

ClearSpeed[™]

ClearSpeed 1.3 Patch for Amber 9

Installation guide

Document No: 06-UG-1301 **Revision:** 3.A

September 2008

Table of contents

1	Before you start	4
1.1	Prerequisites	4
1.2	Content of patch	4
2	Installation	5
2.1	Preliminary steps	5
2.2	Installation instructions	5
2.3	Test installation	6
2.4	Using sander.CS	6
2.5	Reversing the ClearSpeed patch	7
Appendix A	Details of scripts provided	8
Appendix B	Files added or modified by the ClearSpeed patch	9
Appendix C	Building card-side code from source	14
Appendix D	Notes on configuration for pgf90	15
Appendix E	Notes on configuration option -openmp	16
Appendix F	Notes on building Amber with MKL version 10.0.1.014	17

1 Before you start

Before installing, read through this Installation Guide to ensure that the installation goes smoothly.

1.1 Prerequisites

To install the Amber patch, you need a system with:

- ClearSpeed runtime version 3.0, installed as `rpm`. Available from the ClearSpeed Support website: <http://support.clearspeed.com>
- Amber 9 source code
- Any of the following compilers:
 - Fortran compiler:
 - Intel Fortran, ifort 9.0, 9.1, 10.0 or 10.1.
 - Portland Group, pgf90 version 6.1-1, 6.1-2, or 7.0-2.
 - gfortran 4.1.1 or higher
- Other Fortran 90 compliant compilers may work, but have not been tested.
- Optional: ClearSpeed SDK version 3.0 installed as an `rpm`. This is only needed to recompile the card-side code. Precompiled binaries are provided in this distribution.

1.2 Content of patch

The contents of `amber9_cs_1_3.tgz` are shown in [Table 1](#).

File name	Description
<code>ClearSpeed_Amber_Installation_Guide.pdf</code>	A PDF version of this Installation Guide.
<code>README.CS</code>	Readme file.
<code>bugfix.all</code>	Amber 9 patches to patch level 41. This is an unaltered copy from the AMBER web site: http://amber.scripps.edu/bugfixes90.html
<code>amber9_CS.patch</code>	ClearSpeed's Amber patches.
<code>patch_amber9_ClearSpeed.bash</code>	Script that patches the Amber 9 distribution, first with the Amber bug fixes, and then with ClearSpeed's modifications
<code>build_amber9_ClearSpeed.bash</code>	Script to build the modified Amber 9 version for Advance™ X620 and the Advance e620 Accelerator cards.
<code>cs_env.bash</code>	Script that sets up part of the environment needed to run ClearSpeed's version of <code>sander</code> . This file should also be sourced before running <code>sander.CS</code> , possibly as part of a login script.

Table 1. Contents of Amber patch

ClearSpeed 1.3 Patch for Amber 9

File name	Description
src/sander/clearspeed/csx/gb.csx	Prebuilt board side binary.
src/sander/clearspeed/csx/ gb6.csx	Prebuilt board side binary.
src/sander/clearspeed/csx/ explicit.csx	Prebuilt board side binary.

Table 1. Contents of Amber patch

2 Installation

This chapter describes how to install the Amber 9 patch.

2.1 Preliminary steps

This patch can be installed on top of your existing Amber 9 installation. The ClearSpeed enabled version of Amber creates a new binary, `sander.CS`.

It may be easier to first try out the patch on a clean copy of Amber 9. If installing on top of an existing Amber 9 installation, it is recommended that you save a copy of the current `src/config.h` configuration file:

```
cd $AMBERHOME/src
cp config.h config.h.orig
```

If anything goes wrong when compiling the host code, the original configuration file will help you find out what went wrong.

The following must be set before running the script `patch_amber9_ClearSpeed.bash`:

- `$AMBERHOME` should point to the top directory in the Amber 9 source tree. If you are using a fresh copy of Amber 9, reset `$AMBERHOME` accordingly.
- A Fortran compiler must be in the path, either `ifort`, `pgf90` or `gfortran`.
- An existing ClearSpeed installation. It is expected to be in `$CSHOME`. `$CSHOME` is usually set to `/opt/clearspeed/csx600_m512_le`, and defined by sourcing `$CSHOME/bin/bashrc`

The following is optional. You only need to set this when compiling the card-side code. Card-side binaries are distributed.

