AMX®

AMX™ CFire Tool Guide

First Printing: June 1, 1999
Last Printing: November 1, 2007

KADAK Products Ltd.
206 - 1847 West Broadway Avenue
Vancouver, BC, Canada, V6J 1Y5
Phone: (604) 734-2796
Fax: (604) 734-8114

TECHNICAL SUPPORT

KADAK Products Ltd. is committed to technical support for its software products. Our programs are designed to be easily incorporated in your systems and every effort has been made to eliminate errors.

Engineering Change Notices (ECNs) are provided periodically to repair faults or to improve performance. You will automatically receive these updates during the product's initial support period. For technical support beyond the initial period, you must purchase a Technical Support Subscription. Contact KADAK for details. Please keep us informed of the primary user in your company to whom update notices and other pertinent information should be directed.

Should you require direct technical assistance in your use of this KADAK software product, engineering support is available by telephone, fax or e-mail. KADAK reserves the right to charge for technical support services which it deems to be beyond the normal scope of technical support.

We would be pleased to receive your comments and suggestions concerning this product and its documentation. Your feedback helps in the continuing product evolution.

KADAK Products Ltd.
206 - 1847 West Broadway Avenue
Vancouver, BC, Canada, V6J 1Y5

Phone: (604) 734-2796
Fax: (604) 734-8114
e-mail: amxtech@kadak.com

DISCLAIMER

KADAK Products Ltd. makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability and fitness for any particular purpose. Further, KADAK Products Ltd. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of KADAK Products Ltd. to notify any person of such revision or changes.

TRADEMARKS

AMX CFire TOOL GUIDE

Table of Contents

This page left blank intentionally.

1. Selecting a Tool Set

Available Toolsets

AMX™ CFire and the *KwikLook*™ Fault Finder have been developed on a PC with Microsoft® Windows® using the software development tools described in this guide.

To simplify the selection process, KADAK has prepared this Tool Guide. This chapter introduces the tools and defines the subsets which KADAK has used with success. Subsequent chapters provide specific guidelines for using each of the supported toolset combinations with AMX CFire.

Note that AMX CFire is delivered to you ready to use with each of the supported toolsets. Should you wish to rebuild the AMX CFire Library for any reason, follow the construction guidelines provided in Appendix D of the AMX User's Guide.

To construct your embedded application, you will require a C or C++ compiler, an assembler, a librarian (optional), a linker and/or locator and a remote debugger. The vendors listed below provide these tools. The tool name listed is the vendor's product name or the name of the executable program used to run the tool. The tool name listed will be used throughout this manual to reference the specific tool from a particular vendor.

Vendor	**C/C++**	**Assembler**	**Librarian**	**Linker**	**Locator**	**Debugger**
Diab-SDS	*DCC*	*DCC*	*DAR*	*DCC*		*SingleStep*
Metrowerks	*MWCCE68K*	*MWASME68K*	*MWLDE68K*	*MWLDE68K*		*CodeWarrior*
Metrowerks	*MWCCMCF*	*MWASMMCF*	*MWLDMCF*	*MWLDMCF*		*CodeWarrior*

Supported Toolsets

Unfortunately you cannot arbitrarily use any combination of the listed tools. Of all the tools listed, KADAK has identified several combinations which can be used with AMX CFire. The supported toolsets are divided into major classes according to the C/C++ compiler vendor and then, if necessary, into sub-classes, one for each locator and/or debugger.

Each supported toolset is given a three character mnemonic called a **toolset id** which is used by KADAK to identify the toolset combination. The first two characters of the mnemonic identify the compiler vendor. The third character, if needed, identifies the locator and/or debugger used.

	Compiler
DA	Diab-SDS, Inc. C/C++
ME	Metrowerks Inc. C/C++

The following toolset combinations are supported by KADAK.

Toolset id:	**DA**	**ME**	**ME**
Vendor:	Diab-SDS	Metrowerks	Metrowerks
C/C++	*DCC*	*MWCCE68K*	*MWCCMCF*
Assembler	*DCC*	*MWASME68K*	*MWASMMCF*
Librarian	*DAR*	*MWLDE68K*	*MWLDMCF*
Linker/ Locator	*DCC*	*MWLDE68K*	*MWLDMCF*
Debugger	*SingleStep*	*CodeWarrior*	*CodeWarrior*

2. Diab-SDS (DA) Tool Guide

AMX™ CFire has been developed on a PC with Windows® NT v4.0 using the Diab-SDS tools listed below. The AMX libraries and object modules on the product disks have been generated using the most recent tools listed. If you are not using this toolset, you may have to rebuild the AMX libraries in order to use your out-of-date tools.

