Chapter 1

Introduction to CMX11

The Connection Machine system is designed to be an integral part of a heterogeneous supercomputing network. In these environments, the X Window System has become the dominant network protocol for distributed graphics programming. The CMX11 library allows you to link the computational power of the Connection Machine system to the flexibility and portability of the X Window System directly from your CM Fortran or C* program without any X programming. You can display the results of a computation that are stored in parallel arrays on the CM system on an X Window System display anywhere on the network.

For example, you might visualize a fluid flow simulation with a particle system that generates parallel arrays of point coordinates (x, y) recording the movements of particles within the system. CMX11 allows you to establish an X window as a display and then, with a single subroutine call to **`CMXDrawPoint`**, draw the entire array of points to the X window.

1.1 The CMX11 Library

The CMX11 library consists of three main groups of routines:

- *Display routines*: A functional CM Fortran or C* interface that makes it easy to create and control one or more windows on an X11 server as a CM display.
- *Drawing routines*: Parallel extensions to the standard X drawing primitives that accept parallel arrays of coordinate and color information. These routines enable you to draw large sets of points, lines, arcs, rectangles, filled polygons, text strings, or an image array with a single

Copyright © 1993 Thinking Machines Corporation

subroutine call. There are also several wrappers that enable CM Fortran programmers to use serial drawing routines.

- *Mouse routines:* Routines that provide a mouse interface, including the ability to draw, to access information about the mouse cursor, and to allocate and deallocate the mouse point object.

These basic capabilities require only a few subroutine calls and allow you to get a visualization up and running quickly.

If you need to refine the operation of your visualization or integrate it with an existing X or Motif application, CMX11 also provides supporting routines that return information about the X display capabilities and that also control the display from your CM Fortran or C* program. In particular, CMX11 provides a full set of routines that support color management on the display. You can determine the color capabilities of the display, allocate colors on the display colormap, and load the colormap with standard colormaps provided by CMX11 or with color arrays of your own.

Coming Up

The remainder of this chapter describes the CMX11 library and describes how to get started using it to set up a display and write to it.

Chapter 2 describes the routines that support displays, including those that return information about the display, manage color, and control the application resources.

Chapter 3 describes the CMX11 drawing routines.

Chapter 4 describes the CMX11 mouse routines.

Chapter 5 provides man pages for each CMX11 routine.

Chapter 6 provides man pages for each scalar X11 wrapper.

1.2 A Simple CM Fortran Example

The following example demonstrates the basic use of CMX11 to create a display and draw to it. The program creates a spray of particles in the center

Version 1.6, October 1993
Copyright © 1993 Thinking Machines Corporation

of the window. The particles radiate out from the center and fall toward the bottom of the window.

The program is discussed in detail in Sections 1.4 and 1.5.

Before running this program you must specify the display to connect to by setting the **DISPLAY** environment variable, as explained in Section 1.4.1, or by using the **-display** *name* command line argument.

Note that the visualization itself involves only five subroutines:

- **CMXCreateSimpleDisplay** to open the display.
- **CMXGetDisplay**, **CMXGetDrawable**, and **CMXGetGC** to get identifiers for the display's X display, drawable, and graphics context. (These must be passed to the drawing routine.)
- **CMXDrawPoint** to draw the points from parallel arrays of point coordinates.

```
      PROGRAM DRAW_POINTS
      IMPLICIT NONE

      INCLUDE '/usr/include/cm/cmx-cmf.h'

      INTEGER ARRAYSIZE
      PARAMETER (ARRAYSIZE = 1024)
      INTEGER,DATA:: NUM_ITERATIONS=1000
      INTEGER,DATA:: XSIZE = 256
      INTEGER,DATA:: YSIZE = 256
      INTEGER,DATA:: NCOLORS = 0
      INTEGER CMXDISPLAY
      INTEGER XDISPLAY, WINDOW, GC
      INTEGER I
      REAL FX(ARRAYSIZE), FY(ARRAYSIZE)
      REAL DX(ARRAYSIZE), DY(ARRAYSIZE)
      INTEGER X(ARRAYSIZE), Y(ARRAYSIZE)
CMF$LAYOUT FX(:NEWS), FY(:NEWS), DX(:NEWS), DY(:NEWS)
CMF$LAYOUT X(:NEWS), Y(:NEWS)

C     Enable command line options for Fortran

      CALL CMXSetArgcArgvF()
```


