Archimedes TDS Server
User Guide

Gnome Computers Ltd
16 Histon Road
Cambridge
CB4 3LE
England

Tel. (0223) 461520

Issue 1

Introduction

The Transputer Development System (TDS) is a program development environment for writing Occam programs for the Inmos Transputer. The TDS runs on a transputer system with some RAM memory and uses one of the transputer's links to communicate with the outside world via a host computer. The host computer provides terminal input/output and filing system facilities for the TDS.

The host computer must be equipped with a peripheral device to allow it to communicate with the transputer using the Inmos link protocol. Such a device is termed a link adapter. In addition, the host must be running a program which can communicate with the TDS over the link and make the facilities of the host available to the TDS. This program is known as a server. The server program which supports the TDS is known as the TDS Server and there are other types of server to support different development environments.

This document describes the TDS Server program for the Acorn Archimedes workstation. This is designed to run with a range of transputer expansion boards for the Archimedes which have been designed by Gnome Computers. The TDS system which this environment supports is the same as that supplied for the IBM PC and compatibles. If you already have this system, it can be transferred to the Archimedes with the Kermit file transfer utility. Otherwise, it is available from Gnome Computers on Archimedes format discs.

The system will run on an Archimedes with 512 kilobytes of memory and a hard disc with at least 3 megabytes of free space. The current system has been developed and tested under release 1.2 of the Arthur operating system. It will also run under the RISC OS operating system but is currently unable to make use of the multi-tasking features of this environment.

Running the TDS

In order to run the TDS, the link adapter must be connected to a transputer system with (ideally) 2 megabytes of memory. Development of small programs is possible with a 1 megabyte system. You should consult the user guide for your transputer expansion card for details of how to connect to a suitable transputer system. The TDS expects to talk via link 0 of the transputer system and the link adapter must also have control of the transputer's reset, analyse and error signals.

The TDS is best run from within a directory specifically created to hold occam programs. If you have done the installation (see appendix I) and are just starting to use the system, the next thing to do is to create such a directory. For example:

```
cdir $.occam
dir $.occam
```

Within this directory, various other directories will be created by the server as the TDS runs. The directory `top`contains top level files, the directory `tsr`contains Occam source, etc. One directory which will normally have to be created manually is the toolkit directory (`tkt`) which contains toolkit files. To create this directory and copy the default toolkit file into it type the following:

```
cdir tkt
copy $.tds2.system.tkt.toplevel tkt.toplevel ~cq
```

Finally, the server can be started as follows:

```
server14
```

This starts the server which will boot the transputer and enter the TDS. If a new top level file is required this should be introduced on the command line as follows.

```
server14 -t newtop.top
```

Differences between TDS on PC and Archimedes

The major difference between the PC and Archimedes implementations of the TDS occurs in the handling of filenames. In the vast majority of situations, the differences are hidden from the user. For the odd occasion where a problem arises the details of filename mapping are given in Appendix II.

A very useful feature of the Archimedes implementation of the server is that the mouse is may be used to assist in moving around the folding editor. The

mouse cursor is an arrow which moves around the screen as the mouse is moved. This has no effect on the editor's text cursor until a mouse button is pressed. When this occurs, the text cursor is moved to join up with the mouse cursor. If the left mouse button was pressed, an ENTER FOLD keystroke is now generated. If the right button was pressed, an EXIT FOLD keystroke is generated. If the centre button was pressed no keystroke is generated unless the text cursor is on the first or last line of the display in which case either a PAGE DOWN or a PAGE UP keystroke is generated.

A further difference is that the Archimedes screen has 32 rows as opposed to the 25 of a PC.

At present neither the TDS Suspend facility nor the use of the RS423 port are supported. It is intended that future releases of the server software will correct this deficiency.

The server can be interrupted at any time by typing CTRL/C. This will normally result in a message prompting to either reboot the transputer or leave the server. If the transputer has crashed it is possible that no message will appear. In this case, typing CTRL/C again will shut down the server to allow the system to be restarted.

The server requires that the floating point emulator program for the Archimedes be installed. This is normally provided with the machine but an additional copy is provided on the distribution disc.

Default Command Line Parameters File

When the server program starts up it gets its command line arguments from two places. It first reads them from the first line of the argument file `$.tds2.args`. This file need not exist but in practice its presence is normally required. Further arguments are then read from the command line which invoked the server. The argument file may conveniently contain arguments which are normally given while additional or overriding arguments may be given on the command line. The content of the argument file as supplied with the system is as follows:

```
-b $.tds2.system.tdsload.b4 -s #200000 -f $.tds2.system.tds2.xsc
```

Access to Graphics Facilities

It is possible to access the Archimedes VDU drivers directly in order to use the graphics operations, change the screen mode, etc. This is done by placing the keyboard in raw mode by sending a TT_KEY_RAW operation. In this mode all characters passed to the TT_OUT_STRING operation go directly to the Archimedes VDU driver. This state of affairs is reversed by placing the keyboard back in cooked mode with the TT_KEY_COOKED operation. A sample program which accesses the Archimedes graphics facilities is listed in Appendix III.

