TABLE OF CONTENT

Š

********** N O T E S **********

NOTE 1 :

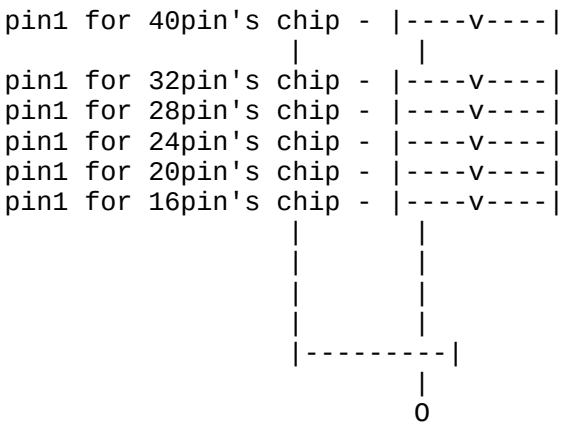
If there is any unclear statement in this manual, please print the 'MANUAL.DOC'file from the disk for reference. Should this manual be modified, MAMUAL.DOC will be also corrected immediately.

NOTE 2:

First of all please read this manual from CHAPTER I to CHAPTER III before you start to instal and use this programmer.

NOTE 3:

Pin assignment of the SOCKET as below:



NOTE 4:

With your computer turned on and off, or your executed files replaced, please don't put the chips on the testing socket. Wait until the main display screen appears. Thus, you can avoid destroying your chips because of error operation.

NOTE 5:

In the programming process, the BUSY LED will light. please don't put the chip on or take it from the socket at this moment. The programming voltage on the socket may destroy the chip, so please wait until the BUSY LED is turned off then start next step.

NOTE 6:

After your computer is turned on, whether the programmer is connected with D-25 CABLE or not, it won't affect the computer and won't destroy the programmer, too.

Š

NOTE 7:

This programmer owns a warrant to succeed a 99 % up performance. Suppose your programmer can't perform well, please contact us. We will help solve problem for you. Should any problem arise in the programming process, please don't waste too many chips to test. First read "DAMAGE CHECK", make sure where the problem is, solve it then program again.

NOTE 8:

Our company will continue offering new softwares. If any chips were found with softwares unavailable for programming from us, please contact us. Wemay have new software at hand. However if we don't have, we will write it for you at our soonest.

Š

CHAPTER I : INTRODUCTION

1.1 Product Overview

(1) This programmer is not individually used. It has to be connected with PC XT/AT. With the help of different softwares in the diskette, it can be able to program different ICs.

(2) This programmer is an Universal Programmer & Tester. It can program ICs as the following list:

* EPROM / EEPROM
* PAL / FPL / GAL / PEEL / EPLD
* BPROM
* Microcomputer

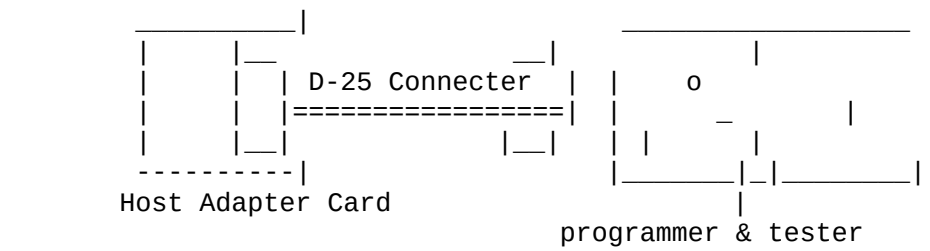
it can also test ICs as the following list:

* 74 series TTL

* 40/45 series CMOS
* SRAM / DRAM

(3) Our company will continue supplying new softwares to meet your requirement to program new ICs.

1.2 Name of the Parts



1.3 Specifications

(1) Host Adapter Card:

The Host Adapter Card includes two types (CSA-03 ,CSA-04). The function of these two types are same and transferable.

*OUTPUT VOLTAGE : +5V, regulated & current limited 1.5A

+12V, -12V.

*OUTPUT SIGNAL : D0 -D7, A0 -A4, /IOR, /IOW. CLK, /RESET & I/O READY.

*I/O ADDRESS SETUP: according to properly adjusted I/O PORT

a. MODEL CSA-03 can select one from the 8 sets of I/O PORTs .(200--2FF), such as

(200-21F); (220-23F); (240-25F); (260-27F);
(280-29F); (2A0-2BF); (2C0-2DF); (2E0-2FF).

Š

b. MODEL CSA-04 can select one from the 16 sets of I/O PORTs(200--3FF), such as

(200-21F); (220-23F); (240-25F); (260-27F);
(280-29F); (2A0-2BF); (2C0-2DF); (2E0-2FF);
(300-31F); (320-33F); (340-35F); (360-37F);
(380-39F); (3A0-3BF); (3C0-3DF); (3E0-3FF).

* PCB SIZE: 105 mm (L) * 90 mm (W)
* CABLE LENGTH: 100 cm (D type, 25 PINs connector)
* WEIGHT: 250 gm.

(2) PROGRAMMER & TESTER

* Supply power

STAND BY	ACTIVE
+5v 470mA(MAX)	+5V 650mA(MAX)

+12V 150mA(MAX)	+12V 1A (MAX)

* Three sets of programming voltage
  VL : 1.5V -- 9V 800mA (MIN) 0.15 V/div
  VM : 1.5V --25V 400mA (MIN) 0.10 V/div
  VH : 1.5V --25V 400mA (MIN) 0.10 V/div
* Each pin driver power : 400 mA & 800 mA
* Standard timer for hardware programming:16uS,128uS & 512uS
* 2MHz frequency is supplied for 8748, 8751, Z8 ... micorcomputer.
* A 40-PINs Universal Narrow ZIP Socket
* Programmable Pin Driver
  * Each Pin Driver has its own short circuit protected

(3) Software

Available softwares are listed as below. New softwares will have a continuous increase. Please contact us anytime for a requiry of programming any special IC.

EPP02	EXE	.... for EPROM & EPPROM
SETUP02	EXE	.... for I/O address setting
48P02	EXE	.... for 8748/42/48/49
51P02	EXE	.... for 8744/51/52
BPP02	EXE	.... for Bipolar PROM
DMT02	EXE	.... test program for 74/54 TTL 40/45 CMOS, SRAM, DRAM
PAP02	EXE	.... for PAL(MMI,NS,TI)
PAPA02	EXE	.... for PAL(AMD/CYPRESS/TICPAL)
FPL02	EXE	.... for Signetics PLHS 153/173 FPLA

Š

S18P8-02	EXE	.... for Signetics PLHS18P8
PALFORM (DIR)		.... fuse map for PAL's
PEEL02	EXE	.... for PEEL(AMI/GOULD/ICT/HYUNDAI)
S-GAL02	EXE	.... for GAL 16V8/20V8(Signetics)
GAL02	EXE	.... for GAL 16V8/20V8(LATTICE /NS / SGS /VLSI)
22V10-02	EXE	.... for GAL 22V10 (AMD / ATMEL / CYPRESS / TI / TICPAL)
PARTS02	LST	.... parts list for programmable & testable ICs
HEXOB02	EXE	.... to convert .HEX file to .OBJ file for INTEL 80/86 & MOTOLORA S1/S2
DASM48	EXE	.... dis-assembler for 8748/49
DASM51	EXE	.... dis-assembler for 8751/52
TEST02	EXE	.... voltage adjust & hardware check for programmer
IOCHK02	EXE	.... I/O address check
MANUAL	DOC	.... user's manual
VERSION	DOC	.... list for software upgraded

Š

CHAPTER II EQUIPMENT

2.1 Unpacking & Inspection

Becareful to unpack the package and check details as below:

* Host Adapter Card
* Connecting Cable
* Programmer Unit
* Utility Disk
* User's Menual

(Please check in detail to see if you got everything in it. Should there be any damage or shortage, please contact the dealer for replacement.)

2.2 Computer System Required

* IBM PC XT/AT or compatible system (Min. 384K RAM)
* PC DOS V2.0 later or compatible DOS

2.3 I/O ADDRESS SETTING

You can skip over this section and direct to CHAPTER III. If you meet problem when you execute CHAPTER III, you can return to this section for answer.

I/O address setting includes two parts: one is the setting on Host Adapter Card and the other is the setting on the software-SETUP.DAT. Anyway these two settings must be same.

(1) I/O address setting on the Host Adapter Card:

There are 2 different models for the Host Adapter Card. They have the same function and any one of them can connect with the programmer to be used but their I/O address settings are different.

a. Model CSA-03 model :
There is one set of JUMPER (JP1) on the Host Adapter Card. The relation between the JUMPER position and the system I/O address (200H-2FF) as below:

Š

JUMPER (JP1)	I/O ADDRESS
1	200-21F
2	220-23F
3	240-25F
4	260-27F
** 5 *****	280-29F ** preselected
6	2A0-2BF in our factory
7	2C0-2DF
8	2E0-2FF

b. Model CSA-04:
There are two sets of JUMPERs (JP1, JP2). The relation between the JUMPER position and the system I/O address (200H--3FF) is as below:

JP2 (1--2 3) -> I/O ADDRESS 200 --2FF

JUMPER (JP1)	I/O ADDRESS
1	200-21F
2	220-23F
3	240-25F
4	260-27F
** 5 ***	280-29F ** preselected
6	2A0-2BF in our factory
7	2C0-2DF
8	2E0-2FF

JP2 (1 2-3) -> I/O ADDRESS 300-3FF

1	300-31F
2	320-33F
3	340-35F
4	360-37F
5	380-39F
6	3A0-3BF
7	3C0-3DF
8	3E0-3FF

* As finished , the setting way of JP1 & JP2 is also the I/O address on the programmer.

(2) SOFTWARE SETTING

With every kind of programming softwares executed, softwares will call to the I/O address in SETUP. DAT, therefore the I/O address on the Host Adapter Card and in SETUP.DAT must be same.

Execute SETUP02.EXE to select the JUMPER's setting way on the Host Adapter Card. After executing, I/O address will be saved in SETUP.DAT file automatically.

Š

For example:

A> SETUP02 <enter>

```
I/O address setup for Host Adaptor Card    V2.0
                      (C) JAN 25   '89
 _____________________________________________
|Current I/O address is:     280-29FH         |
|      JP1 at     PIN 5 .     JP2 at PIN 1&2              |
|_____________________________________________|

       Change     I/O address
(A) For no JP2 on the card, or JP2 is pin 1 & 2
  shorted
(B) For JP2 is pin 2&3 shorted
 <ESC> Exit & Save I/O address
 _____________________________________________
| I/O address is from 200H    to 2FFH              |
|       The jumper of JP1 at which position(1-8) :  |
|_____________________________________________|
```

REMAKE : If the Host Adapter Card is Model CSA-03, please select "A". If the Host Adapter Card is Model CSA-04, you may select "A" or "B". The relation with JUMP 2 is as below:

1. Please choose "A" as JP2(1--2 3) on the Host Adapter Card. (200H-2FFH)
2. Please choose "B" as JP2(1 2--3) on the Host Adapter Card. (300H-3FFH)

<ESC> : save file, return to DOS

* As finished in our factory, the I/O address of the software will be preselected 280H--29FH.
For example:

```
A>      TYPE  SETUP.DAT
A5
||
||__ JP1 :   PIN 5 (the 5th set)
|___ JP2 :   (1--2 3)
```

Š

CHAPTER III INSTALLATION & TESTING

3.1 Hardware Installation

1. Please turn off the power of your computer.
2. Plug the Host Adapter Card into any slot on your computer.
3. Use the cable to connect the Host Adapter Card with your programmer.

