APPLICATION NOTE

AP-242

October 1985

Additional Printer Support for the NDS-II System

CHRIS FEETHAM
DSO APPLICATIONS ENGINEERING

Order Number: 231478-001

INTRODUCTION

Using printers for hard copy of data has long been necessary in most computer systems. Software engineers use printers primarily for software program listings, but increasingly, letter quality printers are being used to generate memos, reports, and other business documents, rather than queueing them up at the secretary's typewriter. Additionally, with the cost of computer terminals and network connections declining, it is becoming rare for the business professional not to have immediate or direct access to a terminal with some type of word processor available. The ability to send hard copy directly to a printer rather than waiting for a typist to re-type the input is a productive benefit for everyone.

THE NDS-II NETWORK

With Intel's advanced Network Development System II (NDS-II), development systems are connected into a network using Ethernet. Additionally, each development system has the ability to host several ISIS Clusters that use low cost serial lines to support the terminals. The complete product line is described in the NDS-II System description (refer to Appendix D for complete details).

With low cost terminals available to everyone, including engineers, managers, and secretaries, files and data can be shared and manipulated directly on the network, reducing the many intermediate steps required in producing a final document. The addition of CPM/80 coupled with the industry standard Wordstar word processing package, available for the NDS-II system (refer to Appendix D for details), further increases secretarial efficiency.

Engineers, managers and secretaries all benefit from the advanced editors and tools provided with Intel's systems. Getting the output to a printer is the next step in the process, and is the subject of this application note.

GETTING THE DATA PRINTED

Virtually every computer sold today, from the most inexpensive PC to the largest mainframe, has serial and/or parallel ports for connections to printers and other devices. Intel's development systems are no exception, providing hardware ports for both serial and parallel printer types.

Intel's operating systems supplied with the NDS-II network and development mainframes, INDX and ISIS respectively, provide software "devices" which the user can copy files to. The software device designations are :LP: for the parallel line printer, and :TO: for the serial device. However, varying types of serial printers and their associated protocols render the simple "Copy file to :TO:" inadequate. Additionally, printers are somewhat expensive and noisy. The desired method of operation is to provide one or two printers accessible by a group of people, located in a separate room away from the immediate working area.

This application note shows how Intel's NDS-II network, combined with ISIS Clusters and terminals provide a solution for the desired method of operation. The NDS-II's INDX operating system provides a print spooler that allows users to copy files to a central spool printer (:SP:). Files copied from the remote stations (ISIS Clusters , Series-II/III and Series-IV development systems) are then copied to a parallel line printer connected to the NDS-II.

This print spooling feature is not a new concept for computers, and is only one of many excellent features of the NDS-II system. Many users would like to support additional printers on the network, both parallel and serial, but the NDS-II's built in spooler does not provide for this.

SOLUTION–Prince

Prince is a versatile spooling program designed for use with Intel's Series-II, Series-III, and Series-IV development systems, either in standalone or network mode, and for ISIS Clusters operating with an NDS-II network. Using a dedicated ISIS Cluster is perhaps the most effective and efficient method of operation. The ISIS Cluster solution provides for the cheapest and most automatic operation, which is detailed in Appendix C.

HOW IT WORKS

Prince is an ISIS-based program operating in the 8085 environment of the Development System or ISIS Cluster. After extensive initialization, Prince continually polls the directory that is ASSIGNed to :F8:, and any files in this directory are PRINTED, then DELETED. As this is an ISIS based program, files to be printed must conform to the ISIS file name format:a maximum of six characters, plus an optional three character extension, separated by a period.

:F8: can be assigned to a directory created on a Series-IV for standalone operation, or to a directory on the Network Resource Manager. If the Network Resource Manager is used, and the NRM has no parallel printer attached, you may assign :F8: to /(root)/SPOOL, the main print spooler directory. A workstation could then copy directly to :SP: instead of :F8:. This saves each workstation from having to assign :F8: to a specific directory.

Prince has been designed for optimal use of network resources, and provides additional capabilities and flexibility above and beyond the automatic print spooler provided with the NDS-II. Prince also provides useful capabilities for Series-IV system operating in standalone mode.