- `$CLEARSP_LICENSE_FILE` should point to the ClearSpeed license server.

2.2 Installation instructions

The following steps are typically run from the account that maintains `AMBER`. A detailed description of the scripts is available in [Appendix A: Details of scripts provided on page 8](#). To install the patch, follow these procedures:

1. Patch the code using the Amber 9 patch and ClearSpeed's Amber patch by moving the `amber9_cs_1_3.tgz` tarball to the Amber home directory, `untar` and running the setup script:

```
mv amber9_cs_1_3.tgz $AMBERHOME
cd $AMBERHOME
tar -zxvf amber9_cs_1_3.tgz
source ./patch_amber9_ClearSpeed.bash
```
2. Configure the build for the Advance X620 or the Advance e620 by adding the `-cs` flag, and optionally the `-openmp` flag, to a serial configuration. Look at the top of an existing `src/config.h` for previously used settings. When comparing the old settings, add

the `-cs` flag and remove any parallel flags (`-mpi`, `-lam`, `-mpich`, `-mpich2`, `-openmpi`, `-scali`)

```
cd src
./configure -cs pgf90 (as an example)
cd ..
```

If you are using `pgf90` on a 64-bit operating system, see [Appendix D: Notes on configuration for pgf90 on page 15](#).

Note that `-openmp` is different to `-openmpi`. See [Appendix E: Notes on configuration option -openmp on page 16](#) for a discussion on the use of OpenMP.

If using MKL version 10.0.1.014 consider if it is possible to upgrade to MKL version 10.0.3.020 or later. If not see [Appendix F: Notes on building Amber with MKL version 10.0.1.014](#) before proceeding. This version of MKL requires static linking of the MKL libraries.

3. Build the new version. You do not need an Advance card to build the software but you will need one for running the tests.

```
./build_amber9_ClearSpeed.bash
```

The build step usually finishes in less than 15 minutes. It is useful to keep the output from the `build_amber9_ClearSpeed.bash` script, for example:

```
./build_amber9_ClearSpeed.bash >& | tee build_log
```

On successful execution of the `build_amber9_ClearSpeed.bash` script, the following files will become available:

```
$AMBERHOME/exe/sander.CS
$AMBERHOME/exe/clearspeed/gb.csx
$AMBERHOME/exe/clearspeed/gb6.csx
$AMBERHOME/exe/clearspeed/explicit.csx
```

2.3 Test installation

Run the tests for the Advance X620 or the Advance e620, to confirm that this build of `sander.CS` uses one of these cards and gives the correct results. This requires that an Advance X620 or the Advance e620 is present, which may not be the case on the build machine.

```
cd test
make test.sander.CS
```

There are ten tests that will finish in approximately 2 minutes and should report that the tests "PASSED".

If there is no card available, an error message will be written to the screen and the test will be run on the host instead.

2.4 Using sander.CS

To set up the ClearSpeed environment and extend paths to include both host and card executables, run the `cs_env.bash` setup script. It may be useful to add this to a login script.

```
${AMBERHOME}/cs_env.bash
```

To use the `sander` executable for the Advance X620 or the Advance e620, use `sander.CS` rather than `sander` as the executable. For example:

```
sander.CS -i mdin -o mdout -p prmtop -c inpcrd
```

`sander.CS` can use more than one Advance X620 or Advance e620 on the host machine. The number of cards and processors are controlled through environment variables. For example, to use four processors on two cards:

```
export CS_NPROCS=4
```

The default setting is to use one card and two processors. The number of cards and processors used is written to the `sander` output file.

If using the `openmp` configuration flag, set the `OMP_NUM_THREADS` environment variable:

```
export OMP_NUM_THREADS=2
```

2.5 Reversing the ClearSpeed patch

The ClearSpeed patches can be reversed with the following command:

```
patch -p1 -R -N -r cs_reverse <CS_amber9.patch
```

This is useful if you are repeating the patch and build process to avoid spurious messages.

