Chapter 8

Initialization

8.1 Calling CM Fortran Subprograms from Serial Languages

The utility library procedure **CMF_INIT** initializes the parallel processing unit. You must call this procedure before calling any CM Fortran subprogram from a program compiled with a serial language compiler (not **cmf**). This requirement applies to programs that run globally (on the CM-5 partition manager) or locally (on the individual nodes).

The **CMF_INIT** procedure is useful for calling CM Fortran kernels from serial Fortran, C, or other languages with C-compatible calling conventions. Both Fortran and C versions of the initialization procedure are provided. To call CM Fortran from C*, see the instructions given in the *C* User's Guide*.

Fortran Syntax:

```
CALL CMF_INIT (PADDING, AXISREORDER)
```

C Syntax:

```
void CMF_init (int padding, int axisreorder)
```


Copyright © 1994 Thinking Machines Corporation

CMF_INIT

Initializes the parallel processing unit in preparation for calling a CM Fortran subprogram from a serial program.

Fortran Syntax

```
CALL CMF_INIT( PADDING, AXISREORDER )
```

C Syntax

```
CMF_init(padding, axisreorder);
```

Arguments

PADDING	A scalar logical (Fortran) or integer (C) that specifies whether the CM Fortran subprogram(s) to be linked in were compiled to expect vector-length padding in the allocation of CM arrays. The value **.TRUE.** in Fortran or nonzero in C signifies **-padding** (the **cmf** default); the value **.FALSE.** in Fortran or zero in C signifies **-nopadding**.
AXISREORDER	A scalar logical (Fortran) or integer (C) that specifies whether the CM Fortran subprogram(s) to be linked in were compiled to expect axis reordering in the allocation of CM arrays. The value **.TRUE.** in Fortran or nonzero in C signifies **-axisreorder** (the **cmf** default); the value **.FALSE.** in Fortran or zero in C signifies **-noaxisreorder**.

Result

None.

Version 2.2, October 1994
Copyright © 1994 Thinking Machines Corporation

Description

CMF_INIT must be called to enable data parallel code to execute on the CM system. When the main program unit is compiled by **cmf** (by virtue of having the filename extension **.fcm** or **.FCM**), the compiler generates a call to this procedure. When the main program unit is compiled by a serial language compiler, the program must call **CMF_INIT** explicitly before calling any CM Fortran subroutine or function.

The arguments indicate certain switch settings used in the compilation of the CM Fortran units. These arguments are passed to the run-time system where they affect the layout of CM arrays. All CM Fortran subprograms must have been compiled with the switch settings indicated in the arguments. Any subsequent calls to **CMF_INIT** after the first one are ignored.

The **cmf** switches **-[no]padding** and **-[no]axisreorder** are used to influence performance, particularly when the program specifies detailed layouts for CM arrays. See the *CM-5 CM Fortran Performance Guide* for an explanation of the significance of these switches. The defaults are **-padding** and **-axisreorder**.

The requirement to call **CMF_INIT** and to identify the switch settings used in compiling the subprograms is the same regardless of whether the calling program is running globally (on the partition manager) or nodally (copies on each of the nodes).

For information on calling CM Fortran from C*, please see the *C* User's Guide*. C* programs need not call **CMF_init**, but Version 7.1 requires that the CM Fortran subprograms be compiled with **-padding** (the default).

Examples

These two examples appear on-line in **/usr/examples/cmfortran/f77_main_1** and **c_main_1**, along with **make** files for compilation and linking. See your site system administrator for the location if these files have been moved.

Copyright © 1994 Thinking Machines Corporation

A Fortran 77 program calling a CM Fortran subroutine:

```
      PROGRAM TST1          ! in f77_main_1_f.f
      REAL A(1000)
      INTEGER I

      CALL CMF_INIT(.TRUE., .TRUE.)

      DO I = 1,1000
        A(I) = I
      END DO

      DO I = 1,901,100
        CALL MYCMFSUB(A(I))
      END DO

      PRINT 10, A
   10 FORMAT(10F6.2)
      END

      SUBROUTINE MYCMFSUB(A)     ! in f77_main_1.fcm
      REAL A(100)
CMF$  LAYOUT A(:SERIAL)
      REAL A_CM(100), B
CMF$  LAYOUT A_CM(:NEWS)

      CALL CMF_FE_ARRAY_TO_CM(A_CM, A)
      B = SUM(A_CM)
      A_CM = (A_CM * A_CM) / B
      CALL CMF_FE_ARRAY_FROM_CM(A, A_CM)

      PRINT *, ' The sum is ', B
      END
```


Version 2.2, October 1994
Copyright © 1994 Thinking Machines Corporation

A C program calling a CM Fortran subroutine:

```
#include <stdio.h>        /* in c_main_1_c.c */

#define TRUE 1
#define FALSE 0

main()
{
      float a[1000];
      int i;

      CMF_init(TRUE,TRUE);

      for (i = 0; i < 1000; i++)
        a[i] = i+1;

      for (i = 0; i < 1000; i += 100)
        mycmfsub_(&a[i]);

      for (i = 0; i < 1000; i++)
        printf(" %f\n", a[i]);
}

      SUBROUTINE MYCMFSUB(A)    ! in c_main_1.fcm
      REAL A(100)
CMF$  LAYOUT A(:SERIAL)
      REAL A_CM(100), B
CMF$  LAYOUT A_CM(:NEWS)

      CALL CMF_FE_ARRAY_TO_CM(A_CM, A)
      B = SUM(A_CM)
      A_CM = (A_CM * A_CM) / B
      CALL CMF_FE_ARRAY_FROM_CM(A, A_CM)

      PRINT *, ' The sum is ', B
      END
```


Copyright © 1994 Thinking Machines Corporation