Other applications might include operation of a parallel printer at a development system host for ISIS Cluster users, or even communication interface that automatically copies files from one system or network to another system connected via a serial or parallel line.

Upon invocation, Prince automatically checks its environment to determine the type of system it is loaded on. Valid systems are Series-II, Series-III, Series-IV, and the ISIS Cluster. Prince then sets up the appropriate serial channel for output, unless output has been directed elsewhere. For the Series-II and Series-III, this is serial channel 1. The Series-IV uses serial channel 2, and the ISIS Cluster uses the on-board serial channel normally used for the console.

Series-IV systems can use serial printers, but the control interface for the serial device, specifically the XON/XOFF (cntl-s / cntl-q) protocol, is currently not provided with a simple copy to the system serial file (designated :TO:). Prince solves this problem by providing the XON/XOFF protocol, and optionally checks for a hardware printer ready signal if desired, by selectively monitoring Data Set Ready (DSR) on the serial line.

The Intel development systems set the serial channel used for the serial device (:TO:) to a specific file transfer rate, better known as baud rate. Prince can selectively output serial data at user specified baud rates of 110, 300, 600, 1200, 2400, 4800, 9600, and 19200. This allows faster devices and devices that can "buffer up" data to take advantage of the full capabilities of the serial line, while the controls mentioned previously (XON/XOFF and DSR) provide the desired control protocol to run the serial devices and the development systems at their fastest rate.

For management tracking and control, Prince keeps a log of all activity, including error messages, initialization defaults, and information about each file printed. File PRINT.LOG is created in the directory assigned to :F9:, and contains relevant information about the files being printed: the file name, time that the file was printed, owner of the file, and the number of bytes actually printed. The log information can be re-directed to another file, including the console, line printer, or disk file. If the log file specified is a disk file, it can be viewed, copied, or deleted at any time. If the log file is deleted, Prince creates the log file again, using the original log file name, at the next file detected for printing.

Prince allows re-direction of the output to a file rather than the printer connected to the serial line. Spooling to a parallel line printer is accomplished by specifying :LP: as the output path. The output re-direction can also go to a disk file, or any other valid ISIS output file name except :TO:. If a disk file is specified for output, it can be viewed, copied, or deleted at any time. Files being copied to the output file are added to the end of the file. For orderly printing, Prince automatically outputs a form feed before printing each file.

This version is initialized for use with a Diablo 630 serial interface and supports the XON/XOFF protocol at 2400 baud. These parameters may be changed by command line controls.

The ISIS.INI, or submit file that invokes this program, must contain a directory assignment to :F8:, for the files to be printed, and also an assignment to :F9: for the log file, unless it has been re-directed.

The default log file name, if none other is specified, is :F9:PRINT.LOG. Any file specified for the optional re-direction of the log file and/or the output path must be a valid ISIS output file name (refer to the NDS-II ISIS III User's Guide #121765-004 for a definition of valid ISIS output filenames).

INVOCATION AND CONTROL OPTIONS

Invocation of Prince is best accommodated in a command file, or SUBMIT file. For Intel systems, use of a user 'init' file is recommended, and essential for automatic use with an ISIS Cluster. User Init files are automatically submitted for execution upon LOGON to the system. This file contains assignments, and the command line that starts Prince.

Control options are all single letter characters, followed immediately by an "=" sign, then the actual option. Controls can be entered in any order, upper or lower case, can be separated by spaces or commas, but must contain no imbedded spaces. If Output is redirected to a file, as opposed to the default serial channel, then DSR and Baudrate controls have no effect, and the serial channel is not initialized.

