```

@EXTLOAD 4;

4. EXTSTORE

format:	@EXTSTORE symbol;
symbol:	only variable
function :	A value from the current address fo the external buffer will be stroed into a variable.

ex) @EXTSTORE tmp;

5. @SET

format:	@SET expr1, expr2
expr1:	In representing the four byte buffer expr1 represents the two most significant bytes.
expr2:	In representing the four byte buffer address expr2 represents the tow least significant bytes.
function:	This will set the current address of the data buffer with the four byte address.

ex) @SET 3A,2F; This means the current address of 3A2F In the data buffer

@SET 0, 14364 + row + (I<<1) + 1; The result fo the expr2 will be the value of the low two bytes.

6. @INC/@DEC

format:	@INC expr;
	@DEC expr;

exp2:	A value to be encreased or decreased
function:	The current address for the data buffer will be Increased or decreased.

ex) @INC 1;
@DEC temp;

7. @EXTINC/@EXTDEC

This is the same as @INC/@DEC except that the increase or decrease will happen in the external buffer.

8. @Checksumon

format:	@Checksumon;
funciton:	The checksum is calculated.

ex) If (FUNCTION == Read) @checksumon; If the function is "Read", the checksum mode will be turned on.

9. @COMPARE

format:	@ COMPARE expr1 expr2;
expr1;	a variable or a constant
expr2;	a variable or a constant
function:	The data comparison will be performed between the data of the device and the data of the buffer.

ex)
```
int tmp, tmp2;
@in D[0..7], tmp;
if (FUNCION = = Data Compare)
{
@Store tmp2;
@Compare tmp, tmp2;
}
```

C. Delay Command

The commands, which are used for time delay or pulse width control, will be explained.

1. @DELAY

format:	@DELAY expr, @timeunit;
expr:	an expression or a constant. The max is 65535.
tmeunit:	Time unit is specified. Only micro second and milli second will be allowed.
function:	A designated time period will be delayed.

ex) @DELAY 50, @MS; 50 milli second will be delayed.
@DELAY DUR, @US

2. @PINDELAY

format:	@PINDELAY pin, constant1, constant2, constant3;
pin;	pin name
constan1;	Initial value:@H (Logic High), @L (Logic Low), @O (turn on), or @F (turn off)
constant 2;	Final value: @H (Logic High), @L (Logic Low), @O (turn on), or @F (turn off)
constant 3;	Time delay raingling 1 to 65535 with the unit of micro second.
function:	A pin(s) is(are) given an initial value (constant1) and hold for the time (constant3) and changed to the final value.

ex) @PINDELAY CE,@L, @H, 100;

The chip enable will be pulled low for 100 us and pulled high.

D. Voltage Command

1. @VOUT

format:	@BOUT voltage_source, voltage;
voltage_source:	one of the three voltage sources (number)
voltage:	Voltage to be output
function:	Turns on one of the three voltage sources with the voltage pecified.

```
ex)   chips = Vpp, A7,......,Vcc;
      @ V2 = VPP;
      @ V3 = VCC;

      Int function A()
      {
      @VOUT 2, 12.75
      }
```

This turns on the second voltage source with 12.75 volts.

2. @RAISING

format:	@RAISING pin_name, constant1, constant2, constant3;
pin_name:	name of a pin
constant1:	intial voltage
constant2:	final voltage
constant3:	time delay
function:	Set a pin from the initial value to the final value in the amount of time given by the constant3.

ex) @RAISING VPP, 5.0, 12.75, 10; In 10 us the voltage of the VPP pin will be raised from 5 volts to 12.75 volts.

3. @Vector Test

format	: @Vector Test;
function	: Executes vector testing for PLD's

E. Environment Command

1. @Display

format:	@Display column, row, expr, option;
column:	column number for the screen
row:	row number for the screen
expr:	Message to be displayed. A string of characters, a constant, or a variable is allowed. Characters should be enclosed between a pair of double quotation marks.
option:	[,d]; decimal [,x]; hexadecimal

```
ex)   @Display 1, 2, "programming....";
      @Display 1, 7, address;
```

2. @FAIL

format:	@FAIL expr1, expr2, expr3;
expr1:	the two high bytes of the address which is 4 byte long

expr2:	the two low bytes of the address which is 4 byte long
expr3:	the data of the address where the execution failed.
function:	During the execution of a funcion, if an error ocurrs this outputs the current address and its content.

ex)
```
test( )
{
int data1, data2;
int high,  low;
@In D[0..7], data2 ; /* reads from the chip into data 2 */
if (FUNCTION = = Verify)
{
@store data1;
if (data ! = data2)
@FAIL high, low, data2;
/* The address of the failure and its content will be displayed
*/
}
.
.
}
```

3. @EXIT

format : @EXIT;
function: : Exits from the program.

ex) if(data1 != data2) @EXIT;

F. Condition/ Repetition Statement

These are not Device Control Statements, but it is very essential to know conditional/repetition statement.

1. IF

There are two formats.

format1 : if (expression) statement(s)
format2 : if (expression) statement(s) else statement(s)

2. FOR

format : for (expression; relation; expression) statement(s)

3. Continue

format : continue;

function : Abandons the current execution of a for-loop and goes to the next turn of the for-loop.

ex)
```
for (i=0; i<30; i++)
{
@Store tmp;
@OUT A[0..7], i;
if(tmp = = 0XFF) CONTINUE; /* if tmp is  OXFF then go to
the beginning of the for-loop */
@OUT CE, @L;
@OUT OE, @L;
@IN D[0..7], tmp2;
if (tmp != tmp2) Break; /* Exits the for loop */
}
```

4. Break

format : BREAK;

function : Stops the execution and exits.

6. APPENDIX DCL

6.1 System Global Variable

There are some options or parameters which are passed down to the program environment from the user through the keyboard. There are three kinds: a chip address, a buffer address, and a byte split. The system variables listed below will show up in programs without declaration.

A. Chip address

The standard chip address is 4 byte long as below. In the notations below we used abbreviations.

Example of chip address : FFFFFFFF

(S: Start, E: End, H: High, L: Low, W: Word, B: Byte)

1.	S_HWHB	: The first byte or the first two F's
2.	S_HWLB	: The second byte or the third and fourth F's
3.	S_LWHB	: The third byte or the fifth and sixth F's
4.	S_LWLB	: The fourth byte or the 7th and 8th F's
5.	E_HWLB	: The 5th byte or the 9th and 10th
6.	E_HWLB	: The 6th byte or the 11th and 12th F's
7.	E_LWHB	: The 7th byte ro the 13th and 14th F's
8.	E_LWLB	: The 8th byte or the 15th and 16th F's

A through D describes and composes the start address of the chip, and E through H describes and composes the end address of the chip.

B. Buffer Address

ex) standard buffer address: FFFFFFFF

In the notation below we used abbreviations as below.

(B: Buffer H: High L: Low)

1. B_HADDR : High address. This is normally used with @set command. The first two bytes or the first 4 F's.
2. B_LADDR : Low address. This is normally used with @set command. The last two bytes or the last 4 F's.
3. _B_START : This values varies depending on the word format selected. This value will be 0 for all the word format except "Odd" and "Double". When the word format is odd, _B_START_ is 1. When the word format is double, the value of _B_START_ is as follows.

Word format	The value of _B_START_
double 0	0
double 1	1
double 2	2
double 3	3

ex) @INC _B_START_;

4. Split: This value depends on the word format selected.

Word format	The value of Split
byte	1
word	2
Double(0, 1, 2, 3)	3

"Split" indicates the amount of increment in the buffer address.

6.2 DCL EXPRESSIONS

DCL expression will be listed to help users understand syntax of DCL.

