**R.E.D. FOR WINDOWS/CYGWIN INSTALL GUIDE**

**by Thomas A. D. Patko – California State University Long Beach, USA**

**REQUIRES R.E.D.-III.3 & HIGHER**

*The notes below may not be the only way to get the R.E.D. program working on your Window PC, but they are the way that I got it going. If you have any corrections or alternative methodology that you wish to suggest, it is encouraged that you share your thoughts with the R.E.D. development team by posting on the q4md, CCL or AMBER mailing list so that others may benefit from your feedback. If you should have any questions about the instructions in this guide, it is also encouraged that you post to these same lists for assistance.*

## INSTALL CYGWIN ON YOUR WINDOWS MACHINE

The default shell for Cygwin will be bash and all further instructions will assume that you are using a bash shell. In this case you will want to create a .bashrc or .profile file in the root of your user account (if not already present) to setup all of the necessary environmental variables. There are instructions on how to setup these necessary environmental variables later in this guide to support each of the *ab initio* package that you may wish to use and any other R.E.D. dependencies.

You will need to download and install a version of Cygwin (see link below). Cygwin will work on any 32-bit or 64-bit version of Windows. I installed version 1.5.25-15 although newer versions should also work.

http://www.cygwin.com

You do not strictly speaking have to install everything in the Cygwin package although this is exactly what I did for simplicity and since I have plenty of free disk space (Cygwin will use up well over 1GB), but all of the necessary POSIX utilities (such as which, diff, cp, mv etc...), the perl package, shells (tcsh and bash) and developer tools (gcc, g++, g77, g95 and gfortran) should be installed to ensure that you can compile the RESP program, and properly run R.E.D. While you can choose to install into a different directory than the default C:\cygwin that is suggested by the installer, unless you have disk space issues on your boot OS drive, why not just use the default one (that is what I did although I suppose other locations should work OK as well). Some good details about what will be required from our Cygwin install (mainly to compile the RESP program later in this guide) are summarized in the following link:

http://ambermd.org/mswindows.html

The remainder of this install guide will assume that you have some reasonable familiarity with LINUX / UNIX shell and command line usage. If you do not, you should probably find a good online UNIX tutorial or "Intro to Linux". The Cygwin specific usage is covered in the online documentation:

http://cygwin.com/cygwin-ug-net/ov-ex-win.html

*Comments about Cygwin path issues germane to using R.E.D. for Windows under Cygwin:*

If you wish to call a path with the normal Cygwin file system (anything under the root installation location which is typically "C:\cygwin" by default) you can use the regular UNIX like path and R.E.D. will automatically take care of any necessary conversions for you (by use of the cygpath utility). If you wish to address any path that is NOT a subdirectory of the main install directory, you will need to address that path (most likely a scratch drive directory path) by using the /cygdrive directory. For example if you have your OS drive as C:\ and your Cygwin installation as "C:\cygwin" (default) and you want to use the "D:\scratch" directory as your scratch path then it can be properly addressed as /cygdrive/d/scratch in your environmental variable for that scratch directory path designation.

You will need to add some environmental variables to user profile in Cygwin. One simple way is to create a .bashrc file in the root of your user directory (if not already present). For example:

```
$ cd ~
$ touch .basrhc
```

or

```
$ touch .profile
```

This will create a file called .bashrc or .profile in the root of your user directory. In some cases your .bashrc or .profile will not be automatically sourced by Cygwin. In this case simple source it from a command line.

```
$ cd ~
$ source .basrhc
```

or

```
$ source .profile
```

Open up your .bashrc or .profile file using one of the text editors recommended later in this guide and add the appropriate environmental variables for your AMBERTOOLS installation and any of the *ab initio* software that you wish to use with R.E.D. (detailed instructions for each set of required environmental variables are provided in the section below).

## INSTALL THE *AB INITIO* SOFTWARE ON WINDOWS

Currently PC GAMESS / Firefly, GAMESS-US (WinGAMESS) and Gaussian for Windows are supported *ab initio* packages for R.E.D. on the Windows/Cygwin platform implementation. Please note that you will get essentially IDENTICAL charge sets no matter which *ab initio* software that you choose to use with R.E.D., so the selection is one much of personal preference (although one software or another may be better suited for your particular systems). Find in the sections below links to download each of these *ab initio* packages along with some comments that will be useful configuration tips to make each of these packages to work with R.E.D. for Windows/Cygwin. My approach was to install each *ab initio* package as usual and then port a known working installation into a logical directory that is in the Cygwin $PATH. I chose the /usr/local/bin directory of Cygwin to drop all of the working binaries for PC GAMESS / Firefly, GAMESS-US (as the WinGAMESS version) and Gaussian for Windows (G03W).