Copyright © 1993 Thinking Machines Corporation

```
C    Open the display window

     CMXDISPLAY = CMXCreateSimpleDisplay(NCOLORS, XSIZE, YSIZE)

C    Get X11 Identifiers

     XDISPLAY = CMXGetDisplay(CMXDISPLAY)
     WINDOW = CMXGetDrawable(CMXDISPLAY)
     GC = CMXGetGC(CMXDISPLAY)

C    Initialize coords and deltas

     FX = XSIZE/2
     FY = YSIZE/2
     DX = SIN(2.0 * 3.14159 * [1:ARRAYSIZE]/ARRAYSIZE)
     DY = COS(2.0 * 3.14159 * [1:ARRAYSIZE]/ARRAYSIZE)

C    Compute Point Coordinates

     DO I=1,NUM_ITERATIONS
       FX = FX + DX
       FY = FY + DY
       DY = DY + 0.02
       X = FX
       Y = FY

C    Draw Points

       CALL CMXDrawPoint(XDISPLAY, WINDOW, GC, X, Y)
     END DO

STOP
END
```

1.3 A Simple C* Example

This example demonstrates the use of CMX11 calls interspersed in C* code. The program draws lines in a window created by **`CMXCreateSimpleDisplay`**.

The program is discussed in detail in the sections that follow.

Version 1.6, October 1993
Copyright © 1993 Thinking Machines Corporation

Before running this program you must specify the display to connect to by setting the **DISPLAY** environment variable, as explained in Section 1.4.1, or by using the command line argument.

Like the CM Fortran example, this program uses only five subroutines for the visualization itself:

- **CMXCreateSimpleDisplay** to open the display.
- **CMXGetDisplay**, **CMXGetGC**, and **CMXGetDrawable** to get identifiers for the display's X display, drawable, and graphics context. (These must be passed to **CMXDrawLine**.)
- **CMXDrawLine** to draw the lines.

```
/*
 * CMX11 C* example: draw lines in a SimpleDisplay window
 */

#include <cm/cmxdisplay.h>
#include <cm/cmx-cs.h>
#include <stdio.h>

shape [32]One_d;

int:One_d xc0,xc1,yc0,yc1;

main (int argc, char **argv)
{
    Display *ex_display;
    GC ex_gc;
    Window ex_window;

    int width,height,colors;

    CMX_display_t cmxdisplay;

    width = 400;
    height = 400;
    colors = 1;

    CMXSetArgcArgv(argc,argv);
    cmxdisplay = CMXCreateSimpleDisplay(colors,width,height);
```


Copyright © 1993 Thinking Machines Corporation

```
    /* Define window properties for X */

    ex_display = CMXGetDisplay(cmxdisplay);
    ex_gc = CMXGetGC(cmxdisplay);
    ex_window = CMXGetDrawable(cmxdisplay);

    with(One_d)
    {
        xc0 = 50;
        xc1 = 350;

        yc0 = 12*pcoord(0);
        yc1 = yc0;

        CMXDrawLine(ex_display,ex_window,ex_gc,xc0,yc0,xc1,yc1);
    }
    /* End the demo */

    printf(" Press Return key to quit\n");
    getchar();
}                               /* end of main */
```

1.4 Opening a Display

A CMX11 display is a connection to an X server and an associated window. Opening the display requires only a single subroutine call in your program, but before running the program you must specify the X server you wish to connect to.

