C-Motion® Engine Development Tools Manual

Performance Motion Devices, Inc.
80 Central Street
Boxborough, MA 01719

Revision 1.2, April 18 2012

NOTICE

Warranty

PMD warrants performance of its products to the specifications applicable at the time of sale in accordance with PMD's standard warranty. Testing and other quality control techniques are utilized to the extent PMD deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Performance Motion Devices, Inc. (PMD) reserves the right to make changes to its products or to discontinue any product or service without notice, and advises customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

Safety Notice

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage. Products are not designed, authorized, or warranted to be suitable for use in life support devices or systems or other critical applications. Inclusion of PMD products in such applications is understood to be fully at the customer's risk.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent procedural hazards.

Disclaimer

PMD assumes no liability for applications assistance or customer product design. PMD does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of PMD covering or relating to any combination, machine, or process in which such products or services might be or are used. PMD's publication of information regarding any third party's products or services does not constitute PMD's approval, warranty or endorsement thereof.

Related Documents

ION/CME Digital Drive User's Manual

Complete description of the ION/CME Digital Drive including installation/getting started section, operational overview, detailed connector information, and complete electrical and mechanical specifications.

Prodigy® /CME Stand-Alone Card User's Guide
Prodigy® /CME PCI User's Guide
Prodigy® /CME PC/104 User's Guide

Complete description of the Prodigy/CME motion cards including getting starting section, operational overview, detailed connector information, and complete electrical and mechanical specifications.

Magellan® Motion Processor User's Guide

Complete description of the Magellan Motion Processor features and functions with detailed theory of its operation.

Magellan® Motion Processor Programmer's Command Reference

Descriptions of all Magellan Motion Processor commands, with coding syntax and examples, listed alphabetically for quick reference.

PMD Resource Access Protocol Programmer's Reference

Descriptions of all PRP device commands, with software architecture overview, command syntax, and examples.

Pro-Motion® User's Guide

User's guide to Pro-Motion, the easy-to-use motion system development tool and performance optimizer. Pro-Motion is a sophisticated, easy-to-use program which allows all motion parameters to be set and/or viewed, and allows all features to be exercised.

Table of Contents

Chapter 1. Introduction 7
1.1 C-Motion Engine Device Overview 7
1.2 Introduction 7
1.3 CME Device Software Overview 8
1.4 Getting Started 9
1.5 Software Installation 10
1.6 Programmer's Notepad 11
1.7 Downloading the Code 14
1.8 Executing the Code 15

Chapter 2. Application Development Tools 17
2.1 Programmer's Notebook 17
2.2 Pro-Motion 18
2.3 Software Development 19

Appendix A. – Copyright Notices

This page intentionally left blank.

1. Introduction

1.1 C-Motion Engine Device Overview

Several products in the Prodigy family of Motion Control Cards and the ION family of digital drives provide a programmable C-Motion Engine (CME), which allows user application code to be run in a motion control device instead of on a host computer.

The following table summarizes the PMD products incorporating a CME, and their associated Users Guide documentation.

Prodigy P/N	Product Type	Bus/Format	User Guide
PR8358x20	Card	PC/104	Prodigy/CME PC/104 User's Guide
PR9358x20	Card	PCI	Prodigy/CME PCI User's Guide
PR13x58x20	Card	Stand-Alone	Prodigy/CME Stand-Alone Card User's Guide
DD3x1S0-056/15	Digital Drive	Ethernet/Serial	ION/CME User's Guide

In this manual all Prodigy cards or ION digital drives that include a C-Motion Engine will collectively be called CME devices.

1.2 Introduction

This manual documents the C-Motion Engine Development Tools (also referred to as CME Development Tools), provided with CME devices, that allow user application code to be created and compiled on a host PC, and then downloaded, executed, and monitored on the CME device's C-Motion Engine module.

The C-Motion Engine Development Tools provides the following features:

- Complete toolset for creation of user-specific applications running on the motion card
- Open source compiler & motion control C libraries
- Interactive Development Environment
- Supports PCI bus, PC/104 bus, RS232/RS485, CANbus, and Ethernet communications

The CME Development Tools includes the following major elements:

- Programmer's Notepad source code editor
- Open source C language compiler & linker
- PMD C language motion control and C-Motion Engine binary libraries
- PMD C language motion control and C-Motion Engine libraries in source form for host-based applications.
- PMD Pro-Motion application which is used for code management & monitoring

These tools, utilized together, allow a user to quickly and easily develop applications that will run on a device equipped with a C-Motion Engine.

The remainder of this chapter, section 1.3 and beyond, provides a convenient 'Getting Started' guide that introduces the major elements of the C-Motion Engine Development Tools, and takes the user through installation of the development tools, loading of a simple example program, compilation and linking of that program, and downloading and execution of that example program.

1.3 CME Device Software Overview

Four major software packages are provided with the CME devices:

Pro-Motion®, an interactive Windows-based exerciser & software development tool

C-Motion®, a C-language library for Magellan Motion Processors that allows you to create motion applications using the C programming language

VB-Motion®, a dynamic link library (DLL) callable from many dynamic languages and source code for using the DLL with Microsoft Visual Basic.

C-Motion® Engine Development Tools, a set of development resources that allow you to create, download, and monitor programs loaded in the CME device's C-Motion Engine

Here is more information on each of these software packages:

1.3.1 Pro-Motion

Pro-Motion is a sophisticated, easy-to-use exerciser program which provides support for C-Motion Engine code development, and general purpose motion system setup and exerciser functions. The Pro-Motion features include:

- **C-Motion Engine Functions:**
  - C-Motion Engine console window function
  - C-Motion Engine user application code download
  - Interactive C-Motion Engine code execution and monitoring functions
- **General Purpose Functions:**
  - Axis Wizard to automate axis setup and configuration
  - High performance motion oscilloscope
  - Interactive DC brush and brushless DC tuning
  - Device window for configuring PMD card and digital drive products
  - Project window for accessing motion resources and connections
  - Ability to save and load current settings
  - Distance and time units conversion
  - Motion network setup & management tools

Pro-Motion is described in more detail in the *Pro-Motion® User's Guid*e.