```
program
      DECLARATION
            definelist
   ENDD
   definitions

definelist
      DATAUNIT ':'               datatype       sc
      FUNCTIONS                  ftname         sc
      MANUFACTURERS              mfgname        sc
      DEVICES                    devname        sc
      CHIPS '='                  pinassign      sc
      voltage_connect

dataunit
      : BYTE
      | BIT

functions
      : Identifier
      | ftname Identifier

manufacturers
      : Identifier
      | mfgname Identifier

device
      : Identifier
      | devname Identifier

chips
      : Identifier
      | pinassign ',' Identifier
      | pinassign ':' Identifier

voltage_connect
      : empty
```

```
        | voltage_source '=' pinlist sc voltage_connect


voltage_source
        : V1
        | V2
        | V3

pinlist
        : Identifier
        | pinlist ',' Identifier

definitions
        : definition
        | definitions definition
        ;

definition
        : function_definition
        | declaration sc
        ;

function_definition
        : base_type Identifier '('
          parameter_listopt ')'
          parameter_declarations
          compound_statement

parameter_list
        : empty
        | Identifier
        | Identifier ',' parameter_list

parameter_declarations
        : empty
        | parameter_declarations parameter_declaration

parameter_declaration
        : base_type parameter_declarator_list sc

parameter_declarator_list
        : Identifier
        | parameter_declarator_list ',' identifier

declarations
        : empty
        | declarationmain  declarations

declarationmain
        : declaration sc

declaration
        : base_type storagelist


storagelist
        : storageid
        | storagelist ',' storageid


storageid
        : identifier initdata
        | identifier '[' arraysize ']' arrayinitdata

arraysize
        : empty
        | constant


initdata
        : empty
        | '=' O constant

arrayinitdata
        : empty
        | '=' '{' O initdatalist '}'

initdatalist
        : constant
        | initdatalist ',' constant

base_type
        : INT
        | CHAR
```

```
        | PIN
        | VOLTAGE

statements
        : empty
        | statements statement

statement
        : sc
        | expression   sc
        | compound_statement
        | RETURN        sc
        | RETURN binary sc
        | BREAK         sc
        | CONTINUE      sc
        | if (expression) statement
        | if  (expression) else statement
        | for (expression-1opt; expression-2 opt;expression-3opt)
          statement
        | while ( expression ) statement
        | pin_control_statement sc
        ;

pin_control_statement
        : IN id_list ',' id_list
        | OUT id_list ',' binary
        | LOAD binary
        | STORE  id_list
        | EXTLOAD  binary
        | EXTSTORE id_list
        | VOUT  constant ',' binary
        | DELAY binary   ',' timeunit
        | PINDELAY identifier ',' constant ',' constant ',' constant
        | SET binary ',' binary
        | INC binary
        | INC binary
        | DEC binary
        | DISPLAY binary ',' binary ',' binary optlist
        | RAISING identifier ',' constant ',' constant ',' constant
        | FALLING identifier ',' constant ',' constant ',' constant
        | CHECKSUMON
        | COMPARE binary ',' binary
        | VECTORTEST
        | FAIL   binary ',' binary ',' binary ',' binary ',' binary
        | EXIT

compound_statement
        : { declarations statements }

timeunit
        : MS (millisecond)
        | US (microsecond)

optlist
        : empty
        | ',' DISPHEX
        | ',' DISPDEC


expression
        ; binary
        | expression ',' binary
        | error ',' binary
        | expression error
        | expression ',' error

binary
        : id_list
        | constant
        | '(' binary ')'
        | '(' error  ')'
        | Identifier
        | '-' binary  '-'
        | '~' binary  '~'
        | '!' binary  '!'
        | ++ id_list
        | -- id_list
        | id_list ++
        | id_list --
        | binary '+'   binary
        | binary '-'   binary
        | binary '*'   binary
        | binary '/'   binary
        | binary '%'   binary
```

```
        | binary '&'   binary
        | binary '|'   binary
        | binary '^'   binary
        | binary '<<'  binary
        | binary '>>'  binary
        | binary '!&'  binary
        | binary '!||' binary
        | binary '!^'  binary
        | id_list '='  binary
        | id_list '+=' binary
        | id_list '-=' binary
        | id_list '*=' binary
        | id_list '/=' binary
        | id_list '%=' binary
        | id_list '&=' binary
        | id_list '^=' binary
        | id_list '<<='binary
        | id_list '>>='binary
        | binary '<'   binary
        | binary '>'   binary
        | binary '>='  binary
        | binary '<='  binary
        | binary '=='  binary
        | binary '!='  binary

id_list
        : Identifier
        | Identifier '[' binary ']'
        | Identifier '[' constant .. constant ']'

constant
        : Constant
        | CHAR
        | Hex
        | Bin
        | Vol
        | LOW
        | HIGH
        | OFF
        | ON
        | GROUNDOUT
        | CLOCKOUT
        | IN
        | OUT


