Z01AAFP

NAG Parallel Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details.

1 Description

Z01AAFP defines a Library Grid and returns a grid context (see Blackford *et al.* [1]) which is subsequently used by other library routines. Further calls to this routine are not possible unless a call to Z01ABFP, which undefines the Library Grid and invalidates the grid context (see Z01ABFP documentation), is made, therefore only one library context may exist at one time.

Z01AAFP may claim up to the maximum number of logical processors $p_{\max} = m_{p0} \times n_{p0}$, as initialised by MPI, where m_{p0} and n_{p0} are the number of processor rows and columns requested on entry to Z01AAFP. Z01AAFP makes an indirect call to MPI_INIT to ensure that $p_{\max}$ identical processes are executing. Further calls to Z01AAFP with different numbers of processor rows, m_p and columns n_p must meet the constraint $m_p \times n_p \leq p_{\max}$.

In more sophisticated applications it is possible for the user to set up processes before calling any NAG Parallel Library routine. This requires a call to Z01AEFP which explicitly disables the checking carried out in the Library routines. However, caution should be exercised when using this facility (see Section 6 for information).

All processors must make a call to Z01AAFP to set up a grid. Failure to do this is a programming error which cannot be detected by the library mechanism and will cause unpredictable results.

2 Specification

```
SUBROUTINE Z01AAFP(ICNTXT, MP, NP, IFAIL)
INTEGER            ICNTXT, MP, NP, IFAIL
```

3 Usage

3.1 Definitions

The following definitions are used in describing the logical processor grid within this document

m_p	–	the number of rows in the Library Grid.
n_p	–	the number of columns in the Library Grid.

3.2 Global and Local Arguments

The routine will ensure that the values set on the root processor overwrite the values set on other processors in the grid defined within the grid context, ICNTXT.

The following global **input** arguments must have the same value on entry to the routine on each processor and the global **output** arguments will have the same value on exit from the routine on each processor:

Global input arguments:	IFAIL
Global output arguments:	MP, NP, IFAIL

Note: it is sufficient for the user to set MP and NP on the root processor (usually the {0,0} processor) in the grid (if the user is not multigridding, this is the processor, identifiable by a call to Z01ACFP).

The output argument ICNTXT is local and should be treated as if it were a pointer to the grid (i.e., its value need not be examined and **must not** be modified).

4 Arguments

1: ICNTXT — INTEGER *Local Output*

On entry: the Library context, usually returned by a call to the Library Grid initialisation routine Z01AAFP.

Note: the value of ICNTXT **must not** be changed.

2: MP — INTEGER *Local Input/Global Output*

On entry: m_p, the number of rows in the logical processor grid will be taken to be the value supplied on the root processor (usually the {0,0} processor and identifiable by a call to Z01ACFP).

On exit: the value supplied to this routine on the root processor will be returned on all other processors. The value is unchanged on the root processor.

Constraint: MP > 0.

3: NP — INTEGER *Local Input/Global Output*

On entry: the number of columns, n_p, in the logical processor grid will be taken to be the value supplied on the root processor (usually the {0,0} processor and identifiable by a call to Z01ACFP).

On exit: the value supplied to this routine on the root processor will be returned on all other processors. The value is unchanged on the root processor.

Constraint: NP > 0.

4: IFAIL — INTEGER *Global Input/Global Output*

The NAG Parallel Library provides a mechanism, via the routine Z02EAFP, to reduce the amount of parameter validation performed by this routine. For a full description refer to the Z02 Chapter Introduction.

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this argument (described in the Essential Introduction) the recommended values are:

IFAIL $= 0$, if multigridding is **not** employed;
IFAIL $= -1$, if multigridding is employed.

On exit: IFAIL $= 0$ (or -9999 if reduced error checking is enabled) unless the routine detects an error (see Section 5).

5 Errors and Warnings

If on entry IFAIL $= 0$ or -1, explanatory error messages are output from the root processor (or processor {0,0} when the root processor is not available) on the current error message unit (as defined by X04AAF).

5.1 Full Error Checking Mode Only

IFAIL $= 1$

MP ≤ 0 or NP ≤ 0.

IFAIL $= 2$

The size of the logical processor grid requested was too large, i.e., there were not enough processes available to map the logical processors onto (see Section 6 for further information).

IFAIL $= 3$

A second call to Z01AAFP has been made without undefining the previous logical processor grid by a call to Z01ABFP.

6 Further Comments

A call to Z01AAFP is used to set up a 'Library Grid' which includes a logical processor grid and its associated context. The environment on which the Library is run expects a static parallel machine, hence the available number of processes will not change when MPI is invoked. The first call to Z01AAFP claims enough logical processors to form the requested grid. However, the user is allowed to reshape the grid (i.e., MP and NP can be varied) by further calls to Z01AAFP provided that a call has already been made to Z01ABFP with appropriate arguments and the total number of processes requested (MP ×NP) does not exceed $p_{\max} = m_{p0} \times n_{p0}$.

For more sophisticated applications, the user may set up his own processor grid(s); in this case a call to Z01AEFP must be made before a call can be made to any Library routine. This gives the user the option of multigridding. See the on-line Tutorial available on our web site for an example of a multigridding program.

Every effort has been made to ensure that the library mechanism is as robust as possible. If the user chooses to employ multigridding then extra care must be taken in programming. In particular, the user must ensure that the SPMD model is adhered to. The user should be thoroughly familiar with initialising processes (and the Basic Linear Algebra Communication Subprogram (BLACS), see the on-line Tutorial available on our web site) before attempting to use the library in a multigrid program since some potential programming errors cannot be trapped. (Normally simple programming errors such as failure to initialize global arguments on all processors can be detected.)

Note: a call to Z01AAFP is a **synchronisation point** which involves **all** processors.

7 References

[1] Blackford L S, Choi J, Cleary A, D'Azevedo E, Demmel J, Dhillon I, Dongarra J, Hammarling S, Henry G, Petitet A, Stanley K, Walker D and Whaley R C (1997) ScaLAPACK Users' Guide *SIAM* 3600 University City Science Center, Philadelpia, PA 19104-2688, USA. URL: http://www.netlib.org/scalapack/slug/scalapack_slug.html

[2] Gropp W, Lusk E and Skjellum A (1994) *Using MPI: Portable Parallel Programming with the Message-Passing Interface* Cambridge, MA, MIT Press.

8 Example

None.