3.2 Testing

1. Power on your PC and load DOS.
2. After A> displayed on the screen, execute IOCHK02.EXE program. (There must be SETUP.DAT. in the diskfiles so as to test whether the I/O ADDRESS on the programmer is coordinative with the other cards)
3. If an error occurs, please refer to section 2.3, " I/O ADDRESS SETTING ". Try to modify the I/O address on software & hardware then start again from section 3.1 to test.
4. If no error occurs, the testing is O.K.

==========COMMON USER'S STEPS==========

1. Check up the filename and the manufacturer's name from the IC LIST for the IC which you are going to program.

   REMARK : Different types of ICs may come from the same manufacturer, therefore, you have to check up PARTS02.LST first.

   For example: MMI PAL 16L8B-2 uses the filename of PAP02.EXE, The manufacturer is MMI-A TYPE.

2. Execute this file. (simply enter the filename)

   For example: A>PAP02 <enter>

3. After main menu appears, press key "M" to choose the manufacturer. For example:

```
   PAL SOFTWARE    V3.0          *MFG.:  NS          CHECK SUM
                              *TYPE: NONE-TYP    =:0
       MAIN MENU:             *FUSE MAP: NONE-MAP
============================   _____________________________________
1. DIR                        |  MANUFACTURER:       |
2. LOAD     FUSE MAP FROM DISK |  1.MMI (A TYPE) 4.MMI(B TYPE)|
3. SAVE     FUSE MAP TO DISK  |  2. NS      5.TI(B TYPE)   |
4. EDIT     FUSE MAP          |  3.TI (A TYPE)       |
M. MANUFACTURER               | <ESC>back to   main manu.  |
T. TYPE                       |                      |
B. BLANK CHECK                |______________________________|
P. PROGRAM   A.   AUTO
R. READ          V.     VERIFY
S. SECURITY FUSE BLOW
Q. QUIT
```

   * Press key '1' to choose MMI-A TYPE and press ESC to return to main manu.

4. Press key 'T' to choose TYPE.
   For example:


```
PAL SOFTWARE      V3.0        *MFG.: MMI (A TYPE)   CHECK SUM
```

```
                        *TYPE: NONE - TYP =:0
                        *FUSE MAP: NONE   MAP
     MAIN MANU:          _____________________________________________
=====================  |   TYPE: 24 pins             |
1. DIR                  |1:12L10  A:20X8   L:20R4A/A-2/B/B-2/D    |
2. LOAD     FUSE MAP FROM     |2.14L8       B:20X8A  M:20L8-10          |
   DISK                 |3.16L6       C:20X4   N:20L8A/A-2/B/B-2/D      |
3. SAVE     FUSE MAP TO DISK|4.18L4   D:20X4A  O:20RS8             |
4. EDIT     FUSE MAP    |5.20L2       E:20S10  P:20RS10            |
M. MANUFACTURER         |6.20C1       F:20RS4  Q:20RA10            |
T. TYPE                 |7.20L10  G:20R8-10                 |
B. BLANK CHECK          |8.20X10  H:20R8A/A-2/B/B-2/D       |
P. PROGRAM   A.AUTO     |9.20X10A I:20R6-10                 |
R. READ          V.VERIFY     |       J:20R6-10             |
S. SECURITY FUSE BLOW   |       k:20R4-10             |
Q. QUIT                 |                             |
                  |    <ESC> back   to main     manu.       |
                  |    <PGUP>next   page                |
                  |    SELECT NUMBER?                   |
                  |_____________________________________|
```

     #   Press key 'J' to choose 16L8B-2/-4 and press ESC back to
      main manu.

5. To program the data from the disk files to the blank PAL,
   execute function 2 (LOAD FUSE MAP FROM DISK), followed by
   entering the path and the filename. For example:

```
      PAL SODTWARE  V3.0        *MFG.: MMI (A   TYPE) CHECK SUM
                          *TYPE: 16L8B-2/-4     =:0
        MAIN MANU:              *FUSE     MAP: A:TEST.JED
============================        ___LOAD FUSE MAP:_____________
1. DIR                         |                       |
2. LOAD     FUSE MAP FROM DISK       |  Enter Fuse map file name  |
3. SAVE     FUSE MAP TO DISK   |  <ESC> back to main manu.  |
4. EDIT     FUSE MAP           |                       |
M. MANUFACTURER                |   A:     TEST.JED          |
T. TYPE                        |_____________________________|
B. BLANK CHECK
P. PROGRAM  A. AUTO
R. READ           V. VERIFY
S. SECURITY FUSE BLOW
Q. QUIT
```

To program data from the PAL MASTER to the blank PAL,exectue
function R(READ). For example:

Š


```
   PAL SOFTWARE    V3.0        *MFG.: MMI (A TYPE)  CHECK SUM
                         *TYPE: 16L8B-2/-4     =:1560
                         *FUSE MAP: A:TEST.JED
      MAIN MANU:        _READ : _____________________________
==========================  |       Ready to read (Y/ESC)?        |
1. DIR                      |       Reading now...                |
2. LOAD     FUSE MAP FROM DISK   |   Read complete                 |
```

```
3. SAVE     FUSE MAP TO DISK     |_____________________________________|
4. EDIT     FUSE MAP
M. MANUFACTURER
T. TYPE
B. BLANK CHECK
P. PROGRAM
R. READ
S. SECURITY FUSE BLOW
Q. QUIT
```

6. Put the blank IC on the testing socket, press P (function P), and press Y to start to program.

```
   PAL SOFTWARE    V3.0        *MFG.:     MMI (A TYPE)      CHECK SUM
                           *TYPE:     16L8B-2/-4    =:1560
                           *FUSE MAP: NONE-MAP
      MAIN MANU:         ____PROGRAM : ____________________
==========================    |     Blank check or not (Y/N/ESC)?  |
1. DIR                        |     Checking now ...         |
2. LOAD     FUSE MAP FROM DISK    | Blank check ok.                |
3. SAVE     FUSE MAP TO DISK      |                              |
4. EDIT     FUSE MAP          |     Blowing     now...                      |
M. MANUFACTURER               |     Verify ok.               |
T. TYPE                       |______________________________|
B. BLANK CHECK
P. PROGRAM    A. AUTO
R. READ          V, VERIFY
S. SECURITY FUSE BLOW
Q. QUIT
```

Š

CHAPTER IV EPROM / EEPROM

========== FUNCTION DESCRIPTION ==========

** Files required :

1. EPP02.EXE : the main program to be executed.
2. EPP.DAT : to save the previous running data after function Q: Quit was executed.
3. SETUP02.EXE
   SETUP.DAT
   (This file is for I/O address setting)

REMARK : Please place above files in the same working disk drive.

*** The function menu will be automatically displayed on the screen as showed below after EPP02.EXE is executed.

```
     E(E)PROM  PROGRAMMER  V3.0          *MFG.:AMD      *ZIP.     :1
                             *TYP.:2764A  *PROG.:intelligent
        MAIN       MANU:                 *VPP.: 12.5V *VCC. : 6.0V
==================================
1. DIR
2. LOAD     OBJ FILE TO MEMORY BUFFER
3. SAVE     MEMORY BUFFER TO DISK
4. DEBUG MEMORY   BUFFER
5. GANG     SIZE
6. PROGRAMMING ALGORITHM
7. SET MEMORY BUFFER SIZE
M. MANUFACTURER
T. TYPE
B. BLANK CHECK
P. PROGRAM  A. AUTO
R. READ           V. VERIFY
C. COMPARE  D. DISPLAY & EDIT
Q. QUIT

SELLECT     WHICH NUMBER ?
```

REMARK 1 : Above right angle
- * MFG. stands for manufacturer(refer to function M)
- * TYP. stands for TYPE (refer to function T)
- * VPP. stands for the programming voltage. (refer to function T)
- * ZIP. stands for the number of programming socket. (refer to function 5)
- * PROG. stands for the programming algorithm & time (refer to function 6)
- * VCC. stands for VCC voltage in programming

Š

REMARK 2 : Above data about MFG. TYP. ... etc. will be saved into EPP.DAT file when you execute function Q, and will be read from EPP.DAT when you run EPP02.EXE program next time, and will be automatically selected as the same status as the last time. (MFG. TYP. ... etc.)

REMAKE 3 : After executing EPP02.EXE, PC will open a 64K or 128K memory space as MEMORY BUFFER for user (refer to function 7). The address for MEMORY BUFFER is selected from 00000H--0FFFF (64K) or 00000H--1FFFF (128K). Data read from the EPROM or the DISK will be saved in MEMORY BUFFER.

*** FUNCTION 1 : DIR
To list the files directory of the disk. (under the same DOS system)

FORMAT : [d:] [path] [filename {.ext}] [/p] [/w]
<ESC> back to main menu.
REMARK : COMMAND.COM must be placed in drive A so as to execute DOS COMMAND.

*** FUNCTION 2 : LOAD OBJ FILE INTO MEMORY BUFFER
To load .OBJ file from the disk into the memory buffer, enter the path, followed by .OBJ file name and the buffer starting address or press <ESC> to return to main menu.

FORMAT : Enter file name to be loaded :
[d:] [path] [filename,OBJ]
Enter buffer starting address:
[n] ---- Usually "0" is input and started from the first address in MEMORY BUFFER.
or press <ESC> to return to main menu.

REMARK : Please load the OBJ CODE file. If the .HEX file was loaded, please use HEXOBJ02.EXE to convert it into .OBJ file before you input data.

*** FUNCTION 3 : SAVE MEMORY BUFFER TO DISK
To save data from the memory buffer to the disk, enter the path, followed by .OBJ file name, the buffer starting and end address or press <ESC> to return to main menu.

FORMAT : Enter file name to be saved:
[d:] [path] [filename.OBJ]
Enter buffer starting address:
Š

[n1] ---- the buffer starting address to be saved
Enter buffer end address:
[n2] ---- the buffer end address to be saved
or <ESC> back to main menu.

REMARK : Above data will be saved in OBJ CODE format.