There are some good notes on installing and testing several *ab initio* packages on the Windows platform (including PC GAMESS / Firefly, Gaussian for Windows and WinGAMESS) posted online (see link below) and I would recommend looking at these notes in addition to this installation guide.

http://www.webmo.net/support/binaries_windows.html

### *PC GAMESS / Firefly for Windows*

Download and install PC GAMESS / Firefly for you Windows machine. There are instructions about how to do this on the website. Run the tests to ensure that you have a properly working installation.

http://classic.chem.msu.su/gran/gamess/index.html

If you plan to run PC GAMESS / Firefly on a single node (machine) for parallel SMP jobs only, then the instructions below is a simplified procedure to get your installation up and running for this configuration. Since this is a rather typical single machine configuration, it is included below.

The following two changes are required to the default PC GAMESS / Firefly for Windows installation to run in parallel on a single node machine to support SMP parallel runs:

1) You need to copy the MPI binding MPIBIND.NT-MPICH-SMP.DLL and rename it as mpibind.dll in the root of the PC GAMESS installation directory (replacing the default mpibind.dll that is provided). The

mpich_smp.dll file must also be present in this same directory to work properly.

http://classic.chem.msu.su/gran/gamess/bindings.html

2) You need to rename the file pcgp2psm.dll to pcgp2p.dll (this will replace the default pcgp2p.dll that is located in the root of the PC GAMESS installation directory).

http://classic.chem.msu.su/gran/gamess/p2p.html

That is it. You should be able to run SMP parallel jobs on a single machine. You can run the benchmarks to ensure that your parallel jobs are working properly. You may want to download some of the graphical job submission tools from the PC GAMESS / Firefly website to make launching your jobs more convenient, although instructions are also provided on how to run jobs from Windows Terminal shell. Please note that running PC GAMESS / Firefly from a Cygwin terminal IS DIFFERENT than running jobs from a Windows TERMINAL. When R.E.D. executes PC GAMESS / Firefly, all of these differences are automatically handled for you.

PC GAMESS / Firefly 7.1.F was tested although other more recent versions should work fine as well. You will be able to make use of Firefly for Windows when called from within R.E.D. under Cygwin exactly as you would from a normal windows command line (all processing cores will be available on your machine and the same memory capabilities are available). You may need to modify the R.E.D. source code as appropriate to use your maximum permissible memory allocation, although the default values are fine for most systems.

Once your PC GAMESS / Firefly installation is working properly in the usual location on your machine, there are really two ways to get it ready to work under Cygwin for use with R.E.D.

Option 1:

Simply append your path for each ab initio software so that the current installation location is included in your path. For example if PC GAMESS / Firefly is installed and working at "C:\Firefly" you could leave it in place and append your $PATH variable to reflect the current installation location:

$ export PATH=$PATH:/cygdrive/c/Firefly

Option 2:

Simply copy the COMPLETE working binary installation directory to somewhere that is already in your $PATH of your Cygwin installation. For example, I chose the /usr/local/bin directory (since this is already

setup in the Cygwin $PATH) and it seems to work perfectly, although other locations are obviously possible.

You need to define the $PCGAMESS_SCRDIR variable in your .bashrc or .profile and source it. Alternatively you can simply declare these required environmental variables for R.E.D. to execute properly from your terminal shell each time that you start Cygwin.
For example, if PC GAMESS / Firefly is installed in "C:\Firefly" and your scratch directory is "D:\scratch" and you chose NOT to copy the COMPLETE working binary directory to /usr/local/bin then you would use:

$ export PCGAMESS_SCRDIR=/cygdrive/d/scratch
$ export PATH=$PATH:/cygdrive/c/Firefly

In the case where you copied your entire working PC GAMESS / Firefly directory to /usr/local/bin and wanted to have your scratch directory on the same drive the you would use:

$ export PCGAMESS_SCRDIR=/tmp/firefly

Note that in the second case you do not need to append your PATH variable (/usr/local/bin is already in the PATH). The scratch directory should exist BEFORE R.E.D. is called.