1.3.2 C-Motion

C-Motion provides a convenient set of callable routines for controlling the Magellan Motion Processor, whether running on a separate host computer such as a PC, or running on the CME device in the C-Motion Engine. C-Motion, which is provided as source code, includes the following features:

- Magellan axis virtualization
- Ability to communicate to multiple PMD motion cards or modules
- Ability to communicate via PCI bus, PC/104 bus, serial, CANbus, or Ethernet
- Provided as source code, allowing easy compilation & porting to various run-time environments including PC, microprocessor, embedded card, or C-Motion Engine
- Can be easily linked to any C/C++ application

C-Motion is described in more detail in the *Magellan® Motion Processor Programmer's Command Reference.*

1.3.3 VB Motion

VB-Motion provides a complete set of methods and properties for developing applications in Visual Basic and other .NET languages. VB-Motion includes the following features:

- Magellan axis virtualization
- Ability to communicate to multiple PMD motion cards or Modules
- Ability to communicate via PCI bus, PC/104 bus, serial, CANbus, or Ethernet
- Provided as a DLL and VB source code for easy porting onto various PC environments

VB Motion is documented in the *PMD REsource Access Protocol Programmer's Reference.*

1.3.4 C-Motion Engine Development Tools

The C-Motion Engine Development Tools includes a source code editor, compiler, linker, and Pro-Motion-based code management & monitoring tools that allow the user to quickly and easily develop applications that will run on a device equipped with a C-Motion Engine. The C-Motion Engine Development Tools includes the following features:

- Complete toolset for creation of user-specific applications running on the motion card
- Open source compiler and C libraries
- Interactive Development Environment
- Supports PCI bus, PC/104 bus, RS232/RS485, CANbus, and Ethernet communications

1.4 Getting Started

The next several sections of this chapter will provide an example code development session that illustrates the basic elements of the CME Development Tools, and provide a foundation for developing your own application for execution on a C-Motion Engine.

This sequence is designed to work with a CME device that has already been connected and initialized to a host PC using the default Pro-Motion communication scheme, and the default console communication scheme for that device type. The following table shows the default Pro-Motion and console connectivity that the device should be set to while using this tutorial, for four types of CME devices:

CME device type	Pro-Motion connectivity to PC	Console connectivity	Power & cabling
Prodigy/CME Stand-Alone	Ethernet	Serial 2	Power, Ethernet, and Serial 2 connectivity provided via cable connections
Prodigy/CME PCI	PCI bus	PCI bus	Card located inside PC, power & connectivity through PCI bus
Prodigy/CME PC/104	Ethernet	Serial 2	Power, Ethernet, and Serial 2 connectivity provided via cable connections
ION/CME	Ethernet	None	Power, Ethernet, and Serial connectivity provided via cable connections

Although it is possible to change both the standard Pro-Motion communication port (the port that will send and receive PRP messages to/from the CME device), and the console port (the destination of printf and similar messages sent from the C-Motion Engine), for simplicity this getting started tutorial will assume the connections indicated in the table above.

Note that for Prodigy/CME Stand-Alone and Prodigy/CME PC/104 cards that were set up using the standard User Guide 'getting started' connection scheme, this means that the Serial 2 port must be connected. This is accomplished using the convenient two-port (Serial 1 and Serial 2) "Y" serial splitter cable provided with the developer's kit version of the cards. For more information on card installation, configuration, and initialization, see the appropriate Prodigy/CME User Guide for the card type that you are using.

1.4.1 Tutorial Sequence Summary

From a host PC that is connected using the communication scheme described above to a properly initialized CME device, here is a summary of the steps that will comprise this C-Motion Engine Development Tools 'getting started' tutorial:

1. Install the C-Motion Engine Development Tools software package on the PC.
2. Invoke the Programmer's Notepad application, load the Hello.prj project, and compile and link this example project.
3. Launch Pro-Motion, and download the Hello project binary file to the CME device's C-Motion Engine.
4. Begin C-Motion Engine code execution, and monitor results on the console window.

Once these steps have been accomplished, the tutorial is complete.

1.5 Software Installation

Before installing the C-Motion Engine Development Tools software, check that your host PC has the following recommended hardware configuration:

- Intel (or compatible) processor, Pentium or better, 300 MB of available disk space, 256MB of available RAM, and a CD-ROM drive. The supported PC operating systems are Windows XP, Windows 7, and Windows Vista.

Two CD-ROMs comprise the software distribution for CME devices. All software applications are designed to work with Windows XP, Windows 7, or Windows Vista.

- Pro-Motion: The Pro-Motion disk is located in its own Pro-Motion box, and contains the software associated with the Pro-Motion Optimized Motion System Development software. This installation procedure will assume that Pro-Motion has already been installed as part of the general card installation and setup as described in the *Prodigy/CME PCI User's Guide* or the *ION/CME User's Manual*. If Pro-Motion is not yet installed on the host PC you will be using for code development, see the User Guide for your card for instructions on installing.
- PMD CME SDK: The PMD CME SDK disk is separate, and contains the C-Motion Engine development tools, the C-Motion source libraries, and VB-Motion Libraries.

To install the PMD CME SDK:

1. Insert the PMD CME SDK disk into the CD-ROM drive of your computer.
   - If autorun is enabled, the installation process will begin when the CD-ROM is inserted.
   - If autorun is not enabled, go to the next step.
2. On the Start menu, click Run.
3. In the Open text box, type D:\setup.exe.

   where D: is the drive letter of your computer's CD-ROM drive.
4. Follow the on-screen prompts to complete the installation process.

Upon completion of the installation process for Pro-Motion and PMD CME SDK, the following components will be installed:

- Programmer's Notepad application
- GNU compiler, linker, C libraries, make utility
- Example code projects
- C-Motion – source code which may be used for developing motion applications in C/C++ based on the Magellan Motion Processor.
- VB-Motion – DLL and example source code which may be used for developing motion applications in Visual Basic based on the Magellan Motion Processor.
- PDF versions of the *Prodigy/CME PCI User's Guide*, *ION/CME User's Manual*, *PMD Resource Access Protocol Programmer's Reference, C-Motion Engine Development Tools, Magellan Motion Processor Programmer's Command Reference*, and *Magellan Motion Processor User's Guide*. The Adobe Acrobat Reader is required for viewing these files. If the Adobe Acrobat Reader is not installed on your computer, it may be freely downloaded from http://www.adobe.com.