Link Adapter Key File

Each link adapter board supplied by Gnome Computers has a unique indentification code built in to it. In order for the TDS to run with a particular link adapter board, a matching software key must be present in a file known as the key file and normally filed as :4.tds2.keys. The key is a 32 bit integer which must be specified as a hexadecimal number in the key file. This file should contain only hex numbers, spaces and newlines. It may contain several numbers allowing the TDS Server to operate with any one of a number of link adapter boards installed in the machine.

The allocation of keys is made by Gnome Computers and one is normally supplied with each copy of the TDS Server.

Appendix I - Installing the TDS from a PC using Kermit

Installing the TDS system on an Archimedes using Kermit is currently done in two stages. The first stage involves installing the Archimedes specific files which are supplied on a single 3.5" floppy disc. The second stage involves copying the Inmos supplied files. Because we do not currently have a licence to duplicate the Inmos software, it must be purchased separately and transferred to the Archimedes. Until the system can be supplied on 3.5" discs, the best method is to install the Inmos software on a PC and then copy the system using a file transfer utility such as Kermit. This is the approach that is currently adopted and the Archimedes Server disc contains a version of Kermit (TDSKermit) to do the transfers.

Installing the Archimedes TDS Server

Place the Archimedes TDS Server disc in the drive and type the following to the Archimedes command prompt:

```
dir :0
install
```

This will create the necessary directory structure on the Archimedes and copy some files to the hard disc. The following files are created:

```
:4.library.server14   - the server program
:4.tds2.args          - default arguments for the server
:4.tds2.keys          - link adapter identification keys
```

The following directories are created on the hard disc:

```
:4.tds2.system.b4
:4.tds2.system.cex
:4.tds2.system.cut
:4.tds2.system.xsc
:4.tds2.system.tkt
```

```
:4.tds2.complibs.tsr
:4.tds2.complibs.top
:4.tds2.complibs.tkt

:4.tds2.tools.cex
:4.tds2.tools.top
:4.tds2.tools.tkt

:4.tds2.tutor.tsr
:4.tds2.tutor.top
:4.tds2.tutor.tkt

:4.tds2.examples.tsr
:4.tds2.examples.top
:4.tds2.examples.tkt

:4.tds2.mathlibs.tsr
:4.tds2.mathlibs.top
:4.tds2.mathlibs.tkt

:4.tds2.mathlibs.ht2.tsr
:4.tds2.mathlibs.ht2.top
:4.tds2.mathlibs.ht2.tkt

:4.tds2.mathlibs.rt2.tsr
:4.tds2.mathlibs.rt2.top
:4.tds2.mathlibs.rt2.tkt

:4.tds2.mathlibs.ht4.tsr
:4.tds2.mathlibs.ht4.top
:4.tds2.mathlibs.ht4.tkt

:4.tds2.mathlibs.rt4.tsr
:4.tds2.mathlibs.rt4.top
:4.tds2.mathlibs.rt4.tkt

:4.tds2.mathlibs.ht8.tsr
:4.tds2.mathlibs.ht8.top
:4.tds2.mathlibs.ht8.tkt

:4.tds2.mathlibs.rt8.tsr
:4.tds2.mathlibs.rt8.top
:4.tds2.mathlibs.rt8.tkt
```



```
:4.tds2.iolibs.tsr
:4.tds2.iolibs.top
:4.tds2.iolibs.tkt

:4.tds2.iolibs.ht2.tsr
:4.tds2.iolibs.ht2.top
:4.tds2.iolibs.ht2.tkt

:4.tds2.iolibs.ht4.tsr
:4.tds2.iolibs.ht4.top
:4.tds2.iolibs.ht4.tkt

:4.tds2.iolibs.ht8.tsr
:4.tds2.iolibs.ht8.top
:4.tds2.iolibs.ht8.tkt
```

Installing the TDS

The TDS software is installed by using TDSKermit to transfer files from a PC to the Archimedes. This release of the server is designed to work with the D700D release of the TDS and the file transfer scripts expect it to be installed in the conventional directories on the PC which are:

```
c:\tds2\system
c:\tds2\complibs
c:\tds2\tools
c:\tds2\tutor
c:\tds2\examples
c:\tds2\mathlibs
c:\tds2\iolibs
```

On the Archimedes, the TDS is kept in the directory :4.tds2 in the subdirectories listed in the previous section. These directories should have been created before attempting to install the TDS.

Installation consists of transferring the TDS from a PC using Kermit. To do this the PC and Archimedes serial ports must be connected and both machines set up with the same serial parameters. A suitable setup would be 9600 baud, 8 data bits, no parity, one stop bit. The suggested data format for serial

transmission can be set up on the Archimedes by typing

```
configure baud 7
configure data 5
```

at the command line and then rebooting the machine. For the PC, type the following at the command line.

```
mode com1:9600,n,8,1
```

Start Kermit running on the PC and place it in server mode. If the Archimedes is running Arthur 1.20, it is necessary to run a program to correct a bug in the serial line driver. This is supplied on the floppy as `rs423drive` and should be run before starting TDSKermit. The following commands should be typed on the Archimedes command line.

```
dir :0
rs423drive
tdskermit
```

and to the TDSKermit prompt type:

```
rem cwd c:\
take takesys
```

The first command will prompt for a password to which just RETURN need be typed. Note that the '\' character needs to be typed twice as it is an escape character. The file transfer will now begin and transfer the following directories:

```
system
complibs
tools
mathlibs
iolibs
```

If you also want the examples and tutor directories, you should give the following command to TDSKermit:


```
take takextra
```

When all the required files have been transferred the installation is complete and the TDS can be run on the Archimedes. The system as installed with the above commands occupies about 2.5 Mbytes. The files obtained via TDSKermit take about 1 hour to transfer at 9600 baud.