### *Gaussian for Windows*

Obtain a version of GXXW (G98W, G03W, G09W) from Gaussian. There is rather extensive documentation posted on the Gaussian website to which I will altogether defer.

http://www.gaussian.com/

If you have a multiprocessor version of G03W you will be able to make use of up to 4 processing cores and approximately 1750MB of memory in aggregate. You will need to modify the R.E.D. source code as appropriate to use your maximum permissible memory allocation, although the default memory allocation should be perfectly fine for most jobs. All tests were run on G03W E.01 multiprocessor version, although other versions should work fine as well.

Once your Gaussian for Windows installation is working properly in the usual location on your machine, there are once again really two ways to get it ready to work under Cygwin for use with R.E.D.

Option 1:

Simply append your path for each ab initio software so that the current installation location is included in your path. For example if Gaussian is installed and working at the default "C:\G03W" directory you could leave it in place and append your $PATH variable to reflect the current installation location:

$ export PATH=$PATH:/cygdrive/c/G03W

Option 2:

Simply copy the COMPLETE working binary installation ("C:\G03W") to somewhere that is in your $PATH of your Cygwin installation. For example, I chose the /usr/local/bin directory (since this is already setup in the Cygwin $PATH) and it seems to work perfectly (although other locations are obviously possible).

In either case you MUST provide the variable GAUSS_EXEDIR environmental variable in the WINDOWS STYLE CONVENTION. You CANNOT use the typical Cygwin style addressing that is OK for all other environmental variables for use with R.E.D.

If your G03W installation is for example in your "C:\G03W" directory then use:

$ export GAUSS_EXEDIR="C:\G03W"

In the case where you copied your entire working "C:\G03W" directory to /usr/local/bin then you would use:

$ export GAUSS_EXEDIR="C:\cygwin\usr\local\bin"

Note that in the second case you do not need to append your PATH variable (/usr/local/bin is already in the PATH). ***Please note that the scratch directory for Gaussian when called from under Windows/Cygwin is always the current working directory (same directory as for RED-vIII.3.pl and the R.E.D. input files).***

### *GAMESS-US for Windows / WinGAMESS*

The simplest way to obtain GAMESS-US for Windows is to select WinGAMESS (*GAMESS version January 12, 2009 R3 for Microsoft Windows*) when download from the Iowa State University / Gordon Group website (see link below). This assumes that you are already registered with the GAMESS and can simply download the updated source code and binaries.

http://www.msg.chem.iastate.edu/GAMESS/download/register/

This guide will assume that you have downloaded the current version of WinGAMESS and copied the binary files to the your /usr/local/bin directory in your Cygwin installation. Some more detailed discussion about GAMESS-US on the Windows platform can be found using this link:

http://www.msg.chem.iastate.edu/gamess/win.html

WinGAMESS is provided as a Windows MSI installer that will supply the precompiled GAMESS-US binary built under Cygwin and all tools to run jobs with WinGAMESS on any Windows system (with or without Cygwin being installed). This standard WinGAMESS distribution will only support running jobs in serial. You can compile GAMESS-US from source yourself under Cygwin if your prefer, using the Linux source code as the starting point. This guide will assume that you are using the version of WinGAMESS that is provided as a precompiled binary, and that only serial jobs are supported. If you wish to compile it yourself from source, find below a couple of links that can assist you do this:

http://www.chemsoft.ch/qc/cygw.htm
http://us.geocities.com/pen7cmc/gamesscygwin.html

If you run the MSI installer that you will download from ISU website ("WinGAMESS.current.msi") and accept the default settings you will end up with a working WinGAMESS installation located at C:\WinGAMESS. This installation will have two subdirectories called "scratch" and "temp" in the location where your WinGAMESS binary is installed (which must be somewhere in your PATH no matter where that might be). You must then have read/write permissions to this installation location for WinGAMESS to run properly.

Once your WinGAMESS installation is working properly in the usual location on your machine, there are really two ways to get it ready to work under Cygwin for use with R.E.D.

