

README: ABOUT THE KDF9 EMULATOR ee9, VERSION 6

CONTENTS

INTRODUCTION

MISCELLANEOUS OS ISSUES

- A. POSIX environment
- B. Getting a GNAT Ada 2012 compiler

BUILDING YOUR OWN VERSION OF ee9

THE USERCODE COMPILER, ka13

ANOTHER KDF9 EMULATOR, kdf9

GETTING AND INSTALLING KIDSGROVE ALGOL UPDATES

INTRODUCTION

ee9, my KDF9 emulator, runs a variety of interesting and important KDF9 programs. Please report any bugs you find to me, using the address kdf9@findlayw.plus.com, to help advance the development effort.

The source code of **ee9** is released under the terms of the GPL, version 3.

You can verify the correct operation of my binary, or a port to another platform, by running (in the `Testing` directory) the command:

```
./ee9_reg_test
```

To be honest, this checks that your **ee9** implementation gives the same results as mine, on a variety of Usercode and Whetstone Algol programs. It runs a set of test cases and allows you to compare expected and actual outputs and execution digital signatures (these are hashes of the register values at the end of each instruction-execution). It automatically compares its results file with a file containing the results from a run on my own computer, taken to be the standard of correctness. If your **ee9** passes these tests we can safely assume that it is working as well as mine.

MISCELLANEOUS OS ISSUES

A. POSIX ENVIRONMENT

You may benefit from ensuring that your system is set up to use the Latin-1 character set. You can find out by issuing the bash command:

```
set | grep LANG
```

If the output does not contain 'ISO8859-1' then you need to reset your locale and character set. The bash command:

```
export LANG=en_GB.ISO8859-1
```

(or its equivalent using another shell) will achieve this for UK users. (This is not relevant to Windows.)

If you are using a pre-10 version of Windows, and you want to run the shell command files that I provide to make life with **ee9** a little easier, or if you want to compile your own version of **ee9**, then you will need to install Cygwin, which provides a POSIX command set and API for Windows, including a bash-compatible shell named `dash`. A suitable version could be obtained from:

www.cygwin.com/install.html

B. GETTING A GNAT ADA 2012 COMPILER

If you want to build **ee9** yourself, you will need a recent version of GNAT, the GNU Ada compiler, because **ee9** is now written in Ada 2012. Current versions of GNAT for several systems, including macOS, Linux, and Windows, can be obtained with a GPL licence from AdaCore at:

www.adacore.com/download

On macOS, with the 2018 or later release of GNAT, you will also need to install **XCode** from the Mac App store.

GNAT is supported by Raspbian on the Raspberry Pi, so use the `apt-get` utility to fetch and install it, thus:

```
apt-get update
```

```
apt-get install gnat
```

Raspbian Ada compilers since the Debian Stretch release support Ada 2012.

There are compilers for other hardware/software combinations at:

www.gnu.org/software/gnat/

A possibility if you have a PowerPC (G3/G4/G5) Mac running OS X Leopard or Tiger is:

sourceforge.net/projects/gnuada/files/GNAT_GPL%20Mac%20OS%20X/2009-tiger-ppc/

Old compilers like this do not support Ada 2012, so you will need to be content with an older, Ada 2005, version of **ee9**.

If your system is not one of these, you may find pointers to a suitable port at the Ada Programming Wikibook.

If none of these meets your needs, try asking in the `comp.lang.ada` USENET newsgroup. People there are very helpful and may already have just the port you need.

BUILDING YOUR OWN VERSION OF **ee9**

The `Build` directory contains a command file called **mk9**, which is used to build **ee9** binaries. **mk9** takes optional parameters that determine the build flags to be used for the compilation. These flags can easily be amended if you find that they are unsuitable for your development environment: see the **mk9** file itself. The object programs are left in `Testing`.