1.6 Programmer's Notepad

Programmer's Notepad is the primary application by which user application code development projects are loaded, edited, compiled, and linked.

To continue the tutorial, execute these steps:

1. On the Start menu (in the lower left of the Windows desktop), click All Programs, PMD Prodigy CME SDK, and then C-Motion CME Code Examples.

The Programmer's Notepad starts, with the project group that contains examples. The following Figure 1-1 shows the resultant view.

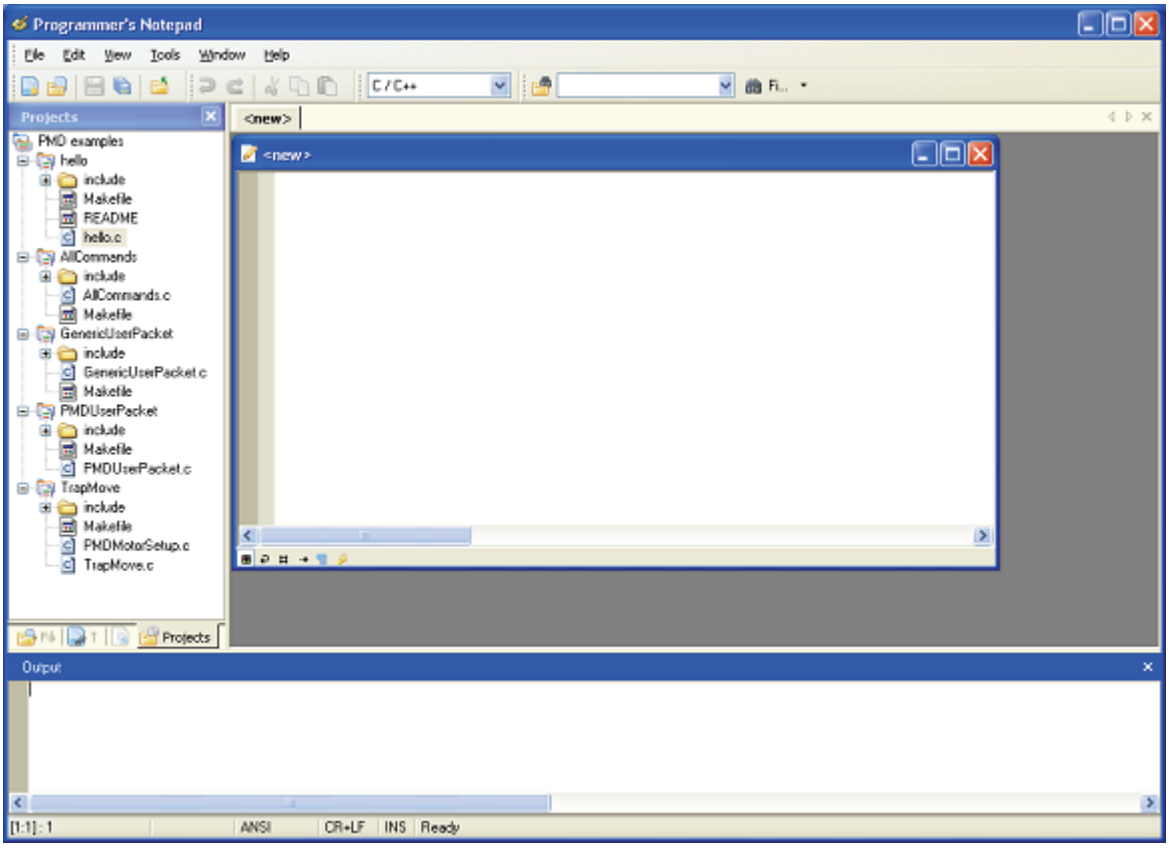


Figure 1-1: Programmer's Notepad Starting Screen

The project pane to the left shows the various available example projects.

2 Right-click the Hello project icon, which is the 'root' icon for the Hello project above the detailed files that comprise the project (Hello.c, etc.), and then in the dialog box that appears, check **Active Project**.

This Hello project is now loaded, and you can double-click specific files to examine them, including README, and Hello.c.

3 For the purpose of this tutorial, double-click Hello.c to display the main() file for this project.

The contents of the C file appears. Figure 1-2 provides an example screen image.

