This document provides a fix for the problem noted by members of the Linrad Google Group since July 2023 where, for Windows the 32-bit version of Linrad compiles fine when the files extra_w3sz.c and users_w3sz.c are used, but the 64-bit compile fails. Also on non-ancient Linux installations such as Ubuntu 20.04.2 with gcc version 9.4.0 Linrad compile fails when the files extra_w3sz.c and users_w3sz.c are used with both 32-bit and 64-bit compiles. On Windows the failure occurs with errors of the form

```
users_extra.c:181:10: error: passing argument 1 to restrict-qualified parameter aliases with argument 2 [-Werror=restrict]
 181 | strncpy(w3sz_buffer,w3sz_buffer,3);//w3sz

users_extra.c:121:2: error: ignoring return value of ‘fgets’, declared with attribute warn_unused_result [-Werror=unused-result]
 121 | fgets(azel_loc,80,Fp);//w3sz

users_extra.c:109:38: error: ‘%s’ directive writing up to 79 bytes into a region of size 58 [-Werror=format-overflow=]
 109 | sprintf(s,"Cannot open to write: %s", azellocfile);//w3sz
```

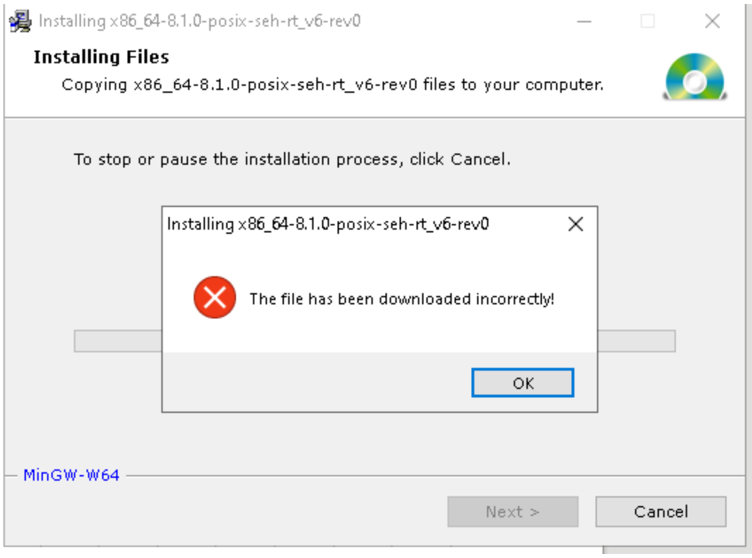
On Ubuntu 20.04.2 with gcc 9.4.0 the errors are of the form

```
users_hwaredriver.c:153:8: error: passing argument 1 to restrict-qualified parameter aliases with argument 2 [-Werror=restrict]
  153 | strcat(zro,zro);

users_hwaredriver.c:357:14: error: this statement may fall through [-Werror=implicit-fallthrough=]
  357 |     *w3szfile="aahsmsfile";

users_hwaredriver.c:392:3: error: ignoring return value of ‘fscanf’, declared with attribute warn_unused_result [-Werror=unused-result]
  392 |   fscanf(Fp,"%d%d%f%f",&ug.xleft,&ug.ytop,&ug.par1,&ug.par2);

In function ‘strncpy’,
    inlined from ‘userdefined_u’ at users_hwaredriver.c:171:1:
/usr/include/bits/string_fortified.h:106:10: error: ‘__builtin_strncpy’ output may be truncated copying 10 bytes from a string of length 79 [-Werror=stringop-truncation]
  106 |   return __builtin___strncpy_chk (__dest, __src, __len, __bos (__dest));

In function ‘strcat’,
    inlined from ‘userdefined_u’ at users_hwaredriver.c:159:1:
/usr/include/bits/string_fortified.h:128:10: error: ‘__builtin___strcat_chk’ source argument is the same as destination [-Werror=restrict]
  128 |   return __builtin___strcat_chk (__dest, __src, __bos (__dest));
```

The above lists are not exhaustive.