Option 1:

Simply append your path for each ab initio software so that the current installation location is included in your path. For example if WinGAMESS is installed and working at "C:\WinGAMESS" you could leave it in place and append your $PATH variable to reflect the current installation location.

```
$ export PATH=$PATH:/cygdrive/c/WinGAMESS
```

You need to define the $GAMESSUS_SCRDIR and $GAMESSUS_TMPDIR environmental variables to your .bashrc or .profile and source it. Alternatively you can simply declare these required environmental variables for R.E.D. to execute properly from your terminal shell each time that you start Cygwin.

$ export GAMESSUS_SCRDIR=/cygdrive/c/WinGAMESS/scratch
$ export GAMESSUS_TMPDIR=/cygdrive/c/WinGAMESS/temp

Option 2:

Simply copy the COMPLETE working binary installation ("C:\WinGAMESS") to somewhere that is in your $PATH. For example, I chose the /usr/local/bin directory (since this is already setup in the Cygwin $PATH) and it seems to work perfectly, although other locations are obviously possible. Note that in the second case you do not need to append your $PATH since /usr/local/bin is already in the $PATH.

You need to define the $GAMESSUS_SCRDIR and $GAMESSUS_TMPDIR environmental variables to your .bashrc or .profile and source it. Alternatively you can simply declare these required environmental variables for R.E.D. to execute properly from your terminal shell each time that you start Cygwin.

$ export GAMESSUS_SCRDIR=/usr/local/bin/scratch
$ export GAMESSUS_TMPDIR=/usr/local/bin/temp

The "scratch" and "temp" directory should exist and "temp" should be empty BEFORE R.E.D. is called.

Ensure that your PC GAMESS / Firefly, GAMESS-US (WinGAMESS) or Gaussian for Windows (GXXW) installations are working properly BEFORE proceeding to test running them with R.E.D.! You only need ONE *ab initio* software working in order to use R.E.D., although there is no harm to have multiple *ab initio* packages configured (in fact it is explicitly supported).

## INSTALL AMBERTOOLS

Strictly speaking you can find the RESP program from sources other than AMBERTOOLS, although it is freely available from this source, and has some nice convenient instructions on how to configure and compile the AMBERTOOLS which includes the necessary RESP program. Once again, you will need to add some environmental variables to your user profile (namely $AMBERHOME and to append your path to include the absolute path to where you built your AMBERTOOLS binaries). The default configure, make and make install should work perfectly and will produce a working RESP binary for R.E.D. under Cygwin.

http://ambermd.org/AmberTools-get.html

Some specific instructions can be found for installing AMBERTOOLS on Windows/Cygwin at:

http://ambermd.org/mswindows.html

Note that in order to ensure that xleap will build correctly, you will need to add the following two symbolic links BEFORE you configure and build AMBERTOOLS on Windows/Cygwin (based upon the AMBER mail reflector post http://archive.ambermd.org/200903/0470.html )

ln -s /lib/libXt.a /usr/lib/libXt.so
ln -s /lib/libXext.a /usr/lib/libXext.so

Once again you will need to setup your environmental variables so that AMBERTOOLS (and most importantly in our case RESP) will work correctly. I chose the following locations for installs, although any valid location should do just fine. Of course this assumes that you build your AMBERTOOLS installation in the root of your home directory. If you picked a different location you will need to adjust your environmental variables accordingly.

$ export AMBERHOME=$HOME/amber10
$ export PATH=$PATH:$AMBERHOME/exe

Typing "which resp" from a Cygwin command line should now return the absolute path to the RESP binary. You will need to test that the RESP program is working properly by running the test suite provided (it should work fine but you can test it for completeness).

**CONFIGURE R.E.D.**

You will need to be sure to have R.E.D.-III.3 or higher to ensure that it works properly on the Windows/Cygwin platform. Follow the instructions on how to setup R.E.D. contained in the download. Once completed, you should be able to run everything in the "HowTo" provided in the R.E.D.-III.3 download package. If these jobs complete correctly, you will be ready to complete your own charge derivation runs on your Windows machine!

http://q4md-forcefieldtools.org/RED/

R.E.D. III.3 was tested on Windows XP, Windows Vista 64-bit and Windows 7 64-bit using PC GAMESS / Firefly version 7.1.F for Windows, Gaussian for Windows G03W E.01 multiprocessor version and GAMESS-

US / WinGAMESS (Jan 2009 R3). More recent versions of these *ab intio* software should also work just fine.