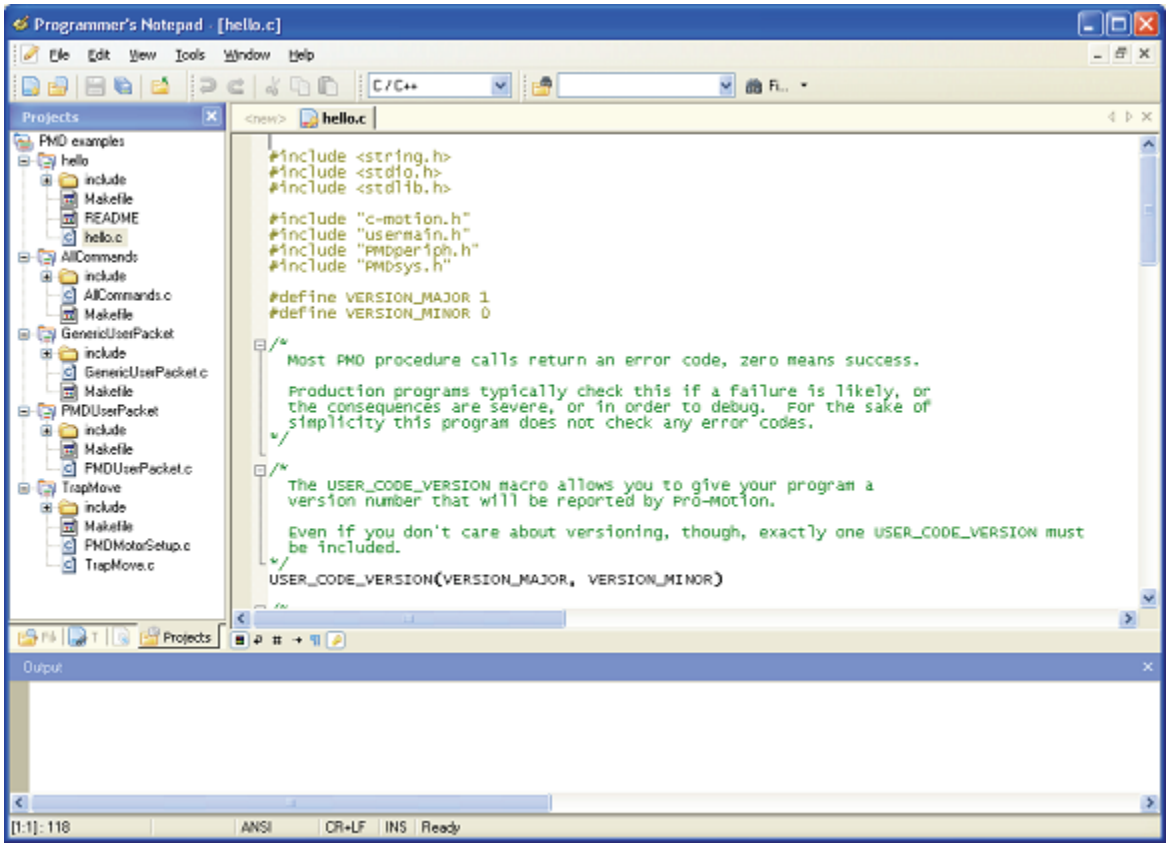


Figure 1-2: Programmer's Notepad with Example Program Shown

Feel free to use the editor controls to scroll through the source code file. Hello.c is a simple application intended for execution on the CME device's C-Motion Engine that:

- Sends "Hello, world" to the console window.
- Communicates to various elements on the card, and sends the version information for the CME device, the C-Motion Engine module, and the card's Magellan Motion Processor, back to the console window.
- Sends the current encoder position of axis #1 of the Magellan Motion Processor to the console window.
- Sends updated values of the actual position, once per second, if the actual encoder position changes.

4 To compile and link the application, use the Programmer's Notepad Tools menu to execute the **Make All** function. This function will compile and link the source code for the entire project.

If the compiling and linking occurs without error, the output window at the bottom of the screen will contain the message 'Process Exit Code: 0' toward the bottom of a list of messages that are output as it is making the binary file image. In addition, a special binary file (also called a bin file) will be created, with the project name and a "**.bin**" extension. So in this case the file will be called "**Hello.bin**". It is this bin file that will be downloaded to the CME device C-Motion Engine.

If one or more error messages occur, double-clicking on a compiler error with a filename and line number will take you to the line where the error is reported. You can try to resolve the error, or call PMD for assistance.

New example projects are added to PMD's CME development tools regularly, so for a complete list of project examples along with brief descriptions, you can use the File menu and the Open function to look at a README file in the project list directory. Most directories within the CME development tools contain a README file which summarizes the contents of that directory.

1.7 Downloading the Code

Once the.bin file has been created, the remaining steps of the development process will occur via Pro-Motion.

A typical development session will involve multiple iterations of code creation, editing, downloading, and execution, so it may be most convenient to put away (minimize) Programmer's Notepad, or arrange it within your screen so that it is available alongside Pro-Motion.

In any case, the following steps are used to continue the tutorial process:

1 Locate the Pro-Motion icon from the PC desktop and launch.

Depending on how Pro-Motion was last used, you may need to click the 'DEV.' (short for development view) icon on the tool bar at the top, right of the application window. Figure 1-3 shows a typical screen view with Pro-Motion in this view mode.