Diab-SDS Tools		**v4.3**	**v4.3A**	**v4.4A**	**v5.0**	**v5.1**	**v5.2**
DCC	ColdFire C/C++ compiler	4.3	4.3A	4.4A	5.0a	5.1.1	5.2.1
DAS	ColdFire assembler	4.3	4.3A	4.4A	5.0a	5.1.1	5.2.1
DAR	ColdFire librarian	4.3	4.3A	4.4A	5.0a	5.1.1	5.2.1
DLD	ColdFire linker	4.3	4.3A	4.4A	5.0a	5.1.1	5.2.1
DDUMP	ColdFire locator	4.3	4.3A	4.4A	5.0a	5.1.1	5.2.1
	SingleStep ColdFire Debugger SingleStep Target Monitor	7.4	7.4	7.6	7.6	7.6	7.6

AMX CFire and *KwikLook* have been tested on the following platforms.

Cadre III M5206EC3 board
Arnewsh SBC5307 board
Motorola M5272C3 board
Motorola M5249C3 board

Motorola M5407C3 board
Motorola M5282EVB board
Motorola M5475EVB board
Motorola M5271EVB board

Environment Variables

Set the following environment variables to provide access to all AMX and Diab-SDS tools, header files, object files and libraries.

	CJPATH	Path to AMX installation directory (*...\AMX512*)
	PATH	Path to AMX and Diab-SDS executable programs
	TMPDIR	Path to a temporary directory for use by Diab-SDS tools
	DIABLIB	Path to Diab-SDS installation directory
	DTARGET	Target ColdFire processor
	DOBJECT=F	Generate ELF object format
	DFP=S	Assume software floating point emulation
or	*DFP=H*	Assume hardware floating point
or	*DFP=N*	Assume no floating point
	DENVIRON=cross	Compiler cross-compiles on PC for ColdFire

The AMX libraries have been constructed using the following Diab-SDS parameters. The resulting AMX CFire libraries are ready for use with all ColdFire implementations.

DOBJECT=F	AMX CFire is generated in ELF object format
DTARGET=MCF5200	AMX CFire target processor is MCF5200
DFP=N	Assume no floating point

Object Formats

The AMX CFire libraries and object modules are provided in ELF format. Your object modules and the AMX and Diab-SDS libraries and object modules, all in ELF format, can be combined to create an executable module in ELF format suitable for use with the Diab-SDS SingleStep Debugger.

Parameter Passing Conventions

AMX CFire uses the stack based parameter passing convention common to all toolsets supported by KADAK.

Register Usage

The Diab-SDS version of AMX makes the following C interface register assumptions. Registers *D0*, *D1*, *A0* and *A1* can always be altered by C procedures. Integers and pointers are returned from C procedures in register *D0*. No register is dedicated for global data access. You must NOT use any C compilation switch which changes these register assumptions.

Using the Diab-SDS C Compiler

All AMX header files `CJ512xxx.H` and the generic AMX include file `CJZZZ.H` must be present in the current directory together with your source file being compiled.

Use the following compilation switches when you are compiling modules for use in the AMX environment.

by default	; no stack checking
by default	; output object module `FILENAME.O`
`-@E=`	; redirect C error messages to `FILENAME.ERR`
`-c`	; compile only
`-XO`	; full optimize for speed
`-W1`	; avoid optimization fault in v4.3 and earlier
`-Xkill-reorder=2`	; avoid optimization fault in v4.3a
`-g`	; (optional) generate debug information

For v4.3 and earlier, the compilation command line is of the form:

```
DCC -c -XO -W1 FILENAME.C -@E=FILENAME.ERR
```

For v4.3a, the compilation command line is of the form:

```
DCC -c -XO -Xkill-reorder=2 FILENAME.C -@E=FILENAME.ERR
```

Compiling the AMX System Configuration Module

Your AMX System Configuration Module *SYSCFG.C* is compiled as follows. All AMX header files *CJ512xxx.H* and the generic AMX include file *CJZZZ.H* must be present in the current directory together with file *SYSCFG.C*.