*** FUNCTION 4 : DEBUG MEMORY BUFFER
To debug and modify your memory buffer.

REMARK : Please load DEBUG.COM into the working disk driver and press any key to start (see DEBUG in MS DOS). The display screen looks like this:

First 64K memory buffer starting address at
4356:0000
|___ data segment

Second 64K memory buffer starting address at
5356:0000
|___ data segment

First enter RDS to change data segment

-RDS
-DS: 1245
-4356 ---- input data segment of the memory buffer start address

*** FUNCTION 5 : GANG SIZE
To select the number of SOCKET in a working batch (1-4 SOCKET(S)).

REMARK : If you want to expand it to 4 , you shall be asked to buy additional 4 ZIP sockets.

*** FUNCTION 6 : PROGRAMMING ALGORITHM
To select the type of the programming algorithm.

REMARK : The programming algorithm is automatically selected when you change the MFG. or TYP. but you can change it by yourself only you make it in accordance with the IC programming rules.

*** FUNCTION 7 : SET MEMORY BUFFER SIZE
To set memory buffer size, you may select 64K
Š

or 128K. As executing this function to change memory buffer size, the computer will return to DOS automatically . At this moment please execute EPP02.EXE program again.

For example : EPROM 128K*8 such as NEC 27C1000, 27C1001 ... etc. must have 128K MEMORY SIZE.

REMARK 1 : As function Q is executed, MEMORY BUFFER SIZE (64K or 128K) will be save in EPP.DAT file.

REMARK 2 : If the 128K is selected, PC's MEMORY SIZE must be at least 384K up so as to execute EPP02 .EXE. If EPP02.EXE fails to be executed, please erease EPP.DAT file then try EPP02.EXE again.

*** FUNCTION M : MANUFACTURER
To select the manufacturer of the chips.

* Please select the correct manufacturer because the programming algorithm of each manufacturer is quite different.
* If IC manufacturer is not found, please choose the first set "DON'T CARE".
* If DON'T CARE is selected, you must assure the TYPE and the programming voltage . If the programming voltage is unknown, please select the low voltage to test. If failed, try high voltage.

*** FUNCTION T : TYPE
To select TYPE and programming voltage, use keys <1><2>...<A><B>...etc.

CAUTION : Please select the exactly correct TYPE and the programming voltage.

* The programming algorithm of National Semiconductor 27CP64 , 27CP128 is comparatively special and different from the other manufacturers' 2764 , 27128 .
* There are two different PIN structure for 32 PINs IC and the IC part number from each manufacturer is quite different.

PINs A : (A16=pin2 , /OE=pin24)
DON'T CARE : 27010
AMD : 27C010
FUJITSU : 27C1001
HITACHI : 27C101
Š

INTEL : 27010 , 27C010
MITSUBISHI : 27C101
TOSHIBA : 27C1000

PINs B : (A16=pin24 , /OE=pin2)
DON'T CARE : 27100
FUJITSU : 27C1000
HITACHI : 27C301
MITSUBISHI : 27C100
NEC : 27C1000
TOSHIBA : 27C1001

*** FUNCTION B : BLANK CHECK
To check whether the chips are blank or not. The length is decided by the section address. (the section address described below) If there is data in the first address, the error will be displayed.

For example :
NO.1 ERROR AT 138
< The first EPROM at address 0138 is not blank)

<Y> : Start to check
<C> : to change section address
<ESC>: back to main menu

Section Address :
CHIP STARTING ADR : the starting address of the IC's on SOCKET.
CHIP END ADR : the end address of the IC's on SOCKET.
BUFFER STARTING ADR: MEMORY BUFFER's starting address.
BUFFER CHECK SUM : accumulated by BYTE value

BUFFER CHECK SUM's expression is accounted from the BUFFER STARTING ADDRESS, its length = (CHIP END ADR) - (CHIP STARTING ADR)

REMARK : Sometimes a damaged IC may be mistaken for a blank IC by doing BLANK CHECK.

*** FUNCTION P : PROGRAM

To program the chip with the data from the memory buffer to EPROM. It will VERIFY automatically after finishing program and display the result. The address and the length are decided by section address.

Š

<Y> : Start to program.
<E> : to program even bytes only
<O> : to program odd bytes only
<C> : to change the address
<ESC>: back to main menu.

For example : Press 'O' to program data addressed at odd bytes only in the memory buffer as showed below:

MEMORY BUFFER		EPROM	
ADDRESS	DATA	ADDRESS	DATA
0000	55	0000	55
0001	38	0001	AA
0002	AA	0002	12
0003	49	:	
0004	12	:	
0005	6C	:	
:		:	

Press 'E' to program data addressed at even bytes only in the memory buffer as showed below:

MEMORY BUFFER		EPROM	
ADDRESS	DATA	ADDRESS	DATA
0000	55	0000	38
0001	38	0001	49
0002	AA	0002	60
0003	49	:	
0004	12	:	
0005	6C	:	
:		:	

*** FUNCTION A : AUTO
To program automatically.

*** steps executed automatically
1. BLANK CHECK.
2. PROGRAM.
3. VERITY.

Š

*** FUNCTION R : READ
To read the data on the chip into the memory buffer .

<Y> : start to read.
<E> : to read data and save it at even bytes only in the memory buffer
<O> : to read data and save it at odd bytes only in the memory buffer
<C> : to change the address
<ESC> : back to main menu.

REMARK : After completing reading data from EPROM to the memory buffer, first you have to assure whether the CHECK SUM is correct or not. If it is not correct, it means that the data read is wrong. If the original CHECK SUM in EPROM is unknown, please check whether the content of the memory buffer is correct or not by using the function D.

*** FUNCTION V : VERIFY
To verify the data in the chips with that in the memory buffer.

<Y> : start to verify
<E> : to verify with data at even bytes only in the memory buffer
<O> : to verify with data at odd bytes only in the memory buffer.
<C> : to change the address
<ESC> : back to main menu.

*** FUNCTION C : COMPARE
To compare the data in the chip with that in the memory buffer. The difference is displayed in the format:

CHIP ADDR : CHIP DATA - (BUFFER ADDR : BUFFER DATA)

As the difference is displayed, you may press <CTRL-S> to hold the display or press any key to continue or press <ESC> to return to main menu.

<Y> : start to compare

<E> : to compare with data at even bytes only in the memory buffer
<O> : to compare with data at odd bytes only in the memory buffer

Š

<C> : to change the address
<ESC> : back to main menu

*** FUNCTION D : DISPLAY & EDIT
To display and edit the content of the memory buffer directly. DEBUG.COM. is not neccessary for this function. Press Q to return to main menu.

*** FUNCTION Q : QUIT
To return to DOS and save the current status of the chips (MFG. TYP. ... etc.) into the EPP.DAT. Next time when you run the EPP02.EXE program, the same IC type as the last time will be automatically selected.

Š

CHAPTER V MICROCONTROLLER (8751 Series)

========== FUNCTION DESCRIPTION ==========

** Files required :
1. 51P02.EXE : the main program to be executed.
2. 51P.DAT : to save the previous running data after function Q: Quit was executed.
3. SETUP02.EXE
SETUP.DAT
(This file is for I/O address setting)

REMAKE : Please place Above files in the same working disk drive.

*** The function menu will be automatically displayed on the screen as showed below after 51P02.EXE is executed.

```
  8751  PROGRAMMER  V3.0     *MFG.: AMD     *ZIP      : 1
                            *TYP.: 8751H  *PROG.: intelligent
  MAIN MANU                 *VPP.: 21.0V  *VCC. : 5.0V
==================================
1. DIR
2. LOAD     OBJ FILE TO MEMORY BUFFER
3. SAVE     MEMORY BUFFER TO DISK
4. DEBUG MEMORY   BUFFER
5. GANG     SIZE
6. PROGRAMMING ALGORITHM
7. ENCRYPTION TABLE SETTING
M. MANUFACTURER
T. TYPE
B. BLANK CHECK
P. PROGRAM   A. AUTO
R. READ            V. VERIFY
C. COMPARE   D. DISPLAY & EDIT
S. SECURITY BIT   PROGRAMMING
E. ENCRYPTION TABLE PROGRAMMING
Q. QUIT

SELLECT     WHICH NUMBER ?
```

REMAKE 1 :
* MFG. stands for Manufacturer (refer to function M)
* TYP. Stands for TYPE (refer to function T)
* VPP. Stands for the programming voltage. (refer to function T)
* ZIP. stands for the number of SOCKET. (refer to function 5)
* PROG. stands for the programming algorithm & time (refer to function 6)
* VCC. stands for VCC voltage in programming.

Š

REMARK 2 : Above data about MFG. TYP. ... etc. will be saved into 51P.DAT file as the function Q is executed. The data will be read from 51P.DAT file as 51P02.EXE is exected next time and be automatically set the same status as the last

time.

REMARK 3 : After 51P02.EXE is executed, the PC will open a 64K memory space as MEMORY BUFFER for user. The address for MEMORY BUFFER is set from 0000H -- FFFF (64K). Data read from the EPROM or the DISK will be saved in MEMORY BUFFER.

*** FUNCTION 1 : DIR
To list the files directory of the DISK. (under the same DOS system)

FORMAT : [d:] [path] [filename {.ext}] [/p] [/w]
or <ESC> back to main menu.

REMARK : COMMAND.COM must be put in drive A so as to execute DOS COMMAND.

*** FUNCTION 2 : LOAD OBJ FILE INTO MEMORY BUFFER
To load .OBJ file from disk into MEMORY BUFFER. You may specify the path, .OBJ file file name, the buffer starting address to be loaded or press <ESC> to return to main menu.

FORMAT : Enter file name to be loaded:
[d:][path][filename .OBJ]
Enter buffer starting address:
[n] ---- Usually '0' is input and started from the first address in MEMORY BUFFER.
or press <ESC> to return to main menu.

REMARK : Please load the OBJ CODE file. If the .HEX file was loaded, please use the HEXOBJ02.EXE file to convert it into .OBJ file before you input data.

*** FUNCTION 3 : SAVE MEMORY BUFFER TO DISK
To save data from MEMORY BUFFER into the disk file. You may specify the path, the .OBJ filename, the buffer starting and end address to be saved or press <ESC> to return to main menu.

Š

FORMAT : Enter file name to be saved:
[d:][path][filename.OBJ]
Enter buffer starting address :
[n1] ---- the starting address
Enter buffer end address :
[n2] ---- the end address
or press <ESC> to return to main menu

REMARK : Above data will be saved with OBJ CODE format.

*** FUNCTION 4 : DEBUG MEMROY BUFFER
To debug and modify your memory buffer.

REMARK : Please place DEBUG.COM in the working disk drive and press any ket to start (refer DEBUG in DOS). The display on the screen shows:

```
Memory     buffer starting   address     at 4356:0000
                             |
                             |__ data
                               segment
<First     enter RDS to change data segment>
--RDS
--DS: 1245
--4356      ---- input   data   segment  of  memory
           buffer start address
```

*** FUNCTION 5 : GANG SIZE
To select the number of SOCKET in a working batch (1-4 SOCKET(S)).

REMAKE : If you want to expand it to 4 SOCKETS, you shall be asked to buy additional 4 ZIP sockets.