Appendix A Details of scripts provided

A.1 The patch_amber9_ClearSpeed.bash script

1. Check that the `$AMBERHOME` environment variable exists. Exit if not found.
2. Go to the `$AMBERHOME` directory.
3. Check that the two patch files `bugfix.all` and `amber9_CS.patch` exists. The former is the `AMBER` patch file, the latter is ClearSpeed's patch to the existing Amber source tree
4. Set up more of the environment. `./cs_env.bash`
5. Apply the Amber 9 patches


```
patch -p0 -N -r patch_rejects <bugfix.all
```
6. If some of the patches have already been applied, warning messages will be seen. They can safely be ignored. For example:


```
Reversed (or previously applied) patch detected! Skipping
patch. 1 out of 1 hunk ignored -- saving rejects to file
patch_rejects patching file test/jar_multi/dist_vs_t.001.save
```
7. Apply ClearSpeed patch of host code


```
patch -p1 -N -r cs_rejects <amber9_CS.patch
```
8. Set executable permission on test scripts


```
chmod 0744 test/cs/lbf7_gb/Run*
chmod 0744 test/cs/jac/Run*
```

A.2 The build_amber9_ClearSpeed.bash script

1. Check that the `$AMBERHOME` environment variable exists. Exit if not found.
2. Go to the `$AMBERHOME` directory.
3. Check that `configure` has been run with `-cs`. Exit if that is not the case.
4. Setup additional environment - this script should be sourced or incorporated by users when using the `sander_CS` executable.


```
source ${AMBERHOME}/cs_env.bash
```
5. Add destination directory for the card-side binaries.
6. The card-side code is distributed as executables. The files are copied to the ClearSpeed subdirectory in `${AMBERHOME}/exe`. For instructions on how to build the executables from source see [Appendix C](#).
7. Remove any remains of previous builds. The `make clean` step does not remove any previously built executables, the build system is set up to be used multiple times with different configurations.


```
make clean
```
8. Build host code. This last step is the time consuming one. It creates a file `sander_CS` in `$AMBERHOME/exe`.


```
make sander_CS
```

Appendix B Files added or modified by the ClearSpeed patch

B.1 Added top level/doc files

Filename	Description
amber9_CS.patch	Patch for ClearSpeed modifications to Amber
Amber9_cs_1_3.tgz	Tar ball containing all ClearSpeed modifications to Amber
bugfix.all	Patch of the normal version of Amber
build_amber9_ClearSpeed.bash	Script to build the sander.CS and associated Advance board binaries
cs_env.bash	Script to set up CS part of environment. Can be run as part of a login script.
patch_amber9_ClearSpeed.bash	Script to patch a clean source tree of Amber 9, first with bugfix.all and then with amber9_CS.patch
README.CS	Short build instructions and list of modified/added files.
doc/ClearSpeed_Amber_Installation_Guide.pdf	Installation guide

Table 2. Files modified by ClearSpeed patch

B.2 Modified source files

Filename	Description
src/Makefile	Added sander.CS target to exclusively build sander.CS. (sander.CS is also built as part of the serial target if configure was run with -cs)
src/configure	Added -cs and -openmp flags and alternative settings for pgf90 to allow a 64-bit built on x86_64 platforms. The -cs flag is for configuring for the Advance X620/e620. The -openmp flag allows the SA part of a GBSA simulation to be overlapped with the nonbonded forces when using the -cs option. No other part of sander uses OpenMP.
src/sander/Makefile	Added target sander.CS
src/sander/def_time.h	Define ClearSpeed specific timers

Table 3. Modified source files

Filename	Description
src/sander/depend	Included the output from <code>makedepend</code> with <code>-cs</code> configuration and the lower optimization level for <code>qm_link_atoms.f</code> .
src/sander/ew_force.f	Use alternative calls to <code>get_nb_energy</code> , and mapping coordinates for adjusted neighbor lists.
src/sander/force.f	Use alternative calls to <code>cs_egb</code> rather than <code>egb</code> , for the two main GB functions.
src/sander/makedepend	Added support for <code>sander.CS</code> .
src/sander/mdread.f	Print ClearSpeed build flag to output file.
src/sander/new_time.f	Setup ClearSpeed specific timers.
src/sander/qm_link_atoms.f	Lowered optimization level so that Amber's test suite pass on 64-bit builds with <code>pgf90</code> .
src/sander/nonbond_list.f	Pack neighbor lists in a format suitable for SIMD.
src/sander/sander.f	Added function call to check if simulation is supported and that at least one Advance X620/e620 is available.

