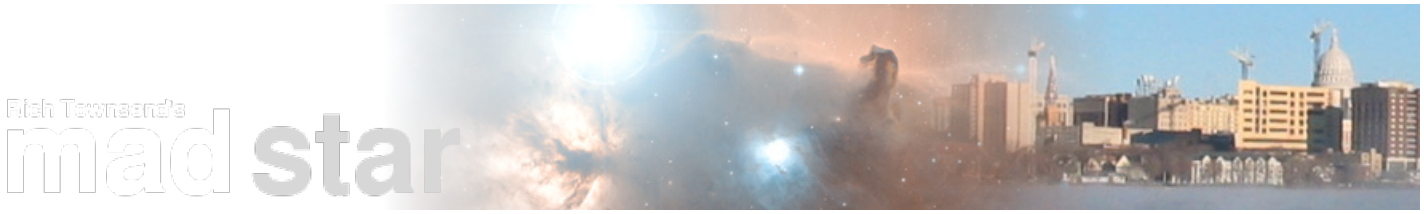

MESA SDK

The MESA SDK is a collection of compilers and run-time libraries that make it *easy* to install and use the MESA stellar evolution code.

Overview

Experience has demonstrated that incompatibilities and bugs in compilers and libraries are significant obstacles in getting MESA up and running with minimal fuss. For instance, MESA makes use of various features in the new(ish) Fortran 2003 standard, which aren't implemented (or are still buggy) in all but the most recent compiler releases.

To help overcome these obstacles, I've put together a unified software development kit (SDK) which contains compilers and libraries *known* to compile MESA correctly. The SDK contains the following components:

- The GNU Comipler Collection (gcc), with support for C, C++, and Fortran
- The GNU Project Debugger (gdb)
- The Valgrind code instrumentation framework
- The Basic Linear Algebra Subprogram (BLAS) library
- The Linear Algebra PACKage (LAPACK) library
- The Hierarchical Data Format v5 (HDF5) library
- The PGPLOT graphics library
- The SE library from the NuGrid project
- The ffmpeg movie encoder
- The ndiff fuzzy comparison tool
- Various helper scripts for use in linking against these libraries and other tasks

Currently, both Linux and Mac OS running on Intel/AMD processors are supported. Although the SDK was initially bundled as part of the standard MESA distribution (from release 3708 onwards), it makes more sense to keep it separate. This page hosts all the necessary information and links to download, install and use the SDK.

Linux (Intel)

Compatibility

The SDK should work on any relatively-recent Linux distribution running on 64-bit Intel-compatible processors (32-bit processors are no longer supported; in any case, MESA itself doesn't work on 32-bit). The GNU C library (also known as GLIBC) included in the distribution must be version 2.5 or more recent; to determine what GLIBC your system uses, run the command `/lib/libc.so.6` (or, possibly, `/lib64/libc.so.6`) and examine the first line of the output for the version number.

Prerequisites

The following components must be installed for the SDK to work on Linux-based systems:

- The 'Binutils' development tools
- The 'Make' dependency/compilation tool
- The 'Perl' scripting language
- The 'X11' windowing library plus development headers
- The 'Z' compression library plus development headers
- The 'C' shell or derivatives

Not all of these components are installed by default on some Linux distributions, and you may have to use the appropriate package management tool (e.g., apt-get, yum, rpm, emerge) to install them. The following table lists the package names of the components (and any other pieces that are required) for some of the more-common distributions:

Package	Fedora / CentOS	Ubuntu	Mint	Gentoo	Arch
Binutils	`binutils`	`binutils`	`binutils`	`sys-devel/binutils`	`binutils`
Make	`make`	`make`	`make`	`sys-devel/make`	`make`
Perl	`perl`	`perl`	`perl`	`dev-lang/perl`	`perl`
X11 library	`libX11`, `libX11-devel`	`libx11-6`, `libx11-dev`	`libx11-dev`	`x11-libs/libX11`	`libx11`
Z library	`zlib`, `zlib-devel`	`zlib1g`, `zlib1g-dev`	`zlib-dev`	`sys-libs/zlib`	`zlib`
C shell	`tcsh`	`tcsh`	`tcsh`	`sys-shells/tcsh`	`tcsh`
Other			`libc6-dev`		`glibc`

If your distribution is not listed here, please contact me and I'll add it to the table.

