Turbo-Plus V1.41 8-bit Installation Guide

April, 1985

Microserve, Inc.
276 Fifth Avenue
New York, N.Y. 10001

TABLE OF CONTENTS

1] Introduction 1-1

Overview 1-2
Checklist 1-3

2] Turbo-Plus Modules 2-1

Program Modules 2-1
Relocatable and executable program files 2-2
Relocatable subroutine files 2-3
.GEN and .PAR files 2-3
System function files 2-3

3] Installing Turbo-Plus 3-1

Generating executable programs 3-1
System generation 3-1
Generating a new system master 3-1
Generating new slaves 3-5

4] Installing Background Batch 4-1

Overview 4-1
Patching 4-1
Slave generation 4-3

5] Appendix

Modifying a Slave Circuit Driver for Turbo-Plus A-1
Installing Turbo-Plus on a multi-circuit system B-1
Definition of the USRSUP calling sequence C-1

INTRODUCTION

This Installation Guide describes the procedure for generating a configuration of Turbo-Plus to meet a given system's specifications.

For information on using the package refer to the User's Guide to Turbo-Plus. That manual includes an overview of the package, and instructions on using each command.

Turbo-Plus is designed with the ability to be patched to run on systems with different search drives. Since it needs to know what drive this is, it is set up as a patchable parameter. Thus, most of Turbo-Plus's modules are distributed as relocatable (.REL) files, so that parameters may be 'plugged in' in the installation procedure using the TurboDOS symbolic patch facility described in the Configuration Guide to TurboDOS.

Section 2 of this guide briefly describes all of the modules sent as part of the Turbo-Plus package, and explains where on the system they should be placed.

Section 3 describes the installation procedure for generating a customized version of Turbo-Plus excluding the background batch processor, which involves some extra steps.

Section 4 describes the installation procedure for the background batch processor.

Overview

Turbo-Plus makes extensive use of the TurboDOS User Defined Function (TurboDOS call 29H) and follows the USRSUP calling protocol, outlined in your TurboDOS 1.41 update notes, and in Appendix C of this manual. Therefore, if you wish to add your own functions with this call, it is imperative that you also follow the USRSUP protocol, which has been adopted by Software 2000 as the standard TurboDOS method.

Serialization

Each copy of Turbo-Plus is serialized to be run only on a particular TurboDOS operating system. The serial number must agree with that of the operating system on which it is to be run. None of the modules in Turbo-Plus will run on any system with a serial number different that its own.

Turbo-Plus Installation Check List

THIS CHECKLIST IS FOR PURE 8 BIT INSTALLATIONS ONLY

[ ] 1. Read the Turbo-Plus Installation Guide.

[ ] 2. Run INSTALL.COM.

This program will ask questions about your TurboDOS version, and your SYSTEM BOOT drive and SEARCH drive. It will then create the proper .PAR files for your configuration and proceed to GEN the Turbo-Plus modules. When GEN is complete, it will copy all of the appropriate files to their correct destinations on your system.

[ ] 3. Run BBINSTAL.COM ¦OPTIONAL¦

If you desire to use the Background Batch commands, it will be necessary to run this installation program. BBINSTAL will ask the same questions as above and in addition will require a particular user area designation where it can reside when idle and maintain it's files.

[ ] 4. COPY all help (.HLP) files from Distribution Disk(s) to user 0 of your search drive.

[ ] 5. COPY TPLUSS.REL, TPLUSM.REL, CON96TP.REL, CONBB.REL, TWXTV.REL and TWXNUL.REL to the area of your disk where you generate your TurboDOS systems.

[ ] 6. GEN all Slaves - Be sure that each OSSLAVEx.GEN includes CON96TP.REL (replacing your existing CON96.REL), TPLUSS.REL and TWXTV.REL (or TWXNUL.REL), as well as USRSUP.REL, NETSVC.REL and NETFWD.REL (all supplied by Software 2000, Inc., but not generally included in slave generation).

[ ] 7. GEN Background Batch slave. ¦OPTIONAL¦

[ ] a. In order to allow the BATCH PROCESSING slave to recover from console input conditions (illegal in BB), substitute CONBB.REL for CON96TP.REL on the OSSLAVEx.GEN designated in the BBINSTAL session.

[ ] b. Include LOGUSR = NN (where NN = user area specified in BBINSTAL - i.e. LOGUSR = 1E for area 30) in this OSSLAVEx.PAR file.

(Cont'd)

[ ] 8. GEN the Master (OSMASTER.SYS).

Be sure to include TPLUSM.REL as well as USRSUP.REL, NETFWD.REL, NETREQ.REL and MSGFMT.REL (supplied by Software 2000, Inc.) in the OSMASTER.GEN file.

[ ] 9. Reset and test your system. You should first notice the new Turbo-Plus LOGON program if everything is properly installed.

Turbo-Plus Modules

Turbo-Plus will arrive on two CP/M format single-sided single-density diskettes. One of these disks will be a "flippy", giving you a total of three disk sides. Side one contains all of the relocatable modules; side two contains .DO, .GEN, and .PAR files used to generate your Turbo-Plus installation; side three contains all of the .HLP files necessary for the HELP command.

Program Modules

Side one contains three types of files:

1) **Relocatable program files:**
   Those files which constitute the main bodies of the Turbo-Plus utility programs.

2) **Relocatable subroutine files:**
   Those files which contain subroutines called by the programs above.

3) **System function files:**
   Files containing extensions to the normal set of TurboDOS operating system calls which must be genned into the operating system.

Side two contains all of the supporting files used for generating your installed version of Turbo-Plus. This includes .GEN and .PAR files for your programs, auxiliary data files, two installation .COM files, and .DO files referenced by the install programs.

Side three contains the text files (????????.HLP) for the HELP utility. All of these files should be moved to user 0 of the system search drive.

Relocatable and executable program files

These are all of the files containing the assembled source code for the Turbo-Plus utilities. They are distributed in relocatable form, to allow the patching of parameters.

DIRDUMP.REL	Program which gives a master directory of any disk, sorted by user area.
GO.REL	Program which moves users to a user area specified by a user-defined name.
GONAME.REL	Utility which allows users to define names for user areas on the system.
HELP.REL	TurboDOS on-line help facility providing help on all TurboDOS and Turbo-Plus commands. Users may add their own help files.
LOCATE.REL	Utility to search certain or all system drives for given file or template.
LOG.REL	Utility to make entries in a date and time stamped log file.
LOGOFF.REL	Enhanced version of system logoff, notifying users of pending mail, and displaying system bulletins.
LOGON.REL	Enhanced version of system logon, notifying users of pending mail, displaying system bulletins, and providing additional levels of security.
MAIL.REL	TurboDOS mail facility to allow electronic mail to be sent among users on the system.
MASTER.REL	Enhanced version of the TurboDOS 'MASTER' command, providing better control of access to the master.
PROFILE.REL	Program to maintain USERID.SYS file.
RESET.REL	Program to reset a slave from another slave.
STATUS.REL	Facility to continuously monitor activity of system users, printers, and buffers.
TWX.REL	TWX facility to allow users to send immediate messages to other consoles on the system.
USER.REL	Allows user to change user areas. Performs the same function as the TurboDOS USER command of versions 1.3 and earlier.
WHO.REL	System status facility to display all current users on the system, processes they are running and other current system characteristics.