For v4.3 and earlier:

```
DCC -c -XO -W1 SYSCFG.C -@E=SYSCFG.ERR
```

For v4.3a:

```
DCC -c -XO -Xkill-reorder=2 SYSCFG.C -@E=SYSCFG.ERR
```

Assembling the AMX Target Configuration Module

Your AMX Target Configuration Module *HDWCFG.S* is assembled as follows. The generic AMX header file *CJZZZK.DEF* must be present in the current directory together with file *HDWCFG.S*.

The Diab-SDS C command line driver is used to invoke the assembler. Some of the command line switches match those used for C. Others are as follows.

by default	; assemble with case sensitivity
by default	; output object module *HDWCFG.O*
-@E=	; redirect assembler error messages to *HDWCFG.ERR*
-Wa,-x	; omit local symbols from object module
-c	; assemble only

```
DCC -c -Wa,-x HDWCFG.S -@E=HDWCFG.ERR
```

Making Libraries

To make a library from a collection of object modules, create a library specification file *YOURLIB.LBM*. Use the Diab-SDS version of the AMX library specification file *CJ512.LBM* as a guide.

Use the following command line switches when using the Diab-SDS librarian.

YOURLIB.A	; output library module *YOURLIB.A*
>YOURLIB.LBE	; redirect librarian error messages to *YOURLIB.LBE*
-qc	; create a new library; use quick append mode

Make your library as follows.

```
DAR -qc YOURLIB.A -@YOURLIB.LBM >YOURLIB.LBE
```

Linking with the Diab-SDS Linker

When used with Diab-SDS C, the modules which form your AMX system must be linked in the following order.

Your *MAIN* module
Other application modules

```
SYSCFG.O          ; AMX System Configuration Module
HDWCFG.O          ; AMX Target Configuration Module

CHxxxxxT.O        ; AMX chip-specific clock driver or your equivalent

CJ512UF.O         ; Launch and leave AMX (may be customized)

CJ512RAC.O        ; AMX ROM Access Module (customized)
                  ; (only if AMX placed in a separate ROM)
                  ; (see Appendix C in AMX CFire Target Guide)


CJ512CV.A         ; AMX CFire vc Conversion Library
                  ; (only if converting an AMX 86 v3, AMX 386 v1 or
                  ;         AMX 68000 v2 application)

CJ512.A           ; AMX CFire Library
```

Diab-SDS C Runtime Libraries for target hardware

Create a link specification file *YOURLINK.LKS*. Use the Diab-SDS version of the AMX Sample Program link specification file *CJSAMPLE.LKS* as a guide.

Start with the sample link specification file for the board which most closely resembles your hardware configuration.

> Note
>
> If you decide to omit any of the link and locate commands from the sample specification, you may encounter link errors or run-time faults.

Link and locate with the Diab-SDS linker and locator using the following command line switches.

```
-m                 ; create section summary
-Wm                ; no default link command file
-o                 ; direct link output to file YOURLINK.OUT
-@E=               ; direct link error messages to file YOURLINK.LKE
>YOURLINK.MAP      ; direct section summary to file YOURLINK.MAP

-t                 ; create summary of symbol values
-v                 ; inhibit output of .bss section to minimize
                   ; the size of the resulting HEX file.
-R                 ; generate Motorola S-record format
                   ; other formats can be generated
                   ; (see Diab-SDS manual)
-o                 ; direct locate output to file YOURLINK.HEX
>YOURLINK.SYM      ; direct symbol summary to file YOURLINK.SYM
```

The link and locate command lines are therefore of the form:

```
DCC -m -Wm -o YOURLINK.OUT YOURLINK.LKS -@E=YOURLINK.LKE >YOURLINK.MAP
DDUMP -t -v -R -o YOURLINK.HEX YOURLINK.OUT >YOURLINK.SYM
```

The resulting load module *YOURLINK.OUT* is suitable for use with the Diab-SDS SingleStep ColdFire debugger.

The resulting load module *YOURLINK.HEX* is ready for burning into EPROM.

Linking a Separate AMX ROM

AMX can be committed to a separate ROM as described in Appendix C of the AMX Target Guide. Use the AMX Configuration Manager to edit your Target Parameter File *HDWCFG.UP* to define your ROM option parameters. Then use the Manager to generate your ROM Option Module *CJ512ROP.S*, ROM Access Module *CJ512RAC.S* and ROM Option link specification file *CJ512ROP.LKS*.