Control Description and Examples

```
Controls:                    Control Description:
L=logfile              ; Valid ISIS filename - log file re-direction
                       ; :F9:PRINT.LOG is the default
P=output$file          ; Valid ISIS filename - output re-direction
                       ; can be :LP: for the local line printer, etc.
D=T                    ; DSR control. Any character other than 'T' will
                       ; not set the DSR control - pin 6 on the RS-232
                       ; line is monitored for printer ready. No DSR is
                       ; the default.
B=baudrate             ; valid number. Only the first two characters
   110                 ; are checked to determine uniqueness.
   300                 ; Any following characters are ignored.
   600
   1200
   2400
   4800
   9600
   19200
```

Examples:

1. To set log file to console out and output to line printer:

   :F9:PRINCE l=:co: p=:lp:

2. To set baudrate to 9600 and initialize DSR control (defaults to :F9:print.log):

   :F9:PRINCE b=9600 d=t

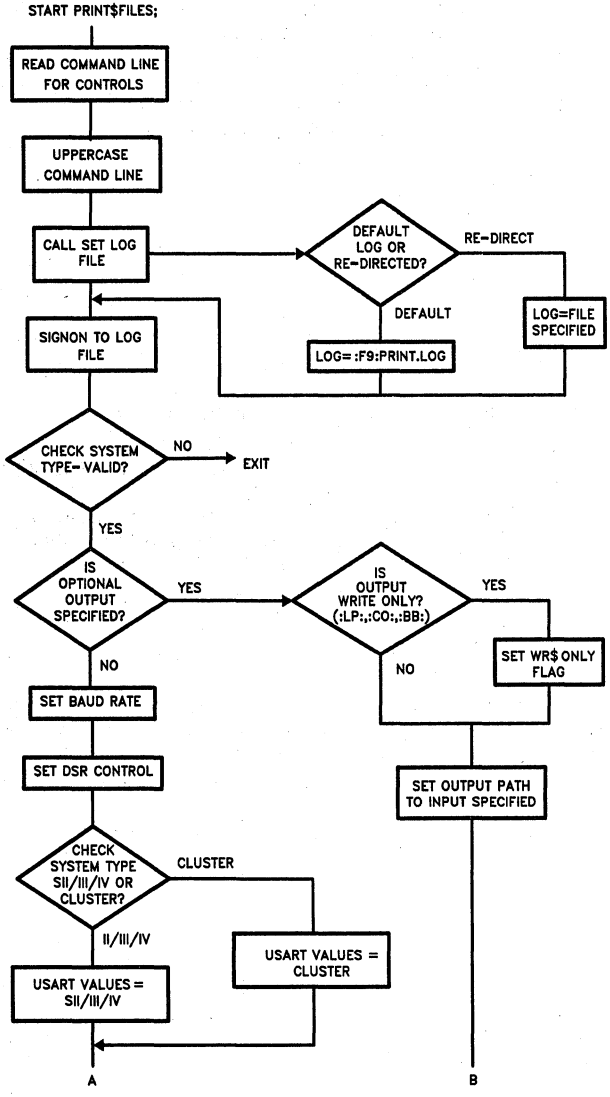
```
Example ISIS.INI:                  ; ISIS.INI file for Series-II/III
                                   ; and ISIS Cluster
ASSIGN 8 to /w/prntspool.dir       ; copy files to be printed to :F8:
ASSIGN 9 to /w/printlog.dir        ; :f9:also contains the program
ISIS                               ; Invoke ISIS-IV - for Series-IV
:F9:PRINCE                         ; Invoke print spooler
```

CONCLUSION

Prince is a versatile utility that enhances the operation of standalone Series-IV systems or NDS-II networks. Prince is available separately from Intel's INSITE Library, (order PRINTS, Insite order code BG61) and is also available along with many other useful tools in Intel's NDS-II Software Tool Box.