optional_argument_list
        : empty
        | argument_list


argument_list
        : binary
        | argument_list ',' binary

rp      : ')'
sc      : ';'
rr      : '}'
```

BOOK III

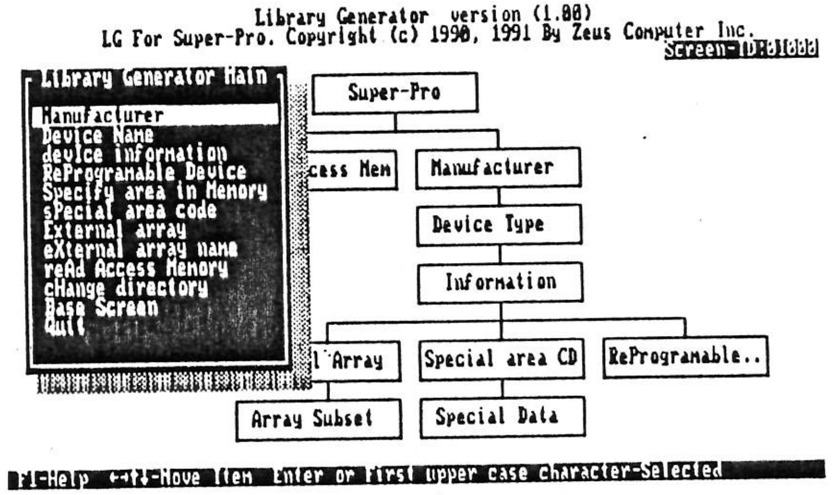
LG (Library Generator)

This will receive the object file generated from AG and generate library files for the users of regular version of SUPERPRO. After the registration, SP.EXE will be able to access the algorithm for programming.

INTRODUCTION

This software will deal with all the data pertinent to the manufacturers and the devices. After the registration of the object file which is generated in the algorithm generator software (AG. EXE), the user interface software (SP. EXE) will be able to access and select the chip in question after the registration in the existing library using LG.EXE.

There are 12 root level submenus as the figure below.



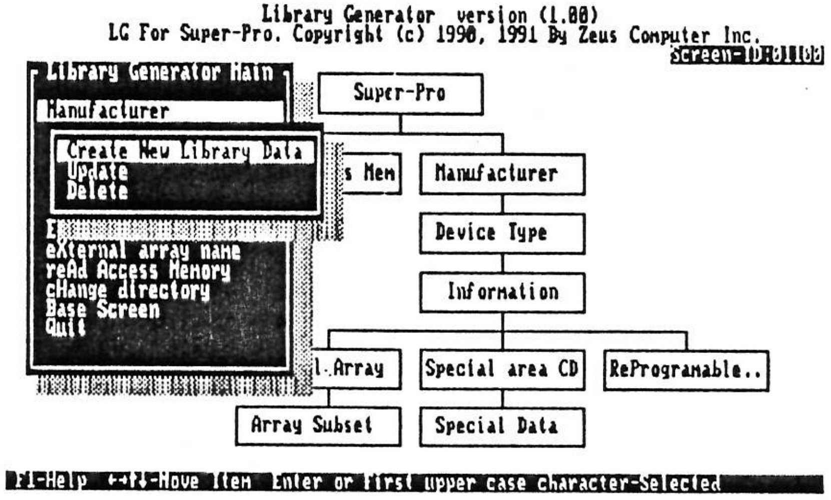
```
              Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                   Screen-ID:01000

Library Generator Main
  Manufacturer
  Device Name
  device information
  ReProgramable Device
  Specify area in Memory
  sPecial area code
  External array
  eXternal array name
  reAd Access Memory
  cHange directory
  Base Screen
  Quit

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Main Menu

1 MANUFACTURER

This manages the data base of manufacturers and has three submenus.

Create new library Data	: Adds the name of a manufacturer
Update	: Edits the name of a manufacturer
Delete	: Deletes the name of a manufacturer



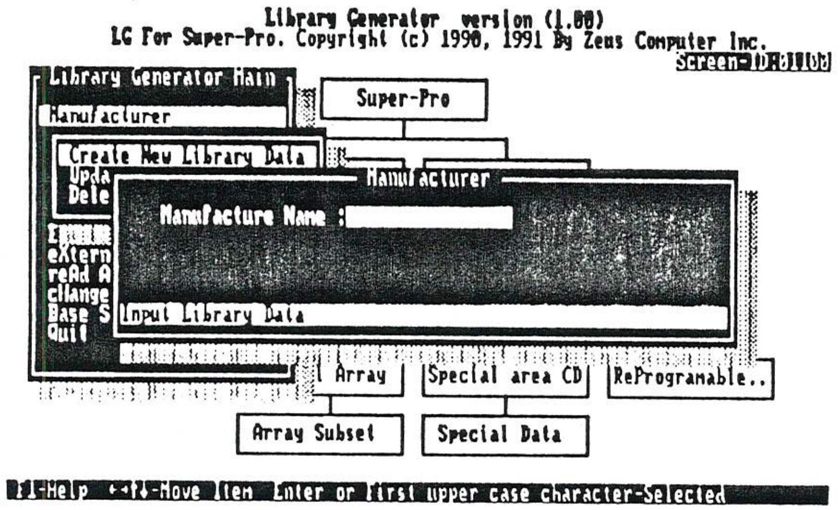
```
Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
```

Figure 1.1 Manufacturer

1.1 Create New Library Data

This registers a new manufacturer in the library. If this menu is selected the menu as in the figure below will show up. If there is the manufacturer already in the library an error message will show up. Otherwise, this will create a new manufacturer in a library. This menu will ask a user whether to add more names of manufacturers.

Names of manufacturers will be composed of combinations of letters and numbers up to 15 characters.



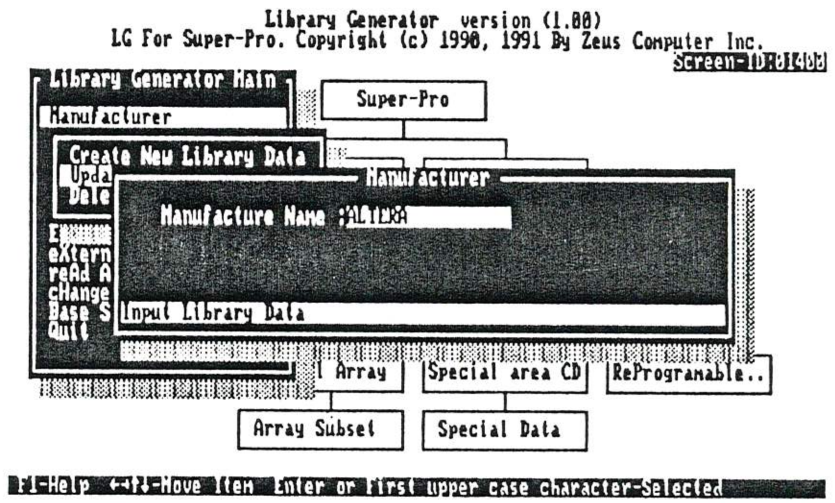
```
Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
Manufacture Name :
F1-Help ←→↑↓-Move Item Enter or First upper case character-Selected
```

Figure 1.2 Create New Library Data

1.2 Update

This will let users edit the name of a manufacturer. If this is chosen a manufacturer's name should be selected. To select a manufacturer, type * and press the return key and highlight the entry to be edited and press the return key.

Then it will show the screen with the name of the manufacturer selected and users can type over or edit the name into the desired name. There are two ways of abandoning the change mode. One way is to pressing the escape key and the other is answering NO to the question which is asked at the end of editing. To save the change users can answer YES.