Relocatable subroutine files

A number of routines are shared by various program modules. They include the following files:

DBUFF.REL	GBUFF.REL	LOGCHK.REL	LOGDAT.REL
MBUFF.REL	MROUTE.REL	PTABLE.REL	TABLES.REL
	TPMOD.REL	TPSCAN.REL	

System function files

These files must be moved to the user area on the system where your system's .GEN and .PAR files reside, and where your system generation takes place. Some of them must be genned into your system in order for Turbo-Plus to work. There are six such files, all on disk 1:

TPLUSS.REL	TPLUSM.REL
TWXNUL.REL	TWXTV.REL
CON96TP.REL	CONBB.REL

.GEN and .PAR files

These files are necessary to patch the modules to work under your system configuration.

All of the following programs have .GEN files, some of which are accompanied by .PAR files:

DIRDUMP	GO	GONAME	HELP
LOCATE	LOG	LOGOFF	LOGON
MAIL	MASTER	PROFILE	RESET
STATUS	TWX	USER	WHO

The following files are necessary for the installation procedure:

TPLUS.DO	INSTALL.COM

Installing Turbo-Plus

Generating executable programs

Before you begin your Turbo-Plus installation, make a backup of the distribution diskette(s). If you received Turbo-Plus on a single TurboDOS format disk, you may run the install procedure directly from that disk. If not, you must copy the first two disks onto any user area on the system **other than user zero of the search drive.**

To customize Turbo-Plus to your system configuration, execute the **INSTALL** command. This program will prompt you for your TurboDOS version number, system search drive and the drive which currently contains your system boot disk (**do not include the colon after the drive letters**); all of the necessary .COM files will be generated and moved down to the search drive. Then, all of the .HLP files should be moved from the distribution disk to user zero of the search drive. During execution of the INSTALL process, it is very possible that certain stages will return with system error messages such as 'File not found'. This is due to the fact that the procedure must make sure that if any of these programs were already present, in an older version, they are deleted. Thus, if the programs were not there, trying to delete them will yield error messages.

Note: If you do not have a system search drive, you must still give some drive parameter to be used by Turbo-Plus as the drive on which to maintain all of its files.

System Generation

Before Turbo-Plus may be brought up, it is necessary to generate a new operating system. You should start with the .GEN and .PAR files which you are currently using for both your slave(s) and your master, but some additions will be necessary.

Generating a new system master

The following changes must be made to the .GEN file for your system master (usually STDMASTR.GEN or OSMASTER.GEN). Using your system editor, insert the following lines after the line for the hardware initialization.

```
USRSUP
TPLUSM
```

The following three lines, containing modules which are supplied by Software 2000, but not generally included in the master configuration, should also be added after STDMASTR:

```
NETFWD
NETREQ
MSGFMT
```

If you have the ability to use the TWX and RESET commands, it is recommended that you use modified circuit drivers. Many existing circuit drivers have already been modified appropriately; if your dealer says that yours has not been, a revision will be necessary. There should be no change made to the master circuit driver, and the source for your slave circuit driver should be changed, following the instructions in Appendix A.

Certain functions performed by Turbo-Plus utilities call for routing tables to be set up in the master. Turbo-Plus handles these changes automatically for single-circuit systems, but if you are utilizing more than one circuit in your system, (i.e. two or more types of slave boards) you will also need to add a Turbo-Plus slave table to your master .PAR file. If this is the case for your system, please refer to Appendix B for instructions on creating this table.

Figure 3.1 shows a sample OSMASTER.GEN file prepared for Turbo-Plus.

Figure 3.1
Sample OSMASTER.GEN file

```
STDMASTR        ; STANDARD NETWORK MASTER CONFIGURATION
NETREQ          ; NETWORK REQUEST PROCESSOR
MSGFMT          ; NETWORK MESSAGE FORMAT TABLE
NETFWD          ; NETWORK MESSAGE FORWARDING
HDWNIT          ; HARDWARE INITIALIZATION
USRSUP          ; USER FUNCTION INTERFACE
TPLUSM          ; TURBO-PLUS FUNCTION EXTENSIONS
CONREM          ; REMOTE MASTER CONSOLE
LSTPAR          ; DRIVER FOR HIGH SPEED PRINTER
LSTETX          ; DRIVER FOR LETTER QUALITY PRINTER
LSTTAB          ; DRIVER FOR HIGH SPEED PRINTER EXPANDING TABS
SPDXXX          ; SERIAL & PARALLEL DRIVERS
BRTXXX          ; BAUD-RATE TABLES
RTCXXX          ; REAL-TIME CLOCK DRIVER
DSKXXX          ; FLOPPY DISK DRIVER
DST58F          ; FLOPPY DISK SPECIFICATION TABLES
DSKHHH          ; HARD DISK DRIVER
MCDXXX          ; MASTER CIRCUIT DRIVER
```

Figure 3.2
Sample OSMASTER.PAR file

```
  COMPAT = 0F0              ; Compatibility flags
  NMBSVC = 9                ; Number of slaves
  NMBXXX = 9                ; Number of slaves (9)
  NMBMBS = 1B               ; Number of message buffers (27)
  NMBUFS = 10               ; 16 I/O buffers
  NMBRPS = 1B               ; Number of network reply packets (27)
  PATXXX = 60,62,64,66,68,6A,6C,6E,70,72,74,76,78,7A,7C,7E
                            ; Slave Port assignment table
  DSKAST = 00,DSKDRA,01,DSKDRA,00,DSKDRB,01,DSKDRB,02,DSKDRB,03,DSKDRB,
           04,DSKDRB,05,DSKDRB,06,DSKDRB,07,DSKDRB,08,DSKDRB,09,DSKDRB,
           0A,DSKDRB,0B,DSKDRB
                            ; Disk assignment table:
                            ;    A,B = floppy drives
                            ;    C-N = Winchester disk
  PTRAST = 00,LSTDRA,01,LSTDRB,00,LSTDRC
                            ; Printer assignment table:
                            ;    A = High speed with raw output
                            ;    B = Letter quality with raw output
                            ;    C = High speed w/ formatted output
  MEMRES = (1000)           ; Reserved memory above TPA
  DSPPAT = 1,2,0,0,0,0,0,0,0,0,0
                            ; De-Spool table:
                            ;    Printer A --> Queue A
                            ;    Printer B --> Queue B
  AUTUSR = 80               ; Log master on to user 0, privileged
  QUEAST = 00,(0),00,(0),00,(0),00,(0),00,(0),00,(0),00,(0),00,(0)
                            ; Define eight valid queues (A-H)
  SRHDRV = 8                ; System search drive = H
  ETXBR = 0E                ; Baud rate on Printer B = 9600
```

Generating new slaves

Next, in your slave .GEN files, add lines containing **USRSUP** and **TPLUSS** following the hardware initialization module. Also include **NETSVC** and **NETFWD** immediately after the line for STDSLAVE. Also, to optimize the performance of the TWX command, you need a special console driver, modified circuit drivers, and a separate module to handle the shift-in shift-out produced by TWX.