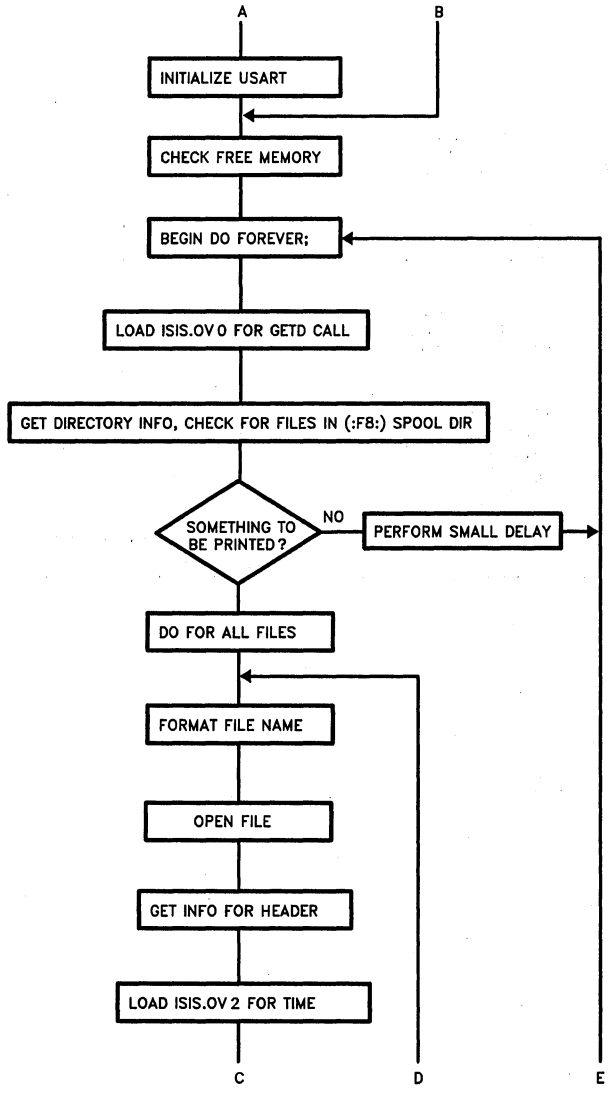
APPENDIX A

PROGRAM FLOW CHART



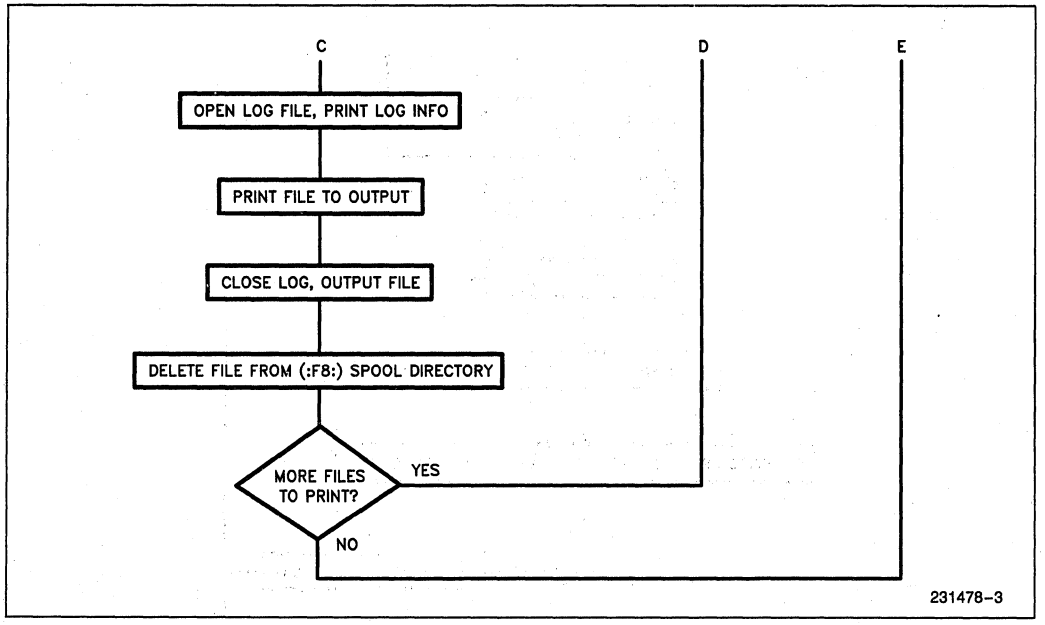
231478-1

PROGRAM FLOW CHART (Continued)



231478-2

PROGRAM FLOW CHART (Continued)