*** FUNCTION 6 : PROGRAMMING ALGORITHM
To select the type of the programming algorithm.

REMAKE : The programming algorithm is automatically selected when you change the MFG. or TYP. but you can also change it by yourself only you make it in accordance with the IC programming rules.

*** FUNCTION 7 : ENCRYPTION TABLE SETTING
To set 32 byte encryption table for protection . You may enter into another subfunction menu as below in executing function 7 .

Š

ENCRYPTION TABLE :

```
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF =>   contect  of
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF      32     byte
                                        encryption
                                        table.
```

ENCRYPTION TABLE MENU :
1. Edit encryption table .
2. Load encryption table from disk . => subfunction menu
3. Save encryption table to disk .
4. Enable/Disable encryption table .

SELECT WHICH ONE
or PRESS <ESC> TO MAIN MENU ?

*** ENCRYPTION TABLE DISABLE *** ==> show whether encryption

table is programmed or not .

* SUBFUNCTION 1 : Edit encryption table
To edit , the 32 byte encryption table will be displayed in the top two lines on the screen. As soon as every one byte is edited, the encryption table will follow to change . Remember to save encryption table into disk after completing editing. (use subfunction 3)

* SUBFUNCTION 2 : Load encryption table from disk
To read the edited encryption table from disk.

* SUBFUNCTION 3 : Save encryption table to disk
To save the edited encryption table to disk.

* SUBFUNCTION 4 : Enable/Disable encryptiion table
To set programmable or non-programmable encryption table . Every time this function key is pressed , the opposite setting will show on the screen,
*** ENCRYPTION TABLE ENABLE ***
OR *** ENCRYPTION TABLE DISABLE ***

If the encryption table is set programmable , as soon as you return to main menu , on the bottom of the screen will show message as below :
*** ENCRYPTION TABLE ENABLE ***

REMAKE : 1. 8751H/8752H have only one Security fuse to prevent inside ROM from being read. Besides the Security fuse , 87C51 / 87C51FA /
Š

87C252 have the 32 byte encryption table programming.

2. After the Security fuse is programmed not only the inside ROM can't be read but also the outside ROM program can not be executed . If only the 32 byte encryption table is programmed , the outside ROM program can be executed and the inside ROM can be read.

However, if original 32 byte encryption table is unknown, the data read from the inside ROM would be in error . You may gain two functions described below if you program encryption table only.

a. You may use Verify mode to know whether the inside program is corrent or not. (Of course you have to know the original setting of the 32 byte encryption table)

b. You can protect the inside program from being copied.

*** FUNCTION M : MANUFACTURER
To select the manufacturer of the chips.

* Please select the correct manufacturer because the programming algorithm of each manufacturer is quite different.
* If IC's manufacturer is not found .please select the first set "DON'T CARE".
* If DON'T CARE is selected, you must assure the TYPE and the programming voltage. If the programming voltage is unknown, please select the low voltage to test. If failed, try high voltage.

*** FUNCTION T : TYPE
Use keys <1><2>...<A><B>...etc, to select TYPE.
The correct programming voltage and algorithm will be automatically set as you select TYPE .

CAUTION : Please don't feel free to change the programming voltage and algorithm.

*** FUNCTION B : BLANK CHECK
To check whether the chips are blank or not.
The length is decided by the section address. (
Š

the section address is described as below). If there is data in the first address, the display will show error.
For example:

NO. 1 ERROR AT 138

(The first address at 0138 is not blank)

<Y> : start to check
<C> : to change the address
<ESC> : back to main menu

Section address : CHIP STARTING ADR.: the starting address of the chip on the socket
CHIP END ADR.: the end address of the chip on the socket
BUFFER STARTING ADR.: the starting address of memory buffer
BUFFER CHECK SUM : accumulated by BYTE value.
< Above is presented by ASCII >

BUFFER CHECK SUM's expression is accounted from the BUFFER STARTING ADDRESS, its length = (CHIP END ADR) - (CHIP STARTING ADR.)

REMARK : Sometimes a damaged chip will be mistaken for a blank chip by using BLANK CHECK.

*** FUNCTION P : PROGRAM

To program the chip with data from the memory buffer to EPROM. It will VERIFY automatically after finishing program and display the result. The address and the length are decided by section address.

<Y> : start to program
<C> : to change the address
<ESC>: back to main menu

*** FUNCTION A : AUTO

To program automatically.

** steps executed automatically :

1. BLANK CHECK
2. PROGRAM
3. VERIFY
4. SECURITY FUSE BLOW

Š

REMAKE : if the function 7 *** ENCRYPTION TABLE ENABLED *** is set, it will be programmed automatically.

*** FUNCTION R : READ

To read the data on the chip into memory buffer.

<Y> : start to reat
<C> : to change the address
<ESC> : back to main menu

REMAKE 1 : If the encryption table is programmed into the chip , the original 32 byte encryption table and the *** ENCRYPTION TABLE ENABLE *** must have been set by using function 7 before reading , so the data read will be correct.

REMAKE 2 : After completing reading data from EPROM to MEMORY BUFFER, first you have to assure whether the CHECK SUM is correct or not. If it is not correct, it means that the data read is wrong. If the original CHECK SUM in EPROM is unknown, please check whether the content of the MEMORY BUFFER is correct or not by using the function D.

*** FUNCTION V : VERIFY

To verify the data in the chips with that in the memory buffer.

<Y> : start to verify
<C> : to change the address
<ESC> : back to main menu.

REMAKE : If the encryption table is programmedinto the chip , the orginial 32 byte encryption table and *** ENCRYPTION TABLE ENABLE *** must have been set by using function 7 before verifying , so the verify data will be correct.

*** FUNCTION C : COMPARE

To compare the data in the chip with that in the memory buffer. The difference is displayed in the format:

CHIP ADDR : CHIP DATA - (BUFFER ADDR : BUFFER DATA)

As the difference is displayed, you may press <CTRL-S> to hold the display or press any key
Š

to continue or press <ESC> to return to main menu.

<Y> : start to compare
<C> : to change the address
<ESC> : back to main menu

REMAKE : If the encryption table is programmedinto the chip , the orginial 32 byte encryption table and the *** ENCRYPTION TABLE ENABLE *** must have been set by using function 7 before comparing , therefore the compared data will be correct.

*** FUNCTION S : SECURITY BIT PROGRAMMING

To program security bit to protect inside ROM from being read . If the U.V. eraser is used to move the protection of the Security bit , the inside ROM programs will be erased , too.

*** FUNCTION E : ENCRYPTION TABLE PROGRAMMING

To program 32 byte encryption table for protection . Before you use this function , you have to set as below by function 7:

*** ENCRYPTION TABLE ENABLE ***

*** FUNCTION D : DISPLAY & EDIT

To display and edit the content of the memory buffer directly. DEBUG.COM. is not neccessary for this function. Press Q to return to main menu.

*** FUNCTION Q : QUIT

To return to DOS and save the current status of the chips (MFG. TYP. ... etc.) into the 51P.DAT . Next time when you execute 51P02.EXE, the

same IC TYPE as the last time will be automatically set .

Š

CHAPTER V MICROCONTROLLER (8748 Series , Z8 Series ..)

========== FUNCTION DESCRIPTION ==========

** Files required :
1. filename02.EXE : the main program to be executed.
2. filename.DAT : to save the previous running data after function Q was executed.
3. SETUP02.EXE
   SETUP.DAT
(This file is for I/O address setting)

REMARK : Please place above files in the same working disk drive.

*** The function menu will be automatically displayed on the screen as showed below after filename02.EXE (like 48P02.EXE) is executed.

```
  8748 PROGRAMMER   V3.0      *MFG.:INTEL *ZIP:1
                        *TYP.:8748È *PROG.:intellegent
  MAIN MANU             *VPP.:21.0V *VCC.:5V
==================================
1. DIR
2. LOAD     OBJ FILE TO MEMORY BUFFER
3. SAVE     MEMORY BUFFER TO DISK
4. DEBUG MEMORY   BUFFER
5. GANG     SIZE
6. PROGRAMMING ALGORITHM
M. MANUFACTURER
T. TYPE
B. BLANK CHECK
P. PROGRAM   A. AUTO
R. READ            V. VERIFY
C. COMPARE   D. DISPLAY & EDIT
S. SECURITY FUSE BLOW
Q. QUIT

SELLECT     WHICH NUMBER ?
```

REMAKE 1 : Above right angle
* MFG. stands for manufacturer(refer to function M)
* TYP. stands for TYPE (refer to function T)
* VPP. stands for the programming voltage. (refer to function T)
* ZIP. stands for the number of SOCKET.(refer

to function 5)
* PROG. stands for the programming algorithm & time (refer to function 6)
* VCC. stands for VCC voltage in programming.

Š

REMAKE 2 : Above data about MFG. TYP. ... etc. will be saved into filename.DAT file as the function Q is executed . The data will be read from filename.DAT file as filename02.EXE is executed next time and be automatically set the same status as the last time.

REMAKE 3 : After filename02.EXE is executed,the PC will open a 64K memory space as MEMORY BUFFER for user . The address for MEMORY BUFFER is set from 0000H--FFFF (64K) . Data read from the EPROM or the DISK will be saved in MEMORY BUFFER.

*** FUNCTION 1 : DIR
To list the files directory of the DISK. (under the same DOS system)

FORMAT : [d:] [path] [filename {.ext}] [/p] [/w]
<ESC> back to main menu.

REMAKE : COMMAND.COM must be put in drive (A:)so as to execute DOS COMMAND.

*** FUNCTION 2 : LOAD OBJ FILE INTO MEMORY BUFFER
To load .OBJ file from DISK into MEMORY BUFFER. You may specify the path, .OBJ file file name, the buffer starting address to be loaded or press <ESC> to return to main menu.

FORMAT : Enter file name to be loaded:
[d:][path][filename .OBJ]
Enter buffer starting address:
[n] ---- Usually '0' is input and started from the first address in MEMORY BUFFER.
or press <ESC> to return to main menu.

REMARK : Please load the OBJ CODE file. If the .HEX file was loaded, please use the HEXOBJ02.EXE file to convert it into .OBJ file before you input data.

*** FUNCTION 3 : SAVE MEMORY BUFFER TO DISK
To save data from MEMORY BUFFER into the disk file . You may specify the path, the .OBJ filename, the buffer starting and end address to be saved or press <ESC> to return to main menu.

Š

FORMAT : Enter file name to be saved:
[d:][path][filename.OBJ]
Enter buffer starting address :
[n1] ---- the starting address
Enter buffer end address :
[n2] ---- the end address
or press <ESC> to return to main menu

REMARK : Above data will be saved in OBJ CODE format.