Table 3. Modified source files

B.3 Added files host side

Filename	Description
src/sander/cs_communication.c	Communication layer between host and card(s).
src/sander/cs_egb.f	Layer that converts data structure to suit CS architecture. Handles the intercepted function calls from <code>force.f</code> .
src/sander/cs_explicit.f	Converts neighbor lists and other data structures to suit CS architecture. Handles intercepted calls from <code>ew_force.f</code> and <code>nonbond_list.f</code> .
src/sander/cs_support.f	Check if simulation can be run on the Advance X620 or the Advance e620, call to initial connection to card(s).

Table 4. Files added host side

B.4 Added files card side

Filename	Description
src/sander/clearspeed/FILES	List of files in card side code
src/sander/clearspeed/Makefile	Card side make file
src/sander/clearspeed/cs_explicit.cn	Source code for explicit solvent, port of short_ene.f
src/sander/clearspeed/cs_gb.cn	Source code for GB models 1, 2, 5, and 7, port of egb.f
src/sander/clearspeed/cs_gb.h	Declarations for cs_gb.cn/cs_gb6.cn
src/sander/clearspeed/cs_gb6.cn	Source code for GB model 6
src/sander/clearspeed/cs_gbneck.h	Lookup table used by GB model 7
src/sander/clearspeed/cs_host_board.h	Constants and structs shared between host and board
src/sander/clearspeed/ew_directe.h	Port of ew_directe.h
src/sander/clearspeed/ew_directe_lj.h	Lennard-Jones part of ew_directe.h/ew_directe2.h
src/sander/clearspeed/ew_directp.h	Port of ew_directp.h

Table 5. Files added card side

B.5 Added source files for card side library, contains code for chip-to-chip communication and specialized reduction

Filename	Description
src/sander/clearspeed/csapps_lib/csapps.cn	Declarations of processor id's and counts
src/sander/clearspeed/csapps_lib/csapps.h	Header file
src/sander/clearspeed/csapps_lib/csapps_broadcast.cn	Broadcast data from first to second chip
src/sander/clearspeed/csapps_lib/csapps_broadcast.h	Header file
src/sander/clearspeed/csapps_lib/csapps_machine.cn	Machine specific code, mainly about semaphores
src/sander/clearspeed/csapps_lib/csapps_machine.h	Header file
src/sander/clearspeed/csapps_lib/csapps_memcpy_dma.cn	Chip to chip memory copy code
src/sander/clearspeed/csapps_lib/csapps_memcpy_dma.h	Header file

Table 6. Files added on card side library

Filename	Description
src/sander/clearspeed/csapps_lib/csapps_reduction.cn	Reduce array of doubles from two to one chip
src/sander/clearspeed/csapps_lib/csapps_reduction.h	Header file
src/sander/clearspeed/csapps_lib/csapps_reductionp.mst	Microcode for poly to mono reduction
src/sander/clearspeed/csapps_lib/csapps_reductionp_ucude.inc	Microcode for poly to mono reduction
src/sander/clearspeed/csapps_lib/asm/csapps_reductionp_asm.h	Inline assembler, reducing poly to mono
src/sander/clearspeed/csapps_lib/asm/csapps_reductionp.inc	Assembler, faster swizzling

Table 6. Files added on card side library

B.6 Prebuilt card-side binaries

Filename	Description
src/sander/clearspeed/csx/gb.csx	Card side binary for GB models 1, 2, 5, and 7
src/sander/clearspeed/csx/gb6.csx	Card side binary for GB model 6
src/sander/clearspeed/csx/explicit.csx	Card side binary for explicit solvent simulations

Table 7. Prebuilt card-side binaries

B.7 Modified test files

Filename	Description
test/Makefile	Added section test.sander CS

Table 8. Modified test files

B.8 Added test files

Filename	Description
test/cs/1bf7_gb/1bf7_gb1.out.save	Baseline for GB 1 test
test/cs/1bf7_gb/1bf7_gb5.min.out.save	Baseline for GB 5 minimization test
test/cs/1bf7_gb/1bf7_gb5.out.save	Baseline for GB 5 test