1.4.1 Specifying the Display

In CMX11, you can specify the X11 server to connect to by setting the **DISPLAY** environment variable, or you can specify a server at run time by using the **-display** option on the command line when you invoke your program if you have used **CMXSetArgcArgvF** or **CMXSetArgcArgv**.

Copyright © 1993 Thinking Machines Corporation

The display is named using the standard X convention: the hostname of the machine supporting the server followed by a colon, then the server number followed by a period, and then the screen number; for example, *hostname:0.0*. If you are displaying to the screen of a single-user workstation, the server number and screen number are usually 0.

To set the **DISPLAY** environment variable from a C shell, type

```
setenv DISPLAY hostname:0.0
```

To set the **DISPLAY** environment variable from a Bourne shell, type

```
DISPLAY=hostname:0.0; export DISPLAY
```

To use the command line option from a CM Fortran or C* program, you must call **CMXSetArgcArgvF()** in the main portion of your program. Then you can name the host on which you wish to display by adding **-display** followed by the hostname on the command line when you invoke your program:

```
my_prog -display hostname:0.0
```

The CMX11 display creation routines—**CMXCreateSimpleDisplay** and **CMXCreateDisplayClass**—automatically parse the command line and attempt to open the X server on the host specified after **-display**. A hostname specified on the command line overrides any specified in the **DISPLAY** environment variable.

NOTE

The CMX11 display creation routines also respond to any of the command line arguments recognized by the X routine **XtOpenDisplay**. For example, **-background** can be used to set the display background color. See your X Window System documentation for details.

Copyright © 1993 Thinking Machines Corporation

1.4.2 Connecting to the Display

To connect to the display specified you need only call **CMXCreateSimpleDisplay**:

From the CM Fortran example:

```
CMXDISPLAY = CMXCreateSimpleDisplay(NCOLORS, XSIZE, YSIZE)
```

From the C* example:

```
cmxdisplay = CMXCreateSimpleDisplay(colors,width,height);
```

The system opens a window on the display sized to **XSIZE** by **YSIZE** (**width** by **height** for C*) pixels by default. The actual size and appearance of the window is determined by the window manager running on the X server. If the window has a title bar, the default name displayed is "CM Display."

CMXCreateSimpleDisplay uses the value of **NCOLORS** (**colors** in the C* example) to determine the appropriate visual class on the display. You might not actually be able to allocate that many colors. Setting **NCOLORS** (or **colors**) to 0 means that any number of colors is okay. The drawing routine draws in the window's current foreground color. Section 2.3 describes how to determine and manage the display's color characteristics.

The return value **CMXDISPLAY** (or **cmxdisplay** for C*) is an identifier that can be passed to other CMX11 routines that return information about or operate on the CMX11 display.

1.5 Drawing to a Display

Before actually drawing to the display, you must get identifiers for the underlying X11 display, drawable, and graphics context:

In the CM Fortran example:

```
XDISPLAY = CMXGetDisplay(CMXDISPLAYID)
WINDOW = CMXGetDrawable(CMXDISPLAYID)
GC = CMXGetGC(CMXDISPLAYID)
```


Copyright © 1993 Thinking Machines Corporation

In the C* example:

```
ex_display = CMXGetDisplay(cmxdisplay)
ex_gc = CMXGetGC(cmxdisplay)
ex_window = CMXGetDrawable(cmxdisplay)
```

These integer return values identify the X11 **Display**, **Drawable**, and **GC** (graphics context) structures of the display created by **CMXCreateSimpleDisplay**. You use these identifiers with CMX11 routines or X11 routines (in C*) that operate on the display.

For example, the values returned by these routines are passed as arguments to the CMX11 parallel drawing routines.

In the CM Fortran example, the values are passed to the drawing routine **CMXDrawPoint**:

```
CALL CMXDrawPoint(XDISPLAY, WINDOW, GC, X, Y)
```

X and **Y** are parallel arrays containing the coordinates of the points to be drawn. The arrays must be conformable: that is, they must be the same size and shape. **CMXDrawPoint** draws a point for each data element in the arrays; the first point is drawn at $(x(1), y(1))$, the second at $(x(2), y(2))$, etc. As with XDrawPoint, the points are drawn in the foreground color specified in the display's graphics context. (Managing the display color is discussed in Section 2.3.)