If your standard console driver is CON96, you may use the CON96TP driver provided with Turbo-Plus. (To do so, simply replace the CON96 line in your .GEN file with CON96TP). If not, you should modify your driver such that before every console output, it performs a WAIT operation on the global semaphore: TWLOCK, and after each console output, it performs a SIGNAL operation on the same semaphore. It should also allow for a character to remove the TWX message from the 25th line of the screen, by calling the external routine TWXRST when this character is received. CON96TP uses the ESC character by default, and if there is no message on the status line, it allows the escape to passed through normally. Figure 3.5 shows the CON96TP driver, which may be used as a guideline.

Furthermore, for those of you using TWX and RESET, your circuit driver may require modification. Consult appendix A for the necessary changes.

The second module necessary for TWX handles the placing of TWX messages on the screen without interrupting normal console input/output. If you are using a Televideo terminal, you may use the **TWXTV** module, which places all received TWX messages on the terminal status line. For any other terminal you may use the **TWXNUL** module, which simply prints each line at the current cursor position, followed by a carriage return-line feed sequence. A source listing of this module is provided and explained in Figure 3.6, in case you wish to modify it for your specific terminal. Modification may be done either by writing your own driver, or patching TWXNUL in the slave .PAR file.

Figure 3.4 shows a sample slave .GEN file and figure 3.3 shows the corresponding .PAR file.

Once all of these changes are complete, you are ready to generate the new master and slaves using the GEN command in the usual way. (Refer to the TurboDOS configuration Guide.) Once all of these steps are done, copy the newly created .SYS files down to user zero of your boot disk, and Turbo-Plus will be ready to come up.

Figure 3.3
Sample STDSLAVE.PAR file

```
COMPAT = 0F0            ; File Compatibility flags
SRHDRV = 8              ; System search drive = H
PRTMOD = 01             ; Print mode = Spooled
QUEPTR = 1              ; Default Queue = A
SPLDRV = 8              ; Spool Drive = I
```

Figure 3.4
Sample STDSLAVE.GEN file

```
STDSLAVE        ; STANDARD NETWORK SLAVE CONFIGURATION
NETSVC          ; NETWORK SERVICE PROCESS
NETFWD          ; NETWORK MESSAGE FORWARDING
NITXXX          ; HARDWARE INITIALIZATION
USRSUP          ; USER FUNCTION CALLING MODULE
TPLUSS          ; TURBO-PLUS FUNCTION EXTENSIONS
CON96TP         ; TURBO-PLUS ASCII CONSOLE AT 9600 BAUD
TWXTV           ; TWX CONSOLE MANAGER FOR TELEVIDEO 950/925/800
SPDXXX          ; SERIAL & PARALLEL DRIVERS
SLVRES          ; SUBROUTINE FOR KEYBOARD RESET OF SLAVE
SCDXXX          ; SLAVE CIRCUIT DRIVER
```

Special hardware-dependent modules, described in the paragraphs above, are in boldface.