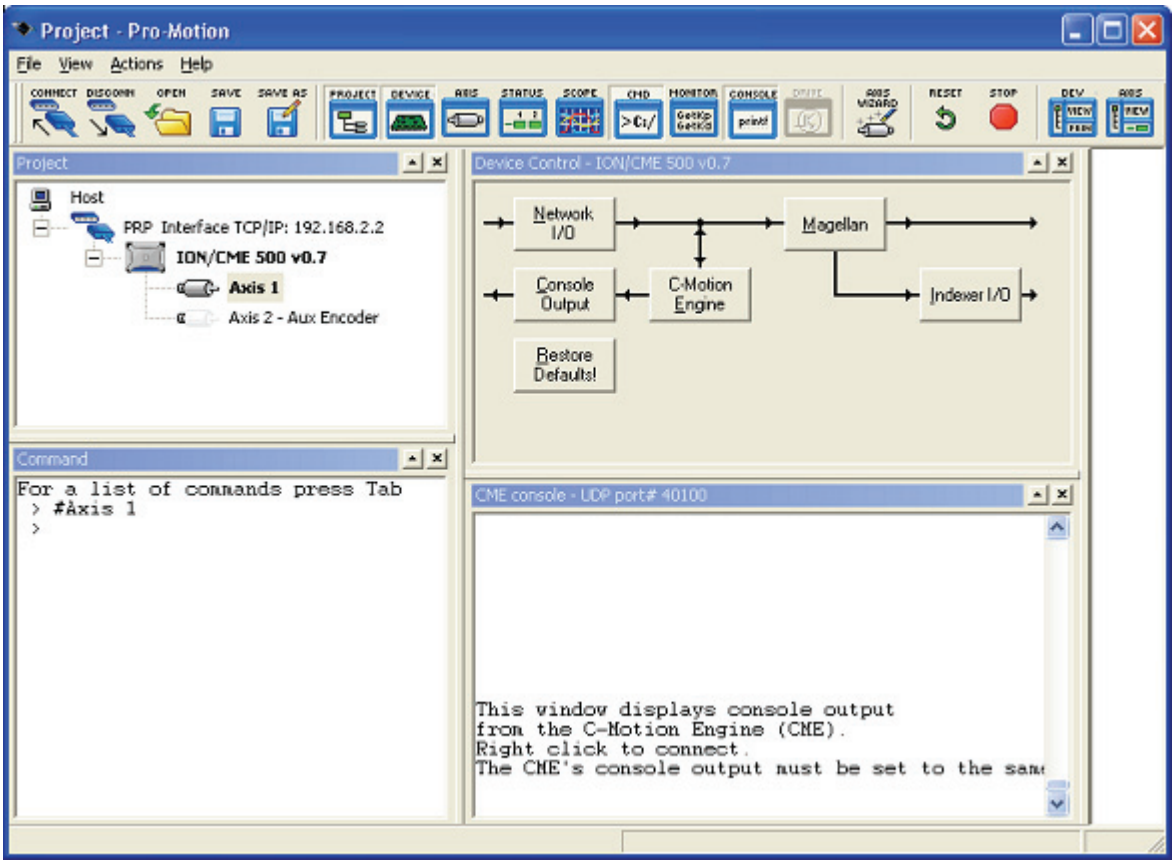


Figure 1-3: Pro-Motion Starting Screen in Dev.View

At the center of the screen you will see the **Device Control** window. This convenient graphical representation of the CME device allows you to manage card-level functions. The **Console** window is to the bottom. This will show messages output from the C-Motion Engine.

- Click the **C-Motion Engine** icon.

  A dialog box as shown in Figure 1-4 appears providing various status and version information for the C-Motion Engine and information on any user application code file (.bin files) that may be loaded into it.

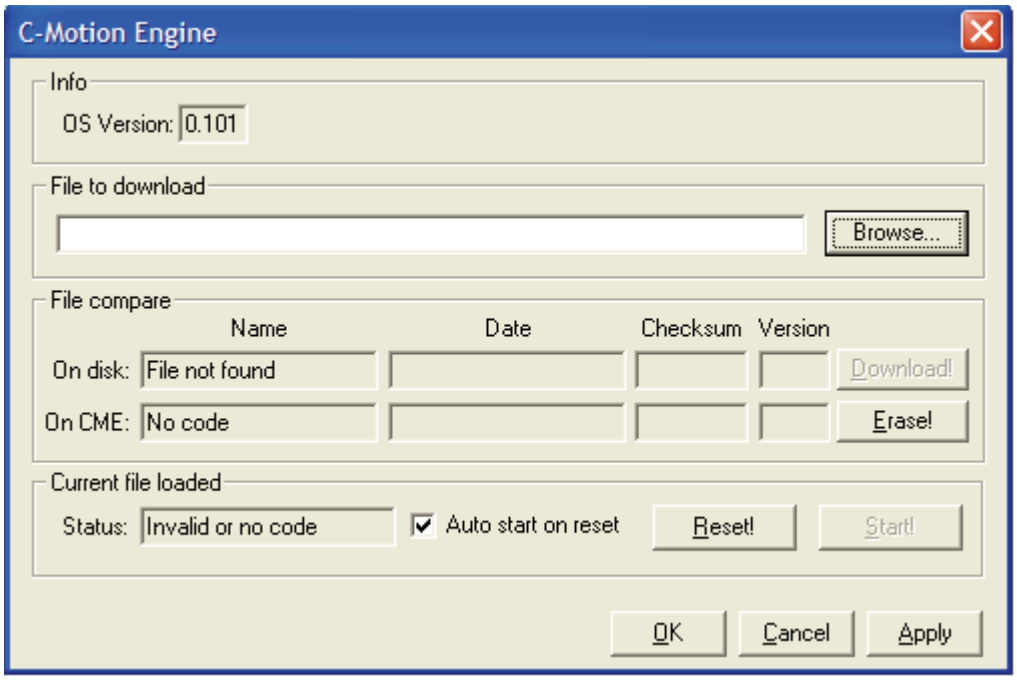


Figure 1-4: C-Motion Engine Dialog Box

2. In the C-Motion Engine window, use the **File To Download** window or the **Browse** button to locate the "**Hello.bin"** file (the examples are typically located in the folder "**C:\My Documents\PMD\CMESDK\CMECode**").
3. Click the **Download** button, and the download (also called Flash) process begins.
4. After a successful flash, the name, user version number, and checksum of the program may be output in the **C-Motion Engine** window.
5. An additional dialog box asks you whether you would like to begin executing the code.

1.8 Executing the Code

Upon successfully downloading the Hello.bin file in the steps above, the code is loaded into the C-Motion Engine and ready to execute. The following instructions complete the process of executing and monitoring code execution on the C-Motion Engine.

1. From step 4 above, click **OK** to begin code execution. As shown in Figure 1-5 you should immediately see a number of console screen messages appear in the window to the bottom of the Pro-Motion screen.

