Introduction to the T3E
Draft version

S. Booth

September 2, 1997

1 Introduction

This document is intended for new users of the Cray T3E-900 at EPCC. The assumption for the present is that you are already a user of the T3D, and that your need is for a brief description of the differences between the two systems. We shall follow the structure of the *Introduction to the Cray T3D at EPCC*, which you can find at this URL:

`http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro.html`

This should be regarded as a draft version of this document. Certain decisions regarding the configuration of the machine and the organisation of the batch queues still remain to be finalised at the date of writing. A new version will be available shortly, and the latest version of this document will always be available from this URL:

`http://www.epcc.ed.ac.uk/t3d/documents/t3e-intro.html`

1.1 The Cray T3E

The T3E-900 is a member of the new generation of MPP systems from Cray Research. Though the operating system and many of the underlying details are radically different the user environment will seem very familiar to users of the T3D and J90.

One of the main differences fromthe T3D is that the T3E has no separate host, as the T3D has in the Y-MP. The T3E handles its own IO, and all operations from large parallel simulations to editors and compilers run on the T3E processors themselves.

The T3E processors are divided into two groups:

- **Command processors.** These are organised as a conventional multi-processor Unix machine like the J90. All single-proccessor programs including the compilers, editors and your login shell run on command processors. Each command processor can support multiple processes by time-sharing in the conventional manner. Processes may also migrate between processors during their life. When you log into the T3E you are running on the command processors.
- **Application processors.** The remaining processors are reserved for running parallel application codes. A multiple PE program is called an *apteam*. See `man 7 apteam` for more details. At any time arbitrary groups of processors may be allocated to interactive or batch use or even dedicated to particular groups or acids.

Further Information

- *Introduction to the Cray T3D at EPCC*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro.html`
- Manual page on apteams: `man 7 apteam`

2 Getting started

Your account on the T3E will have the same home directory and password as your J90 account. The T3E is currently called `sn6710.ed.ac.uk`. Log into this machine just as you would `kelvin` or `darwin`. As before you should only change your password or default shell on `kelvin`.

The operating system on the T3E is `UNICOS-MK`. This has been developed from the `UNICOS` operating system used on Cray PVP systems such as the J90, and most commands work in the same way.

Note that programs compiled for the J90 or Y-MP will not run on the T3E, so if you keep any compiled utility programs in your home directory you should make two directories (one for the T3E one for the J90/Y-MP) and add something like the following to your `.profile`.

```
TYPE='uname -m'
if [ "$TYPE" = "CRAY T3E" ]
then
  PATH=$PATH:$HOME/bin/t3e
else
  PATH=$PATH:$HOME/bin/pvp
fi
```

2.1 Finding out about the system

All processes running on the T3E (including multi-processor applications) can be viewed using the `ps` command. If you only wish to view programs running on the application PEs you should use:

```
sn6710$ ps -flaMe
```

or

```
sn6710$ mppusers
```

Note that the T3E supports several different user communities, and some of them may have some processors dedicated to their use, so processors which are currently unused are not necessarily available to you. See the systems web-pages to see the current operating schedule.

Further Information

- "Getting started" in the *Introduction*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro/node7.html`
- Systems page on the T3E: `http://www.epcc.ed.ac.uk/systems/t3e.html`

3 Filespace and file structure

Filespace on the T3E is organised along the same lines as on the T3D and the J90.

- The home directories which you will use are physically located on the J90, even though you login to the T3E itself. The `/arch` area for communicating with the mass store is also still on the J90.
- Workspace directories on the T3E can be accessed via `/ework/`*project*; access to these is by arrangement with your PI. Note that `/work` directories used by T3D programs continue to live on the Y-MP.

  The workspace on the T3E is *not* backed up in any way. In the near future this space will be re-configured as a RAID disk to improve data security, but at present you should ensure that valuable data is copied back to `/arch`.

As before data can be efficiently copied between the Cray systems using the `xcp` script.

Further Information

- "Filespace and file structure" in the *Introduction*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro/node16.html`

4 Compiling

The T3E compilers are only available on the T3E. You cannot cross compile on the J90 for the T3E or on the T3E for the J90. The `TARGET` environment variable will be automatically set to `cray-t3e` when you log-in. Make sure that you do not re-set this in your own `.profile` file.