Figure 3.5
Sample Turbo-Plus Console Driver

```
CON96T - TURBODOS OPERATING SYSTEM NULL CONSOLE DRIVER
COPYRIGHT (C) 1982, SOFTWARE 2000, INC.


                        ;
                        ; COPYRIGHT (C) 1982, SOFTWARE 2000, INC.
                        ;
                        ; AUTHORS: RONALD E. RAIKES
                        ;          MICHAEL D. BUSCH
                        ;
                        ; VERSION: 01/20/82
                        ;
                        ;       Edit History: JBG : 18-Jan-83 : TWX MX-Locks ins
                        ;                     JBG :    Jun-83 : Control code loc
                        ;                     JBG :  9-Nov-83 : 1.30 Optimized o
                        ;                     JBG : 17-Jul-84 : Optimized output
                        ;                     JBG : 28-Sep-84 : Included with V.
                        ;                     JBG : 30-Nov-84 : Clear status lin
                        ;
                        .IDENT  CON96T                  ; MODULE ID
                        ;
                        .INSERT DREQUATE                ; DRIVER SYMBOLIC EQUIVA
                        ;
0000"                           .LOC    .DATA.#         ; LOCATE IN DATA AREA
                        ;
0000"   8E              CONBR:: .BYTE   8EH             ; CONSOLE BAUD RATE CO
0001"   0C              FFCHR:: .BYTE   AFF             ; FORM FEED CHARACTER
0002"                   R25CHR::
0002"   1B                      .BYTE   1BH             ; RESTORE 25TH LINE CHAR
0003"   00              INITC:  .BYTE   0               ; INITIALIZATION COMPLET
0004"   00              EFLAG:  .BYTE   0
0005"   00              BCOUNT: .BYTE   0
0006"   0000            SCOUNT: .WORD   0
                        ;
0000'                   .LOC    .PROG.#         ; LOCATE IN PROGRAM AREA
                        ;
0000'   21 0003"        CONDR%::LXI     H,INITC         ; GET INITIALIZATION COM
0003'   7E                      MOV     A,M
0004'   B7                      ORA     A               ; INITIALIZATION COMPLET
0005'   CC 0106'                CZ      CNINIT          ; IF NOT, INITIALIZE CON
0008'   7B              ..CDRV: MOV     A,E             ; GET FUNCTION NUMBER
0009'   D608                    SUI     8               ; FUNCTION NUMBER=8?
000B'   CA 0123'                JZ      CONSO           ; IF SO, ERROR SHIFT OUT
000E'   3D                      DCR     A               ; FUNCTION NUMBER=9?
000F'   CA 0123'                JZ      CONSI           ; IF SO, ERROR SHIFT IN
0012'   3D                      DCR     A               ; FUNCTION NUMBER = 10?
0013'   CA 0129'                JZ      OPT             ; IF SO, JUMP TO OPTIMIZ
0016'   7B                      MOV     A,E             ; GET BACK IN A
0017'   B7                      ORA     A               ; IF 0, GO TO CONSTAT
0018'   2838                    JRZ     CONST           ; .
001A'   3D                      DCR     A               ; IF 1, CONIN
001B'   2807                    JRZ     CONIN           ; .
```

```
CON96T - TURBODOS OPERATING SYSTEM NULL CONSOLE DRIVER
COPYRIGHT (C) 1982, SOFTWARE 2000, INC.


001D'   3D                      DCR     A               ; IF 2, CONOUT
001E'   CA 00A6'                JZ      CONOUT          ; .  ******
0021'   C3 0000:04              JMP     SERIAL#         ; .
                        ;
0024'                   CONIN:
0024'   3A 0004"                LDA     EFLAG           ; IF ESCAPE FLAG IS NOT
0027'   B7                      ORA     A               ; .
0028'   CA 002F'                JZ      ..SER           ; JUST GO TO SERIAL, ELS
002B'   AF                      XRA     A               ; RESET FLAGS AND SEMAPH
002C'   CD 0093'                CALL    RELEAS          ; .
002F'                   ..SER:
002F'   C5                      PUSH    B               ; SAVE CHANNEL NUMBER
0030'   D5                      PUSH    D               ; AND FUNCTION NUMBER
0031'   CD 0000:04              CALL    SERIAL#         ; GET THE CHARACTER
0034'   F5                      PUSH    PSW             ; SAVE IT
0035'   E67F                    ANI     7FH             ; STRIP PARITY
0037'   47                      MOV     B,A             ; SAVE IT
0038'   3A 0002"                LDA     R25CHR          ; COMPARE TO 25TH LINE R
003B'   B8                      CMP     B               ; .
003C'   C2 004E'                JNZ     ..RET           ; IF SO
003F'   3A 0000:05              LDA     LINE25#         ;    CHECK FOR PRESENCE
0042'   B7                      ORA     A               ;    .
0043'   2809                    JRZ     ..RET           ;    IF NONE, SKIP THIS
0045'   CD 0000:06              CALL    TWXRST#         ;    RESTORE THE LINE
0048'   F1                      POP     PSW             ;    GET REGISTERS OF TH
0049'   D1                      POP     D               ;    . BECAUSE WE NEED T
004A'   C1                      POP     B               ;    . AND THE CHANNEL I
004B'   C3 0000:04              JMP     SERIAL#         ;    GO AHEAD TO SERIAL#
004E'                   ..RET:                          ; ELSE
004E'   F1                      POP     PSW             ;    GET CHAR BACK IN A
004F'   D1                      POP     D               ;    RESTORE THE STACK
0050'   C1                      POP     B               ; ENDIF
0051'   C9                      RET                     ; RETURN
                        ;
0052'                   CONST:
0052'   3A 0004"                LDA     EFLAG           ; IF ESCAPE FLAG IS NOT
0055'   B7                      ORA     A               ; .
0056'   CA 0067'                JZ      ..SER           ; JUST GO TO SERIAL, ELS
0059'   E5                      PUSH    H               ; SAVE HL
005A'   2A 0006"                LHLD    SCOUNT          ; CHECK CON STAT COUNT
005D'   23                      INX     H               ; BUMP IT
005E'   22 0006"                SHLD    SCOUNT          ; SAVE NEW STAT COUNT
0061'   7C                      MOV     A,H             ; IF 0, RELEASE ESCAPE F
0062'   B5                      ORA     L               ; .
0063'   CC 0093'                CZ      RELEAS          ; .
0066'   E1                      POP     H               ; RESTORE HL
0067'                   ..SER:
0067'   C5                      PUSH    B               ; SAVE CHANNEL NUMBER
0068'   CD 0000:04              CALL    SERIAL#         ; AND GO TO SERIAL
006B'   D1                      POP     D               ; GET CHANNEL NUMBER BAC
006C'   B7                      ORA     A               ; IF NOTHING AVAILABLE,
```

```
CON96T - TURBODOS OPERATING SYSTEM NULL CONSOLE DRIVER
COPYRIGHT (C) 1982, SOFTWARE 2000, INC.


006D'   C8                      RZ                      ; .
006E'   F5                      PUSH    PSW             ; SAVE REGISTERS FOR RET
006F'   C5                      PUSH    B               ; .
0070'   79                      MOV     A,C             ; .
0071'   E67F                    ANI     7FH             ; STRIP PARITY
0073'   47                      MOV     B,A             ; SAVE IT
0074'   3A 0002"                LDA     R25CHR          ; COMPARE TO 25TH LINE R
0077'   B8                      CMP     B               ; .
0078'   C2 0090'                JNZ     ..RST           ; IF SO
007B'   3A 0000:05              LDA     LINE25#         ;    CHECK FOR PRESENCE
007E'   B7                      ORA     A               ;    .
007F'   280F                    JRZ     ..RST           ;    IF NONE, SKIP THIS
0081'   CD 0000:06              CALL    TWXRST#         ;    RESTORE THE LINE
0084'   42                      MOV     B,D             ;    GET CHANNEL NUMBER
0085'   1E01                    MVI     E,1             ;    CEALL CONIN TO FLUS
0087'   CD 0000:04              CALL    SERIAL#         ;    .
008A'   C1                      POP     B               ;    GET REGISTERS OFF S
008B'   F1                      POP     PSW             ;    .
008C'   AF                      XRA     A               ;    FLAG NO CHARACTER A
008D'   C3 0092'                JMP     ..RET           ; ELSE
0090'                   ..RST:                          ;    .
0090'   C1                      POP     B               ;    RESTORE REGISTERS N
0091'   F1                      POP     PSW             ;    .
0092'                   ..RET:                          ; ENDIF
0092'   C9                      RET                     ; RETURN
                        ;
0093'                   RELEAS:
0093'   32 0004"                STA     EFLAG           ; .
0096'   32 0005"                STA     BCOUNT          ; .
0099'   C5                      PUSH    B               ;    SAVE REGISTERS
009A'   D5                      PUSH    D               ;    .
009B'   E5                      PUSH    H               ;    .
009C'   21 0000:07              LXI     H,TWLOCK#       ;   RELEASE CONSOLE
009F'   CD 0000:08              CALL    SIGNAL#         ;    .
00A2'   E1                      POP     H               ;    RESTORE REGISTERS
00A3'   D1                      POP     D               ;    .
00A4'   C1                      POP     B               ; ENDIF
00A5'   C9                      RET                     ; RETURN
                        ;
00A6'                   CONOUT:
00A6'   3A 0004"                LDA     EFLAG           ; IF WE ARE IN THE MIDDL
00A9'   B7                      ORA     A               ;    .
00AA'   280E                    JRZ     ..NSEQ          ;    .
00AC'   3A 0005"                LDA     BCOUNT          ;    GET BYTE COUNTER
00AF'   3D                      DCR     A               ;    DECREMENT IT
00B0'   32 0005"                STA     BCOUNT          ;    STORE IT
00B3'   2024                    JRNZ    ..CONT          ;    IF IT'S ZERO
00B5'   32 0004"                STA     EFLAG           ;       TURN OFF ESCAPE
00B8'   181F                    JMPR    ..CONT          ;    ENDIF
00BA'                   ..NSEQ:                         ; ELSE
00BA'   79                      MOV     A,C             ;    GET BYTE IN A
```

```
CON96T - TURBODOS OPERATING SYSTEM NULL CONSOLE DRIVER
COPYRIGHT (C) 1982, SOFTWARE 2000, INC.

00BB'   E67F                    ANI     7FH             ;    STRIP PARITY
00BD'   FE1B                    CPI     1BH             ;    CHECK FOR ESCAPE
00BF'   200C                    JRNZ    ..NESC          ;    IF ESCAPE
00C1'   F6FF                    ORI     0FFH            ;       SET FLAG
00C3'   32 0004"                STA     EFLAG           ;       .
00C6'   3C                      INR     A               ;       .
00C7'   32 0006"                STA     SCOUNT          ;       AND INITIALIZE S
00CA'   32 0007"                STA     SCOUNT+1        ;       .
00CD'                   ..NESC:                         ;    ENDIF
00CD'   C5                      PUSH    B               ;    SAVE REGISTERS
00CE'   D5                      PUSH    D               ;    .
00CF'   E5                      PUSH    H               ;    .
00D0'   21 0000:07              LXI     H,TWLOCK#       ;  WAIT FOR CONSOLE FRE
00D3'   CD 0000:09              CALL    WAIT#           ;    .
00D6'   E1                      POP     H               ;    RESTORE REGISTERS
00D7'   D1                      POP     D               ;    .
00D8'   C1                      POP     B               ;    .
00D9'                   ..CONT:                         ; END IF
00D9'   79                      MOV     A,C             ; GET BYTE IN A
00DA'   E67F                    ANI     7FH             ; STRIP PARITY
00DC'   FE1B                    CPI     1BH             ; IF IT'S AN ESCAPE
00DE'   2011                    JRNZ    ..NE2           ;    .
00E0'   3E06                    MVI     A,6             ;    INITIALIZE BYTE COU
00E2'   32 0005"                STA     BCOUNT          ;    .
00E5'   F6FF                    ORI     0FFH            ;    SET FLAG
00E7'   32 0004"                STA     EFLAG           ;    .
00EA'   3C                      INR     A               ;    AND INITIALIZE STAT
00EB'   32 0006"                STA     SCOUNT          ;    .
00EE'   32 0007"                STA     SCOUNT+1        ;    .
00F1'                   ..NE2:                          ; ENDIF
00F1'   CD 0000:04              CALL    SERIAL#         ; PRINT THE BYTE
00F4'   3A 0004"                LDA     EFLAG           ; CHECK ESCAPE FLAG
00F7'   B7                      ORA     A               ; IF IT WAS NOT SET
00F8'   C0                      RNZ                     ;    .
00F9'   C5                      PUSH    B               ;    SAVE REGISTERS
00FA'   D5                      PUSH    D               ;    .
00FB'   E5                      PUSH    H               ;    .
00FC'   21 0000:07              LXI     H,TWLOCK#       ;  RELEASE CONSOLE
00FF'   CD 0000:08              CALL    SIGNAL#         ;    .
0102'   E1                      POP     H               ;    RESTORE REGISTERS
0103'   D1                      POP     D               ;    .
0104'   C1                      POP     B               ; ENDIF
0105'   C9                      RET                     ; RETURN
                        ;
0106'   35              CNINIT: DCR     M               ; SET INITIALIZATION COM
0107'   D5                      PUSH    D               ; SAVE FUNCTION NUMBER
0108'   C5                      PUSH    B               ; SAVE CHANNEL NUMBER/CH
0109'   3A 0000"                LDA     CONBR           ; GET CONSOLE BAUD RATE
010C'   4F                      MOV     C,A             ; TELEVIDEO BAUD RATE CO
010D'   1E03                    MVI     E,3             ; SET FUNCTION NUMBER=3
010F'   CD 0000:04              CALL    SERIAL#         ; SET CHANNEL BUAD RATE
```

```
CON96T - TURBODOS OPERATING SYSTEM NULL CONSOLE DRIVER
COPYRIGHT (C) 1982, SOFTWARE 2000, INC.


0112'   3A 0001"            LDA     FFCHR           ; GET FORM FEED CHARACTE
0115'   B7                  ORA     A               ; FORM FEED CHARACTER=0?
0116'   2808                JRZ     ..NITX          ; IF SO, CONTINUE
0118'   C1                  POP     B               ; ELSE, RESTORE CHANNEL
0119'   C5                  PUSH    B               ; SAVE CHANNEL NUMBER
011A'   4F                  MOV     C,A             ; FORM FEED CHARACTER TO
011B'   1E02                MVI     E,2             ; SET FUNCTION NUMBER=2
011D'   CD 0000:04          CALL    SERIAL#         ; OUTPUT FORM FEED
0120'   C1          ..NITX: POP     B               ; RESTORE CHANNEL NUMBER
0121'   D1                  POP     D               ; RESTORE FUNCTION NUMBE
0122'   C9                  RET                     ; DONE
                    ;
0123'               CONSO:
0123'   CD 0000:0A  CONSI:  CALL    DMS#            ; POSITION TO NEXT LINE
0126'   0D8A                .ASCIS  [ACR] [ALF]
0128'   C9                  RET                     ; DONE
                    ;
0129'               OPT:
0129'   3A 0000:07          LDA     TWLOCK#         ; LOOK AT TWX LOCK
012C'   B7                  ORA     A               ;    IF IN USE, RETURN U
012D'   C8                  RZ                      ;
012E'   3A 0004"            LDA     EFLAG           ; ELSE, IF IN MIDDLE OF
0131'   B7                  ORA     A
0132'   2802                JRZ     ..NSEQ          ;      .
0134'   AF                  XRA     A               ;      RETURN UNSUCCESS
0135'   C9                  RET                     ;      .
0136'               ..NSEQ:                         ; ELSE IF THE CHARACTER
0136'   79                  MOV     A,C             ;      .
0137'   E67F                ANI     7FH             ;      .
0139'   D61B                SUI     AESC            ;      RETURN UNSUCCESSFUL
013B'   C8                  RZ                      ;      .
013C'   C3 0000:04          JMP     SERIAL#         ; ELSE, JUST DO THE OUTP
                    ;
                    .END
```