The ROM Option and ROM Access source modules are assembled as follows.

```
   DCC -c -Wa,-x CJ512ROP.S -@E=CJ512ROP.ERR

   DCC -c -Wa,-x CJ512RAC.S -@E=CJ512RAC.ERR
```

The AMX ROM is linked using link specification file *CJ512ROP.LKS* as follows.

```
   DCC -m -Wm -Ws -Wc -o AMXROM.OUT CJ512ROP.LKS
                                   -@E=AMXROM.LKE >AMXROM.MAP
   DDUMP -t -v -R -o AMXROM.HEX AMXROM.OUT >AMXROM.SYM
```

This example generates file *AMXROM.HEX* in Motorola S-record format suitable for transfer to ROM. Other formats supported by Diab-SDS can be selected with the appropriate command switch.

Note that command line switch *-Ws* is used to prevent loading of the default C startup module. Command line switch *-Wc* is used to prevent loading of the default C runtime library.

When you link your AMX application, be sure to include your customized AMX ROM Access Module *CJ512RAC.O* (created above) in your system link specification file.

Using the AMX Configuration Generator

If you cannot use the AMX Configuration Manager, you may still be able to use the stand-alone AMX Configuration Generator to generate the ROM Option Module *CJ512ROP.S*, ROM Access Module *CJ512RAC.S* and ROM Option link specification file *CJ512ROP.LKS*.

Copy the ROM Option and ROM Access template files *CJ512ROP.CT* and *CJ512RAC.CT* to the current directory. Also copy the ROM Option Link Specification Template file *CJ512ROP.LKT* to the current directory.

Use the AMX Configuration Generator to generate the ROM option source modules as follows.

```
   CJ512CG HDWCFG.UP CJ512ROP.CT CJ512ROP.S
   CJ512CG HDWCFG.UP CJ512RAC.CT CJ512RAC.S
   CJ512CG HDWCFG.UP CJ512ROP.LKT CJ512ROP.LKS
```

Once the ROM option source modules have been created, you can proceed to build your AMX ROM image and your AMX application as described above.

Diab-SDS SingleStep Debugger

The Diab-SDS SingleStep™ ColdFire Debugger supports source level debugging of your AMX CFire system.

The SingleStep Debugger can operate by simulating a ColdFire or by using a BDM or JTAG connection to the ColdFire processor.

SingleStep can also operate using a serial (or other) connection to the target ColdFire system under test. When used in this fashion, you must install the SingleStep Target Monitor in your target hardware. Instructions for doing so are provided in the SingleStep Reference Manual. Your version of the SingleStep Target Monitor must provide a device driver for the serial (or other) device used for communication with the SingleStep Debugger. It is recommended that your driver use polled I/O so that the SingleStep Target Monitor can operate with interrupts disabled.

Using the *KwikLook* Fault Finder

The *KwikLook*™ Fault Finder is compatible with the SingleStep Debugger providing full screen, source level, task-aware debugging from within the Microsoft Windows® environment. *KwikLook* can be invoked directly from the debugger while at breakpoints giving you finger tip access to your application from the AMX perspective. Note that *KwikLook* and SingleStep share a common link to the target system.

3. Metrowerks (ME) Tool Guide

AMX™ CFire has been developed on a PC with Windows® NT v4.0 using the Metrowerks tools listed below. The AMX libraries and object modules on the product disks have been generated using the most recent tools listed. If you are not using this toolset, you may have to rebuild the AMX libraries in order to use your out-of-date tools.

Metrowerks Tools		**v2.5**	**v3.2**	**v4.0**	**v5.1**	**v6.1**
MWCCE68K	ColdFire C/C++ compiler	2.5	3.2			
MWASME68K	ColdFire assembler	2.5	3.2			
MWLDE68K	ColdFire linker/librarian	2.5	3.2			
MWCCMCF	ColdFire C/C++ compiler			4.0	5.1	6.1
MWASMMCF	ColdFire assembler			4.0	5.1	6.1
MWLDMCF	ColdFire linker/librarian			4.0	5.1	6.1
	CodeWarrior IDE					
	CodeWarrior ColdFire Debugger					
	MetroTRK Target Resident Kernel					

AMX CFire and *KwikLook* have been tested on the following platforms.