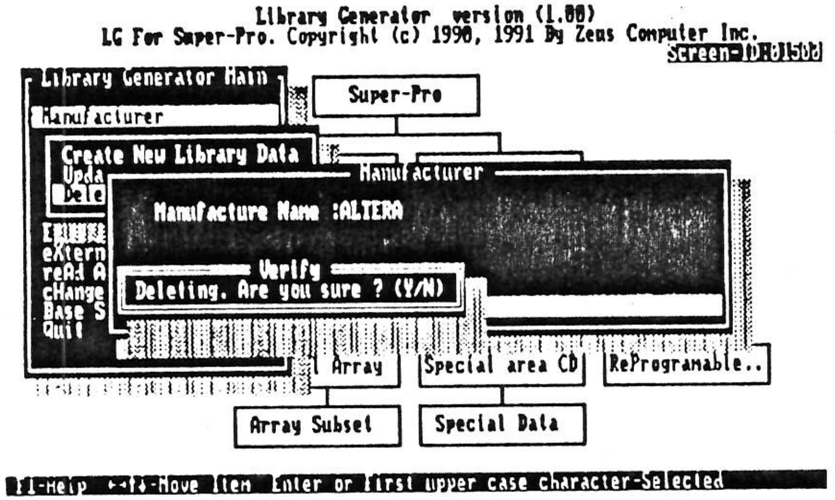


```
Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
Manufacture Name :ALTERA
F1-Help ←→↑↓-Move Item Enter or First upper case character-Selected
```

Figure 1.3 Updating a manufacturer

1.3 Delete

This will let users delete the names of manufacturers. The procedures of "Delete" is same as "Update" procedures described in "Update". Just select the name of a manufactures as in "Update" and press the return key. Then a message for deletetion will show up. Answering YES will delete the entry selected. Once it is deleted all the devices under it will not be able to be selected. Thus, extreme caution is required.



```
                 Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                  Screen-ID:01500

Manufacture Name :ALTERA

Verify
Deleting. Are you sure ? (Y/N)

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

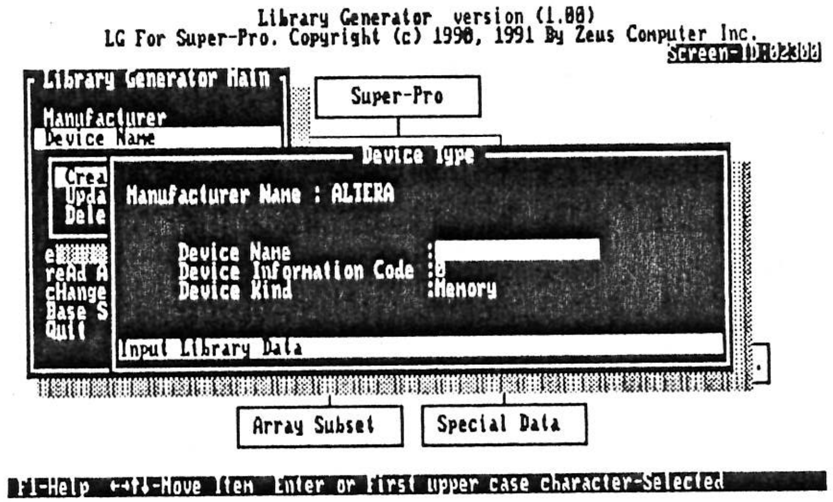
Figure 1.4 Deleting Manufacturers

2 DEVICE NAME

This manages the names of devices and the information codes which have all the information about the devices in its tables.

2.1 Create new Library Data

This will register the new device. If this name is selected the manufacturer of the device will be entered. If the manufacturer of the device is not existing in the menu, manufacturer "Create New Library Data" should be selected and the manufacturer should be created. After the manufacturer has been selected the menu as below will be displayed.



```
                 Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                  Screen-ID:02300

Manufacturer Name : ALTERA

     Device Name                :
     Device Information Code    :0
     Device Kind                :Memory

Input Library Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

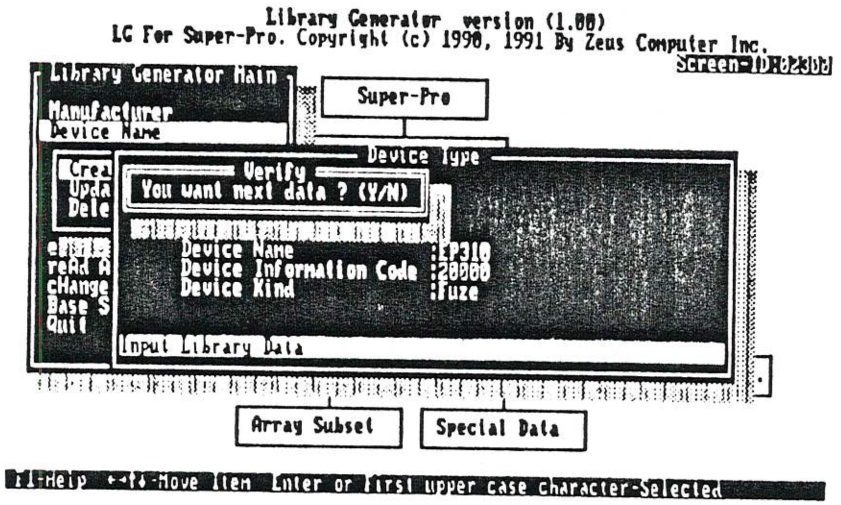
Figure 2.1 Device Name Create

A. Manufacturer Name

This shows the manufacturer of the device.