In the C* example, the values are passed to the drawing routine **CMXDrawLine**:

```
CMXDrawLine(ex_display,ex_window,ex_gc,xc0,yc0,xc1,yc1);
```

Lines are drawn from [0]xc0,[0]yc0 to [0]xc1,[0]yc1, and so on.

CMX11 also supplies versions of the drawing routines that let you specify different colors for the points or lines, such as:

```
CALL CMXDrawCPoint(XDISPLAY, WINDOW, GC, X, Y, COLOR)
```

COLOR is an array of color values conforming to **X** and **Y**. The color values are indices into the display colormap. Each point is drawn in the color specified in the corresponding element of the **COLOR** array.

Copyright © 1993 Thinking Machines Corporation

1.6 Compiling and Running

CM Fortran

To use the CMX11 routines in CM Fortran you must

- include the appropriate header file in your program:

```
INCLUDE '/usr/include/cm/cmx-cmf.h'
```

- link with the supporting libraries when compiling:

```
cmf prog.fcm -lcmx_cm5_{sparc|vu}_sp -lXm -lXt -lXext -lX11
```

NOTE

If you are not using the display functions (**`CMXCreateSimpleDisplay`**, etc.) or making direct calls to Motif or X Toolkit functions (names beginning with Xm or Xt), then you should *not* link with **`-lXm`** or **`-lXt`**. Remove those libraries from the link line, leaving:

```
cmf prog.fcm -lcmx_cm5_{sparc|vu}_sp -lXext -lX11
```

C*

To use the CMX11 routines in C* you must

- include the appropriate header files in your program:

  if you are using the display functions:

```
#include <cm/cmxdisplay.h>
#include <cm/cmx-cs.h>
```

  if you are not using the display functions:

```
#include <cm/cmx.h>
#include <cm/cmx-cs.h>
```


Version 1.6, October 1993
Copyright © 1993 Thinking Machines Corporation

- link with the supporting libraries when compiling:

```
cs prog.cs -lcmx_cm5_{sparc|vu}_sp -lXm -lXt -lXext -lX11
```

NOTE

If you are not using the display functions (**CMXCreateSimpleDisplay**, etc.) or making direct calls to Motif or X Toolkit functions (names beginning with Xm or Xt), then you should *not* link with **-lXm** or **-lXt**. Remove those libraries from the link line, leaving:

```
cs prog.cs -lcmx_cm5_{sparc|vu}_sp -lXext -lX11
```

1.7 On-Line Examples

Several examples are available on-line.

1.7.1 CM Fortran Examples

The default directory for the CM Fortran examples is **/usr/examples/cmx11/cmf**. Contact your system administrator if you do not find them there.

- **/usr/examples/cmx11/cmf/image/image.FCM**

  Puts up a very simple color image of diagonal stripes on a color display. CMX11 functions used include:

CMXSetArgcArgvF	**CMXGetColormap**
CMXCreateSimpleDisplay	**CMXAllocateColors**
CMXGetDisplay	**CMXLoadColorArrays**
CMXGetDrawable	**CMXStoreColors**
CMXGetGC	**CMXPutImage**

Copyright © 1993 Thinking Machines Corporation

- **/usr/examples/cmx11/cmf/cmxdemo/cmxdemo.FCM**

  Demonstrates many of the CMX11 drawing functions. CMX11 functions used include:

  CMXGetXAppName
  CMXSetXAppName
  CMXSetXWindowTitle
  CMXCreateSimpleDisplay
  CMXGetDisplay
  CMXGetDrawable
  CMXGetGC
  CMXGetColormap
  CMXSetForeground
  CMXDrawPoint
  CMXClearDisplay
  CMXDrawLine
  CMXDrawCLine
  CMXFillCRectangle
  CMXDrawCString
  CMXAllocMBuffers
  CMXFillCPolygon
  CMXSwitchMBuffers
  CMXPutImage
  CMXAllocateColors
  CMXStoreColors