Figure 3.6
TWX Null Console Manager

```
/XNUL - Turbo-Plus TWXNUL driver

                        ;
                        ;       Shift-In/Shift-Out controls for null console
                        ;
                        ;       AUTHOR: Jim Gabriel
                        ;               Microserve, Inc.
                        ;
                        ;       Edit History: JBG : 24-Aug-83 : Revised for Turb
                        ;                     JBG : 24-Jul-84 : Revised for Turb
                        ;                     JBG : 28-Sep-84 : Equates changed
                        ;                     JBG : 30-Nov-84 : Clear status lin
                        ;
                        .IDENT  TWXNUL
                        ;
                        .INSERT DREQUATE
                        .INSERT TEQUATE
                        ;
0000"                           .LOC    .DATA.#
                        ;
0000"                   LINE25::
0000"   00                      .BYTE   0
0001"                   SICODE::
0001"   0D0A00000000            .BYTE   ACR,ALF,0,0,0,0,0,0,0,0
000B"                   SOCODE::
000B"   0D0A00000000            .BYTE   ACR,ALF,0,0,0,0,0,0,0,0
0015"                   RSTCOD::
0015"   000000000000            .BYTE   0,0,0,0,0,0,0,0,0,0
                        ;
0000'                           .LOC    .PROG.#
0000'                   TWXSI::
0000'   F5                      PUSH    PSW             ; SAVE REGISTERS
0001'   C5                      PUSH    B               ; .
0002'   D5                      PUSH    D
0003'   E5                      PUSH    H               ; .
0004'   21 0001"                LXI     H,SICODE        ; SET HL FOR SHIFT IN
0007'   1819                    JMPR    SCONT           ; .
                        ;
0009'                   TWXSO::
0009'   F5                      PUSH    PSW             ; SAVE REGISTERS
000A'   C5                      PUSH    B               ; .
000B'   D5                      PUSH    D
000C'   E5                      PUSH    H               ; .
000D'   F6FF                    ORI     0FFH            ; SIGNAL THAT A MESSAGE
000F'   32 0000"                STA     LINE25          ; .
0012'   21 000B"                LXI     H,SOCODE        ; SET HL FOR SHIFT OUT
0015'   180B                    JMPR    SCONT           ; .
                        ;
0017'                   TWXRST::
0017'   F5                      PUSH    PSW             ; SAVE REGISTERS
0018'   C5                      PUSH    B               ; .
`019'   D5                      PUSH    D
```

```
TWXNUL - Turbo-Plus TWXNUL driver
Microserve, Inc.

001A'   E5                      PUSH    H               ; .
001B'   AF                      XRA     A               ; SIGNAL THAT NO MESSAGE
001C'   32 0000"                STA     LINE25          ; .
001F'   21 0015"                LXI     H,RSTCOD        ; SET HL FOR SHIFT OUT
0022'                   SCONT:
0022'   3A 0000:04              LDA     CONAST#         ; GET CONSOLE CHANNEL IN
0025'   57                      MOV     D,A             ; .
0026'                   CLOOP:                          ; FOR EACH BYTE DO
0026'   7E                      MOV     A,M             ;    GET BYTE IN E
0027'   B7                      ORA     A               ;    .
0028'   280E                    JRZ     SRET            ;    .
002A'   5F                      MOV     E,A             ;    .
002B'                   CMOUTO::                        ;    .
002B'   0E24                    MVI     C,24H           ;    SET PARM FOR CONOUT
002D'   D5                      PUSH    D               ;    SAVE CHANNEL NUMBER
002E'   E5                      PUSH    H               ;    SAVE POINTER
002F'   AF                      XRA     A               ;    .
0030'   CD 0000:05              CALL    XTNTRY#         ;    SEND TO COM CHANNEL
0033'   E1                      POP     H               ;    RESTORE POINTER
0034'   D1                      POP     D               ;    RESTORE CH NO.
0035'   23                      INX     H               ;    INCREASE POINTER
0036'   18EE                    JMPR    CLOOP           ; END DO
                        ;
0038'   E1              SRET:   POP     H               ; RESTORE REGISTERS
0039'   D1                      POP     D
003A'   C1                      POP     B               ; .
003B'   F1                      POP     PSW             ; .
003C'   C9                      RET                     ; RETURN
                        ;
                        .END
```

