C ports of PL/M 80 and Fortran applications

This repository contains my ports of several PL/M 80 applications to C and my port of the old Fortran based PL/M80 compiler, again to C.

There is also a historical port of the ISIS-II PL/M 80 compiler to C++, directly done from the disassembled code, i.e. before I reversed engineered the compiler into PL/M 80 source.

Originally the code was distributed as part of my Intel80Tools repository, however I have refactored into its own repository as a cleaner option, as the number of ports increases. Prebuilt Windows 32bit binaries of these tools (except the historic C++ port) are still distributed as part of the Intel80Tools repository.

See the doc directory for information on each of the applications, including usage differences from the original applications.

Visual studio solution files are provided for all the tools, along Linux Makefiles.

Note if you get a warning message when building files, it may be because you are using an older or possibly newer version than I have set the project to. Visual Studio provides simple options to retarget to any version you have.

There are also a three of script files from my versionTools repository in the Scripts directory. Of these getVersion.cmd and getVersion.pl are the main ones that creates the version numbers. The other install.cmd is used to auto install Windows executables to your desired directory. This can be replaced with your own installation scripts. For Linux all the built files can be found in the subdirectory Linux/Install, from here you can install to your required directory.

Recent major changes

4-May-2023

- I have replaced the versioning model with one that is simpler to generate. A description can be found in the document Scripts/versionTools.pdf.
- Related to the above I have standardised the inclusion of supplementary version information and it's display. Details can be found in Scripts/showVersion.pdf
- I have modified the licence information to reflect original owners and that the code is released for hobbyist use and academic interest.
- Linux build support has been added. To avoid source tree pollution, a separate directory Linux has been added. Within this are subdirectories one for each application. Whilst each application can be built individually, the Makefile in the Linux directory can be used to build all the applications.
  The Makefiles assume gnu make is being used and the use of the -j (parallel builds) is supported. As I test using WSL on Windows, due to performance issues of git access from WSL to Windows,

the default build does not force a version check or timestamp. In development this is ok, for release the publish target can be used to force the check.

- A number of bugs were identified by Andrey Hlus and have been fixed. Thanks Andrey.
- The code has been significantly modified to allow clean builds with gcc -Wall and cl /W3.
  Note using cl /W4 creates lots of additional minor warnings which I have checked. Most of the warnings are about assigning a larger integer to a smaller one, adding the explicit casts would make the code much harder to read.
  Also note cl /Wall generates lots of additional information, e.g. which functions are in-lined.
- binobj, hexobj, lib, link, locate and objhex should now work on non little endian processors, let me know if you find problems. The work on fixing asm80 and plm80c has started.
- tex has been modified to work on Linux, although care is needed to ensure file names are the correct case on the command line and within tex input. The changes which impact Windows and Linux are
  - On the command line only a - can also be used as the parameter marker instead of $. This avoids shell expansion under Linux. I suggest avoiding filenames starting with $ or - to avoid any confusion.
  - getch, which is used during paging, now only enters raw input whilst getting the single character. This hopefully will allow console input during tex processing to use standard line handling.
  - kbhit has been stubbed. Both operating systems support control C to abort a program. Although this does not give the opportunity to continue, entering control C is unlikely to be accidental.
    A limitation on this is when using a key hit to pause output to a printer to say handle a paper jam. As modern printer interfaces are usually spooled, direct printer output will be uncommon. If it does occur, run tex with the $I command to start output from a given page.

```
Updated by Mark Ogden 4-May-2023
```