B. Device Name

The maximum number of 15 characters are allowed.



```
                 Library Generator  version (1.00)
 LC For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                     Screen-ID:02300
Library Generator Main
Manufacturer
Device Name
                              Device Type
          Verify
 You want next data ? (Y/N)

        Device Name               :EP310
        Device Information Code   :20000
        Device Kind               :Fuze

Input Library Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 2.2 Device Name Selected

C. Device Information Code

This is the information code of the table which has the detailed information about the device in question. The range of the information code is from 1 to 32767 in decimal.

D. Device Kind

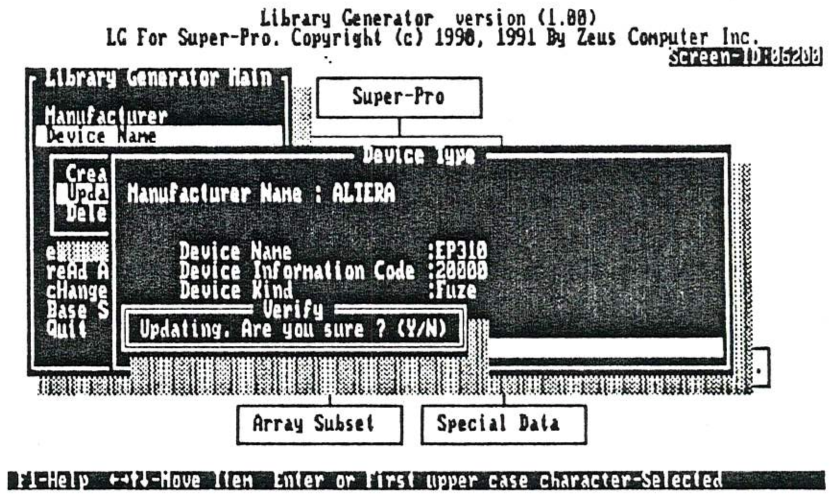
This indicates the type of a chip which is registered. If the chip is ROM or Single, this should be set "Memory" and if the chip is PAL or GAL this should be set "Fuse".

To abandon the device registration press the escape key. If the registration of the device is done the menu will ask for another registration. By answering "Y" continuous update for more devices is possible. This is useful since normally one algorithm can implement numerous devices. Especially, our DCL can utilize the "if" statement to encompass the other similar algorithms. In the case of ROM, it is common for one algorithm to program 20 or 30 chips of different specifications.

2.2 Update

This can be used to update or edit the information of the device generated in the menu of "Device Name Create New Library Data". If this is selected, type * in the space for the name of the manufacturer and press the return key. All the preregistered manufacturers will show up. Highlight an entry then a box for a device name will be displayed.

Either type the exact part name or type * to display all the entries. In case all the entries are displayed the desired entry should be highlighted and the return key should be pressed for selection. After the selection, an information table will be displayed. Edit the name of the device, the information code, and the kind of the device. The question for affirmation will be displyed.

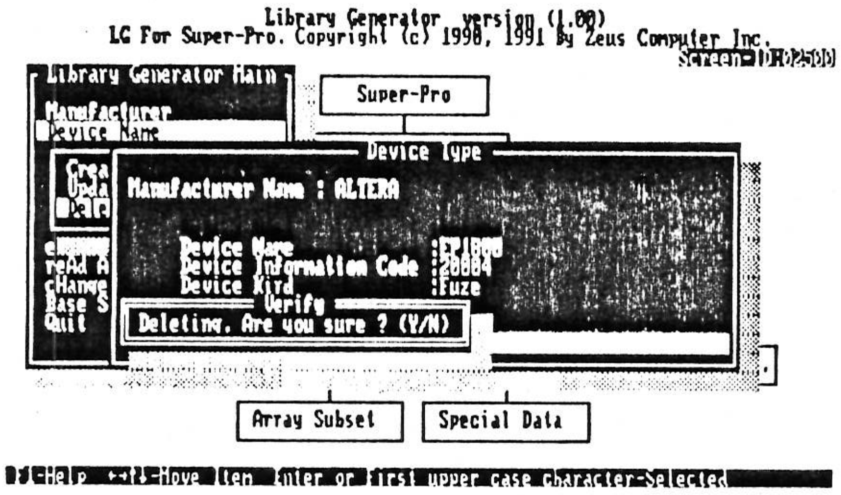


```
                 Library Generator  version (1.00)
 LC For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                     Screen-ID:06200
Library Generator Main
Manufacturer
Device Name
                              Device Type
 Manufacturer Name : ALTERA

        Device Name               :EP310
        Device Information Code   :20000
        Device Kind               :Fuze
          Verify
 Updating. Are you sure ? (Y/N)

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 2.3 Update

2.3 Delete

This is very similar with "Device Name -> Update". The only difference is that this menu is deleting the entry instead of editing it. Please refer to the menu of "Device Name -> Update".



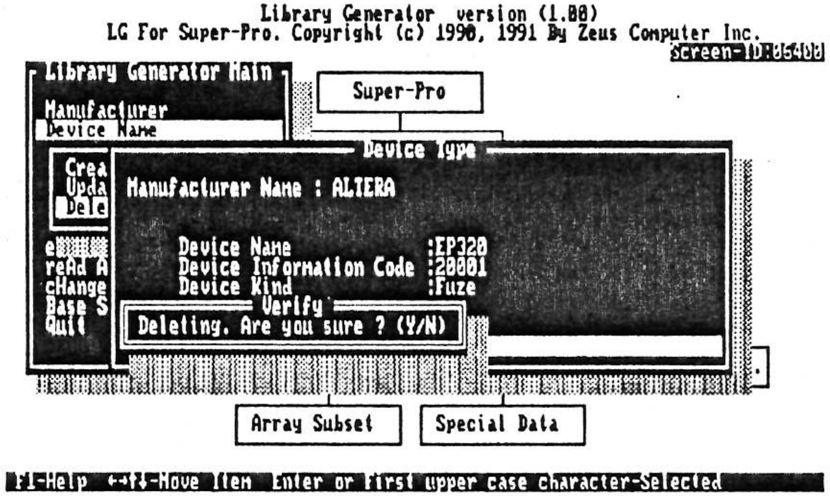
```
                Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                  Screen-ID:02500
Library Generator Main
Manufacturer          Super-Pro
Device Name
                      Device Type
Crea  Manufacturer Name : ALTERA
Upda
Dele
          Device Name                  :EP1800
          Device Information Code      :20004
          Device Kind                  :Fuze
                   Verify
      Deleting. Are you sure ? (Y/N)

                 Array Subset    Special Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 2.4 Delete

3 DEVICE INFORMATION

After the name and the outline of the device is registered the detailed information will be specified in this menu. There are three submenus; "Create New Library Data", "Update", & "Delete". "Update" and "Delete" are operated in the same way as this menu. Thus only the menu of "Create New Library Data" will be described.

3.1 Create New Library Data



```
                Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                  Screen-ID:06400
Library Generator Main
Manufacturer          Super-Pro
Device Name
                      Device Type
Crea  Manufacturer Name : ALTERA
Upda
Dele
          Device Name                  :EP320
          Device Information Code      :20001
          Device Kind                  :Fuze
                   Verify
      Deleting. Are you sure ? (Y/N)

                 Array Subset    Special Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 3.1 Device Information

A. Information Code

This ranges from 1 to 32767 in decimal number. If there is an existing information code an error message will be displayed, and the data input will be ignored.

B. Chip Size or Max Fuse

For ROM's the size of memory will be entered and for PAL's and GAL's the number of fuses will be entered. Hexa decimal numbers will be accepted and the maximum 8 digits will be allowed.

C. Max of Pin

The number of pins in a device will be entered. It will take a decimal number and 40 is max.

D. Max of Input Line

In a PAL and a GAL, the number of input lines will be entered.

E. Limit of Program

In an algorithm a number of trials per byte for successful programming is given. If programming is seccessful within that limit, the next byte will be programmed. If a byte cannot be programmed within the given limit the software will exit the system giving up programming the chip. We named this number "Limit of Program".