*** FUNCTION 4 : DEBUG MEMROY BUFFER
To debug and modify your memory buffer.

REMARK : Please place DEBURG.COM in the working disk drive and press any key to start (see DEBURG in DOS). The display on the screen show :

```
Memory     buffer starting   address     at 4356:0000
                             |
                             |__ data
                               segment
<First     enter RDS to change data segment>
--RDS
--DS:1560
--4356      ---- input data segment of memory buffer
            start address
```

*** FUNCTION 5 : GANG SIZE
To select the number of SOCKET in a working batch (1-4 SOCKET(S)).

REMAKE : If you want to expand it to 4 SOCKETS, you shall be asked to buy additional 4 ZIP sockets.

*** FUNCTION 6 : PROGRAMMING ALGORITHM
To select the type of the programming algorithm.

REMAKE : The programming algorithm is automatically selected when you change the MFG. or TYP. but you can change it by yourself only you make it in accordance with the IC programming rules.

*** FUNCTION M : MANUFACTURER
To select the manufacturer of the chips.

* Please select the correct manufacturer because the programming algorithm of each manufacturer is quite different.

Š

* If IC's manufacturer is not found, please select the first set "DON'T CARE".
* If DON'T CARE is selected, you must assure the TYPE and the programming voltage. If the

programming voltage is unknown, please select the low voltage to test. If failed, try high voltage.

*** FUNCTION T : TYPE
Use keys <1><2>...<A><B>...etc, to select TYPE. The correct programming voltage and algorithm will be automatically set as you select TYPE .

CAUTION : Please select the exactly correct programming voltage and algorithm.

*** FUNCTION B : BLANK CHECK
To check whether the chips are blank or not. The length is decided by the section address. (the section address described as below). If there is data in the first address, the displayed will show error.
For example:

NO. 1 ERROR AT 138
(The first EPROM at 0138 is not blank)

<Y> : start to check
<C> : to change the address
<ESC> : back to main menu

Section address : CHIP STARTING ADR. : the starting address of the chip on the socket
CHIP END ADR. : the end address of the chip on the socket
BUFFER STARTING ADR.: the starting address of memory buffer
BUFFER CHECK SUM : accumulated by BYTE value.
< Above is presented by ASC II >

BUFFER CHECK SUM's expression is accounted from the BUFFER STARTING ADDRESS, its length = (CHIP END ADR) - (CHIP STARTING ADR.)

REMARK : Sometimes a damaged chip will be mistaken for a blank chip by using BLANK CHECK.

Š

*** FUNCTION P : PROGRAM
To program the chip with data from the memory buffer to EPROM. It will VERIFY automatically after finishing program and display the result. The address and the length are decided by section address.

<Y> : start to program
<C> : to change the address
<ESC> : back to main menu

*** FUNCTION A : AUTO
To program automatically.
** steps executed automatically :
1. BLANK CHECK
2. PROGRAM
3. VERIFY
4. SECURITY FUSE BLOW

*** FUNCTION R : READ
To read the data on the chip into memory buffer.

<Y> : start to reat
<C> : to change the address
<ESC> : back to main menu

REMAKE : After completing reading data from EPROM to the MEMORY BUFFER, first you have to assure whether the CHECK SUM is correct or not. If it is not correct, it means that the data read is wrong. If the original CHECK SUM in EPROM is unknown, please check whether the content of the MEMORY BUFFER is correct or not by using the function D.

*** Function V : VERIFY
To verify the data in the chips with that in the memory buffer.

<Y> : start to verify
<C> : to change the address
<ESC> : back to main menu.

*** FUNCTION C : COMPARE
To compare the data in the chip with that in the memory buffer. The difference is displayed in the format:

Š

CHIP ADDR : CHIP DATA - (BUFFER ADDR : BUFFER DATA)

As the difference is displayed, you may press <CTRL-S> to hold the display or press any key to continue or press <ESC> to return to main menu.

<Y> : start to compare
<C> : to change the address
<ESC> : back to main menu

*** FUNCTION S : SECURITY BIT PROGRAMMING
To program security bit to protect inside ROM from being read . If the U.V. eraser is used to move the protection of the Security bit , the

inside ROM programs will be erased , too.

*** FUNCTION D : DISPLAY & EDIT
To display and edit the content of the memory buffer directly. DEBUG.COM. is not neccessary for this function. Press Q to return to main menu.

*** FUNCTION Q : QUIT
To return to DOS and save the current status of the chips (MFG. TYP. ..etc.) into the filename. DAT . Next time when you execute filename02.EXE , the same IC TYPE as the last time will be automatically set .

Š

CHAPTER VII Bipolar PROM

========== FUNCTION DESCRIPTION ==========

** Files required :

1. BPP02.EXE : the main program to be executed.
2. BPP.DAT : to save the previous running data after function Q was executed.
3. SETUP02.EXE
   SETUP.DAT
   (This file is for I/O address setting)

REMARK : Please place above files in the same working disk drive.

*** The function menu will be automatically displayed on the screen as showed below after BPP02.EXE is executed.

BPROM PROGRAMMER V3.0 * MFG.: AMD
* TYP.: 27S20-255*4
MAIN MENU

```
==============================
   1. DIR
   2. LOAD OBJ FILE TO MEMORY BUFFER
   3. SAVE MEMORY BUFFER TO DISK
   4. DEBUG MEMORY BUFFER
   5. SWAP BUFFER DATA
   M. MANUFACTURER
   T. TYPE
   B. BLANK CHECK
   P. PROGRAM         A. AUTO
   R. READ      V. VERIFY
   C. COMPARE         D. DISPLAY & EDIT
   Q. QUIT

SELECT WHICH NUMBER?
```

REMARK 1 : above right angle
* MFG. stands for manufacturer (see function M)
* TYP. stands for type (see function T)

REMARK 2 : Above data about MFG. TYP. ... etc. will be saved into BPP.DAT file as the function Q is executed. The same data will be read from BPP.DAT file as BPP02.EXE is executed next time and be automatically set the same status as the last time.

REMARK 3 : After BPP02.EXE is executed, the PC will open a 64K memory space as MEMORY BUFFER for user. The address for MEMORY BUFFER is set from 0000H --

Š

FFFF (64K). Data read from the EPROM or the DISK will be saved in MEMORY BUFFER.

*** FUNCTION 1 : DIR
To list the files directory of the disk.
(under the same DOS system)
FORMAT : [d:][path][filename {.ext}][/p][/w]
or <ESC> back to main menu

REMARK : COMMAND.COM must be put in drive A so as to execute DOS COMMAND.

*** FUNCTION 2 : LOAD OBJ FILE INTO MEMORY BUFFER
To load .OBJ file from DISK into MEMORY BUFFER.
You may specify the path, .OBJ file file name, the buffer starting address to be loaded or press <ESC> to return to main menu.

FORMAT : Enter file name to be loaded:
[d:][path][filename .OBJ]
Enter buffer starting address:
[n] ---- Usually '0' is input and started from the first address in MEMORY BUFFER.
or press <ESC> to return to main menu.

REMARK : Please load the OBJ CODE file. If the .HEX file

was loaded, please use the HEXOBJ02.EXE file to convert it into .OBJ file before you input data.

*** FUNCTRION 3 : SAVE MEMORY BUFFER TO DISK

To save data from MEMORY BUFFER into the disk . You may specify the path, the .OBJ filename, the buffer starting and end address to be saved or press <ESC> to return to main menu.

FORMAT : Enter file name to be saved:
[d:][path][filename.OBJ]
Enter buffer starting address :
[n1] ---- the starting address
Enter buffer end address :
[n2] ---- the end address
or press <ESC> to return to main menu

REMARK : Above data will be saved in OBJ CODE format.

*** FUNCTION 4 : DEBUG MEMROY BUFFER

To debug and modify your memory buffer.

Š

REMARK : Please place DEBUG.COM in the working disk drive and press any ket to start (see DEBUG in DOS). The display screen looks like this:

```
Memory     buffer starting   address     at 4356:0000
                             |
                             |__ data
                               segment
<First     enter RDS to change data segment>
--RDS
--DS:1560
--4356      ---- input data segment of memory buffer
           start address
```

*** FUNCTION 5 : SWAP BUFFER DATA

To exchange data in HIGH NIBBLES with that in LOW NIBBLES in the memory buffer. The length is decided by section address.

For example : the orinigal memory buffer is 0010: C3
after this function is executed 0010: 3C

*** FUNCTION M : MANUFACTURER

To select the manufacturer of the chips.

* Please select the correct manufacturer because the programming algorithm of each manufacturer is quite different.

*** FUNCTION T : TYPE

Use keys <1><2>...<A><B>...etc, to select TYPE.

CAUTION : Please select the exactly correct programming voltage and algorithm.

*** FUNCTION B : BLANK CHECK

To check whether the chips are blank or not. The length is decided by the section address. (the section address described as below). If there is data inside the address, the error will be displayed. For example:

NO. 1 ERROR AT 138

(BPROM at 0138 is not blank)

<Y> : start to check

Š

<C> : to change the address

<ESC> : back to main menu

Section address : CHIP STARTING ADR. : the starting address of the chip on the socket

CHIP END ADR. : the end address of the chip on the socket

BUFFER STARTING ADR.: the starting address of memory buffer

BUFFER CHECK SUM : accumulated by BYTE value.

< Above is presented by ASC II >

BUFFER CHECK SUM's expression is accounted from the BUFFER STARTING ADDRESS, its length = (CHIP END ADR) - (CHIP STARTING ADR.)

REMARK : Sometimes a damaged chip will be mistaken for a blank chip by using BLANK CHECK.

*** FUNCTION P : PROGRAM

To program the chip with data from the memory buffer to EPROM. It will VERIFY automatically after finishing program and display the result. The address and the length are decided by section address.

<Y> : start to program

<C> : to change the address

<ESC> : back to main menu

*** FUNCTION A : AUTO

To program automatically.

** steps executed automatically :

1. BLANK CHECK
2. PROGRAM

3. VERIFY

*** FUNCTION R : READ
To read the data on the chip into memory buffer to be saved.

<Y> : start to reat
<C> : to change the address
<ESC> : back to main menu

REMARK : If there is only 4 Bit data from BPROM, the
Š

data will be read into the 4-bit only in low nibbles of the memory buffer as the function READ is executed. It won't affect the 4 bit in high nibbles.

REMARK : After completing reading data from BPROM to MEMORY BUFFER, first you have to assure whether the CHECK SUM is correct or not. If it is not correct, it means that the data read is wrong. If the original CHECK SUM in BPROM is unknown, please check whether the content of MEMORY BUFFER is correct or not by using the function D.

*** FUNCTION V : VERIFY
To verify the data in the chips with that in memory buffer.
<Y> : start to verify
<C> : change the address
<ESC> : back to main menu

REMARK : If there is only 4 bit data from BPROM, as the function VERIFY is executed the computer will verify with the 4 bit data only in low nibbles of the memory buffer.

*** FUNCTION C : COMPARE
To compare the data in the chip with that in the memory buffer. The difference is displayed in the format:

CHIP ADDR : CHIP DATA - (BUFFER ADDR : BUFFER DATA)

As the difference is displayed, you may press <CTRL-S> to hold the display or press any key to continue or press <ESC> to return to main menu.