Two languages are supported:

- Fortran 90 (`f90`)
- C++ (`CC`)

Unless you are an advanced user these compilers are essentially identical to the equivalent T3D compilers.

Specifying the number of processors

As with the T3D you can specify a number of processors at compile time using the `-X` flag. Alternatively, you can use an option to the `mpprun` command to specify the number of processors at run time. Unlike the T3D, there is no disadvantage to doing it this way.

```
mpprun -n 32 my_program
```

If you do not specify the number of processors at compile time the resulting program will run on one processor by default.

The hardware does not restrict the number of processors to powers of 2, as on the T3D; `mpprun` can start any number of PEs. However, awkward numbers of PEs make it impossible to run an efficient batch service, so we are restricting the number of PEs in batch jobs to 16, 32 and 64. Interactive jobs may use any number of PEs up to eight.

The same compilers are used for command PEs and application PEs. The only difference between the two is the number of processors you use; all single-PE jobs are run on a command processor.

Linking

The T3E uses a linker called `cld`, as opposed to `mppldr` on the T3D. As a result, some of the linker options may be slightly different. Read the manual page for `cld` for more information.

All of the inter-processor communication libraries `PVM`, `MPI` and `shmem` are available in the `mpt` module. You must have this module loaded before you can compile against these libraries. By default it is loaded for you. If necessary you can also load it by hand:

```
sn6710$ module load mpt
```

4.1 F90 Compiler flags and directives.

Use of the T3D Fortran 90 compiler is described in section 10 of *Tools and Techniques for the Cray T3D*. Tough the T3E's F90 compiler is essentially the same, there are a number of additional features available. See *CF90 Commands and Directives Reference Manual, SR-3901 3.0* for more details.

Fortran 90 compiler flags

The following F90 compiler flags may be useful:

- `-a pad` Pad static storage to try and reduce cache conflicts. This will not help with conflicts between 2 parts of the same array. *Not available on T3D*.
- `-e 0` Initialise all stack variables to zero. *Not available on T3D*.
- `-e A` Enable Apprentice instrumentation.
- `-e i` Generate a run-time error if uninitialised variables are used. *Not available on T3D*.
- `-e I` All variables must be explicitly typed. *Not available on T3D*.
- `-d p` Disable double precision.
- `-e v` Implicit `SAVE` statement on all variables.
- `-f free/fixed` Force free or fixed source form.
- `-g/-G 0` Full debugging enabled. Optimisiation disabled.
- `-G 1` Partial debugging and partial optimisation.
- `-O opt, opt` Optimisation levels. Check the manual page for more details. Note that though the T3E does not have vector hardware it can make use of vector optimisation techniques and some vector compiler directives. *There are more optimisation options on the T3E than the T3D.*
- `-p module-file` Set search location for compiled modules. All modules must be compiled before any source file that uses them. In most cases the compiler will be able to locate compiled modules automatically but it is quicker and more reliable to use the `-p` flag.
- `-R flags` Run time argument and array shape checking. This will slow down code execution but can be sueful for locating programming errors. (see manual page). *Not available on T3D*.

Compiler directives

The following compiler directives may be useful. They should be preceded on the line by the special comment marker, `!DIR$`

- `SYMMETRIC` *variable list*

  This forces stack based variables to be allocated at the same address on all processors for use in `shmem` calls. Unfortunatly this directive does not work with variables allocated via the F90 `ALLOCATE` statement. *This Directive is not available on the T3D.*

- IVDEP

  Marks the next loop as having no vector dependencies. This allows extra optimisation such as the use of vector intrinsic functions. *This Directive is not available on the T3D.*

- NOUNROLL/UNROLL

  Request loop unrolling on the next loop. (You must also specify `-O unroll1` or higher on the command line).

- NOINLINE/INLINE

  Request code in-lining. (You must also specify `-O inline1` or higher on the command line). *This Directive is not available on the T3D.*

- INLINEALWAYS/INLINENEVER *routine*

  Request/suppress inlining for a particular subroutine. (You must also specify `-O inline1` or higher on the command line). *This Directive is not available on the T3D.*

- CACHE_BYPASS *array*

  Access *array* via the communication hardware rather than the cache. This can be faster if there is no data re-use.