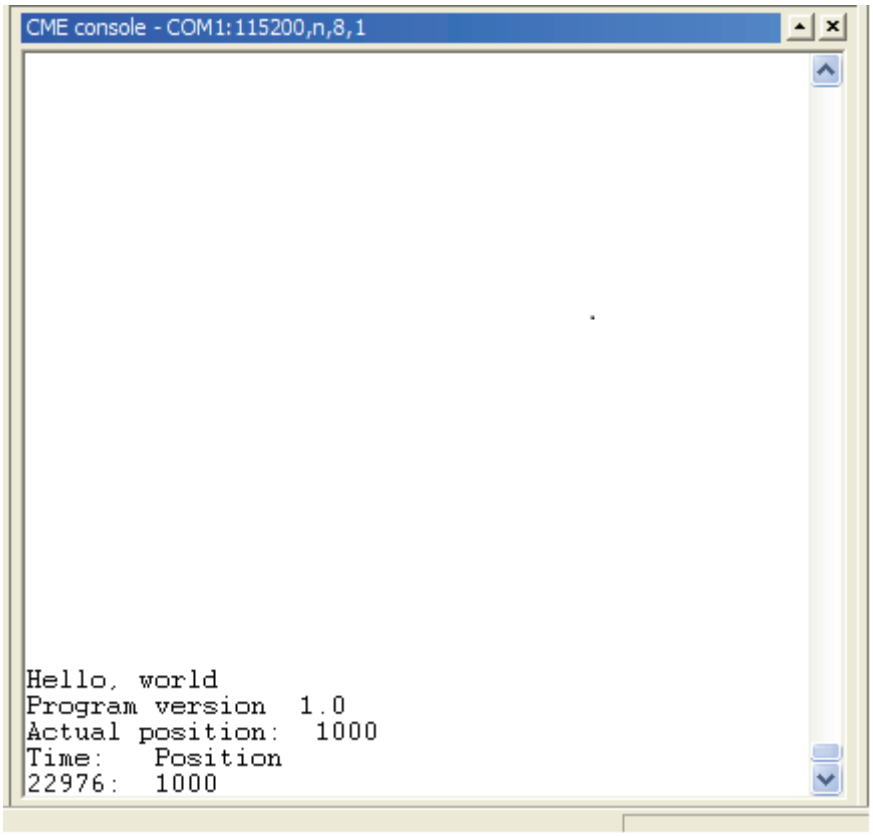


Figure 1-5: Pro-Motion Console Window

These message were sent by the C-Motion Engine to the Console window as it executed the downloaded code. Console messages are special communications that use the standard output stream (stdio) to display message from the C-Motion Engine.

2 If an encoder is connected to axis #1 of your CME device, turn the encoder and updated actual position values will be output to the Console window.

This program continues executing indefinitely until you either stop execution, load another .bin file to the C-Motion Engine, or reset the card.

Congratulations! You have completed the basic 'Hello, world' tutorial and can now continue code development by creating your own projects or expanding on the example projects included with your SDK.

2. Application Development Tools

2.1 Programmer's Notebook

Programmer's Notepad is a utility for editing source code and for running build tools. It is included with the Prodigy/CME developer's kit software and provides a convenient way of working on the C-Motion Engine (CME) user code examples, and also on real CME user code applications. Programmer's Notepad is free software, and may be downloaded from http://www.pnotepad.org.

The use of Programmer's Notepad is not required, all CME code building may be done from the command line, or integrated into some other tool if desired.

2.1.1 Group Organization

Programmer's Notebook relies on the concept of a project to organize logically related software elements. A project is a set of files needed to build a program, organized into a unit. Information about a project is saved in a file with a suffix of ".pnproj." Project files may be opened by clicking Open Project(s) on the File menu, and may be saved by right-clicking on the project name in the Projects pane and clicking Save Project.

Projects may be collected together into a project group, which may be saved as a file with a suffix of ".ppg." To make your own project group, on the File menu, click New, and then Project Group. Projects, either new or existing, may be added to the project group by right clicking the New Project Group name. To save the project group to disk, on the File menu, click Save All; you will be prompted for a file in which to save the project group. To use the project group file, either click Open on the File menu, or start Programmer's Notepad by clicking the .peg file in an Explorer window.

The individual files in a project are listed below its name in the Projects pane. In order to read or edit a file, double click on its name, and a window with the file contents will appear in the large file area on the right side of the Programmer's Notepad window.

2.1.2 Editing

Using Programmer's Notepad to edit a file is similar to using Windows Notepad. The arrow keys, or the slider on the right side of the window, or the page up and down keys may be used to navigate, and text may be inserted at any time by typing. The Edit menu has a variety of choices for searching through text, cutting and pasting, and manipulating text in various ways.

2.1.3 Building

In order to build a project, right-click on the project name in the Projects pane, and select Set Active Project. The project just selected remains "active" until changed; this means that the Tools menu will apply to that project. The Tools menu is used for building a project – to try building, click Make All on the Tools menu, or press the **F7** key.

The pane at the bottom of the window shows the output of the build process, including the process exit code at the end. An exit code of zero means that the build succeeded, and any other value that it failed. Normally in case of failure some useful error messages will be shown in the Output pane. When the error messages are compiler errors with file name and line number information Programmer's Notepad is smart enough to show exactly where the compiler thinks the error occurred; just double-click an error message with a file name and line number and a window showing the location will pop up.

In addition to building projects with Make All, Programmer's Notepad by default can remove all built files by using the Tools menu option Make Clean. By using the Tools menu option, Options, one can get a window showing various configuration choices for the Tools menu. The **Tools** and **Project Tools** names in the lower part of the right hand pane allow one to add new entries to the Tools menu, and to edit the Make All and Make Clean entries, which are found by choosing the Scheme "(None – Global Tools)."

By following the Tools configuration trail, one quickly sees that Make All and Make Clean just call an external program called "**pmdmake.exe,**" with appropriate options. **pmdmake.exe,** is really GNU make, renamed to make it less likely to conflict with already installed programs. The make utility performs the build by calling the compiler, linker, and other utilities in a way dictated by the Makefile. Each of the makefiles for the CME device examples uses the same Makefile in the parent folder, and that Makefile may be used as a starting point for user application projects. Writing Makefiles is beyond the scope of this manual, but is described in the GNU make documentation, which is included.

2.1.4 Example Projects

There are some example projects already included with the C-Motion Engine Development Tools SDK. In order to try some of the example CME programs, run Programmer's Notepad from the Start menu, using the shortcut "**C-Motion CME Code Examples**." This will bring up a window that lists the CME Code Examples. The collection of projects that are CME device examples is saved as the file Examples.ppg.

You should notice that each example includes a file called README, which describes what the example does, any configuration or special hardware that might be required, and so forth.

The examples can be compiled as is, or can be used as the basis of a customized system through further modification.

The examples provided with the C-Motion Development Tools SDK are intended purely for illustration purposes. They are not warranted to be error free, nor are they warranted to be suitable for use in a particular application. It is up to the user to insure the quality of their developed code.

2.2 Pro-Motion

Pro-Motion is a software prototyping and control tool used for all Magellan family products. Most Pro-Motion features are described in the *Pro-Motion User's Guide*, but features used only with CME devices are described here.

When used with a CME device, Pro-Motion provides a Device Control pane, which may be turned on and off with the device icon in the left half of the toolbar. The right half of the toolbar includes a Dev (short for development) icon which may be used to set Pro-Motion to a view that includes the Device Control pane, and other panes useful for CME code development.