### XRED support under Windows/Cygwin

Currently the support for XRED is quite limited when used under Windows/Cygwin, although all features are nominal using the text only terminal shell. The reason for this limited XRED support is that the Tcl/Tk binaries that are provided from the Cygwin repositories are built against the Win32 API, and NOT against X11 API as are all other Tcl/Tk installations for any other POSIX like OS (Linux, IRIX, OS X, ….etc). For XRED to work properly, you MUST have a Tcl/Tk installation that has been built against X11 (meaning that the Tcl/Tk from Cygwin is not acceptable for XRED). Using the instructions to install some precompiled Tcl/Tk binaries built against X11 provided below, at this time XRED is only known to work on Windows XP 32-bit, and known not to work on Windows Vista 64-bit and Windows 7 64-bit. Other operating systems may, or may not, work using the instructions below (feedback on other working Windows OS are welcome).

### *INSTALL Tcl/Tk built against X11 on Cygwin*

I simply downloaded the two precompiled binaries from the webpage http://www.opencircuitdesign.com/cygwin/ which contains more details and the two links provided below.
http://www.opencircuitdesign.com/cygwin/archive/tcltk_x11_win.tgz
http://www.opencircuitdesign.com/cygwin/archive/tcltk_x11_vista.tgz

Note that for the XRED testing my machine was Windows XP 32-bit.

The two different Tcl/Tk builds as contained in tcltk_x11_win.tgz and tcltk_x11_vista.tgz can actually be installed on the same Cygwin installation. I presume that for Windows Vista that only the tcltk_x11_vista.tgz version will work (although it did not work for my Vista 64-bit machine). Perhaps the version contained in tcltk_x11_vista.tgz works properly for Vista 32-bit machine (I do not have such a test machine but feedback from those that do are welcome). To install both of these binary packages, simply copy each of the *.tgz files into the root of the Cygwin install (default is "C:\cygwin") and execute the following commands:

```
$ cd /
$ tar zfx tcltk_x11_win.tgz
$ tar zfx tcltk_x11_vista.tgz
```

STARTING XRED USING TCL/TK BUILT WITH X11 SUPPORT

Now, bring up a new Cygwin terminal and append your PATH as follows:

$ export PATH=$PATH:/usr/local/lib

Before calling XRED you will need to start the XServer by typing "startxwin.sh" as usual from the text only Cygwin terminal. This will launch the XServer (an icon will appear in your bottom right taskbar) and a xterm terminal shell (white background of xterm rather than black of Cygwin text only terminal is an easy to tell).

To call XRED using the older 8.4.11 version of Tcl/Tk built against X11
$ wish.exe XRED-III.3.tcl &

To call XRED using the newer 8.5.0 version of Tcl/Tk built against X11
$ wish8.5.exe XRED-III.3.tcl &

Both versions worked fine on Windows XP although the differences in the graphics rendering engine is immediately clear. Since the newer 8.5.0 build seems to work fine (this came from tcltk_x11_vista.tgz) I do not particularly see any reason to use the older 8.4.11 build that came from tcltk_x11_win.tgz. The install notes and commands to call both versions are provided above mainly for completeness.

In the case the precompiled binary installation of Tcl/Tk built against X11 for Windows/Cygwin approach described above does not work for your Windows/Cygwin installation, your best approach would be to try and build Tcl/Tk against X11 using the instructions as post at:

http://www.opencircuitdesign.com/cygwin/tclcompile.html

Any reports of successful Tcl/Tk builds using the procedure as detailed in the link above on operating systems other than Windows XP 32-bit would be appreciated.

## TEXT EDITORS

You will want to have a good text editor that supports UNIX and Windows text file formats. I use EditPad Pro (commercial) although the free EditPad Lite should be perfectly fine as well. The Open Source Crimson Editor for Windows is also an excellent choice (see links below).

http://www.editpadpro.com/
http://www.editpadpro.com/editpadlite.html
http://www.crimsoneditor.com/

There are many other text editors for Windows (too many to list here), but if you have a particular favorite pass it along and it will get added to the next revision.

**Windows specific installation notes and usage guide**

**Thomas Patko, Dept. of Chemistry & Biochemistry, California State University, Long Beach, USA**

**Last Revised October 1st, 2009**