Cadre III M5206EC3 board
Arnewsh SBC5307 board
Motorola M5272C3 board
Motorola M5249C3 board
Motorola M5407C3 board
Motorola M5282EVB board
Motorola M5475EVB board
Motorola M5271EVB board

Environment Variables

Set the following environment variables to provide access to all AMX and Metrowerks tools, header files, object files and libraries.

CJPATH	Path to AMX installation directory (*...\AMX512*)
PATH	Path to AMX and Metrowerks executable programs
TMPDIR	Path to a temporary directory for use by Metrowerks tools
CWFolder	Path to Metrowerks installation directory
MWCIncludes	Path to Metrowerks include directories
MWLibraries	Path to Metrowerks library directories
MWLibraryFiles	Filenames of Metrowerks C libraries to be searched

Warning!

You must not use Metrowerks v4.0 or later tools with any version of AMX CFire prior to v1.10a. The previous AMX libraries, even if rebuilt, are not compatible with the revised register passing conventions introduced by Metrowerks with their v4.0 release.

Command Line Tools

The Metrowerks CodeWarrior Integrated Development Environment (IDE) provides a software development environment within which you can readily create a project which incorporates AMX. However, the AMX library construction process is independent of the CodeWarrior IDE.

To make the AMX libraries and to construct an AMX application as described in this Tool Guide, you must use the Metrowerks command line tools. It is assumed that the following Metrowerks tools have been copied from the Metrowerks installation directory to the Metrowerks BIN directory and renamed as follows.

For Metrowerks tools prior to v4.0:

...\E68K_Tools\Command_Line_Tools\mwcce68k.exe	to MWCC.EXE
...\E68K_Tools\Command_Line_Tools\mwasme68k.exe	to MWASM.EXE
...\E68K_Tools\Command_Line_Tools\mwlde68k.exe	to MWLD.EXE

For Metrowerks tools v4.0 and later:

...\E68K_Tools\Command_Line_Tools\mwccmcf.exe	to MWCC.EXE
...\E68K_Tools\Command_Line_Tools\mwasmmcf.exe	to MWASM.EXE
...\E68K_Tools\Command_Line_Tools\mwldmcf.exe	to MWLD.EXE

Object Formats

The AMX CFire libraries and object modules are provided in ELF format. Your object modules and the AMX and Metrowerks libraries and object modules, all in ELF format, can be combined to create an executable module in ELF format suitable for use with the Metrowerks CodeWarrior Debugger.

Parameter Passing Conventions

AMX CFire uses the stack based parameter passing convention common to all toolsets supported by KADAK.

Register Usage

The Metrowerks version of AMX makes the following C interface register assumptions. Registers *D0*, *D1*, *D2*, *A0* and *A1* can always be altered by C procedures. For Metrowerks tools prior to v4.0, integers and pointers are returned from C procedures in register *D0*. For Metrowerks v4.0 tools and later, integers and pointers are returned from C procedures in register *A0*. Register *A5* is dedicated for global data access. You must NOT use any C compilation switch which changes these register assumptions.

Using the Metrowerks C Compiler

All AMX header files *CJ512xxx.H* and the generic AMX include file *CJZZZ.H* must be present in the current directory together with your source file being compiled.

Use the following compilation switches when you are compiling modules for use in the AMX environment.

by default	; no stack checking
by default	; target processor is big endian
-c	; compile only
-proc MCF5206e	; generic ColdFire target
-sdata 0	; small mutable data section is empty
-nopic -nopid	; no position independent code or data
-intsize 4	; use 32-bit integers
-model far	; use 32-bit addressing
-Cpp_exceptions off	; disable C++ exceptions
-o	; output object module *FILENAME.O*
>FILENAME.ERR	; redirect C error messages to *FILENAME.ERR*
-Op	; optimize for speed
-g	; (optional) generate debug information

The compilation command line is therefore of the form:

```
MWCC -c -proc MCF5206e -sdata 0 -nopic -nopid
         -intsize 4 -model far -Cpp_exceptions off -Op
         -o FILENAME.O FILENAME.C >FILENAME.ERR
```

If the command line becomes too long, use a command file. For example, create a text file *CCSW.CMD* which contains the following command string.

```
-c -proc MCF5206e -sdata 0 -nopic -nopid
-intsize 4 -model far -Cpp_exceptions off -Op
```

The compilation command line can then reduce to the form:

```
MWCC @CCSW.CMD -o FILENAME.O FILENAME.C >FILENAME.ERR
```