Download

To download the SDK for Linux, click on the appropriate link in the table:

Release Date MESA Version at Release	File	Notes
April 7 2021 (current) 15140	mesasdk-x86_64-linux-21.4.1.tar.gz	Added fypp preprocessor; enabled float128 support in mpfr
December 18 2020 15140	mesasdk-x86_64-linux-20.12.1.tar.gz	Updated gcc to 10.2
March 25 2020 12778	mesasdk-x86_64-linux-20.3.2.tar.gz	Added ODEPACK library
March 1 2020 12778	mesasdk-x86_64-linux-20.3.1.tar.gz	Added math slots functionality

Note that versions of the SDK older than the current one are not formally supported but are provided here as a courtesy; if you run into problems using an older version, you should first try upgrading to the current version.

Installation

On Linux the SDK can be installed anywhere (this is different from previous releases, where it had to be put in the `/opt` directory). However, for simplicity the following instructions will assume you're installing in your home directory. The steps are as follows:

- Download the package from the table above
- Extract it using the command `tar xvfz` *`package_name`* `-C ~/` (note that's a tilde in front of the slash!)
- Set the path to the SDK:
  - For the C shell: `setenv MESASDK_ROOT ~/mesasdk`

    - For the Bourne shell: `export MESASDK_ROOT=~/mesasdk`
- Initialize the SDK (also checks compatibility):
    - For the C shell: `source $MESASDK_ROOT/bin/mesasdk_init.csh`
    - For the Bourne shell: `source $MESASDK_ROOT/bin/mesasdk_init.sh`
- Check that the SDK is properly installed by running `gfortran --version`. The first line of the output should look something like this:
  `GNU Fortran (GCC) 9.2.0`
  If it *doesn't*, then you should consult the Troubleshooting section below.

Steps 3 and 4 need to be repeated each time you begin a new shell session; alternatively, they can be added to the appropriate shell start-up file (`~/.cshrc` for the C shell, and `~/.profile` for the Bourne shell).

Usage

Nothing special needs to be done in order to use the SDK to build MESA. Simply change into the top-level `mesa` directory (the one created when you download and unpack MESA) and then run `./install`.

Mac OS (Intel)

Compatibility

The SDK should work on any relatively-recent OS X distribution (10.10, Yosemite or later) running on Apple computers with 64-bit Intel-compatible processors (32-bit processors are no longer supported; in any case, MESA itself doesn't work on 32-bit).

It should also work on Apple computers with new M1 (ARM) processors, using the Rosetta 2 translation layer. A native M1 SDK is still under development.

Prerequisites

The following components must already be installed before the SDK can be installed on Mac OS systems:

- The Xcode command-line tools
- The XQuartz X-windows infrastructure

The Xcode command-line tools can be installed by running `xcode-select --install` from a Terminal prompt. (Note that a full installation of the Xcode development environment is not necessary). Likewise, XQuartz can downloaded and installed from here. Note that it is often necessary to *reinstall* the command-line tools and/or XQuartz after upgrading to a new release of OS X.

Download

To download the SDK for Mac OS, click on the appropriate link in the table:

Release Date MESA Version at Release	File	Notes
November 18 2021 (current) 15140	mesasdk-x86_64-macos-21.11.1.pkg	Added support for MacOS 12 (Monterey)
September 10 2021 n/a	mesasdk-x86_64-macos-gcc9-21.9.1.pkg	Special SDK release based on gcc 9.3; intended for compiling older releases of MESA (e.g., 11701 and 12778) on newer versions of Mac OS
April 7 2021 15140	mesasdk-x86_64-macos-21.4.1.pkg	Added fypp preprocessor; enabled float128 support in mpfr
February 4 2021 15140	mesasdk-x86_64-macos-21.2.1.pkg	Added support for MacOS 11.2 (Big Sur)
December 18 2020 15140	mesasdk-x86_64-macos-20.12.2.pkg	Updated to gcc 10.2
December 16 2020 15140	mesasdk-x86_64-macos-20.12.1.pkg	Added support for MacOS 11.1 (Big Sur)
November 19 2020 12778	mesasdk-x86_64-macos-20.11.1.pkg	Added support for MacOS 11.0 (Big Sur)
September 28 2020 12778	mesasdk-x86_64-macos-20.9.1.pkg	Fixed 'library not found for -lSystem' issue, arising on fresh Xcode 12 deployments

April 7 2020 12778	mesasdk-x86_64-macos-20.4.1.pkg	Added md5sum and other checksum utilities from coreutils

Note that versions of the SDK older than the current one are not formally supported but are provided here as a courtesy; if you run into problems using an older version, you should first try upgrading to the current version.