Table 9. Added test files

Filename	Description
test/cs/lbf7_gb/lbf7_gb6.out.save	Baseline for GB 6 test
test/cs/lbf7_gb/lbf7_gb7.out.save	Baseline for GB 7 test
test/cs/lbf7_gb/Run.lbf7_gb1	Script to run GB 1 test
test/cs/lbf7_gb/Run.lbf7_gb5	Script to run GB 5 test
test/cs/lbf7_gb/Run.lbf7_gb5.min	Script to run GB 5 minimization test
test/cs/lbf7_gb/Run.lbf7_gb6	Script to run GB 6 test
test/cs/lbf7_gb/Run.lbf7_gb7	Script to run GB 7 test
test/cs/lbf7_gb/inpcrd	Coordinates common for all GB tests
test/cs/lbf7_gb/prmtop	Parameter file common for all GB tests
test/cs/jac/Run.jac	Script to run jac MD test
test/cs/jac/Run.jac.eedmeth2	Script to run modified jac test with eedmeth = 2
test/cs/jac/Run.jac.eedmeth4	Script to run modified jac test with eedmeth = 4
test/cs/jac/Run.jac.eedmeth5	Script to run modified jac test with eedmeth = 5
test/cs/jac/Run.jac.min	Script to run jac minimization test
test/cs/jac/inpcrd	Coordinates common for all jac tests
test/cs/jac/mdout.jac.save	Baseline for jac MD test
test/cs/jac/mdout.jac.eedmeth2.save	Baseline for eedmeth =2 modified jac test
test/cs/jac/mdout.jac.eedmeth4.save	Baseline for eedmeth =4 modified jac test
test/cs/jac/mdout.jac.eedmeth5.save	Baseline for eedmeth =5 modified jac test
test/cs/jac/mdout.jac.min.save	Baseline for jac minimization test
test/cs/jac/prmtop	Parameter file common for all jac tests

Table 9. Added test files

Appendix C Building card-side code from source

The card-side code is provided in two formats: as precompiled binaries and as source code. The binaries are installed by running the `build_amber9_ClearSpeed.bash` script. The card-side binaries are installed if there is no ClearSpeed SDK installed on the system, when running the `build_amber9_ClearSpeed.bash` script. The binaries are built from source if the SDK has been installed. The following is a set of instructions on how to build the card-side binaries from source.

```
cd ${AMBERHOME}/src/sander/clearspeed
make clean
make install
```

The executables shown in [Table 10](#) will be created and copied to `${AMBERHOME}/exe/clearspeed`.

Filename	Description
<code>gb.csx</code>	Executable for GB models 1, 2, 5, and 7
<code>gb6.csx</code>	Executable for GB models 6
<code>explicit.csx</code>	Executable for explicit solvent

Table 10. Executables created from source code

Appendix D Notes on configuration for pgf90

The configure script distributed with Amber 9, enforces a 32-bit build for the pgf90 compiler.

On 64-bit operating systems, ClearSpeed's libraries and runtime environment is set up for 64-bit. The configure script distributed with ClearSpeed's patch sets up a 64-bit build on 64-bit operating systems for pgf90. In order to get the build to pass the Amber test suite, the optimization level was lowered for the file `src/sander/qm_link_atoms.f`.

The default installation for the pgf90 compiler has also been changed. It used to be `/usr/pgi` and it is now `/opt/pgi`. The configure script has been made more flexible so that it can find the associated BLAS and LAPACK libraries. Previously, the libraries were expected to be in `/usr/pgi/linux86/lib`.

Appendix E Notes on configuration option `-openmp`

The `-openmp` option is only used to overlap the surface area calculation on the host of a GBSA (`gbsa=1`) simulation with the nonbonded force calculation on the Advance X620 or the Advance e620. This option is not used anywhere else for the sander executable. When configuring with `-openmp`, only a couple of files are compiled for OpenMP. During testing it was found that some of the standard DIVCON tests with ifort failed if all of the sander source files were compiled with the OpenMP flag.