APPENDIX B
PROGRAM LISTING

```
PL/M-80 COMPILER    PRINT FILES              PAGE 1
ISIS-II PL/M-80 V4.0 COMPILATION OF MODULE PRINTFILES
COMPILER INVOKED BY:    :f1:plm80 :F3:prince.p80 PAGEWIDTH(80)
                $TITLE ('PRICE') PAGEWIDTH(80)
  1             Prince: do;
     $nolist include(:f3:procs.p80)
     $list
    /*************************** Program Start ************************/
    /*
     Read input line and set log file. Signon to log file, then determine
     system type. Check for optional baud rate control, and DTR/DSR control.
     Then set up the 8251 USART per the system type.
                */
450  1          call read (1,.input$buffer,128,.in$buffer$actual,.status);
451  1          do i = 0 to in$buffer$actual-1; /* UPPER CASE the input buffer.*/
452  2            input$buffer(i) = set$upper$case(input$buffer(i));
453  2            end;
454  1          call set$log$file;             /* Set up the log file. */
455  1          call print$message(0);         /* Signon to log file. */
456  1          if (high$byte<1) or (high$byte>5) then do;/* Exit if invalid */
458  2             call print$message(11);                 /* system type. */
459  2             end;
                /* Check if Line Printer specified in command.*/
460  1          call set$device;
461  1          if lp$flag <> true then do;/* If not line printer, set USART. */
463  2               call set$baud;         /* Set baud rate. */
464  2               call set$dsr;          /* Check for DTR/DSR control. */
465  2               if system$is$SII or system$is$SIV then do;
467  3                    serial$output=.s$serial$output; /* Use SeriesII/IV */
468  3                    wait$for$printer=.s$wait$for$printer; /* serial chn. */
469  3                    end;
470  2               call initialize$usart;
471  2               end;
                /* How much free memory below the Overlay base? */
472  1          limit = 0E87FH - .memory;
473  1          do forever;
                /* Any files to be printed ? */
```

```
474  2              call load(.('ISIS.0V0 '), 0, 0, .entry, .status);
475  2              call check$status(1);
476  2              start = 0;
477  2              call getd(8, .start, 1000, .actual, .dir$dump, .status);
478  2              call check$status(2);
479  2              if actual <> 0 then do index = 0 to actual - 1;
                  /* Something to be printed */
                  /* Format the filename */
481  3                      j = 4;
482  3                      do i = 0 to 5;
483  4                           if dir$dump(index).filename(i) <> 0 then do;
485  5                                filename(j) = dir$dump(index).filename(i);
486  5                                j = j + 1;
487  5                                end;
488  4                           end;
489  3                      if dir$dump(index).filename(6) <> 0 then do;
491  4                           filename(j) = '.';
492  4                           j = j + 1;
493  4                           do i = 6 to 8;
494  5                                if dir$dump(index).filename(i) <> 0 then do;
496  6                                     filename(j) =dir$dump(index).filename(i);
497  6                                     j = j + 1;
498  6                                     end;
499  5                                end;
500  4                           end;
501  3                      filename(j) = ' ';
                  /* Filename formatted, get the file */
502  3              do while status <> e$file$open;
503  4                   call open(.aftn, .filename, read$only, no$line$edit,
.status);
504  4                   end;
                  /* Get information for the header */
505  3                   file$table.aftn = aftn;
506  3                   call spath(.file$name, .file$table.device$number, .status);
507  3                   call check$status(4);
508  3                   call load(.('ISIS.0V1 '), 0, 0, .entry, .status);
509  3                   call check$status(3);
510  3                   call filinf(.file$table, 1, .file$info, .status);
511  3                   call check$status(6);
                  /* Load ISIS.0V2 to get the TIME! */
512  3                   call load(.('ISIS.0V2 '), 0, 0, .entry, .status);
513  3                   call check$status(5);
PL/M-80 COMPILER          PRINT FILES          PAGE 3
                  $eject
/* Print the header - form feed to printer, header to log file or :co:*/
514  3              call print(.(FF), 1);
515  3              call open$file$safely (.aftnl,.logfile,wr$only$log);
516  3              call write(aftnl,.header$1, length(header$1),.status);
517  3              call write(aftnl,.filename(4), (j-4),.status);
518  3              call write(aftnl,.header$2, length(header$2),.status);
519  3              call move(4, .zero$time, .dt.system$time);
520  3              call de$time(.dt.system$time, .status);
```

```
521  3          call write(aftnl,.dt.time(0), 8,.status);
522  3          call write(aftnl,.(' on '), 4,.status);
523  3          call write(aftnl,.dt.date(0), 8,.status);
524  3          call write(aftnl,.header$3, length(header$3),.status);
525  3          call write(aftnl,.fileinfo.owner(1), fileinfo.owner(0),.status);
526  3          call write(aftnl, .(cr,lf),2,.status);
527  3          call close (aftnl,.status);
                /* Print the file */
528  3          file$bytes = 1;
529  3          do while file$bytes <> 0;
530  4               call read(aftn, .memory, limit, .file$bytes, .status);
531  4               call check$status(8);
532  4               if memory(0) = FF then memory(0) = 0;
534  4               call print(.memory, file$bytes);
535  4               end;
                /* File has been printed */
536  3               call close(aftn, .status);
537  3               call check$status(9);
538  3               call open$file$safely (.aftnl,.logfile,wr$only$log);
539  3               call write(aftnl,.header$4,length(header$4),.status);
540  3               call print$size (file$info.len$hi,file$info.len$lo);
541  3               call write(aftnl, .(cr,lf),2,.status);
542  3               call close(aftnl, .status);
543  3               call delete(.filename .status);
544  3               call check$status(10);
545  3               end; /* Look for next file */
                /* No files to be printed , Wait a minute or so */
546  2          else do i = 0 to 60;
547  3               do j = 0 to 500;
548  4                    call time(10);
549  4                    end;
550  3               end;
551  2          end; /* of Do forever */
552  1     end Prince;
```