<Y> : start to compare
<C> : to change the address
<ESC> : back to main menu

REMARK : If there is only 4 bit data from BPROM, as the function COMPARE is executed the computer will

compare with the 4 bit data only in low nibbles of the memory buffer.

*** FUNCTION D : DISPLAY & EDIT

To display and edit the content of the memory

Š

buffer directly. DEBUG.COM. is not neccessary for this function. Press Q to return to main menu.

*** FUNCTION Q : QUIT

To return to DOS and save the current status of the chips (MFG. TYP. ... etc.) into the BPP.DAT file. Next time as you execute BPP02 .EXE, the same IC TYPE will be automatically selected.

CHAPTER VIII PAL, FPL, GAL, PEEL, EPLD

========== FUNCTION DESCRIPTION ==========

**** NOTES *****

1. To use the chips such as PAL / FPL / GAL / PEEL / EPLD etc., you have to select the correct manufacturer and the TYPE fisrt before you read / compare / program. It is possible to demage IC even in READ MODE or VERIFY MODE if the wrong manufacturer and TYPE are selected.

2. If you want to bulk program PAL, please try one or two first. Should there be no problem in actural practice, you can start to bulk program PAL. Unneccessary damage can be avoided due to error operation by following this instruction.

** File required :

1. (filename.EXE) : the main program to be executed such as GAL02.EXE
2. PALFORM\fusemap.FRM : the subdirectory including the blank form files of all PAL chips available for the package.
3. (filename.DAT) : to save the previous running data while the the function QUIT is executed, such as GAL.DAT
4. SETUP02.EXE
   SETUP.DAT
   (This file is for I/O address setting)

REMARK : Please place above files in the same working disk drive.

*** The function menu will be automatically displayed on the screen as showed below after filename.EXE (such as GAL02.EXE) is executed.

GAL PROGRAMMER V3.00 *MFG.: LATTICE CHECK SUM
*TYPE: NONE-TYPE =:0
MAIN MENU *FUSE MAP: NONE-MAP

```
============================
1. DIR
2. LOAD     FUSE MAP FROM DISK
3. SAVE     FUSE MAP TO DISK
4. EDIT     FUSE MAP
M. MANUFACTURER
T. TYPE
B. BLANK CHECK
P. PROGRAM    A. AUTO
R. READ             V. VERIFY
E. BULK     ERASE
S. SECURITY FUSE BLOW
Q. QUIT


      REMARK 1 : above right angle
             * MFG. stands for manufacturer (see function M)
             * TYP. stands for TYPE (see function T)
             * FUSE MAP. stands  for     the filename loaded into
                    FUSE     BUFFER
             * CHECK SUM. stands for the CHECK SUM ofthe FUSE
                     BUFFER

      REMARK 2 : Above  running      data  such as MFG. TYP. ...etc.
             will  be  saved into filename.DAT and be loaded
             from  filename    .DAT next time   when you execute
             filename.EXE.    The  data  will   be automatically
             selected the same as the last time.

      REMARK 3 : The  PC will open a memory space as FUSE BUFFER
             after executing filename.EXE.


*** FUNCTION 1 : DIR
             To list the files directory from the disk.
             (under     the same DOS system)

      FORMAT : [d:][path][filename{.rxt}][/p][/w]
             or <ESC> bact to main menu

      REMARK :  COMMAND.COM  must be put in driver A so as to
             execute DOS COMMAND,


*** FUNCTION 2 : LOAD FUSE MAP FROM DISK
             To  load  the    JEDEC  FORMAT file from the disk
             into  FUSE  BUFFER, enter the path, followed by
             the  file  name  for FUSE MAP or press  <ESC> to
             return     to main     menu.
Š
                                                    page 8-3

      FORMAT : [d:][path][filename.jed]
             <ESC> back to main menu

      REMARK : Please    use the standard format for JEDEC FILE,
             such  as  the    JEDEC  FORMAT assembled by ABEL,
             PALASM, AMAZE,   CUPL, PLAN... etc.


*** FUNCTION 3 : SAVE FUSE MAP TO DISK
```

To save fuse map in JEDEC FORMAT from FUSE BUFFER to the disk, enter the path and the JEDEC FORMAT filename or press <ESC> to return to main menu.

FORMAT : [d:][path][filename.jed]
<ESC> back to main menu

*** FUNCTION 4 : EDIT FUSE MAP
To display or edit the FUSE MAP.
<ESC> back to main mene.

To move the cursor : <UP><DN><LF><RT><HOME><END>
To change the page : <PGUP><PGDN>
To enter the data : <0><1><F>

	GAL. FPL. PEEL(18CV8)	PAL, 22V10
0	fuse blown	fuse not blown
1	fuse not blown	fuse blown

* key F : set the whole EDIT FUSE MAP(FUSE BUFFER) as BLANK
* (N) in EDIT MAP stands for no program function.

REMARK : This is a slow way to edit FUSE MAP and easy to make mistake, so it is better for you to try to use assembler software to edit FUSE MAP such as ABEL, PALASM, AMAZE, CUPL PLAN ... etc.
(Please refer to the user manual of kinds of assembler software.)

*** FUNCTION M : MANUFACTURER
To select the manufacturer for the chips.
For example: GAL02.EXE

Š

MANUFACTURER
1. LATTICE
2. NS
3. SGS
4. VLSI
<ESC> back to main menu
SELECT NUMBER ?

Please select the correct manufacturer.
REMARK : The programming algorithm of each manufacturer is quite different.

*** FUNCTION T : TYPE

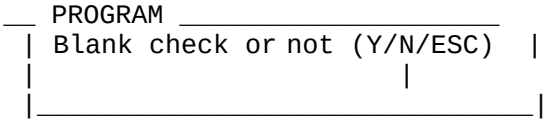
To select TYPE.

For example : there are two types for GAL02.EXE TYPE
1. 20 PIN
 such as GAL16V0-15/25/35
2. 24 PIN
 such as GAL20V8-15/25/35
* Press <PGUP><PGDN> to select the chip of 20 pin or 24 pin.
* Press <1><2>...<A><B>... etc. to select TYPE.

*** FUNCTION B : BLANK CHECK
To check whether the chips are blank or not. If there is data inside the chip, the display will show 'error'.

REMARK : The PAL which SECURITY FUSE is blown may show blank in doing BLANK CHECK.

*** FUNCTION P : PROGRAM
To program the chips.



<Y> : BLANK CHECK. If the chip is blank, it will be programmed. If the chip is not blank, the error will be displayed and the chip will not be programmed.
<N> : No BLANK CHECK. Program directly.
<ESC> : back to main menu

* If as you press key <Y> , the BLANK CHECK is
Š

O.K. but errors occur in programming, the IC may be damaged.
* Although the PAL which SECURITY FUSE is blown is checked blank, it is not programmable any more.
* If PEEL (18CV8 ,22CV10) and GAL (16V8, 20v8) of EEPROM TYPE are not blank, please use the function E : BULK ERASE to clear the content before you program them.
* Please use ERASER to clear first the chips of EPROM TYPE such as CPAL.

*** FUNCTION A : AUTO
** steps executed automatically
1. BLANK CHECK
2. PROGRAM
3. VERIFY
4. SECURITY FUSE BLOW
(The error will be showed if anyone of above can't be executed)

*** FUNCTION R : READ
To read the data in the chips. The CHECK SUM is
on the top right angle of the screen .

REMARK : You can check the data loaded from the chips by
using the function 4.

*** FUNCTION V : VERIFY
To verify the data in the chips with that in
the FUSE BUFFER.

*** FUNCTION S : SECURITY FUSE BLOW
To protect the chips from being read or writted.

*** FUNCTION Q : QUIT
To return to DOS and to save the running data
(MFG. TYP. JEDEC FILE) into filename.DAT (such
as GAL.DAT, PAP.DAT etc.)

*** EEPROM TYP such as
PEEL (18CV8 , 22CV10) & GAL (16V8 , 20V8) special function ***

Š

*** FUNCTION E : BULK ERASE
To clear data in the chips and do BLANK CHECK.

REMARK : After ERASE, the protection is also erased.

Š

CHAPTER IX : DIGITAL IC & MEMORY TESTER

========== FUNCTION DESCRIPTION ==========

** Files required :

1. DMT02.EXE : the main program to be executed.
2. DMT.DAT : to save the previous running data after function Q is executed.
3. SETUP02.EXE
   SET.DAT
   (This file is for I/O address setting)

REMARK : Please place above files in the same working disk drive.

*** The function menu will be automatically displayed on the screen as showed below after DMT02.EXE is executed.

DIGITAL IC & MEMORY TESTER V3.0

MAIN MENU

1. DIR
2. LOAD TEST PATTERN & TESTING
3. SAVE TEST PATTERN
4. EDIT TEST PATTERN
5. DEBUG TEST PATTERN
T. TTL TESTER
C. CMOS TESTER
M. MEMORY TESTER
Q. QUIT

SELECT WHICH NUMBER?

*** FUNCTION 1 : DIR
To list the files directory from the disk.
(under the same DOS system)

FORMAT : [d:][path][filename{,ext}][/p][/w]
<ESC> back to main menu

REMARK : COMMAND.COM must be place into the drive A so as to execute DOS COMMAND.

*** FUNCTION 2 : LOAD TEST PATTERN & TESTING
To load the TEST PATTERN file from the disk into the memory buffer. (Please refer to the format in function 4) Enter the path, file name and press any key to test or press <ESC> to return to main menu.
Š

FORMAT : [d:][path][filename]
<ESC> back to main menu

*** FUNCTION 3 : SAVE TEST PATTERN
To load the TEST PATTERN from the memory buffer into the disk to be filed. Enter the path and the filename or press <ESC> to return to main menu.

FORMAT : [d:][path][filename.jed]
<ESC> back to main menu

REMARK : Above is saved in BINARY CODE format.

*** FUNCTION 4 : EDIT TEST PATTERN
To edit the TEST PATTERN.
Please place DEBUG.COM into the working disk drive before you press this function key, then press any key to start. (Please see DEBUG. in MS DOS)

<Q> back to main menu

REMARK : see examples

*** FUNCTION 5 : DEBUG TEST PATTERN
To debug and display the test pattern established by <function 4> and test by step.

REMARK : see examples

*** FUNCTION T : TTL TESTER
To test the chips of TTL series.

<C> : change number (change the tested IC number)

<A> : auto search IC number (search the IC number
ready for test automatically and test it)
<SPACE> : testing (test selected IC)
<ESC> : return to main menu.

REMARK 1 : change number : simply enter the number such as
138, 00, 07..etc. (at most
three digitals)

REMARK 2 : auto search IC number : will or won't display
the IC number with the
same function.