- **/usr/examples/cmx11/cmf/cmap/cmap.FCM**

  Puts up all four named CMX11 colormaps in separate windows (on a color display). CMX11 functions used include:

  CMXSetArgcArgvF
  CMXSetXWindowTitle
  CMXSetXIconName
  CMXCreateSimpleDisplay
  CMXGetDisplay
  CMXGetDrawable
  CMXGetGC
  CMXGetColormap
  CMXAllocateColors
  CMXLoadColorArrays
  CMXStoreColors
  CMXPutImage

- **/usr/examples/cmx11/cmf/cmap/mouse/mouse.FCM**

  Demonstrates the use of the mouse routines and scalar drawing functions. Functions used include:

  CMXSetArgcArgvF
  CMXCreateSimpleDisplay
  CMXGetDisplay
  CMXGetDrawable
  CMXGetGC
  CMXGetColormap
  CMXGetWidget
  CMXAllocateMousePoint
  XLoadFont
  XWhitePixel
  XBlackPixel
  CMXAllocateColors
  XSync
  CMXGetMousePoint
  XFillRectangle
  CMXSetBackground
  XDrawLine
  XDrawImageString
  XDrawPoint
  XDrawArc
  XFillArc
  CMXGetMouseLine
  CMXMousePointX
  CMXMousePointY

Copyright © 1993 Thinking Machines Corporation

```
CMXStoreColors          CMXGetMouseRectangle
XSetWindowBackground    XDrawRectangle
CMXClearDisplay         CMXCurrentMousePoint
CMXSetForeground        CMXProcessEvents
XDrawString
```

1.7.2 C* Examples

The default directory for the C* examples is **/usr/examples/cmx11/cstar**. Contact your system administrator if you do not find them there.

- **lines_by_simple_window.cs**

  Draws lines in a SimpleDisplay window. CMX11 functions used include:

  ```
  CMXCreateSimpleDisplay  CMXGetDrawable
  CMXGetDisplay           CMXDrawLine
  CMXGetGC
  ```

- **lines_by_standardx.cs**

  Draws lines in parallel in a standard X window. This example illustrates that C* programmers can use X calls directly. X11 functions used include:

  ```
  XOpenDisplay            XSelectInput
  BlackPixel              XMapRaised
  WhitePixel              XNextEvent
  XLoadFont               XDrawImageString
  XCreateSimpleWindow     XLookupString
  XSetStandardProperties  XClearWindow
  XCreateGC               XFreeGC
  XSetBackground          XDestroyWindow
  XSetForeground          XCloseDisplay
  XSetFont
  ```

  CMX11 functions used include:

  ```
  CMXDrawLine
  ```


Copyright © 1993 Thinking Machines Corporation

- **motif_demo.cs**

  Demonstrates Motif interface with CMX11 drawing. Draws various primitives in parallel in a Motif drawing area widget. Motif, Xt, and X functions used include:

  XtVaAppInitialize
  XtVaCreateManagedWidget
  XtVaSetValues
  XLoadQueryFont
  XmFontListCreate
  XmStringCreateLtoR
  XtAddCallback
  XmStringFree
  XmFontListFree
  XtRealizeWidget
  XtAppMainLoop
  XtDisplay
  XtGetGC
  XClearArea

  CMX11 functions used include:

  CMXDrawLine
  CMXDrawArc
  CMXDrawRectangle

- **putimage.cs**

  Puts up a simple rainbow image. CMX11 functions used include:

  CMXCreateSimpleDisplay
  CMXGetDisplay
  CMXGetDrawable
  CMXGetGC
  CMXGetColormap
  CMXAllocateColors
  CMXLoadColorArrays
  CMXStoreColors
  CMXPutImage

Version 1.6, October 1993
Copyright © 1993 Thinking Machines Corporation