2.2.1 The Console

The CME has the notion of a *console,* a peripheral used for displaying messages from C-Motion Engine user programs. This is useful for debugging, reporting progress, and sanity checking.

For Prodigy/CME Stand-Alone and Prodigy/CME PC/104 cards, the second serial channel (Serial 2) is the default output console channel. For Prodigy/CME PCI cards, the PCI bus is the default console output channel. For ION/CME, the single serial port is the default console output channel.

To change where the CME device outputs its console information the Console Output box in the Device Control pane is used. By clicking on this box, you can select one of the options: serial, PCI (for Prodigy/PCI cards), Ethernet (UDP/IP), or none.

Changing these values changes the power up default console settings only, so you must reset the card for these changes to take effect. This can also be accomplished in the Device Control pane.

To enable Pro-Motion displaying messages on the console window, right-click the CME Console window and select the console channel that matches what the card uses for console output.

After reset, if a user program is running, then the console shows the output of any PMDprintf, PMDputs, or PMDputch calls.

2.2.2 Downloading and Running User Programs

The C-Motion Engine in a CME device may be programmed with a single user program at a time, and downloading, sometimes called "flashing" because the program is stored in flash RAM, may be done using the C-Motion Engine box in the Device Control pane.

2.3 Software Development

2.3.1 CME Development Software Libraries

C-Motion Engine user programs may use standard C library and math library calls, which are provided by the Red Hat newlib C library, which is intended specifically for embedded applications. Floating point support is provided in software, so there is a definite performance penalty for floating point arithmetic when compared to fixed point.

Dynamic memory allocation is supported, using the standard malloc and free calls. Approximately 7 kilobytes of heap space are available. Because of the possibilities of heap fragmentation and memory leaks in application code it is strongly recommended that dynamic allocation be kept as static as possible, for example by allocating all necessary data structures at initialization time.

Standard I/O functions are supported only for the standard output streams stdout and stderr, which are connected to the console. Because there are no file systems available to CME device user programs there is no way to open a file.

2.3.2 PMD Software Libraries

Almost all operations performed by C-Motion Engine user programs that deal with physical devices and resources are accomplished using PMD C language libraries.

It may not be obvious at the outset of development exactly which parts of an application should run in the CME and which on a host computer. In order to make it simpler to change this division of labor, and to allow prototyping of CME code on a PC, almost all of the PMD library call sequences are the same in both host-based and C-Motion Engine environments, although the internal details of implementation and data structures differ substantially.

There are several categories of physical device that CME user programs may deal with:

- Magellan Motion Processors, either the CME device processor, or Magellan attached devices, such as non-CME ION modules or non-CME Prodigy cards. These are controlled using the PMD C-Motion library, which is documented in the *Magellan Motion Processor Programmer's Reference*. C-Motion uses a representation of a Motion Processor control axis called an axis handle for all commands. Although the means of obtaining the axis handle are different for the various Magellan devices, the use of the axis handle once obtained is identical.
- Peripheral connections, meaning communication links to off-card devices, using serial, Ethernet, CANBus, or PC/104 bus channels. Peripheral handling is supported by a PMD C language library which is used only with CME devices and documented in the *PMD Resource Access Protocol Programmer's Guide*.

Peripheral connections are represented by a simple peripheral handle object, which may be read or written. There are special procedures for using peripheral connections to address Magellan attached devices or other CME devices.

- Other on-card resources, such as dual-ported RAM, and general purpose digital I/O pins, are accessed either by using the Prodigy/CME library or C-Motion calls for non-Magellan resources documented in the *Magellan Motion Processor Programmer's Reference.*

2.3.3 User Code Architecture

A CME program is normally in the form of a non-terminating loop, because it must continue doing its job as long as the CME device is powered. If it is necessary to exit because of an error condition the PMDAbort procedure should be used.

The C-Motion Engine is a specialized computer that does almost nothing but motion control. This means that a CME user program may freely use, and rely upon, almost all the resources (memory, processor cycles) available to the CME, but it also means that the programmer is responsible for not exceeding the limits on any resource. These resources include:

- Stack space – deeply nested procedure calls, and large data structures in auto storage should be avoided.
- Processor cycles – the developer should use timing procedures such as PMDGetTickCount to verify that each part of an application can run within its time budget. Handling peripheral input and output also requires processor cycles, so heavy network traffic, particularly Ethernet traffic, may overload the processors.
- Data space plus heap space is limited to approximately 7 kilobytes.
- Heap space is limited to approximately 7 kilobytes. It is suggested that all dynamic allocation be done in the initialization phase of a user program, so that unexpected allocation failures due to memory leaks or heap fragmentation do not cause problems.
- Code space is limited to approximately 256 kilobytes. Fortunately code size limitations can be found at build time. One of the best ways of reducing code size is to factor code so that the same procedure performs its task in several places. Cut and paste coding bloats code size.

CME programs require a small amount of boiler plate, specifically the USER_CODE_VERSION and USER_CODE_TASK macros. USER_CODE_VERSION is used to specify major and minor version numbers for the user program, but is required even if you don't care about versioning. It must be used once, at top level (outside any procedure body). USER_CODE_TASK is used to define the procedure used as the main entry point in the user program. A skeleton user program might look like this:

```
#include "usermain.h"        // user program macros, (required).
#include "C-Motion.h"       // C-Motion library calls, for Motion Processor control
#include "PMDsys.h"         // Timing, program control, and console procedures.

#define MAJOR_VERSION 0 // These are for the user, and may be freely chosen.
#define MINOR_VERSION 9

USER_CODE_VERSION(MAJOR_VERSION, MINOR_VERSION) // Required

USER_CODE_TASK(myjob)
```

```
{
   // Get ready to do something useful here.
    while (!0) { …
        // Do something useful here.
  }
}
```

The first tool for discovering what your user program is doing, and why it might be failing, is the console. Information on progress, status, and debugging values may be printed using **PMDprintf**, **PMDputs**, and **PMDputch**. Remember that **PMDprintf** may not be used to format floating point numbers, instead **sprintf** using a temporary buffer must be used.