To modify this driver you may either write your own, or use the symbolic patch facility. The primary reason to write your own would be to perform operations other than a simple console output of a string of bytes, such as code to also keep track of the cursor position before the message.

If you wish to do this, the module must meet the following specifications: It must have the global entry points TWXSI, which will be called before every TWX line, to position the cursor as desired; TWXSO, which will be called after every TWX line, to restore the cursor; and TWXRST, which will be called to remove the TWX message from the 25th line. All console output must be done via calls to the comm channel, which is defined in register D upon entry to the routine.

If your only modifications involve changing the string of bytes to be sent out before and after each message, it will probably be more convenient to use the TurboDOS symbolic patch facility. The routine allows for up to ten bytes to be patched at locations SICODE, SOCODE, and RSTCOD for the sequences to be sent out before the message, after the message, and to remove the message respectively. For example, if you wish to send out five bells and a clear screen at the beginning, five bells and a carriage return-line feed sequence at the end, and an ESCAPE, control-G sequence to remove the message, your .PAR file for the slave could be patched as follows, using TWXNUL:

```
SICODE = 07,07,07,07,07,0C
SOCODE = 07,07,07,07,07,0D,0A
RSTCOD = 1B,07
```

However, if you are using one type of terminal frequently, it may be easiest to write a special driver for it, even if it only involves changing the bytes, so that you need not change every .PAR file which you use. An example of such a driver is TWXTV, shown in Figure 3.7, written for the Televideo 800, 925, and 950 terminals. This driver is designed to take advantage of the status line of the terminal. All TWX messages will appear on this line, leaving the user's screen intact.