Installation

On Mac OS the SDK is installed into the Applications folder; you will likely need Administrator privileges to do this.

- Download the installer from the table above
- Open it by double clicking on it in the Finder
- Follow the on-screen prompts until the installation completes
- Set the path to the SDK:
  - For the C shell: `setenv MESASDK_ROOT /Applications/mesasdk`
  - For the Bourne shell: `export MESASDK_ROOT=/Applications/mesasdk`
- Initialize the SDK (also checks compatibility):
  - For the C shell: `source $MESASDK_ROOT/bin/mesasdk_init.csh`
  - For the Bourne shell: `source $MESASDK_ROOT/bin/mesasdk_init.sh`
- Check that the SDK is properly installed by running `gfortran --version`. The first line of the output should look something like this:
  `GNU Fortran (GCC) 10.2.0`
  If it *doesn't*, then you should consult the Troubleshooting section below.

Steps 4 and 5 need to be repeated each time you begin a new shell session; alternatively, they can be added to the appropriate shell start-up file (`~/.cshrc` for the C shell, and `~/.bash_profile` for the Bourne shell).

Usage

See the usage instructions above for Linux (they are the same for Mac OS).

Math Slots

It's often convenient to have different versions of the same math libraries available through a single SDK installation — for instance, one optimized for speed, one for accuracy, and so on. To this end, recent releases (>= 20.3.1) of the SDK support a feature known as 'math slots'. By setting the `MESASDK_MATH_SLOT` environment variable, you can chose which set of math liberaries ('slot') gets used at compile time. Possible choices are as follows:

`MESASDK_MATH_SLOT` choice	Description
`default`	Default (no special optimizations or modifications)
`crmath`	Correctly-rounded math, via the crlibm library

If `MESASDK_MATH_SLOT` is not set, then the `default` slot is used. Note that when compiling MESA it isn't necessary to set `MESASDK_MATH_SLOT` yourself; the build scripts take care of it.

Making Movies

The SDK includes the ffmpeg encoder and a simple script, `images_to_movie`, which uses ffmpeg to create movies from PNG files produced by MESA.

To illustrate the script in action, suppose the `&pgstar` section of the MESA inlist file contains the following parameters:

```
⋮
Grid6_file_flag = .true.
Grid6_file_dir = 'png'
Grid6_file_prefix = 'grid6_'
⋮
```

This will make MESA write a sequence of PNG images into the `png` subdirectory, with filenames `grid6_NNNNN.png` (where `N` represents a single digit). To combine these files together into a movie, run the following command from the same directory MESA was run in:

```
images_to_movie 'png/grid6_*.png' movie.mp4
```

This will produce an MPEG4 movie, with the filename `movie.mp4`. The type of movie produced is determined from the file extension. Other choices are possible (e.g., `mpg` for MPEG2), but the `images_to_movie.sh` script is specifically targeted at producing MPEG4 output, so it might be best to stick with the `mp4` extension unless you know what you're doing.

IMPORTANT NOTE: In the example above, the quotes (' ') are necessary to prevent the shell from prematurely expanding the wildcard (*) character.

Installing as Root User

If you wish to install the SDK as the root user, that's fine — but you must run the initialization script at least once as root, before you can use the SDK as an ordinary user. This is because configuration files are written into the `$MESASDK_ROOT/etc` directory on first initialization; and these files must be written as root if you installed as root.

Troubleshooting

If you encounter an error during the build process, please consult the FAQ first. If your problem is not resolved, then you should submit a bug report to the MESA Forum. Be sure to include the following information in your bug report:

- The version of MESA you are trying to build
- The version of the SDK you are using (use the `mesasdk_version.sh` command to determine this)
- Your platform (machine type, operating system, version)

Also, it would be helpful if you could post the output of the following commands:

- `uname -a`
- `gfortran -v`
- `echo $MESASDK_ROOT`
- `echo $PATH`

Frequently Asked Questions (FAQ)

Q: When trying to download the SDK using wget, I get the error

```
403: Forbidden
```

A: Our web server is set up to reject requests from wget. As a workaround, add the flag `--user-agent=""` to your wget invocation.

Q: How can I download the SDK from the command line?

