Movidius
an Intel company

moviTools Intro

Version 00.88.0 / 2017-12-15

Table of Contents

1. Introduction **2**

1.1. Overview. . . . 2

1.2. Supported platforms 2

1.3. moviTools 2

1.4. Conventions used in Tool User Manuals 2

2. Desktop environment prerequisites **4**

3. Documentation **5**

3.1. Overview. . . . 5

4. Directory Structure **6**

5. File Formats **8**

5.1. ELF File 8

5.2. TCL File 8

5.3. MVCMD File 8

5.4. HEX File. . . . 9

Movidius
an Intel company

Copyright and Proprietary Information Notice

Movidius Ltd.
1730 South El Camino Real, Suite 200
San Mateo, CA 94402
http://www.movidius.com/

Movidius
an Intel company

1. Introduction

1.1. Overview

Movidius Tools are a set of tools used to build, run and debug applications for Movidius Myriad processors. A proprietary set of tools, moviTools, are used for the SHAVE processors, debugging and system simulation. These tools are used alongside the gnu sparc-elf toolsuite to create build binaries for the LEON sparc v8 processor, and link a final binary.

1.2. Supported platforms

Currently, the Movidius tools support the following platforms:

- MA2100
- MA2x5x (MA2150, MA2155, MA2450, MA2455)
- MA2x8x (MA2480, MA2485)

1.3. moviTools

moviTools comprise the following set of tools, and are the focus of this documentation package.

- moviCompile Movidius SHAVE Compiler
- moviAsm Movidius Assembler
- moviDebug2 Movidius Debugger v2
- moviDebugServer Movidius Debug Server
- moviSim Movidius Simulator
- moviDump Movidius Object Dumper
- moviConvert Movidius Object Converter
- moviUsbBoot Movidius USB boot tool

1.4. Conventions used in Tool User Manuals

1.4.1. Code text

Source code and command line examples are showing in Courier New font.

1.4.2. Command line specification

The words enclosed by < and > are not actually keywords, but values from a set. For example: <parameter> is one of the parameters

The { } denote a series of possible values, | separates the values of the series. The [ ] denote an

optional parameter.

2. Desktop environment prerequisites

The {emsp} Movidius tools are compatible with the following operating systems:

- Windows 7, 8, 8.1, 10
- Linux: CentOS, Ubuntu

They can be used on other Linux platforms as well, but the aforementioned are considered officially supported.

For using the tools within a Windows environment, several prerequisites must be present:

- Cygwin – 32bit version – required only by the Sparc GCC toolchain and the MDK.
- VS2015 redistributable package – 32-bit version – required by all tools binaries other than the Sparc GCC toolchain. The redistributable package can be downloaded from here:

```
https://www.microsoft.com/en-us/download/details.aspx?id=48145
```

3. Documentation

3.1. Overview

Each tool has its own dedicated manual which is included in same directory as this document. In addition release notes and licensing information are provided in separate documents.