Further Information

- CF90 section of *Tools and Techniques*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-tandt/node13.html`
- CF90 Commands and Directives Reference Manual:
  `http://cdochost.epcc.ed.ac.uk/library/all/3901_3.0`
- The linker: `man cld`

5 Running

Normally T3E applications just need to be run in the standard way, by typing the name of the executable. However, the `mpprun` command can be used if you need to add run-time options.

For example, you can specify the number of PEs to use. If you don't do this, the program will use the number of PEs you specified at compilation time (1, by default).

`mpprun -n` *number executable*

When a program fails, a `core` file is created, not an `mppcore` as on the T3D, and by default it contains information about only one PE. You can make it contain information about all the PEs by using this `mpprun` option:

`mpprun -c` *executable*

There is no `mppexec` command on the T3E.

Further Information

- "Running programs" in the *Introduction*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro/node31.html`

6 ACIDs and Accounting

As with the T3D service, ACIDs are used to store accounting credits for use on the T3E. T3E credits and T3D credits are entirely separate. To view the current state of your ACID on the T3E you should use the `t3eacct` command; this will return the number of T3E hours left in your ACID.

When discussing time allocations on the T3E, EPSRC and other bodies often work in T3D-equivalent hours, at a conversion rate of 4 T3D hours to 1 T3E hour. However, the software on the machine itself works exclusively in real T3E hours.

Further Information

- "Charging" in the *Introduction*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro/node36.html`

7 Batch Jobs under NQS

Batch jobs are submitted to a single queue called `T3E`. The values of some of the QSUB parameters, including the ACID, number of PEs, CPU time and memory, are then used to select the appropriate internal queue. If the number of PEs to use is not explicitly requested the job will be assumed to contain single-processor programs to be run on the command PEs.

The T3E at EPCC will have a variety of memory sizes. The batch system on the T3E currently does not provide a flag to allow you to request a specific amount of memory for a parallel job.

Note that some of the QSUB parameters are interpreted differently on the T3E to the T3D.

- `-l mpp_t` On the T3E this sets the maximum MPP CPU time usable by each apteam PE in the request. In practice as most of the message passing libraries "busy-wait" this is almost the same as wall-clock time, unless a large amount of disk IO is performed.
- `-l p_mpp_t` As above but for each individual process.
- `-lt` On the T3E this sets the maximum CPU time useable on the command PEs by a single process in a batch request.
- `-lT` On the T3E this sets the maximum cumulative CPU time useable on the command PEs in a batch request.

Further Information

- "Batch jobs under NQS" in the *Introduction*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro/node37.html`

8 Tools

The Apprentice, XBrowse and TotalView tools are all available on the T3E and work in exactly the same way as they do on the T3D/J90 system. In fact, you can move `core` and `app.rif` files from the T3E to the J90 and debug or analyse them there, or vice versa.

Note that the T3E generates `core` files instead of `mppcore` files. By default only one PE generates a core file when an error occurs. You can force information about all PEs to be included in the core file by specifying the `-c` flag to `mpprun`.

An additional tool: PAT

There is an additional profiling tool available on the T3E called PAT. This is a very much simpler profiler than Apprentice but can generate subroutine-level profiles with very low overhead.

PAT is a PC-sampling profiler (similar to `prof`) that also uses the hardware performance counters to collect information about instruction counts. It can also be used to add extra instrumentation to the program which can gather additional information about selected subroutines.

PAT has the following advantages:

- It can be enabled at link time, you don't need to recompile the entire code.
- It has very low overhead.
- It displays information about library routines as well as your own code.
- It displays simple load-balance information.
- You can add explicit event tracing to your code that can then be displayed by PAT.

To enable PAT, you should relink your code with the `pat.cld` file and the `pat` library:

```
f90 -o program pat.cld *.o -lpat
```

Then run the program to generate the profile. This will generate a program data file `pdf.`*nnnn* which can be analysed using PAT itself:

```
pat program pdf.nnnn
```

Further Information

- "Tools Overview" in the *Introduction*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-intro/node47.html`
- "TotalView" in *Tools and Techniques*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-tandt/node58.html`
- "Apprentice" in *Tools and Techniques*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-tandt/node59.html`
- "XBrowse" in *Tools and Techniques*:
  `http://www.epcc.ed.ac.uk/t3d/documents/t3d-tandt/node60.html`