Modifying the code so that it successfully 64-bit compiles linrad64.exe in Windows and xlinrad or xlinrad64 in linux turned out to be pretty simple. The problem preventing a successful compile process was just that more "modern" compilers have progressively tightened their restrictions so that code that was acceptable to older compiler versions is now associated with warnings/errors that prevent compilation. Mitigating this and achieving successful compilation was quickly done, but then there was a second issue in Windows caused by the particular version of the mingw-w64 compiler that I was using.

This second compiler-version issue occurred because the mingw-w64 installer linked from Leif's page has been broken for a couple of years and wouldn't install mingw-w64 (this is a known problem well documented on the web), and the compiler that I had chosen to install instead compiled the modified code OK, but linrad64 would crash each time the receive page was entered. Specifically, the mingw-w64-install.exe installer at the link https://sourceforge.net/projects/mingw-w64/ as referenced from https://www.sm5bsz.com/linuxdsp/install/compile/wincompile.htm when run gives the error as shown in the image below.



Installing x86_64-8.1.0-posix-seh-rt_v6-rev0

Installing Files

Copying x86_64-8.1.0-posix-seh-rt_v6-rev0 files to your computer.

To stop or pause the installation process, click Cancel.

Installing x86_64-8.1.0-posix-seh-rt_v6-rev0

The file has been downloaded incorrectly!

The steps required to get a version of mingw-w64 that will compile linrad64.exe so that Linrad does not crash when the receive screen is entered are:

1. From https://sourceforge.net/projects/mingw-w64/files/ click on the link labeled x86_64-posix-seh which will download the file: x86_64-8.1.0-release-posix-seh-rt_v6-rev0.7z.

2. Unzip this file into the folder x86_64-8.1.0-release-posix-seh-rt_v6-rev0

3. Create the directory C:\mingw64

4. Copy the folder mingw64 from the folder you created (x86_64-8.1.0-release-posix-seh-rt_v6-rev0) into the directory you created (C:\mingw64), so that you have the directory structures:
C:\mingw64\mingw64\bin,
C:\mingw64\mingw64\etc,
C:\mingw64\mingw64\include, etc., etc.

5. Set the windows environment path to include C:\mingw64 (good practice even though Leif used absolute addressing in his compile/linking code).
6. Download my modified files extra_w3sz.c and users_w3sz.c from

https://w3sz.com/extra_w3sz.c
https://w3sz.com/users_w3sz.c
7. Place the modified files extra_w3sz.c and users_w3sz.c into your linrad directory.

8. Rename the modified users_w3sz.c to wusers_hwaredriver.c and rename the modified file extra_w3sz.c to users_extra.c Note that these modified files will NOT work for 32 bit compiles with the version of the mingw compiler as listed on Leif's webpage at the link given near the beginning of this document, but the original unmodified files work for 32-bit compiles so the modified files are not needed in that case.

9. In your linrad directory type "configure", hit "Enter", and then type "make64" and hit "Enter".

If you do the above, linrad64.exe should compile and run fine...it does here. If there are issues, please let me know.

73,

Roger
W3SZ
Jan 6, 2024

MinGW 64-bit versions that did work:
The mingw version that did work, with links given earlier in this document had
mingw-w64 version 4.3.5
gcc 8.1.0

MinGW 64-bit versions that didn't work:

1. From https://winlibs.com/
https://github.com/brechtsanders/winlibs_mingw/releases/download/13.2.0posix-17.0.6-11.0.1-msvcrt-r4/winlibs-x86_64-posix-seh-gcc-13.2.0-llvm-17.0.6-mingw-w64msvcrt-11.0.1-r4.zip

This has mingw-w64 version 11.0.1-r3
gcc 13.2.0

2. Original download source unknown
Source data 9-17-2016
mingw-w64 4.3.0
gcc 6.2.0