Š

EXAMPLE 1 : Result : 04, 14, 19, 4069, 4584
**** END

EXAMPLE 2 : Result:
**** END

REMARK 3 : testing : test and display GOOD or ERROR

*** FUNCTION C : CMOS TESTER
To test the chips of CMOS seires.
(please refer to <function T>)

*** FUNCTION M : MEMORY TESTER
To test memory .

1. 4164 - 64K*1 2. 4256 - 256K*1
3. 2114 - 1K*4 4. 6116 - 2K*8
5. 6264 - 8K*K 6. 6256 - 32K*8
(There are six sets to select in MEMORY)

<C> : change number (to select MEMORY TYPE)
<SPACE> : testing
<ESC> : back to main menu

*** FUNCTION Q : QUIT
to return to DOS

*** EXAMPLE :
The pin assignment of the testing socket is showed as
below: (only 24 pin used, 3 bi-direction I/O PORTs)

```
    |-------v-------|
PC4 --|1       24|-- PC7 or VCC
PC5 --|2       23|-- PC6 or VCC
PC0 --|3       22|-- PC3 or VCC
PC1 --|4       21|-- PC2 or VCC
PA0 --|5       20|-- PB7 or VCC
PA1 --|6       19|-- PB6 or VCC
PA2 --|7       18|-- PB5
```

```
          PA3 --|8        17|-- PB4
          PA4 --|9        16|-- PB3
          PA5 --|10             15|-- PB2
          PA6 --|11             14|-- PB1
     GND     or PA7 --|12             13|-- PB0
               |---------------|
                 24 PIN SOCKET
```

Š

There are 3 bi-direction I/O PORTs (PA, PB and PC) to connect with socket.

The procedures to establish your own TEST PATTERN is :

1. select VCC and GND type code .(VCC)
2. set direction (DIRA, DIRB & DIRC) or PA, PB and PC.
3. establish test pattern for PORT A .(TPA)
4. establish test pattern for PORT B .(TPB)
5. establish test pattern for PORT C .(TPC)
6. total sets of test pattern. (NO.)

So you have to establish the following data on paper first.

(1). VCC (VCC & GND code) =
(2). DIRA (direction of PA) =
(3). DIRB (direction of PB) =
(4). DIRC (direction of PC) =
(5). TPA (test pattern of PA) =
(6). TPB (test pattern of PB) =
(7). TPC (test pattern of PC) =
(8). NO. (total sets of test pattern) =

Here are two examples to establish the test pattern on paper.

Expamle 1 : to establish TEST PATTERN for 74LS00

```
              |-------v-------|
          PC4 --|1        24|-- PC7
          PC5 --|2        23|-- PC6
          PC0 --|3        22|-- PC3
          PC1 --|4        21|-- PC2
          PA0 --|5_____________20|-- PB7
     INPUT  PA1 --|6(1A)   (VCC)19|-- PB6
     INPUT  PA2 --|7(1B)    (4B)18|-- PB5  INPUT
    OUTPUT  PA3 --|8(1Y)    (4A)17|-- PB4  INPUT
     INPUT  PA4 --|9(2A)    (4Y)16|-- PB3  OUTPUT
    OUTPUT  PA5 --|10(2B)   (3B)15|-- PB2  INPUT
     INPUT  PA6 --|11(2Y)   (3A)14|-- PB1  INPUT
          GND --|12(GND)  (3Y)13|-- PB0  OUTPUT
              |---------------|
```

PIN ASSIGNMENT OF 74LS00

PROCEDURES:

(1). VCC and GND type code : 00
The VCC & GND code are defined as below :

```
 _____________________________________
|                                     |
|  14     pin chip ---  code is : 00  |
|  16     pin chip ---  code is : 01  |
|  18     pin chip ---  code is : 02  |
|  20     pin chip ---  code is : 03  |
|  22     pin chip ---  code is : 04  |
|  24     pin chip ---  code is : 05  |
|_____________________________________|
```

74LS00 is a 14 pin chip, so VCC code is 00

(2) set DIRA, DIRB or PA and PB.(don't care PC)

The code bit of direction is defined as below :

```
 _____________________________________
|                                     |
|       VCC  pin code  bit is : 0     |
|       GND  pin code  bit is : 0     |
|       INPUT  pin code  bit is :  1  |
|      OUTPUT  pin code  bit is :  0  |
|  DON'T CARE  pin code  bit is : 1   |
|_____________________________________|
```

*** Set DIRA of PA ***

```
PORT A :  PA7   PA6  PA5  PA4   PA3  PA2  PA1  PA0
         GND    OUT  IN   IN    OUT  IN      IN  X
          0      0    1    1     0    1       1  1 = 37
```

so DIRA = 37

*** Set DIRA of PB ***

```
PORT B :  PB7   PB6  PB5  PB4   PB3  PB2  PB1  PB0
          X     VCC  IN   IN  OUT   IN   IN   OUT
          1      0    1    1     0    1       1  0 = B6
```

so DIRB = B6

(3). establish TEST PATTERN

level define : 0 = low level
1 = high level

Test pattern of PA :

```
   PORT A :  PA7  PA6  PA5  PA4  PA3  PA2  PA1  PA0
             GND  OUT    IN   IN  OUT    IN   IN      X
.....................................................
(1)  test bit input           0    0    0    0
     expected output       1         1
                  ------------------------------------
     conbination      0    1  0    0   1     0    0       0 = 48
```



```
.....................................................
(2)  test bit input           0    1    0    1
```

```
     expected output        1          1
                  ------------------------------------
     conbination       0    1   0    1   1      0    1        0 = 5A
     ..................................................
(3)  test bit input             1    0    1    0
     expected output        1          1
                  ------------------------------------
     conbination       0    1   1    0   1      1    0        0 = 6C
     ..................................................
(4)  test bit input             1    1    1    1
     expected output        0          0
                  ------------------------------------
     conbination       0    0   1    1   0      1    1        0 = 36
     ..................................................
       so TPA     is : 48     5A 6C 36 ( total 4 sets)

          PORT B :  PB7  PB6  PB5  PB4  PB3  PB2  PB1  PB0
                    X   VCC    IN   IN       OUT  IN   IN     OUT
      ..................................................
(1)  test bit input            0    0           0    0
     expected output                 1           1
                  ------------------------------------
      conbination      0    1  0    0        1  0    0        1 = 49
      ..................................................
(2)  test bit input            0    1           0    1
     expected output                 1           1
                  ------------------------------------
     conbination       0    1  0    1        1  0    1        1 = 5B
      ..................................................
(3)  test bit input            1    0           1    0
     expected output                 1           1
                  ------------------------------------
     conbination       0    1  1    0        1  1    0        1 = 6D
     ..................................................
(4)  test bit input            1    1           1    1
     expected output                 0           0
                  ------------------------------------
     conbination       0    1  1    1        0  1    1        0 = 76
     ..................................................
        so TPB is : 49 5B 6D 76  ( total 4 sets)

       (4). The total sets of test pattern is 4.
       Now,  we have established all the code  and patterns, as
       shown below :

            1. VCC     (VCC & GND code)  = 00
            2. DIRA (direction of PA) = 37
            3. DIRB (direction of PB) = B6
            4. DIRC (direction of PC) = DON'T CARE
            5. TPA     (test pattern of PA) = 48 5A 6C    36
Š
                                                    page 9-7

            6. TPB     (test pattern of PB) = 49 5B 6D    76
            7. TPC     (test pattern of PC) = DON'T CARE
            8. NO.     (total sets of test pattern) = 4


       (5). EDIT TEST   PATTERN

       1. Run      DMT02.EXE  file,  it  will  open  a  BUFFER for
```

editing user's test pattern and the BUFFER is assigned as below :

buffer 0 to 7F used for TPA
buffer 80 to FF used for TPB
buffer 100 to 17F used for TPC
buffer 180 used for DIRA
buffer 181 used for DIRB
buffer 182 used for DIRC
buffer 183 used for VCC
buffer 184 used for NO.

2. Key in 4 to do the edit function, and the screen will shown as,

Memory buffer starting address at 2133:0000 (buffer starting address)

```
-RDS     <-- enter RDS
DS 2C2A
:2133    <-- enter buffer starting address
-E0
2133:0000  00-48  00-5A  00-6C 00-36 <--enter TPA
-E80
2133:0080  00-49  00-5b  00-6D 00-76 <--enter TPB
-E180
2133:0180  00-37    <--enter DIRA
-E181
2133:0181  00-B6    <--enter DIRB
-E183
2133:0183  00-00    <--14 pin code is 00
-E184
2133:0184  00-04    <--4 sets of test pattern
-Q                  <--back     to main     menu
```

(6). DEBUG TEST PATTERN
Type 5 to debug user's test pattern, as below

1. Total pattern sets : 04
2. VCC & GND code : 00
3. port(A) in/out code : 37
4. port(B) in/out code : B6

Š

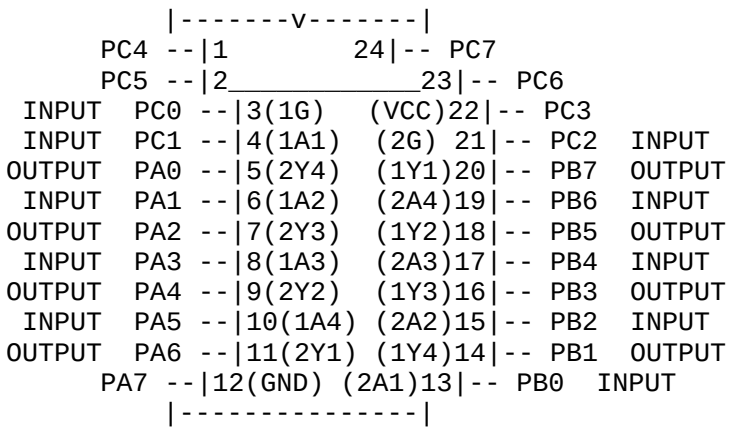
Put IC on socket, then press any key to test by step, or press <ESC> to quit.

```
1. Pattern write   port(A)=48   port(B)=49
           read   port(A)=48   port(B)=49
2. Pattern write   port(A)=5A   port(B)=5B
           read   port(A)=5A   port(B)=5B
3. Pattern write   port(A)=6C   port(B)=6D
           read   port(A)=6C   port(B)=6D
4. Pattern write   port(A)=36   port(B)=76
           read   port(A)=36   port(B)=76

   ****** END.  Press any key to continue.
```

(7). SAVE TEST PATTERN
Type 2 to save user's test pattern.

Expamle 2 : to edit TEST PATTERN for 74LS244



PIN ASSIGNMENT OF 74LS244

1. VCC and GND type code is 03

2, set DIRA, DIRB and DIRC or PORT A ,PORT B and PORT C.