The first parameter of **mk9** should be one of the following:

ee9: make an optimised binary, with a high level of runtime checking enabled (default)
warn: make an unoptimised binary, with many optional compilation warning messages enabled
clean: remove all the auxiliary files used by GNAT so as to make a fresh start with compilations
kal3: compile `Adjuncts/kal3.c`
KAlgol: convert the Kidsgrove components in `Adjuncts` to **ee9** form; compile `Source/kidopt.adb`, `Adjuncts/mkchan.c`, and `Adjuncts/kal3.c`
a2b: compile `Source/a2b.adb`
mtp: compile `Source/mtp.adb`
all: call **mk9** successively with the **a2b**, **mtp**, **KAlgol**, and **ee9** parameters
tidy: establish a standard set of files and permissions, using `Build/tidy`
zip: create a compressed 'zip' archive of emulation, named `<distribution>.zip`, using `Build/zipup`; **zipup** uses `Build/tidy`
distro: call **mk9** with the **all** parameter, and use `Build/pk9` to package the results; **pk9** uses `Build/zipup`

Note that optimised builds of **ee9** run about 4 times faster than unoptimised builds!

The second, *distribution*, parameter is given only if a build type is the first parameter, and is one of the following:

LINUX, **Linux**, or **linux**; or **macOS**, **Macintosh**, **Mac** or **OSX**; or **UNIX**, **Unix** or **unix**:

Build a binary, respectively, for Linux, for macOS, or for some other POSIX system that supports ANSI-terminal escape sequences and has `/dev/tty` as the interactive terminal. **macOS is the default**, so, e.g.:

mk9 ee9

has the same effect as:

mk9 ee9 macOS

This parameter is needed because, although the source code is identical in these three cases, different linker options are needed for best results.

WINDOWS, **Windows**, or **windows**:

Build a binary for Microsoft Windows, e.g.:

mk9 ee9 Windows

This parameter is needed because, as well as the linker issue, one source code file must be very slightly different to accommodate some Microsoft idiosyncrasies.

N.B. The PowerPC target is no longer supported by **mk9**. Raspberry Pi binaries are now created in the same way as all other UNIX/Linux targets, subject to the availability of an Ada 2012 compiler. One such is available in current Debian releases of Raspbian.

The compilation listing is left in a file named `Build/komlog.adb`; it should be free of any compilation warning messages. It starts with a header that identifies the **ee9** version, **mk9** build type, and compilation options used.

If you have difficulty in compiling **ee9** warning-free, please let me know, as it may indicate a portability defect. However, if the problem is only that the **gnatmake** command rejects a parameter of the form `-gnatwsomething` or the form `-gnatysomething`, it is safe to delete that whole parameter from the listed options within **mk9**, as it only controls optional warning messages, and they may differ between GNAT releases.

PACKAGING A VERSION OF ee9

The `Build` directory contains a command file called `pk9`, which is used by `mk9` to package up a version of `ee9` as a zip archive file. The only, optional, parameter is the *distribution* type, as used by `mk9`. It can be used on its own to make zip archives of existing versions of `ee9`.

THE USERCODE COMPILER, ka13

David Holdsworth's `ka13` (a new KDF9 Usercode compiler) is included in both source and binary forms with the appropriate download packages. It is trivial to (re-)compile `ka13`, using the `ka13` option of `mk9`.

ANOTHER KDF9 EMULATOR, kdf9

David Holdsworth's `kdf9` (a more basic KDF9 emulator than `ee9`) is written in C and is available in source form here:

settle.ddns.net/KDF9/

GETTING AND INSTALLING KIDSGROVE ALGOL UPDATES

A copy of Kidsgrove Algol is included with the `ee9` distribution. It may be worth checking David Holdsworth's website to see whether a newer version is available. If so, copy `systape.txt`, `mksys2.bin` and `KAB00.bin`, into the `Testing/Adjuncts` directory. They need to be converted to `ee9` format. Put any updated sources of `ka13.c`, or `mkchan.c` in the `Adjuncts` directory. They need to be compiled. To complete the task, run `mk9` with the `KAlgol` option from within the `Build` directory.

David Holdsworth's material can presently be found at:

settle.ddns.net/KDF9/kalgol/DavidHo/

but a web search should find it if it gets moved to a new server.