Another useful diagnostic tool is the set of general purpose digital I/O pins, which may be read or written using the C-Motion **ReadIO** and **WriteIO** procedures as documented in the *Prodigy/CME Stand-Alone User's Guide, Prodigy/CME PCI User's Guide*, *Prodigy/CME PC/104 User's Guide or ION/CME User's Manual*. With the use of an oscilloscope these pins can send diagnostic output much faster than the console procedures.

CME user programs may cause processor exceptions, for example, by reading from or writing to illegal addresses, or by trying to execute illegal instructions, probably because of stack corruption or bad function pointers. An exception will cause a reset of the entire CME device, because it is impossible to know what data structures were corrupted before the exception occurred. After the reset some diagnostic information, including the code address of the exception, register values at the time, and some stack contents, will be printed to the console. This information may be useful in determining where the program went wrong.

A C-Motion Engine user program has a single thread of control; threading or interrupt handling are not available. Because of this restriction CME user programs have a very simple model, and semaphores or other locking mechanisms are not used, which means that almost everything done by a user program must be driven by polled input, with or without timeouts. The recommended way of structuring a CME user program is as a state machine, which uses one or more variables to represent an enumerated set of states, and transitions from one state to the next in a way determined by the inputs read in each state.

This page intentionally left blank.

Appendix A. – Copyright Notices

A.1 Overview

The following sections provide required copyright notifications for the open source software libraries, LWIP (Lightweight TCP/IP Stack), and Newlib, that PMD uses as part of its C-Motion Engine Development Tools. If you have any questions on these copyright notices please contact PMD for more information.

A.2 Lightweight TCP/IP Stack (LWIP)

lwIP is licensed under the BSD license:

A.3 Newlib

The newlib subdirectory is a collection of software from several sources.

Each file may have its own copyright/license that is embedded in the source
file. Unless otherwise noted in the body of the source file(s), the following copyright
notices will apply to the contents of the newlib subdirectory:

(1) Red Hat Incorporated

(2) University of California, Berkeley

INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

(3) David M. Gay (AT&T 1991, Lucent 1998)

The author of this software is David M. Gay.

The author of this software is David M. Gay.

SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER
IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,
ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF
THIS SOFTWARE.

(4) Advanced Micro Devices

So that all may benefit from your experience, please report any problems
or suggestions about this software to the 29K Technical Support Center at
800-29-29-AMD (800-292-9263) in the USA, or 0800-89-1131 in the UK, or
0031-11-1129 in Japan, toll free. The direct dial number is 512-462-4118.

Advanced Micro Devices, Inc.
29K Support Products
Mail Stop 573
5900 E. Ben White Blvd.
Austin, TX 78741
800-292-9263

(5) C.W. Sandmann

(6) Eric Backus

(7) Sun Microsystems

(8) Hewlett Packard

(9) Hans-Peter Nilsson

(10) Stephane Carrez (m68hc11-elf/m68hc12-elf targets only)

(11) Christopher G. Demetriou

(12) SuperH, Inc.

SuperH, Inc.
405 River Oaks Parkway
San Jose
CA 95134
USA

(13) Royal Institute of Technology

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

3. Neither the name of KTH nor the names of its contributors may be
   used to endorse or promote products derived from this software without
   specific prior written permission.

THIS SOFTWARE IS PROVIDED BY KTH AND ITS CONTRIBUTORS ``AS IS'' AND ANY
EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL KTH OR ITS CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(14) Alexey Zelkin

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(15) Andrey A. Chernov

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(16) FreeBSD

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(17) S. L. Moshier

Author: S. L. Moshier.

Copyright (c) 1984,2000 S.L. Moshier

Permission to use, copy, modify, and distribute this software for any
purpose without fee is hereby granted, provided that this entire notice
is included in all copies of any software which is or includes a copy
or modification of this software and in all copies of the supporting
documentation for such software.

THIS SOFTWARE IS BEING PROVIDED "AS IS", WITHOUT ANY EXPRESS OR IMPLIED
WARRANTY. IN PARTICULAR, THE AUTHOR MAKES NO REPRESENTATION
OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY OF THIS
SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

(18) Citrus Project

Copyright (c)1999 Citrus Project,
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(19) Todd C. Miller

Copyright (c) 1998 Todd C. Miller <Todd.Miller@courtesan.com>
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products
   derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL
THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(20) DJ Delorie (i386)
Copyright (C) 1991 DJ Delorie
All rights reserved.

Redistribution and use in source and binary forms is permitted
provided that the above copyright notice and following paragraph are
duplicated in all such forms.

This file is distributed WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

(21) Mike Barcroft

(22) Konstantin Chuguev (--enable-newlib-iconv)

HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

iconv (Charset Conversion Library) v2.0

(23) Artem Bityuckiy (--enable-newlib-iconv)

Copyright (c) 2003, Artem B. Bityuckiy, SoftMine Corporation.
Rights transferred to Franklin Electronic Publishers.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(24) - Alex Tatmanjants (targets using libc/posix)

Copyright (c) 1995 Alex Tatmanjants <alex@elvisti.kiev.ua>
at Electronni Visti IA, Kiev, Ukraine.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(25) - M. Warner Losh (targets using libc/posix)

Copyright (c) 1998, M. Warner Losh <imp@freebsd.org>
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(26) - Andrey A. Chernov (targets using libc/posix)

Copyright (C) 1996 by Andrey A. Chernov, Moscow, Russia.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

(27) - Daniel Eischen (targets using libc/posix)

Copyright (c) 2001 Daniel Eischen <deischen@FreeBSD.org>.
All rights reserved.

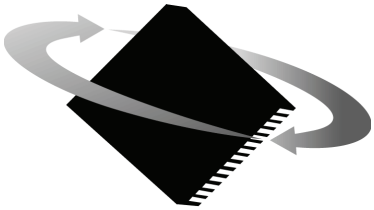
Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

For additional information, or for technical assistance, please contact PMD at (978) 266-1210.

You may also e-mail your request to support@pmdcorp.com

Visit our website at http://www.pmdcorp.com



P M D

Performance Motion Devices
80 Central Street.
Boxborough, MA 01719