One drawback of using the OpenMP flag, is that there will be a thread running on the host at 100% CPU polling, if a synchronization point has been reached. If the `gbsa=1` option is never used, then there is no advantage in using the `-openmp` option.

OpenMP has only been tested with the pgf90 and ifort compilers.

When using the `openmp` configuration flag, set the `OMP_NUM_THREADS` environment variable:

```
export OMP_NUM_THREADS=2
```

Appendix F Notes on building Amber with MKL version 10.0.1.014

There is an issue with this MKL version, in that an internal error handler is not found if the MKL library is linked as a shared object. This can be worked around, by linking in static versions of the MKL libraries. However, the Amber `-static` configure flag, links the whole application statically, this is a problem for the `sander.CS` executable, as the ClearSpeed version uses the dynamic loader to decide which libraries to load at runtime. If `sander.CS` is linked statically, the application will fail with a segmentation fault at the point when attempting to use the dynamic loader.

This issue has only been seen with this particular version of MKL. It has been seen both when using `ifort` and `gfortran` as the Fortran compiler.

The way around these issues is to link in a static version of the MKL library, while linking the application dynamically. This can be accomplished by:

1. Configure Amber with `-static`, for example:

```
configure -static -cs -openmp ifort_x86_64
```

2. Removed the `-static` directive from the load line in `config.h`.

Edit `config.h`, the load line should look like:

```
LOAD= ifort $(LOCALFLAGS) $(AMBERBUILDFLAGS)
```

An alternative solution is to update to MKL version 10.0.3.020 or later.

ClearSpeed Technology, Inc.

800 West El Camino Real
Suite 180
Mountain View, CA 94040

Tel: +1 650 943 2329

Fax: +1 650 962 1188

ClearSpeed Federal Systems, Inc.

228 Hamilton Avenue, 3rd Floor
Palo Alto, CA 94301

Tel: +1 650 798-5027

Fax: +1 650 798-5001

ClearSpeed Technology plc

3110 Great Western Court
Hunts Ground Road
Bristol BS34 8HP
United Kingdom

Tel: +44 (0)117 317 2000

Fax: +44 (0)117 317 2002

Email: info@clearspeed.com

Web: <http://www.clearspeed.com>

Support: <http://support.clearspeed.com>

1. Information and data contained in this document, together with the information contained in any and all associated ClearSpeed documents including without limitation, data sheets, application notes and the like ('Information') is provided in connection with ClearSpeed products and is provided for information only. Quoted figures in the Information, which may be performance, size, cost, power and the like are estimates based upon analysis and simulations of current designs and are liable to change.
2. Such Information does not constitute an offer of, or an invitation by or on behalf of ClearSpeed, or any ClearSpeed affiliate to supply any product or provide any service to any party having access to this Information. Except as provided in ClearSpeed Terms and Conditions of Sale for ClearSpeed products, ClearSpeed assumes no liability whatsoever.
3. ClearSpeed products are not intended for use, whether directly or indirectly, in any medical, life saving and/ or life sustaining systems or applications.
4. The worldwide intellectual property rights in the Information and data contained therein is owned by ClearSpeed. No license whether express or implied either by estoppel or otherwise to any intellectual property rights is granted by this document or otherwise. You may not download, copy, adapt or distribute this Information except with the consent in writing of ClearSpeed.
5. The system vendor remains solely responsible for any and all design, functionality and terms of sale of any product which incorporates a ClearSpeed product including without limitation, product liability, intellectual property infringement, warranty including conformance to specification and or performance.
6. Any condition, warranty or other term which might but for this paragraph have effect between ClearSpeed and you or which would otherwise be implied into or incorporated into the Information (including without limitation, the implied terms of satisfactory quality, merchantability or fitness for purpose), whether by statute, common law or otherwise are hereby excluded.
7. ClearSpeed reserves the right to make changes to the Information or the data contained therein at any time without notice.

© Copyright ClearSpeed Technology plc 2007, 2008. All rights reserved.

Advance is a registered trademark of ClearSpeed Technology plc

ClearSpeed, ClearConnect, Advance and the ClearSpeed logo are trade marks or registered trade marks of ClearSpeed Technology plc. All other brands and names are the property of their respective owners.