Compiling the AMX System Configuration Module

Your AMX System Configuration Module *SYSCFG.C* is compiled as follows. All AMX header files *CJ512xxx.H* and the generic AMX include file *CJZZZ.H* must be present in the current directory together with file *SYSCFG.C*.

```
MWCC -c -proc MCF5206e -sdata 0 -nopic -nopid
         -intsize 4 -model far -Cpp_exceptions off -Op
         -o SYSCFG.O SYSCFG.C >SYSCFG.ERR
```

Assembling the AMX Target Configuration Module

Your AMX Target Configuration Module *HDWCFG.S* is assembled as follows. The generic AMX header file *CJZZZK.DEF* must be present in the current directory together with file *HDWCFG.S*.

The Metrowerks assembler requires the following command line switches.

by default	; assemble with case sensitivity
by default	; target processor is big endian
by default	; assemble only with generic ColdFire code
-o	; output object module *HDWCFG.O*
>HDWCFG.ERR	; redirect assembler error messages to *HDWCFG.ERR*

The Metrowerks command to invoke the assembler is as follows.

```
MWASM -o HDWCFG.O HDWCFG.S >HDWCFG.ERR
```

Making Libraries

To make a library from a collection of object modules, create a library specification file *YOURLIB.LBM*. Use the Metrowerks version of the AMX library specification file *CJ512.LBM* as a guide.

Use the following command line switches when using the Metrowerks linker/librarian.

-library	; create a library
-o YOURLIB.A	; output library module *YOURLIB.A*
>YOURLIB.LBE	; redirect librarian error messages to *YOURLIB.LBE*

Make your library as follows.

```
MWLD -library -o YOURLIB.A @YOURLIB.LBM >YOURLIB.LBE
```

Linking with the Metrowerks Linker

When used with Metrowerks C, the modules which form your AMX system must be linked in the following order.

Your *MAIN* module
Other application modules

```
SYSCFG.O          ; AMX System Configuration Module
HDWCFG.O          ; AMX Target Configuration Module

CHxxxxxT.O        ; AMX chip-specific clock driver or your equivalent

CJ512UC.O         ; AMX minimal C replacement library
                  ; (used to eliminate sprintf I/O support from C library)

CJ512UF.O         ; Launch and leave AMX (may be customized)

CJ512RAC.O        ; AMX ROM Access Module (customized)
                  ; (only if AMX placed in a separate ROM)
                  ; (see Appendix C in AMX CFire Target Guide)

CJ512CV.A         ; AMX CFire vc Conversion Library
                  ; (only if converting an AMX 86 v3, AMX 386 v1 or
                  ;         AMX 68000 v2 application)

CJ512.A           ; AMX CFire Library
                  ; Metrowerks C Runtime Libraries are automatically loaded
                  ; per environment variable MWLibraryFiles
```

Create a link specification file `YOURLINK.LKS`. Use the Metrowerks version of the AMX Sample Program link specification file `CJSAMPLE.LKS` as a guide.

Create a linker command file `YOURLINK.LCF`. Use the Metrowerks version of the AMX Sample Program linker command file `CJSAMPLE.LCF` as a guide.

Start with the sample link specification file and linker command file for the board which most closely resembles your hardware configuration.

> Note
>
> If you decide to omit any of the link and locate commands from the sample specification, you may encounter link errors or run-time faults.

Link and locate with the Metrowerks linker and locator using the following command line switches.

by default	; target processor is big endian
by default	; main entry point is at symbol `__start`
`-proc MCF5206e`	; generic ColdFire target
`-nopic -nopid`	; no position independednt code or data
`-map unused`	; direct section and symbol summary to
	; file `YOURLINK.OUT.XMAP`
	; include unused symbols in map file output
`-g`	; (optional) add debug information to the output to file
`-srec`	; direct `HEX` output to file `YOURLINK.OUT.S19`
	; generate Motorola S-record format
`-o`	; direct link output to file `YOURLINK.OUT`
	; use linker command file `YOURLINK.LCF`
`@`	; use link specification file `YOURLINK.LKS`
`>`	; direct link error messages to file `YOURLINK.LKE`

To avoid an overlength command line, create a command file `LDSW.CMD`, a text file containing the command line switches which you require.

```
-proc MCF5206e -nopic -nopid -map unused -srec
```

The link and locate command line is then of the form:

```
MWLD @LDSW.CMD -o YOURLINK.OUT
         YOURLINK.LCF @YOURLINK.LKS >YOURLINK.LKE
```

The resulting load module `YOURLINK.OUT` is suitable for use with the Metrowerks CodeWarrior ColdFire debugger.