Appendix II - File name transformations

Because the version of the TDS used on the Archimedes is PC based, various transformations have to be applied to filenames to make them compatible with the Archimedes filing system. In practice, these will generally be transparent to the user but are described below for when something goes wrong! The server has a debugging mode which prints filenames when a file is opened.

In general, it is best to make all file name references either to the current directory or to the TDS system directory. If this is done, programs should be readily portable between PC and Archimedes. If explicit references to Archimedes directories are made the program will have to be modified to run on a PC.

The problems are as follows:

1) The Archimedes allows a smaller number of entries per directory than MSDOS

2) The Archimedes has no concept of a filename extension (eg the '.BAT' in 'AUTOEXEC.BAT')

3) The Archimedes directory separator is '.' whereas that of MSDOS is '\'

The Archimedes server performs certain transformations on filenames in order to get around these problems.

The first transformation inspects the beginning of the filename. If it filename begins with `C:`or `C:\`, this is replaced by `:4.`. If the filename begins with `A:\`or `A:`, this is replaced by `:0.`. If the filename begins with `\`, this is prefixed by `$`.

The next transformation replaces all instances of '\' with '.'.

```
dir\file.ext           ->  dir.file.ext
\tds2\iolibs\file.tsr  ->  $.tds2.iolibs.file.tsr
toplevel.top           ->  toplevel.top
```

The third transformation involves the extension. If the extension is a 'recognised' extension, the extension and preceding filename component are swapped. Thus:


```
dir.file.ext           ->  dir.file.ext        !! NB
$.tds2.iolibs.file.tsr ->  $.tds2.iolibs.tsr.file
toplevel.top           ->  top.toplevel
```

Recognised extensions are as follows:

```
tcm tsr dcd cs1 dds ddb cp1 cu1 ce1
dlk csc cpr cut cex dck dbn dmp tci tai
top b4 tsf xsc tkt
```

When a file is being written on the Archimedes, the server ensures that all necessary directories are created before the file is opened for writing. When such a file is deleted, any directories which were created when it was first written remain intact.

The modified version of Kermit supplied with the system does the swapping of filename and extension components automatically when files are transferred to the Archimedes. It can be used to transfer other Occam programs to the Archimedes. For example, to move a program 'primes' from the PC, the following TDSKermit commands would do the job:

```
set file type binary
rem cwd \occam
cwd $.occam
get primes.*
```

This would take files 'primes.top', 'primes.tsr', etc. and store them on the Archimedes as '$.occam.top.primes', '$.occam.tsr.primes', etc. Note that the directories 'top', 'tsr', etc. must already exist.

Server debug mode

The debugging mode causes the server to print two filenames and the access mode whenever a file is opened. The first is the name the transputer system is attempting to access, the second is the transformed name that the server generates. The access mode will be r, w, or u for read, write or update. The success or failure of the open operation is also printed.

The server is placed into this mode by an extra parameter on the command line. This is introduced with the -d key which is accompanied by a numeric argument. 0 generates no debug information, 1 prints filenames and 2 prints filenames and then waits for a key press before continuing. Thus a command line might look like:

```
server14 -d 1
```


Appendix III - Simple graphics demonstration

```
#USE userhdr

INT i, char:

PROC plot (VAL INT k, x, y)          – emulate BASIC plot
  [6] BYTE buf:
  SEQ
    buf [0] := BYTE 25
    buf [1] := BYTE (k /\ 255)
    buf [2] := BYTE (x /\ 255)
    buf [3] := BYTE ((x >> 8) /\ 255)
    buf [4] := BYTE (y /\ 255)
    buf [5] := BYTE ((y >> 8) /\ 255)
    screen ! tt.out.string
    screen ! 6
    screen ! buf
:

PROC mode (VAL INT mode)             – sets screen mode
  [2] BYTE buf:
  SEQ
    buf [0] := BYTE 22
    buf [1] := BYTE (mode /\ 255)
    screen ! tt.out.string
    screen ! 2
    screen ! buf
:

SEQ
  screen ! tt.key.raw                – set keyboard raw mode
  keyboard ? char                    – get response byte

  mode (0)                           – set screen mode 0

  SEQ i = 0 FOR 20                   – draw some ellipses
```

```
    INT j:
    SEQ
      j := i * 20
      plot (4, 640, 512)
      plot (4, 215 + j, 512)
      plot (#C5, 640, 512 + j)

  screen ! tt.key.cooked            – set keyboard cooked
  keyboard ? char                   – get response byte

  keyboard ? char                   – wait for a key
```