PORT A : PA7 PA6 PA5 PA4 PA3 PA2 PA1 PA0
GND OUT IN OUT IN OUT IN OUT
0 0 1 0 1 0 1 0

so DIRA of PORT A is : 2A

PORT B : PB7 PB6 PB5 PB4 PB3 PB2 PB1 PB0
OUT IN OUT IN OUT IN OUT IN
0 1 0 1 0 1 0 1

Š

so DIRB of PORT B is : 55

PORT C : PC7 PC6 PC5 PC4 PC3 PC2 PC1 PC0
X X X X VCC IN IN IN
1 1 1 1 0 1 1 1 = F7

so DIRC of PORT C is : F7

3. establish TEST PATTERN

PORT A : PA7 PA6 PA5 PA4 PA3 PA2 PA1 PA0
GND OUT IN OUT IN OUT IN OUT

(1) test bit input 1 1 1
expected output 1 1 1 1
---
conbination 0 1 1 1 1 1 1 1 = 7F

(2) test bit input 0 0 0
expected output 1 1 1 1

```
     conbination       0    1  0    1        0  1    0        1 = 55
     ..................................................
(3)  test bit input           1      1        1
     expected output        1       1          1       1
                    -------------------------------------
     conbinatioî       0    1  1    1        1  1    1        1 = 7F
     ..................................................
      so TPA is : 7F 55 7F

          PORT B :  PB7  PB6  PB5  PB4  PB3  PB2  PB1  PB0
                   OUT  IN     OUT  IN       OUT  IN   OUT     IN
(1)  test bit input         1       1           1       1
     expected output   1       1       1       1
                    -------------------------------------
     conbination       1    1  1    1        1  1    1        1 = FF
     ..................................................
(2)  test bit input         0       0           0       0
     expected output   1       1       1       1
                    -------------------------------------
     conbination       1    0  1    0        1  0    1        0 = AA
     ..................................................
(3)  test bit input         1       1           1       1
     expected output   1       1       1       1
                    -------------------------------------
     conbination       1    1  1    1        1  1    1        1 = FF
     ..................................................
       so TPB     is : FF     AA FF

          PORT C :  PC7  PC6  PC5  PC4  PC3  PC2  PC1  PC0
                   X    X      X    X       VCC  IN   IN     IN
(1)  test bit input                            1        0
     expected output                    0
Š
                                                        page 9-10

                    -------------------------------------
     conbination       0    0  0    0        1  0    1        0 = 0A
     ..................................................
(2)  test bit input                            0        1
     expected output                   1
                    -------------------------------------
     conbinatioî       0    0  0    0        1  1    0        1 = 0D
     ..................................................
(3)  test bit input                            1        1
     expected output                   1
                    -------------------------------------
     conbination       0    0  0    0        1  1    1        1 = 0F
     ..........................................................
      so TPC is   : 0A,0D,0F

       (4). Total sets of test pattern is 3
       Now,  we have established all the code  and patterns, as
       shown below :

         1. VCC (VCC & GND code)  = 03
         2. DIRA (direction of PA) = 2A
         3. DIRB (direction of PB) = 55
         4. DIRC (direction of PC) = F7
         5. TPA (test   pattern     of PA) = 7F 55 7F
         6. TPB (test   pattern     of PB) = FF AA FF
         7. TPC (test   pattern     of PC) = 0A 0D 0F
```

8. NO. (total sets of test pattern) = 3

(5). EDIT TEST PATTERN

1. Run DMT-02.EXE file , it will open a BUFFER for editing user's test pattern and the BUFFER is assigned as below :

   buffer 0 to 7F used for TPA
   buffer 80 to FF used for TPB
   buffer 100 to 17F used for TPC
   buffer 180 used for DIRA
   buffer 181 used for DIRB
   buffer 182 used for DIRC
   buffer 183 used for VCC
   buffer 184 used for NO.

2. Key in 4 to do the edit function, and the screen will shown as,

   Memory buffer starting address at 2133:0000 (buffer starting address)

   -RDS <-- enter RDS
   DS 2C2A
   :2133 <-- enter buffer starting address

Š


```
-E0
2133:0000   00-7F     00-55  00-7F <--enter TPA
-E80
2133:0080   00-FF     00-AA  00-FF <--enter TPB
-E100
2133:0100   00-02     00-05  00-07 <--enter TPC
-E180
2133:0100   00-2A       <--enter DIRA
-E181
2133:0181   00-55       <--enter DIRB
-E182
2133:0182   00-FF       <--enter DIRC
-E183
2133:0183   00-03       <--20     pin code is 00
-E184
2133:0184   00-03       <--3 sets of test pattern
-Q           <--back to main menu
```

(6). DEBUG TEST PATTERN
Type 5 to debug user's test pattern, as below

1. Total pattern sets : 03
2. VCC & GND code : 03
3. port(A) in/out code : 2A
4. port(B) in/out code : 55
5. port(C) in/out code : FF

   Put IC on socket, then press any key to test by step, or press <ESC> to quit.

1. Pattern write port(A)=7F port(B)=FF port(C)=02
   read port(A)=7F port(B)=FF port(C)=02

2. Pattern write port(A)=55 port(B)=AA port(C)=05
   read port(A)=55 port(B)=AA port(C)=05
3. Pattern write port(A)=7F port(B)=FF port(C)=07
   read port(A)=7F port(B)=FF port(C)=07

****** END. Press any key to continue.

(7). SAVE TEST PATTERN
Type 2 to save user's test pattern.

Š

CHAPTER X OTHER SOFTWARES DESCRIPTION

1. DASM48.EXE & DASM51.EXE

DASM48.EXE is the dis-assembler for 8748/49; DASM51.EXE is the dis-assembler for 8751/52.
For example:

```
To execute DASM48.EXE
A>DASM48
HI-LO SYSTEM RESEARCH CO., LTD
8048/8049 DISASSEMBLER V1.0

-?

Command syntax
Dump memory    : D[start     address[,end address]]
Enter          :     E[start     address]
Disassembler   : U[start     address[,end address]]
Define filename: N[file spec.]
Load file      : L[start     address]
Write file     : W[start     address, end address]
Quit to DOS    : Q
Query command  : H or ?

NOTE: { ...} :   Contain     must be     specified
      [ ...] :   Contain     optional
Address     limit :     2047 (7FF Hex)
comma(,) may be   replaced by blank, dot or TAB
```

The usage of each function will be displayed as soon as you enter ?. The usage of those functions is same to DEBUG in DOS. Please refer to the description of DEBUG in DOS.

2. HEXOBJ02.EXE

HEXOBJ02.EXE is used to convert HEX FORMAT into BINARY CODE. It can also convert INTEL (80/86 HEX FORMAT) and MOTOLORA S1 /S2 FORMAT.

```
 A> HEXOBJ02
    HI-LO  HEX TO OBJECT CONVERTER V3.1   1988
---------------------------------------------------
   HEX file name : test
   OBJ file name : test
   HEX file format {<I>intel 80/86 /<M>MOTOLORA S1/S2 }:I

   Wait ...
   INTEL 80/86 Hex to Binary    converter
   Convert complete
   A>
```

Š

3. PARTS02.LST
   This software is a record of all available programmable ICs. Please print PARTS02.LST to review if you don't know which IC should be programmed by which software.

   PARTSO2.LST contains five parts
   a. EPROM / EEPROM
   b. PAL /FPL /GAL / PEEL
   C. Microcomputer
   d. BPROM
   e. IC & MEMORY TESTER

   Each part is divided as below:

```
Run file     : PAP02.EXE (V3.0)     ----- execute program
Manufacturer : MMI-A TYPE           ----- select MANUFACTURER
Type         :
10H8-2          12H6-2             ----- select TYPE
16C1-2          10L8-2
  :               :
  :               :
```

4. VERSION.DOC
   the list for all softwares upgraded.

5. MANUAL.DOC
   the User' manual.

6. TEST02.EXE
   the hardware check.(Please refter to CHAPTER XI 'DAMAGE CHECK)

7. IOCHK02.EXE
   the I/O ADDRESS check for universal programmer .

Š

CHAPTER XI DAMAGE CHECK

10.1 TEST02.EXE is mainly used to test whether the programmer's VL, VM & VH voltage, ground, indicator light, and I/O BUS are normally performed. (Should there be any problem arise during working, please contact the dealer or our company. We will fix and maintain your programmer at our soonest or solve problem for you. Please don't try to fix your programmer by yourself.)

10.2 In testing, simply test the voltage and the status of each pin on the socket.

10.3 Please prepare the following testing tool required in testing.

a. SCOPE.
b. DVM.

10.4 the user testing procedure After executing TEST02.EXE, the display shows as below:

```
-----   MAIN MENU  --------
User testing procedure:
(1) VL test and adjustment
(2) VM test and adjustment
(3) VH test and adjustment
(4) Ground   test
(5) I/O test
(6) Extra test
(Q) Quit
--------------------------
```

========== FUNCTION DESCRIPTION ==========

The +(-) 0.2 V difference for the following voltage values is allowed.

*** FUNCTION 1 : VL TESTING & ADJUSTMENT
To test and adjust VL voltage. (VR1)

<S>: To select the testing voltage. VL testing voltage is selected as 2.2V , 4.5V, 5.9V 7.7V, 8.3V, 8.7V etc.
-- totally six setting. In testing, you simply test the voltage status at pin 40.
<P>: To test each pin whether it's voltage value is same to that as showed on the screen. VL pin is

assigned at pin 25, 28, 29, 30, 32 34, 36 &40.
<ESC>: back to main menu.
Š

*** FUNCTION 2 : VM TESTING & ADJUSTMENT
To test and adjust VM voltage (VR3)

<S>: to select the testing voltage. VM testing voltage is selected as 1.7V, 3.1V, 4.6V, 9.1V, 10.3V, 12.6V, 17.3V, 18.7V, 21.9V etc. -- totally 9 settings. In testing, you simply test the voltage status at pin 1.
<P>: To test each pin whether it's voltage value is same to that as showed on the screen. VM pin is assigned at pin 1, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, &32.
<ESC>: back to main menu

*** FUNCTION 3 : VH TESTING & ADJUSTMENT
To test and adjust VH voltage (VR2)

<S>: To select the testing voltage. VH testing voltage is selected as 3.0V, 4.5V, 6.0V, 18.6V, 11.9V, 14.2V, 18.9V, 20.3V, 23.7V -- totally 9 setting. In testing, you simply test the voltage status at pin 21.
<P>: To test each pin whether its voltage value is same to that showed on the screen. VH pin is assigned at pin 9, 11, 13, 19, 21, 23, 24, 25, 26, 27, 28, 29, 30 & 32.
<ESC>: back to main menu.

*** FUNCTION 4 : GROUND TEST
To test ground.

To test whether pin 1, 11, 20, 23, 24, 25 28, 29 30 is from 0V to 0.3V or not , execute this function.
<ESC>: back to main menu.

*** FUNCTION 5 : I/O TEST
To test I/O BUS.

'1'
'0'
<CR>: to exchange odd pins with even pins
<ESC> : back to main menu

*** FUNCTION 6 : EXTRA TEST
To test the indicator light & the CLOCK.

Š

<1>: to test whether BUSY LED is normal or not. (

The light will become flashing if it is normal)
<2>: In executing this function, please test whether
pin 2, 3, 18 &19 have 2MHz frequency.
Press any key to quit.
<ESC>: back to main menu.

**** FUNCTION Q : QUIT
Quit this testing.