The resulting load module `YOURLINK.OUT.S19` is ready for burning into EPROM.

Linking a Separate AMX ROM

AMX can be committed to a separate ROM as described in Appendix C of the AMX Target Guide. Use the AMX Configuration Manager to edit your Target Parameter File *HDWCFG.UP* to define your ROM option parameters. Then use the Manager to generate your ROM Option Module *CJ512ROP.S*, ROM Access Module *CJ512RAC.S* and ROM Option linker command file *CJ512ROP.LCF*.

The AMX Configuration Manager must have access to the ROM Option Linker Command Template file *CJ512ROP.LCT*. If you have installed AMX for multiple toolsets, the Manager may not be referencing the Metrowerks toolset directory *TOOLME* for its template files. Go to the File, Templates... menu and, from the list of selectors, choose the selector for the ROM Option Link/Locate File. Adjust the configuration template by browsing for the file *TOOLME\CFG\CJ512ROP.LCT*.

The ROM Option and ROM Access source modules are assembled as follows.

```
MWASM -o CJ512ROP.O CJ512ROP.S >CJ512ROP.ERR

MWASM -o CJ512RAC.O CJ512RAC.S >CJ512RAC.ERR
```

The AMX ROM is linked using linker command file *CJ512ROP.LCF* and link specification file *CJ512ROP.LKS* as follows.

```
MWLD -proc MCF5206e -nopic -nopid -map unused -srec
         -nostdlib -main "_cjksenter" -o AMXROM.OUT
         CJ512ROP.LCF @CJ512ROP.LKS >AMXROM.LKE
```

This example generates file *AMXROM.OUT.S19* in Motorola S-record format suitable for transfer to ROM.

Note that command line switch *-main "_cjksenter"* is used to prevent loading of the default C startup module. Command line switch *-nostdlib* is used to prevent loading of the default C runtime library.

When you link your AMX application, be sure to include your customized AMX ROM Access Module *CJ512RAC.O* (created above) in your system link specification file.

Using the AMX Configuration Generator

If you cannot use the AMX Configuration Manager, you may still be able to use the stand-alone AMX Configuration Generator to generate the ROM Option Module *CJ512ROP.S*, ROM Access Module *CJ512RAC.S* and ROM Option linker command file *CJ512ROP.LCF*.

Copy the ROM Option and ROM Access template files *CJ512ROP.CT* and *CJ512RAC.CT* to the current directory. Also copy the ROM Option Linker Command Template file *CJ512ROP.LCT* to the current directory.

Use the AMX Configuration Generator to generate the ROM option source modules as follows.

```
CJ512CG HDWCFG.UP CJ512ROP.CT CJ512ROP.S
CJ512CG HDWCFG.UP CJ512RAC.CT CJ512RAC.S
CJ512CG HDWCFG.UP CJ512ROP.LCT CJ512ROP.LCF
```

Once the ROM option source modules have been created, you can proceed to build your AMX ROM image and your AMX application as previously described.

Metrowerks CodeWarrior Debugger

The Metrowerks CodeWarrior® ColdFire Debugger supports source level debugging of your AMX CFire system.

The CodeWarrior Debugger can operate by using a BDM connection to the ColdFire target.

The CodeWarrior Debugger can also operate using a serial (or other) connection to the target ColdFire system under test. When used in this fashion, you must install the CodeWarrior *MetroTRK* Target Resident Kernel in your target hardware. Instructions for doing so are provided in the CodeWarrior Reference Manual. Your version of the Target Resident Kernel must provide a device driver for the serial (or other) device used for communication with the CodeWarrior Debugger. It is recommended that your driver use polled I/O so that the Target Resident Kernel can operate with interrupts disabled.

Using the *KwikLook* Fault Finder

The *KwikLook*™ Fault Finder is compatible with the CodeWarrior Debugger providing full screen, source level, task-aware debugging from within the Microsoft Windows® environment. *KwikLook* can be invoked directly from the debugger while at breakpoints giving you finger tip access to your application from the AMX perspective. Note that *KwikLook* and CodeWarrior share a common link to the target system.

This page left blank intentionally.