4. Directory Structure

The directory structure of the package is the following:

```
EULA.txt
|
|→ common                          //OS independent files needed for tools
|  |→ moviCompile
|  |  |→ include                    //Includes for moviCompile
|  |  |→ libraries                  //libraries of moviCompile
|  |→ moviDebug
|  |  |→ ddrinit
|  |  |→ MV153FlashWriter.elf
|  |  |→ include
|  |  |  |→ moviDebugDll.h
|  |→ moviEclipse
|  |  |→ GraphDesigner
|  |→ moviSim
|  |  |→ architectures
|  |  |  |→ myriad.xml
|  |  |  |→ myriad2.xml
|  |  |  |→ //other architecture descriptions
|  |  |→ intrinsics
|→ doc                             //Documentation of current release
|  |→ llvmLicense.txt
|  |→ moviAsm.pdf
|  |→ moviCompile.pdf
|  |→ moviConvert.pdf
|  |→ moviDebug2.pdf
|  |→ moviDebugServer.pdf
|  |→ moviDebugTcl.pdf
|  |→ moviEclipse.pdf
|  |→ moviToolsIntroduction.pdf
|  |→ moviSim.pdf
|  |→ moviDump.pdf
|  |→ ReleaseNotes.pdf
|→ examples
|→ linux64
|  |→ bin
|  |  |→ moviAsm                   //Movidius SHAVE assembler
|  |  |→ moviCompile               //Movidius SHAVE compiler
|  |  |→ moviConvert               //Movidius SHAVE compiler
|  |  |→ moviDebug2                //Movidius debugger
|  |  |→ moviDebugServer           //Movidius debug server
|  |  |→ moviDebugTcl
|  |  |→ moviDump
|  |  |→ moviLink                  //Movidius linker
|  |  |→ moviSim                   //Movidius simulator
|  |  |→ moviUsbBoot               //Movidius Usb boot tool
|  |→ drivers
|  |  |→ libFTCJTAG.so
|  |  |→ libftd2xx.so
|  |→ lib
```

```
|  |  |→ moviDebugTcl.so
|  |→ models
|  |  |→ NALDecoder.so
|  |  |→ NALEncoder.so
|  |  |→ leonDll.so
|  |  |→ myriad2ShaveDll.so
|  |  |→ shavesDll.so
|  |  |→ sippDll.so
|  |  |→ timersDll.so
|  |  |→ //other model shared libraries
|  |→ sparc-myriad-elf-4.8.2          //Sparc V8 64bit Linux tolchain
|→ win32                             //Win32 tools
|  |→ bin
|  |  |→ moviAsm.exe                  //Movidius SHAVE assembler
|  |  |→ moviCompile.exe              //Movidius SHAVE compiler
|  |  |→ moviConvert.exe              //Movidius object converter
|  |  |→ moviDebug2.exe               //Movidius debugger
|  |  |→ moviDebugServer.exe          //Movidius debug server
|  |  |→ moviDump.exe                 //Movidius object dumper
|  |  |→ moviSim.exe                  //Movidius simulator
|  |  |→ moviUsbBoot.exe              //Movidius Usb boot tool
|  |→ drivers                         //FTDI JTAG dongle drivers
|  |  |→ FTCJTAG.dll
|  |  |→ FTD2XX.dll
|  |  |→ install                      // Files for windows driver install
|  |→ lib
|  |  |→ libdwarf.dll
|  |  |→ moviDebugDll.dll
|  |  |→ moviDebugTcl.dll
|  |→ models                          //Models needed for moviSim
|  |  |→ NALDecoder.dll
|  |  |→ NALEncoder.dll
|  |  |→ leonDLL.dll
|  |  |→ myriad2ShaveDll.dll
|  |  |→ shavesDll.dll
|  |  |→ sipDll.dll
|  |  |→ timersDll.dll
|  |  |→ // other model shared libraries
|  |→ sparc-myriad-elf-4.8.2          //Sparc V8 win32 toolchain
```

5. File Formats

This section describes in detail each of the file formats used by Movidius Tools:

- <fileName>.elf (Executable and Linkable Format)
- <fileName>.tcl (Tcl script file)
- <fileName>.hex (Hexadecimal File)
- <fileName>.mvcmd (Movidius Command File)
- <fileName>.mvupd (Movidius Update File)

5.1. ELF File

The .elf file format used in tools is used in the build flow:

- by the LEON toolchain as object format for:
  - output of the compilation stage (gcc tool collection)
  - output of the assembly stage (as tool)
  - input for the link stage (ld tool) in the form of:
    - dynamic libraries (.so)
    - static libraries (.a)
    - simple objects (.o)
  - input for the objdump tool
- as an input to moviLink in the final stage of an application containing Leon code
- as output of the moviConvert tool, if the moviConvert specified command line switch (-elf) is present

The full description of the file format and its components can be found at:

```
https://refspecs.linuxbase.org/elf/elf.pdf
```

5.2. TCL File

The .tcl file is a script written in the Tcl language, useful for moviDebug2. The contents of the file can consist of any standard Tcl commands and also moviDebug2 specific commands. See the moviDebug2 user manual for the details of the debugger command set.