F. Algorithm Name

The max 8 characters are allowed. The file name with the suffix of LEF which is generated in AG.EXE will be entered here without LEF.

G. Sub Device Exist

This is normally used for GAL's which can configure other devices. Press the space bar to indicate the capability of configuring other devices such as RAL's.

H. Extern Array Exist

If there is a MES or an encryption table use the space bar to indicate so.

I. Gang support

Whether the chip is programmed with the 4 socket adaptor (made by XELTEK) or not, it will be indicated here.

J. Usage Algorithm

K. External array code

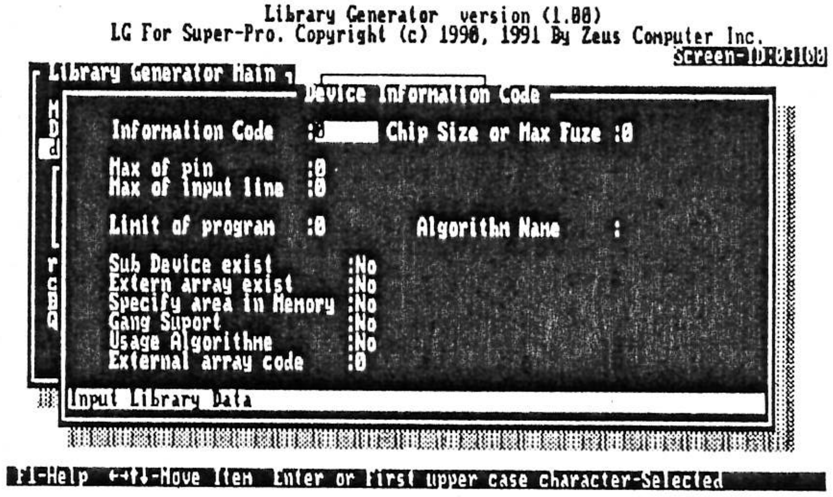
If the menu "Extern Array Exist" has been set YES, a code for an external array table will be entered.

4 REPROGRAMMABLE DEVICE

If the menu, "Device Information -> Create New Library Data" is set YES, the information about the sub devices will be entered. Since "Update" and "Delete" have the same usage and organization the explanation will be omitted. This is used for GAL's which can configure many sub devices such as RAL's.

4.1 Create New Library Data

If this menu is selected the device information code will be asked. In this case only the devices which have the subdevices will be shown. If there's no such devices the error message saying "Search key not found" will be displayed. After the device information code has been selected a submenu as below will show up.



```
                 Library Generator  version (1.00)
   LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                        Screen-ID:03100
 Library Generator Main
                   Device Information Code
     Information Code    :0        Chip Size or Max Fuze :0

     Max of pin          :0
     Max of input line   :0

     Limit of program    :0            Algorithm Name      :

     Sub Device exist           :No
     Extern array exist         :No
     Specify area in Memory     :No
     Gang Suport                :No
     Usage Algorithme           :No
     External array code        :0
  Input Library Data
F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 4.1 Reprogrammable Device

A. Sub Device Name

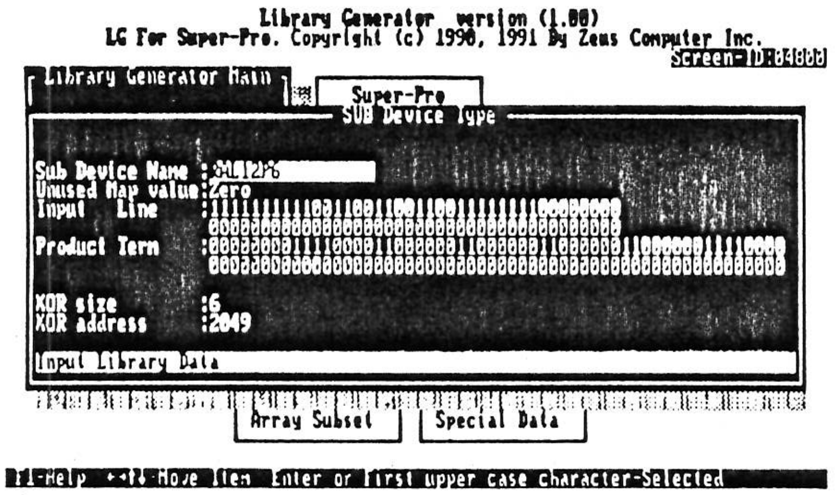
The name of a subdevice up to 15 characters will be entered.

B. Unused Map Value

In configuring RAL's this sets the default values for the unused product terms and input lines. Zero or one will be used to indicate whether the product terms and the input lines are used or not.

C. Input Line

This indicates the used and unused input lines according to the default values set for the unused map value.



```
                     Library Generator  version (1.00)
        LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                           Screen-ID:04800
 Library Generator Main      Super-Pro
                        SUB Device Type

 Sub Device Name :20L12P6
 Unused Map value:Zero
 Input    Line   :111111111100110011001100111111110000000
                  000000000000000000000000000000000000000
 Product Term    :0000000011110000110000001100000011000000110000001111000
                  0000000000000000000000000000000000000000000000000000000

 XOR size        :6
 XOR address     :2049

 Input Library Data

                 Array Subset      Special Data

 F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 4.2 Reprogrammable device

D. Product Term

This indicates the used and unused product terms according to the unused map value.

E. XOR Size

The size of XOR will be specified.

F. XOR Address

When XOR array is used this will indicate the beginning address of XOR array.

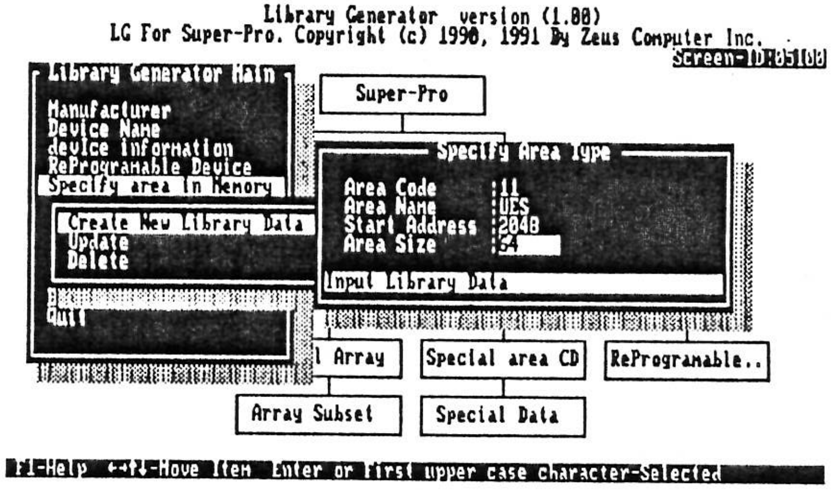
5 SPECIFY AREA IN MEMORY

When there is a need to view and edit the certain area of the memory this is used. There are three sub menus : create New Library Data, Update and Delete. Update and Delete will have the same sub menus and structures. Thus they will not be explained here. The only differences are their functions. Create new library data will create, and Update will edit, and Delete will delete.