A: Use the wget tool. For instance, to download the Linux version dated `YYYYMMDD`, run

```
wget --user-agent="" http://www.astro.wisc.edu/~townsend/resource/download/mesasdk/mesasdk-x86_64-linux-YYYYMMDD.tar.gz
```

(See the question above for a discussion of why the `--user-agent=""` flag is necessary.)

Q: The initialization script produces errors of the sort

```
mesasdk_init.sh: checking architecture
touch: cannot touch '/opt/mesasdk/etc/check_arch.done': Permission denied
```

A: This is occurring because you installed the SDK as a different user (e.g., root). You must run the initialization script at least once as the installation user, before running as any other user.

Q: When compiling, I'm encountering the error

```
file locking disabled on this file system (use HDF5_USE_FILE_LOCKING environment variable to override)
```

A: To fix this, set the `HDF5_USE_FILE_LOCKING` environment variable to `FALSE`. This can be done via the command

```
export HDF5_USE_FILE_LOCKING=FALSE
```

Q: When compiling MESA on MacOS, I'm encountering the error

```
CC ../private/utils_c_system.c
In file included from /Applications/mesasdk/bin/../sysroot/usr/include/sys/wait.h:110,
                 from /Applications/mesasdk/bin/../sysroot/usr/include/stdlib.h:66,
                 from ../private/utils_c_system.c:26:
/Applications/mesasdk/bin/../sysroot/usr/include/sys/resource.h: In function 'getiopolicy_np':
/Applications/mesasdk/bin/../sysroot/usr/include/sys/resource.h:447:34: error: expected declaration specifiers before
'__OSX_AVAILABLE_STARTING'
  447 | int     getiopolicy_np(int, int) __OSX_AVAILABLE_STARTING(__MAC_10_5, __IPHONE_2_0);
```

A: To fix this, uninstall and then re-install the Xcode command line tools via the commands

```
sudo rm -rf /Library/Developer/CommandLineTools
xcode-select --install
```

Then, re-install the SDK.

Updated 2021-11-18 10:53:53

MESA SDK (old releases)

This page hosts old releases of the MESA SDK. These releases are not supported in any way, but are provided here in case they're needed to compile previous versions of the MESA stellar evolution code. Be sure to consult the main MESA SDK page for a discussion of compatibility issues, prerequisites, etc.

Linux

Release Date MESA Version at Release	File i686 (32-bit)	File x86_64 (64-bit)
August 30 2019 12115		mesasdk-x86_64-linux-20190830.tar.gz
May 3 2019 11701		mesasdk-x86_64-linux-20190503.tar.gz
April 4 2019 11554		mesasdk-x86_64-linux-20190404.tar.gz
March 15 2019 11554		mesasdk-x86_64-linux-20190315.tar.gz
August 22 2018 10398		mesasdk-x86_64-linux-20180822.tar.gz
January 27 2018 10108		mesasdk-x86_64-linux-20180127.tar.gz
November 20 2017 10108		mesasdk-x86_64-linux-20171120.tar.gz
September 21 2017 10000		mesasdk-x86_64-linux-20170921.tar.gz
August 2 2017 9793		mesasdk-x86_64-linux-20170802.tar.gz
January 29 2016 7624		mesasdk-x86_64-linux-20160129.tar.gz
September 8 2015 7624		mesasdk-x86_64-linux-20150908.tar.gz
July 15 2014 6596		mesasdk-x86_64-linux-20140715.tar.gz
February 4 2014 5819		mesasdk-x86_64-linux-20140204.tar.gz
September 13 2013 5329	mesasdk-i686-linux-20130913.tar.gz	mesasdk-x86_64-linux-20130913.tar.gz
April 8 2013 4849	mesasdk-i686-linux-20130408.tar.gz	mesasdk-x86_64-linux-20130408.tar.gz
November 6 2012 4631	mesasdk-i686-linux-20121106.tar.gz	mesasdk-x86_64-linux-20121106.tar.gz

August 9 2012 4298	mesasdk-i686-linux-20120809.tar.gz	mesasdk-x86_64-linux-20120809.tar.gz

Mac OS X (10.12 and later)

IMPORTANT NOTE: On OS X 10.14 (Mojave), you may need to install the development header files. These ship as a standard part of Mojave, but must be installed by hand. To do so, run the command

```
open /Library/Developer/CommandLineTools/Packages/macOS_SDK_headers_for_macOS_10.14.pkg
```

This step may need to be repeated whenever you upgrade to a new release of the Xcode command-line tools.