Figure 3.7
TWX Televideo Console Manager

```
TWXTV  - Turbo-Plus TWXTV driver
Microserve, Inc.


                        ;
                        ;       Shift-In/Shift-Out controls for Televideo 925/95
                        ;
                        ;       AUTHOR: Jim Gabriel
                        ;               Microserve, Inc.
                        ;
                        ;       Edit History: JBG : 24-Aug-83 : Revised for Turb
                        ;                     JBG : 24-Jul-84 : Revised for Turb
                        ;                     JBG : 28-Sep-84 : Equates changed
                        ;                     JBG : 30-Nov-84 : Clear status lin
                        ;
                        .IDENT  TWXTV
                        ;
                        .INSERT DREQUATE
                        .INSERT TEQUATE
                        ;
0000"                           .LOC    .DATA.#
                        ;
0000"                   LINE25::
0000"   00                      .BYTE   0
0001"                   SICODE::
0001"   071B671B661B            .BYTE   ABEL,AESC,67H,AESC,66H,AESC,47H,3CH,0
000B"                   SOCODE::
000B"   0D0000000000            .BYTE   ACR,0,0,0,0,0,0,0,0,0
0015"                   RSTCOD::
0015"   1B6800000000            .BYTE   AESC,68H,0,0,0,0,0,0,0,0
                        ;
0000'                           .LOC    .PROG.#
0000'                   TWXSI::
0000'   F5                      PUSH    PSW             ; SAVE REGISTERS
0001'   C5                      PUSH    B               ; .
0002'   D5                      PUSH    D
0003'   E5                      PUSH    H               ; .
0004'   21 0001"                LXI     H,SICODE        ; SET HL FOR SHIFT IN
0007'   1819                    JMPR    SCONT           ; .
                        ;
0009'                   TWXSO::
0009'   F5                      PUSH    PSW             ; SAVE REGISTERS
000A'   C5                      PUSH    B               ; .
000B'   D5                      PUSH    D
000C'   E5                      PUSH    H               ; .
000D'   F6FF                    ORI     0FFH            ; SIGNAL THAT A MESSAGE
000F'   32 0000"                STA     LINE25          ; .
0012'   21 000B"                LXI     H,SOCODE        ; SET HL FOR SHIFT OUT
0015'   180B                    JMPR    SCONT           ; .
                        ;
0017'                   TWXRST::
0017'   F5                      PUSH    PSW             ; SAVE REGISTERS
```

```
TWXTV  - Turbo-Plus TWXTV driver
Microserve, Inc.

0018'   C5                      PUSH    B               ; .
0019'   D5                      PUSH    D
001A'   E5                      PUSH    H               ; .
001B'   AF                      XRA     A               ; SIGNAL THAT NO MESSAGE
001C'   32 0000"               STA     LINE25          ; .
001F'   21 0015"               LXI     H,RSTCOD        ; SET HL FOR SHIFT OUT
0022'                   SCONT:
0022'   3A 0000:04             LDA     CONAST#         ; GET CONSOLE CHANNEL IN
0025'   57                      MOV     D,A             ; .
0026'                   CLOOP:                          ; FOR EACH BYTE DO
0026'   7E                      MOV     A,M             ;    GET BYTE IN E
0027'   B7                      ORA     A               ;    .
0028'   280E                    JRZ     SRET            ;    .
002A'   5F                      MOV     E,A             ;    .
002B'                   CMOUTO::                        ;    .
002B'   0E24                    MVI     C,24H           ;    SET PARM FOR CONOUT
002D'   D5                      PUSH    D               ;    SAVE CHANNEL NUMBER
002E'   E5                      PUSH    H               ;    SAVE POINTER
002F'   AF                      XRA     A               ;    .
0030'   CD 0000:05              CALL    XTNTRY#         ;    SEND TO COM CHANNEL
0033'   E1                      POP     H               ;    RESTORE POINTER
0034'   D1                      POP     D               ;    RESTORE CH NO.
0035'   23                      INX     H               ;    INCREASE POINTER
0036'   18EE                    JMPR    CLOOP           ; END DO
                        ;
0038'   E1              SRET:   POP     H               ; RESTORE REGISTERS
0039'   D1                      POP     D
003A'   C1                      POP     B               ; .
003B'   F1                      POP     PSW             ; .
003C'   C9                      RET                     ; RETURN
                        ;
                        .END
```

INSTALLING BACKGROUND BATCH

Overview

The Turbo-Plus Background Batch System operates on its own dedicated slave board. It requires a number of .COM files and related data files. It allows job scheduling, maintains a log of batch operation, and offers utilities to list current and pending jobs and to delete jobs.

The batch system requires two user areas: one on the system boot disk, and another on user zero of any drive on the system. Furthermore, it requires the presence of supporting .COM files in user 0 of the system search drive. All of the modules can be easily installed in any user area using the background batch installation program, BBINSTAL.

Patching

The program modules which require patching are BB, BBACK, BBCANCEL, BBDEL, BBEGIN, BBLIST, and BBLOG. The patches are needed to tell the batch system on which user area its files will be kept. To do this customization, run the BBINSTAL program included on the distribution diskette. This program will issue a series of questions about the manner in which you want to set up your background batch. It will then proceed to generate the necessary parameter files, and start a DO process to generate the .COM files, and move all of the modules to the necessary user areas on the system. A sample background batch installation session follows. All user input is underlined.

```
5F}BBINSTAL

BB requires one user area on the system boot disk where a
WARMSTRT.AUT file will be placed.  Nobody else should log
on to this area of the boot disk.  Which area would you
like this to be? (1-30): 1

BB requires one user area anywhere else on the system
where it maintains all of its files.  This should be
preferably on the hard disk, if you have one.  It will
use user 0 on the drive you select.

What drive would you like it to use? (A-P): H

What is your system search drive? (A-P): H

The Background Batch processor will require one slave
board dedicated to it.  Which slave will you set up
to service the background batch? (A-P): B
```

```
Copyright (C) 1982, Software 2000, Inc.
BB
TPMOD
Pass 1
BB      TPEQU  TPMOD
Pass 2
BB      TPEQU  TPMOD
Processing parameter file:
DRIVE = 07

Writing output file




5F}COPY
* BBEGIN.COM 01A:WARMSTRT.AUT;N
 5F:BBEGIN  .COM    copied to    1A:WARMSTRT.AUT
* BB.COM OH:;N
 5F:BB      .COM    copied to    0H:BB       .COM
* BBLIST.COM OH:;N
 5F:BBLIST  .COM    copied to    0H:BBLIST   .COM
* BBACK.COM OH:;N
 5F:BBACK   .COM    copied to    0H:BBACK    .COM
* BBLOG.COM OH:;N
 5F:BBLOG   .COM    copied to    0H:BBLOG    .COM
* BBCUR.JOB OH:;N
 5F:BBCUR   .JOB    copied to    0H:BBCUR    .JOB
* BBJOBS OH:;N
 5F:BBJOBS  .       copied to    0H:BBJOBS   .
* BBLOG 01H:;N
 5F:BBLOG   .       copied to    0H:BBLOG    .
*

5F}
```

Slave Generation

Finally, a number of modifications to your system generation must be completed:

Two changes must be made in the system .SYS files: First, a new slave must be generated for the batch system. This slave should have one change made in its .GEN file: Replace its console driver (typically CON96) with CONBB.REL, also supplied on user 0 of the installation disk. The slave's .PAR file should be changed with the **LOGUSR** parameter included, setting up the slave to log automatically onto the user area containing WARMSTRT.AUT. (E.g. If you choose to warmstart into user 1 of the boot drive, the patch should be LOGUSR = 1.) It is also advisable to have this slave printing to some remote queue or to file, rather than directly to a printer or to console, since in the latter two cases it would be easy to lose desired output produced by any jobs running in the batch processor. This is accomplished via the PRTMOD and QUEPTR parameters documented in the TurboDOS Configuration Guide. This slave must then be generated into the system master file, by changing the slave table, NMBSLV parameter, and NMBXXX parameter (where XXX is the particular slave.) The new slave and master must be generated in the normal system generation manner, and the Turbo-Plus batch system will be ready for operation upon system reset.