APPENDIX C
ISIS CLUSTER BOARD PREPARATION

Used with an ISIS Cluster, Prince can be driven from the ISIS Cluster board's serial channel, which is normally used for a terminal. With the addition of the special SERVER PROM for the ISIS Cluster, the Prince program can be automatically invoked and begin copying files from a network spool directory to a serial printer or other serial device, immediatley upon power-up. In this mode of operation, there is no console connected to the ISIS Cluster. Instead, the serial printer or other serial device is connected to the console port, and the SERVER PROM installed on the Cluster board automatically logs the Cluster board onto the NDS-II network, then submits the ISIS.INI command file. This command file contains the necessary assignments, as well as the Prince program invocation.

To prepare the NDS-II to support the Prince spooler, a username and home directory for the Prince program and SERVER prom must exist. To provide trouble free spooling, the username for the SERVER prom should be declared as a Superuser. This way, file access rights need not be set each time a file must be spooled, printed, and deleted.

The SERVER PROM image is included with the Prince program. The SERVER PROM image can be modified to change the username or password. Bytes 0FF0 to 0FFC (inclusive) are reserved for the SERVER username, password, and string terminator (00H). (Note that these are PROM addresses - this PROM image is moved to a different location in RAM on initialization.)

byte 0FFDH: PROM checksum

byte 0FFEH: resreved

byte 0FFFH: system ID (05 for Cluster - DO NOT CHANGE!)

The following strings are stored in the PROM image:

username: SERVER0<CR>

password: ©

checksum: 082H

FF0																
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
53	45	52	56	45	52	30	0D	0D	00	spare			82	00	05	HEX
S	E	R	V	E	R	0	CR	CR								ASCII

If you change the LOGON name and/or password, remember to change the checksum, which is stored in byte 0FFDH. NOTE:The checksum is actually the two's compliment of the checksum calculated by the boot code. Thus, if you change the username to SERVER2 from SERVER1 (increment byte 0FF6H), you must decrement byte 0FFDH. Changing the PROM image can easily be accomplished using Intel's IPPS software, which is supplied with the iUP-200 PROM programmer.

CLUSTER BOARD PREPARATION - PROM BURNING

1. The PROM image is written in 286 format. Remember to initialize the iPPS properly.
2. Read in PROM image, modify LOGON strings, modify checksum, and burn a 2732 or 2732A.
3. Remove the old monitor prom from the Cluster board (A25) and place in the next-door socket (A37) for safe keeping (may be needed by CE).
4. Install new SERVER prom in A25.
5. Install a jumper between pins 67 and 68. This ties Clear-to-Send/ Request-to-Send together on-board. Prince uses XON/XOFF or XON/XOFF and DSR for control, so CTS/RTS is not required between devices.
6. If the printer to be used operates with the hardware DTR/DSR protocol, configure the serial cable such that the printer ready line comes in on pin 6 (DSR) of the serial cable to the Cluster board.
7. Refer to the ISIS Cluster installation manual for further Cluster installation instructions.
8. Set up ISIS.INI file to make assignments and invoke PRINCE, plug it all in and go.