Release Date MESA Version at Release	File OS X 10.12, Sierra OS X 10.13, High Sierra	File OS X 10.14, Mojave
March 25 2019 12778	mesasdk-x86_64-macos-20.3.2.pkg	
March 1 2019 12778	mesasdk-x86_64-macos-20.3.1.pkg	
November 5 2019 12115	mesasdk-x86_64-macos-19.11.2.pkg	
November 1 2019 12115	mesasdk-x86_64-macos-19.11.1.pkg	
August 30 2019 11701	mesasdk-x86_64-osx-10.10-10.14-20190830.dmg	
May 3 2019 11701	mesasdk-x86_64-osx-10.10-10.14-20190503.dmg	
March 15 2019 11554	mesasdk-x86_64-osx-10.12-10.14-20190315.dmg	
November 4 2018 10398		mesasdk-x86_64-osx-10.14-20181104.dmg
January 27 2018 10108	mesasdk-x86_64-osx-10.12-20180127.dmg	
September 21 2017 10000	mesasdk-x86_64-osx-10.12-20170921.dmg	
August 2 2017 9793	mesasdk-x86_64-osx-10.12-20170802.dmg	

Mac OS X (10.11 and earlier)

Release Date MESA Version at Release	File OS X 10.6, Snow Leopard OS X 10.7, Lion OS X 10.8, Mountain Lion OS X 10.9, Mavericks	File OS X 10.10, Yosemite OS X 10.11, El Capitan
March 25 2019 12778		mesasdk-x86_64-macos-20.3.2.pkg
March 1 2019 12778		mesasdk-x86_64-macos-20.3.1.pkg
November 5 2019 12115		mesasdk-x86_64-macos-19.11.2.pkg
November 1 2019 12115		mesasdk-x86_64-macos-19.11.1.pkg
August 30 2019 11701		mesasdk-x86_64-osx-10.10-10.14-20190830.dmg
May 3 2019 11701		mesasdk-x86_64-osx-10.10-10.14-20190503.dmg
March 15 2019 11554	mesasdk-x86_64-osx-10.12-10.14-20190315.dmg	

November 4 2018 10398		mesasdk-x86_64-osx-10.14-20181104.dmg
January 27 2018 10108	mesasdk-x86_64-osx-10.6-10.9-20180127.dmg	mesasdk-x86_64-osx-10.10-10.11-20180127.dmg
September 21 2017 10000		mesasdk-x86_64-osx-10.10-10.11-20170921.dmg
August 2 2017 9793		mesasdk-x86_64-osx-10.10-10.11-20170802.dmg
April 4 2016 8118	mesasdk-x86_64-osx-10.6-10.9-20160404.dmg	mesasdk-x86_64-osx-10.10-20160404.dmg
August 27 2015 7624	mesasdk-x86_64-osx-10.6-10.9-20150827.dmg	mesasdk-x86_64-osx-10.10-20150827.dmg
February 10 2015 7385	mesasdk-x86_64-osx-10.6-10.9-20150210.dmg	mesasdk-x86_64-osx-10.10-20150210.dmg
October 23 2014 7184		mesasdk-x86_64-osx-10.10-20141023.dmg
July 15 2014 6596	mesasdk-x86_64-darwin-20140715.dmg	
February 4 2014 5819	mesasdk-x86_64-darwin-20140204.dmg	
September 13 2013 5329	mesasdk-x86_64-darwin-20130913.dmg	
April 8 2013 4849	mesasdk-x86_64-darwin-20130408.dmg	
November 6 2012 4631	mesasdk-x86_64-darwin-20121106.dmg	
July 28 2012 4219	mesasdk-x86_64-darwin-20120728.dmg	

Frequently Asked Questions (FAQ) for Old Releases

Q: I'm getting compilation errors of the form:

```
/usr/bin/ld: cannot find -lX11
/usr/bin/ld: cannot find -lz
```

A: Have you properly installed the X windows and Z compression libraries, as specified in the prerequisites?

Q: I'm getting compilation errors of the form:

```
libpng warning: Application built with libpng-1.2.10 but running with 1.5.6
PGPLOT /png: error in libpng while writing file...
```

A: This is a known problem, caused by the pgplot library being compiled with the wrong libpng headers. It was fixed in the 20120727 release of the SDK; if you are using an older release, please upgrade.