5.1 Create New Library Data

A. Area Code

A decimal number ranging from 1 to 32767 will be entered. This will be the number of the table.



```
Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
Screen-ID:05100

Specify Area Type
Area Code      :11
Area Name      :UES
Start Address  :2048
Area Size      :64
Input Library Data
```

Figure 5.1 Specify area in Memory

B. Area Name

A name with the max 15 characters will be entered for the name of the area chosen.

C. Start Address

This will indicate the start address of the area which has been designated.

D. Area Size

This specifies the size of the designated area. The size of the buffer will be allocated starting from the address specified above, in the menu, "start address".

6 SPECIFY AREA CODE

Onto the device which will use the designated area, the area code will be registered. First the "Device information code" will be specified. If "specify area exist" in the device informmation table is not set YES, this menu will have no effect. The menu as below will show up.

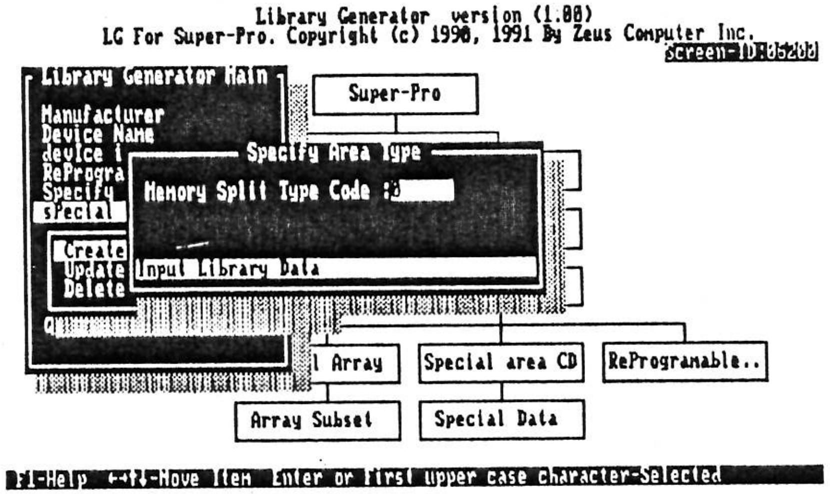
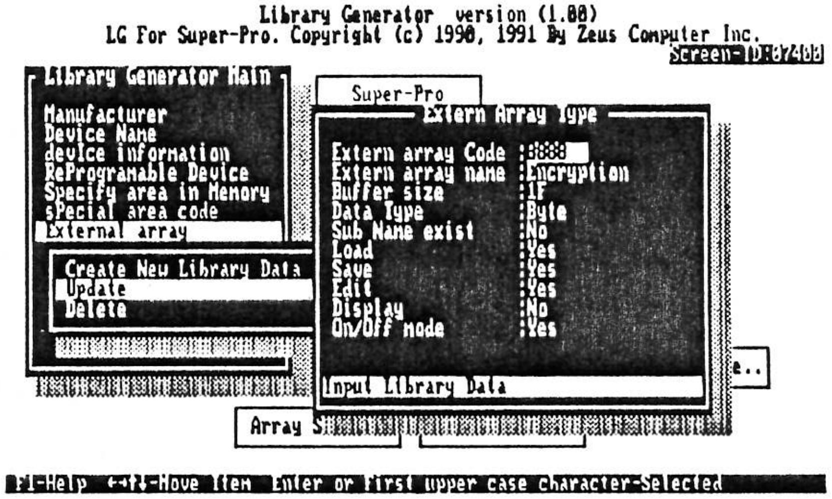

```
Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
Screen-ID:05200

Specify Area Type
Memory Split Type Code :0
Input Library Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 6.1 Special Area Code

7 EXTERNAL ARRAY

If there is an internal area (USE buffer) this menu will deal with that area. When this menu is used, the code here will be entered. In the menu "Device Information -> External Array Code".



```
                 Library Generator  version (1.00)
       LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                           Screen-ID:07400

 Library Generator Main     Super-Pro
                             ── Extern Array Type ──
 Manufacturer
 Device Name                 Extern array Code  :8888
 device information          Extern array name  :Encryption
 ReProgramable Device        Buffer size        :1F
 Specify area in Memory      Data Type          :Byte
 sPecial area code           Sub Name exist     :No
 External array              Load               :Yes
                             Save               :Yes
  Create New Library Data    Edit               :Yes
  Update                     Display            :No
  Delete                     On/Off mode        :Yes

                             Input Library Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 7.1 External Array

A. External Array Code

The decimal number ranging form 1 to 32767 will be allowed.

B. External Array Name

The name with the max of 15 characters will be entered.

C. Buffer Size

The size of the external array will be entered. The max is the five decimal number.

D. Data Type

This determines whether the data is in the units of bits (PLD's) or bytes (Memory).

E. Sub Name Exist

If there is a sub name select YES.

F. Load

If the content of the external array needs to be loaded as a file this menu will be set to YES.

G. Save

If the content of the external array needs to be saved as a file, this menu is set for YES.

H. Edit

If users want to edit or view the external array, this menu should be set YES.

I. Display

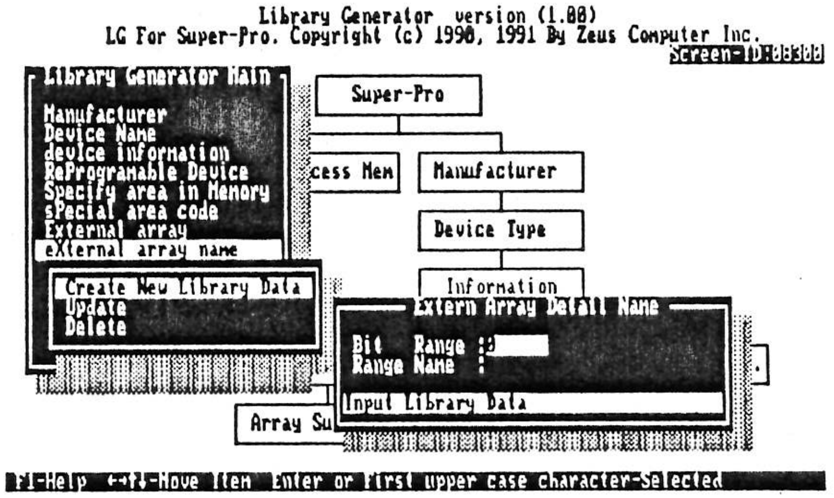
To view the content of the external array in read only mode this is set YES.