APPENDIX D RELATED PUBLICATIONS

1. ISIS Users Guide 9800306
2. Network Development System II iMDX 450 210937-004
3. CP/M-80 on the NDS-II – Application NoteAP 253
4. ISIS Cluster Installation instructions .....

APPENDIX E ERROR MESSAGES

Prince returns messages and error conditions if certain external conditions prevent normal functioning of the spooler. All messages are directed to the log file, unless a fatal ISIS error occurs, preventing Prince from handling the error. ISIS will trap the fatal error and re-boot itself. Some error conditions that are not fatal ISIS errors are considered fatal by Prince, and after logging the error message in the log file, Prince will exit. The 18 messages given by Prince are as follows:

1. Serial printer driver xxx'

   Non-fatal message - The normal sign-on message at Prince invocation.

2. 'ISIS.OV0 not present on system disk'

   Fatal error, Prince will exit. ISIS overlay 0 must be present on :F0:for Prince to function properly.

3. 'GETD system call failed'

   Non-fatal - Prince uses this system call to determine the presence of files to be printed in directory :F8:. Possible causes for failure:a damaged or incorrect ISIS.OV0, file access rights, etc.

4. 'ISIS.OV1 not present on system disk'

   Non-fatal - Prince must load ISIS.OV1 to support the file$info system call. ISIS.OV1 is not on :F0:, or access rights are insufficient.

5. 'SPATH system call failed'

   Non-fatal - SPATH returns information about the file to be printed for log file status of the file.

6. 'ISIS.OV2 not present on system disk'

   Non-fatal - ISIS.OV2 is used to provide time and date information that is placed in the log file for all files printed, and all messages.

7. 'FILINF system call failed'

   Non-fatal - The file$info call returns file information to be used in the log file, such as the owner of the file, etc. If this call fails, meaningful file information will be absent from the log.

8. 'Could not open the file to be printed'

   Non-fatal - Most common causes of this malfunction are insufficient access rights to the file, or an invalid ISIS file name. Rename the file name, or give access rights to the Prince user.

9. 'Could not read the print file'

   Non-fatal - This error will occur only during the printing of the file, before the print has completed, but after the first successful open.

10. 'Could not close the print file'

    Non-fatal - Prince could not close the file just printed.

11. 'Could not delete the print file'

    Non-fatal - Prince attempts to delete the print file after printing. Most common cause of this error is insufficient (delete) access rights. Give Delete Access rights to the Prince user.

12. 'Unknown System Type - not supported'

    Fatal error, Prince will exit. Printfiles checks byte 0FFFFH to discern system type. Valid types are: 01 = Series-II 02 = Series-IV 05 = ISIS Cluster

13. 'BAUD control defaulted to 2400 baud'

    Non-fatal message - An attempt was made to set a different baud rate per the baud rate control (B=number) and number was invalid. Valid numbers are:110, 300, 600, 1200, 2400, 4800, 9600, 19200.

14. 'LOG control defaulted to :F9:print.log'

    Non-fatal message - An attempt was made to redirect the log file, but the log file name was greater than 14 characters. Prince does a gross check on the pathname specified to assure a correct ISIS file name.

15. 'DTR/DSR control activated'

    Non-fatal message - The DSR control is active.

16. 'DTR/DSR control not activated'

    Non-fatal message - An attempt was made to set DSR other than true - DSR not true is the default.

17. 'OUTPUT file defaulted to :F9:print.out'

    Non-fatal message - An attempt was made to re-direct the output file, but the file name was greater than 14 characters. Prince does a gross check on the pathname specified to assure a valid ISIS file name.

18. 'Could not write OUTPUT file'

    Fatal Error, Prince will exit. Prince could not write to the output file specified, so spooling is suspended.