Appendix A

Modifying a Slave Circuit Driver for Turbo-Plus

If you are using slave boards with incorrect circuit drivers, **it is highly recommended that** you patch the slave circuit driver source code (SCD???.ASM) in order to use TWX and RESET commands.

The change to be made occurs at the end of the interrupt service routine in the circuit driver. If the last two lines of your routine are:

```
EI              ; Enable Interrupts
RETI            ; Return
```

replace them with:

```
JMP  ISRXIT#    ; Jump to ISR exit
```

If the last two lines of that routine are different, contact your dealer.

Appendix B

Installing Turbo-Plus on a Multi-circuit System

It is possible that your system has two types of slave boards installed, in which case you will need to add one extra patch to the master .PAR file. This patch is for the Turbo-Plus slave table, and has the label TPSTBL, with the follwing default value:

```
TPSTBL = (0001),(0002),(0003),(0004),(0005),(0006),(0007),(0008)
         (0009),(000A),(000B),(000C),(000D),(000E),(000F),(0010)
```

The sixteen words in this table contain the network addresses of the slaves which Turbo-Plus will refer to as A through P respectively. Thus, under the default configuration, a single circuit with up to sixteen slaves will define those slaves as A-P in the order that they are referred to in your circuit driver's port assignment table.

For a specific example, let us consider a system having three slave circuits, with five slaves on the first, two on the second, and four on the third. In this case, we are only worried about the first eleven entries in the table. We must change the network addresses of slaves F-K, as follows:

```
TPSTBL = (0001),(0002),(0003),(0004),(0005),(0101),
         (0102),(0201),(0202),(0203),(0204)
```

Proposal for the Design of an Application-Independent User Function Interface for TurboDOS Versions 1.2, 1.3, and 1.4

Authors: Jim Gabriel
Microserve, Inc.
276 Fifth Ave. - Suite 1005
New York, NY 10001
(212) 683-2811

Will Poole
Commercial Dynamics, Inc.
91 Sheldon Street
Providence, RI 02906
(401) 274-4038

Following is a proposal for a parameter passing protocol for the TurboDOS user function. This convention will allow for the coexistence of numerous user function packages in a TurboDOS system. At this time it is generally not possible to use more than one user function in a given system configuration. The design of this protocol takes into account all actively used versions of TurboDOS, including V1.40, in which the methods for routing user function requests have been altered.

BACKGROUND

Current Problems

There are currently two primary problems in using multiple user function packages produced by independent authors:

1) TurboDOS provides only one user function entry point (USRFCN), so methods for including multiple functions must be mutually agreed upon by all concerned user function authors.

2) All authors of user functions must choose their own exclusive range of sub-function numbers in order to avoid conflict. To date, this has not been an issue since only one user function may be included in a system generation.

Current Conventions

There are two commonly used user function parameter passing protocols: The first uses the E register as the sub-function number and the HL register pair for other arguments. The second passes sub-functions in the L register and uses the DE pair for word length arguments.

However, TurboDOS version 1.4 uses the DE pair to specify network routing. Therefore, to create user functions compatible with <u>all</u> versions of TurboDOS, only the HL pair may be used for passing parameters to user functions. Any additional parameters must be passed through the DMA transfer area.

DESIGN PROPOSAL

Design Goal

The goal of this design is to create a standard application-independent interface for the TurboDOS user function that will allow multiple user function packages to be included in a system configuration without requiring that the respective authors coordinate their entry point naming and sub-function number usage.

User Function Initialization

A general purpose INIT time entry point for a user function initialization routine is required. While a number of OEM's have incorporated such an entry point into their own hardware initialization routines, our goal is to keep this convention hardware and software independent.

We propose, therefore, to create a user function initialization entry point via the method currently used by the Turbo-Plus software package. This procedure involves patching a jump from the LCLUSR initialization routine to the user function initialization entry point. The patch **LCLNIT + 7 = USRNIT** under versions 1.22 and 1.30, and **LCLNIT + OD = USRNIT** under version 1.40 forces the LCLUSR initialization routine to jump to our initialization routine, instead of CLBNIT, as its last jump. Thus, we may use this convention to support an initialization routine with the entry point USRNIT::. After user function initialization, a jump will be made to CLBNIT#.

Note: Under TurboDOS V1.41, the USRSUP module and a kernel level initialization-time call to USRNIT are being supplied with TurboDOS by Software 2000. Therefore, the preceding paragraph can be ignored for users of V1.41 or later.

The USRNIT initialization routine will support an arbitrary number of user function packages (8 seems reasonable). USRNIT will call USRINA#, USRINB#, USRINC#, etc. If the called USRIN% routine is present, our convention specifies that the routine return the number of subfunctions required in the A register, and an eight byte package name at the address pointed to by the value returned in the HL register pair. Thus, a return code of 0 in the A register will mean that either the user-defined function did not exist (in which case the TurboDOS .UND.:: routine will return a 0) or it was merely a piece of initialization code which did not support any subfunctions. The main initialization routine will store each returned package name and computed function base in an internal table.

User Function Package Entry Points

User function packages conforming to our protocol will use the entry point USRFC%::. One common module, also supplied by the authors of this proposal, will be responsible for dispatching

each included user function package when a call to USRFCN is made.

alling Sequence for Application Programs

Our protocol requires that one change be made to each application program that will call user functions. At the beginning of each program that will call a user function, a call must be made to the user function sub-function 0. The user function package name that the program is to access is passed at the DMA address. The user function 0 call returns with 0 in the A register, indicating that the package is not available, or with a user function sub-function offset. The calling program may, from then on, access the requested user function package by calling USRFCN and adding the returned offset to the package's sub-function number.

When the main USRFCN dispatch routine is called, it will know from the range of the sub-function which user-function package is desired. It will then subtract the appropriate offset from the sub-function and jump to the correct package. In this fashion, all user function packages may be coded with sub-function numbers beginning with 0.

If a program must route user functions to multiple destinations, then an initialization call in the program is necessary for each of those destinations. This procedure is necessary in case the user function is genned in different locations in the various destinations. The sub-function offsets must be saved and re-used whenever calls are made to the corresponding user function packages. Although these initialization calls add a small one time overhead, the added flexibility is well worth the computing time cost.

COMMENTS

Although this protocol is significantly more complicated than the de facto standard of pre-allocating sub-function numbers between authors, a flexible convention must be designated. Therefore it is worth any recoding necessary to guarantee the possibility of user function coexistence in the future.