Q: I'm getting compilation errors of the form:

```
../private/utils_isnan_okay.f:43.36:

        is_real_inf = (2*x==x .and. x /= 0)
                                     1
Warning: Inequality comparison for REAL(4) at (1)
```

A: This is a known issue with the 20130320 (and later) releases of the SDK, caused by the upgrade to gfortran 4.8.0. To fix, add the flag '`-Wno-compare-reals`' to the end of the definition of the `FCwarn` variable in `mesa/utils/makefile_header`.

Q: I'm getting compilation errors of the form:

```
../private/utils_dict.f:429.41:
```

```
        integer, parameter :: multiplier = 31
                                           1
Warning: Unused parameter 'multiplier' declared at (1)
```

A: This is a known issue with the 20130320 (and later) releases of the SDK, caused by the upgrade to gfortran 4.8.0. To fix, add the flag `'-Wno-unused-parameter'` to the end of the definition of the `FCwarn` variable in `mesa/utils/makefile_header`.

Q: On Ubuntu Linux I encounter these errors during compilation:

```
/usr/bin/ld: cannot find crt1.o: No such file or directory
/usr/bin/ld: cannot find crti.o: No such file or directory
collect2: error: ld returned 1 exit status
```

A: This is a known problem, caused by Ubuntu's use of non-standard installation locations. It was fixed in the 20120727 release of the SDK; if you are using an older release, please upgrade.

Q: On Ubuntu Linux I encounter this error during compilation:

```
/opt/mesasdk/lib/gcc/i686-pc-linux-gnu/4.7.0/include-fixed/features.h:338:25: fatal error: sys/cdefs.h: No such file or
directory compilation terminated.
make: *** [btf_order.o] Error 1
```

A: This is a known problem, caused by Ubuntu's use of non-standard installation locations. It was fixed in the 20120727 release of the SDK; if you are using an older release, please upgrade.

Q: On Red Hat Enterprise Linux (RHEL) I encounter this error during compilation:

```
gfortran: /lib/libc.so.6: version `GLIBC_2.11' not found (required by gfortran)
```

A: This is a known problem, caused by the SDK being compiled with a more-recent version of the GNU C Library (GLIBC) than is installed on RHEL systems. It was fixed in the 20120120 release of the SDK; if you are using an older release, please upgrade.

Q: On OS X I encounter this error during compilation:

```
Re: dyld: unknown required load command 0x80000022
```

A: This problem likely stems from trying to use the SDK on an older version of OS X (10.4 Tiger or 10.5 Leopard), as these have difficulty running 64-bit executables. Please contact Rich Townsend for further support.

Q: On OS X 10.8 I find an error such as

```
dyld: lazy symbol binding failed: Symbol not found: ___emutls_get_address

  Referenced from: /usr/local/lib/libgomp.1.dylib

  Expected in: /usr/lib/libSystem.B.dylib

  This is a run-time error.  It shows up, for example, in the output of a module test such as const/test/tmp.txt.
```

A: Try adding both the `$MESASDK_ROOT/lib` and `/usr/local/lib` directories to your `DYLD_LIBRARY_PATH` environment variable. These should appear first and second, respectively, in the path.

Q: I tried to compile MESA on OS X 10.9, and I got the following error

```
/usr/bin/awk: can't open file ../ndiff/share/lib/ndiff/ndiff-2.00/ndiff.awk
source line number 1 source file ../ndiff/share/lib/ndiff/ndiff-2.00/ndiff.awk
```

A: This means ndiff didn't compile correctly, most likely because the command-line tools weren't installed. See the note in the prerequisites.

Q: On Linux systems gcc can't create executables, producing errors of the sort

```
In file included from test.c:1:0:
  /usr/include/stdio.h:320:43: error: missing binary operator before token "("
   #if defined __USE_XOPEN2K8 || __GLIBC_USE (LIB_EXT2)
```

A: This is occurring because the C header files that ship with the SDK need to be updated to work properly with the system header files. To update the header files, run the following commands after setting `MESASDK_ROOT`:

```
GCC_VERSION=`gcc --version | grep ^gcc | sed 's/^.* //g'`

$MESASDK_ROOT/libexec/gcc/x86_64-pc-linux-gnu/$GCC_VERSION/install-tools/mkheaders $MESASDK_ROOT
```

Updated 2020-12-18 12:34:34