J. ON/OFF Mode

If this is set YES, it will initiate exclusive NORing between the data of the chip and the encryption table. In the user interface software, SP.EXE, Extern mode (exclusive NORing) can be enabled or disabled.

8 EXTERNAL ARRAY NAME

If "External Array -> Sub Name Exist" is set YES this table should be filled. First, external array code should be determined.



```
                Library Generator  version (1.00)
    LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                        Screen-ID:00300
```

```
Bit   Range :0
Range Name  :
Input Library Data
```

```
F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 8.1 External Array Name

8.1 Bit Range

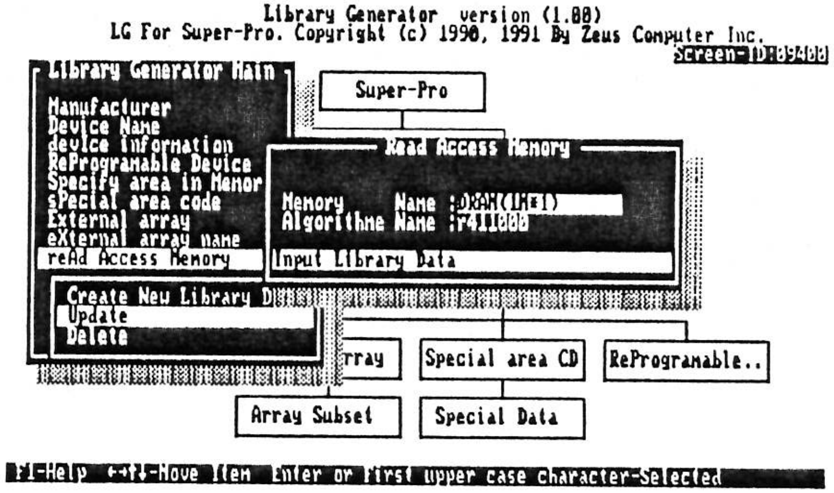
The size of bits will be entered in decimal number.

8.2 Range Name

The range of bits designated above will be named with up to 15 characters.

9 RANDOM ACCESS MEMORY

RAM (Random Access Memory) will be dealt.



```
                 Library Generator  version (1.00)
    LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                          Screen-ID:89400
Library Generator Main
 Manufacturer
 Device Name
 device information
 ReProgramable Device
 Specify area in Memor
 sPecial area code
 External array
 eXternal array name
 reAd Access Memory

   Create New Library D
   Update
   Delete

                         Read Access Memory
              Memory      Name :DRAM(1M*1)
              Algorithme Name :r411000
             Input Library Data

Super-Pro   rray   Special area CD   ReProgramable..   Array Subset   Special Data

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 9.1   Random Access Memory

9.1 Memory Name

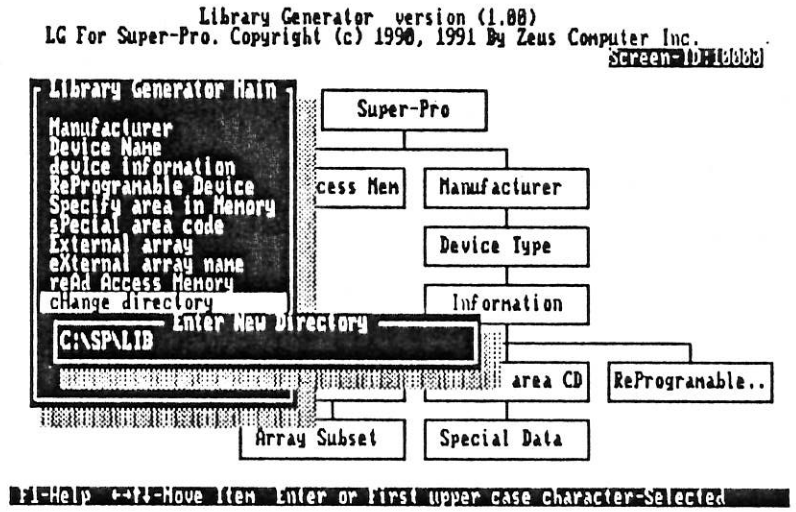
The name of the RAM will be entered

9.2 Algorithm Name

The algorithm generated from Algorithm Generator will be entered without the extension. The max of 8 characters will be allowed.

10 CHANGE DIRECTORY

The files of Libraries will be generated from the software, (LG), and will be saved in a directory designated in this menu.

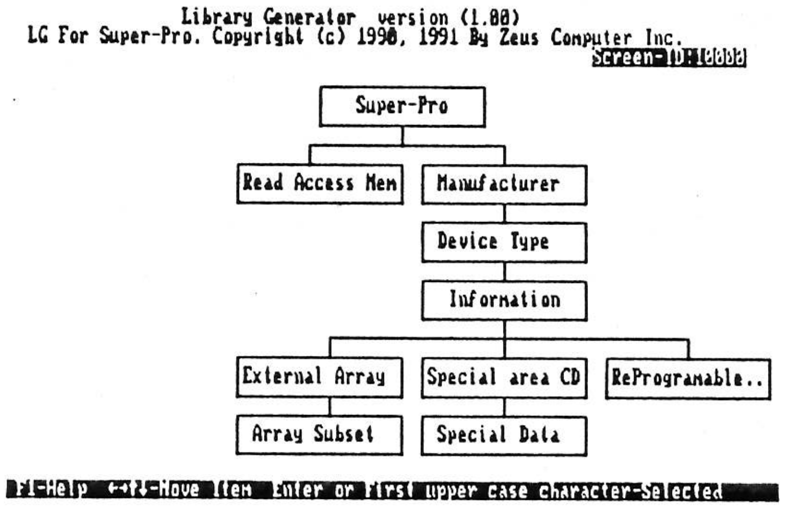


```
                 Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                    Screen-ID:10000

Library Generator Main
 Manufacturer
 Device Name
 device information
 ReProgramable Device
 Specify area in Memory
 sPecial area code
 External array
 eXternal array name
 reAd Access Memory
 cHange directory
        Enter New Directory
 C:\SP\LIB

F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 10.1 Change Directory

11 BASE SCREEN

The base screen shows the top down structure of the library generator. For the users to view the base screen, this menu will remove the main menu on the left. To bring back the main menu press any key.



```
              Library Generator  version (1.00)
LG For Super-Pro. Copyright (c) 1990, 1991 By Zeus Computer Inc.
                                                    Screen-ID:10000
```

```
F1-Help  ←→↑↓-Move Item  Enter or First upper case character-Selected
```

Figure 11.1 Base Screen

12 QUIT

If this is selected, the configuration file which has all the setups will be saved and quit to DOS.