5.3. MVCMD File

The mvcmd file is the boot file and it is produced by the moviConvert tool from an input .elf file, and it contains Boot commands. They are interpreted by the code which resides in ROM, and which are used usually to load and start the code which is present in the mvcmd file. The commands which are present in the .mvcmd file can be viewed in a human readable format using the moviDump tool (-mvcmd command line switch).

Example:

The example bellow was produced from an ELF file invoking the moviConvert tool, and then moviDump tool. The content (in human readable format) is:

```
SKIP16 (0xA8)
     Length : 0x00, 0x16
     Data : 0x00, 0x53, 0x69, 0x6D, 0x70, 0x6C, 0x65, 0x43, 0x61, 0x6D, 0x31, 0x35, 0x35,
0x48, 0x44, 0x4D, 0x49, 0x2E, 0x6D, 0x6F, 0x66,0x00
SETBITS32 (0x8E)
     Address: 0x80, 0x03, 0x00, 0xBC
     Data : 0x00, 0x10, 0x80, 0x00
BLOCKCOPY32 (0xA2)
     Dest : 0x90, 0x10, 0x00, 0x00
     Source : 0x80, 0x03, 0x00, 0x00
     Length : 0x00, 0x01
SETBITS32 (0x8E)
     Address: 0x80, 0x03, 0x00, 0x00
BLOCKWRITE32 (0x8A)
     Address: 0x80, 0x00, 0x00, 0x08
     Length : 0x00, 0x04
     Data : 0x80, 0x00, 0x18, 0x43, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00
SKIP16 (0xA8)
     Length : 0x00, 0x0B
     Data : 0x00, 0x2E, 0x73, 0x79, 0x73, 0x2E, 0x74, 0x65, 0x78, 0x74, 0x00
BLOCKWRITE32 (0x8A)
     Address: 0x90, 0x10, 0x00, 0x00
     Length : 0x00, 0xE8
     Data : 0x8B, 0x58, 0x00, 0x00, 0x8D, 0x50, 0x00, 0x00, 0xA0, 0x09, 0x6F, 0xF0, 0x8F,
0x48, 0x00, 0x00, 0xAA, 0x10, 0x00, 0x01, 0x80, 0xA4, 0x20, 0x50, 0x32, 0x80, 0x00, 0x19,
0x80, 0xA4, 0x20, 0x60, 0x03, 0x24, 0x04, 0x00, 0x83, 0xC0, 0x60, 0x40, 0x81, 0x90, 0x00,
...
```

5.4. HEX File

The .hex files are used for the RTL simulation of a real project which has all the sections in an ELF file. The .mvcmd file is produced by the moviConvert tool (see moviConvert specifications for detail).

For an ELF file, the following .hex files are produced:

- a file for each CMX block named:

  <projectName>_<sliceNumber>_<tileNumber>.hex

- where:
  - <projectName> is the name of the input elf file;
  - <sliceNumber> is the number of the slice (in Myriad there are 8 slices)
  - <tileNumber> is the number of the tile (there are 6 tiles for each slice)
- a file for DDR which is named as follows:
  <projectName>_ddr.hex
- a file for LRAM which is named as follows:
  <projectName>_lram.hex

Each file stores the content in ASCII format as follows:

```
@<offset>
<data>
...
```

Where:

- <offset> is the offset in memory of the following data
- <data> is the actual content of the memory expressed in hexadecimal ASCII

Examples:

Example of CMX file content:

```
@00000000
002C28B1002808B1
400102A82300C040
000321C00338260C
```

Example of DDR content:

```
@00000000
01234567
89ABCDEF
```