



FORCHECK

for

HP-UX

A Fortran Verifier and Programming Aid

version 14

Installation Guide

January 22, 2011

The information in this document is subject to change without previous notice and should not be taken as a commitment by Forcheck b.v. Forcheck b.v. can not assume responsibility for any errors which may appear in this document.

The software described in this document is furnished under a license and may be used, copied or disclosed only when in accordance with the terms of this license.

Copyright ©Forcheck b.v. 1984 through 2011. All rights reserved.
FORCHECK has been developed by Erik W. Kruyt.

FORCHECK is currently available for PC/Windows, PC/Linux and HP-UX.

FORCHECK is a registered trademark of Forcheck b.v.

Absoft is a trademark of Absoft Corporation.

DEC, PDP, VAX, AXP, Alpha, RSX, VMS, OpenVMS, Ultrix and Tru64 UNIX are trademarks of Hewlett Packard Company.

DR Fortran-77 is a trademark of Digital Research, Inc.

FTN77 and FTN95 are trademarks of Salford Software Ltd.

FTN90 is a joint trademark of Salford Software Ltd and the Numerical Algorithms Group Ltd.

Hewlett-Packard, UX, Fortran/9000 are trademarks of Hewlett-Packard Company.

IBM, MVS, VS Fortran, Professional Fortran, RS/6000 and AIX are trademarks of International Business Machines Corporation.

Intel is a trademark of Intel Corporation.

Cray, Unicos, CF77 and CF90 are trademarks of Silicon Graphics, Inc.

Silicon Graphics, IRIX and MIPSpro are trademarks of Silicon Graphics, Inc.

Lahey, F77L, LF90 and LF95 are trademarks of Lahey Computer Systems, Inc.

Linux is a registered trademark of Linus Torvalds.

Microsoft, MS-DOS, MS-Fortran, Microsoft Fortran PowerStation, Windows 95, and Windows NT are trademarks of Microsoft Corporation.

MicroWay and NDP Fortran-386 are trademarks of MicroWay, Inc.

NAG and NagWare are trademarks of The Numerical Algorithms Group Limited.

Prospero Fortran and Pro Fortran-77 are trademarks of Prospero Software.

Ryan-McFarland and RM/Fortran are trademarks of Ryan-McFarland Corporation.

Sun and Solaris are trademarks of Sun Microsystems, Inc.

WATCOM is a trademark of Sybase, Inc.

All other trademarks and registered trademarks are the property of their respective holders.

Website: <http://www.forcheck.nl>

Email: info@forcheck.nl

Contents

1 Introduction	5
2 Distribution	7
3 Installation	9
3.1 Retrieving the distribution files	9
3.2 Installation on a private account	9
3.3 System wide installation	10
3.4 Checkout	10
4 Environment	11
4.1 Operating system	11
4.2 Copyright message	11
4.3 Configuration file	11
4.4 Exit status	12
4.5 Help	12
4.6 User's guide	12
5 Customizing	13
5.1 Compiler emulation	13
5.2 Messages	13
5.3 Remarks	14
5.4 Summary of global variables	14

Chapter 1

Introduction

FORCHECK is a Fortran program development, conversion and maintenance tool. It parses Fortran programs, verifies the syntax and composes documentation. It analyzes both separate program units and the program as a whole. FORCHECK verifies the syntax by parsing the source program. This is done as precisely as possible at compile time. The full Fortran 2003 syntax (which includes the Fortran 95, Fortran 90 and FORTRAN 77 syntax) is supported. Moreover most language extensions of many compilers and most Fortran 2008 features are accepted. As an option the syntax can be checked for strict conformance to the Fortran 2008, Fortran 2003, Fortran 95, Fortran 90, or FORTRAN 77 standard.

Cross-reference tables of all objects within program units are composed. Information and warnings concerning the usage of these objects are provided.

The reference structure (call tree) of the program can be analyzed and presented. Recursive references are traced.

The consistency of the entire program is verified by checking the type of the procedures and the argument lists of all procedure references. Length, data type and structure of the common blocks specified in the various subprograms are compared. Cross-reference tables of all procedures, common blocks, common-block objects, modules, external I/O and include files over the program are composed.

FORCHECK can emulate the language extensions of a specific compiler by reading a configuration file in which all data types and extensions to be supported are enumerated.

The global information of each program unit can be stored in library files which can be referenced and updated in subsequent FORCHECK runs to test program units in the context of the entire program. A librarian utility is supplied to maintain the FORCHECK library files.

Chapter 2

Distribution

The distribution kit contains the following files:

executables:

<code>forchk</code>	FORCHECK executable
<code>fcklib</code>	FORCHECK library utility
<code>interf</code>	FORCHECK interface builder

documentation files:

<code>INSTALL</code>	installation notes
<code>fckinstall</code>	script to install FORCHECK
<code>HPUX_I.pdf</code>	The Installation Guide in PDF format
<code>unix_cmd1.pdf</code>	The user Guide in PDF format
<code>fxdf.txt</code>	List of supported language extensions
<code>libfile.txt</code>	Description of FORCHECK library files
<code>LICENSE.txt</code>	The license agreement

man pages:

<code>forchk.1</code>	unformatted on-line documentation of FORCHECK
<code>fcklib.1</code>	unformatted on-line documentation of <code>fcklib</code>
<code>interf.1</code>	unformatted on-line documentation of <code>interf</code>

on-line help files:

<code>fckhlp.txt</code>	help option text of <code>forchk</code>
<code>fckinhlp.txt</code>	help option text of <code>interf</code>
<code>fcklibhlp.txt</code>	help option text of <code>fcklib</code>

ancillary files:

<code>fckerr.msg</code>	direct access file with messages
<code>_fck_tree.xml</code>	Style sheet for xml output
<code>*.cnf</code>	configuration files for compiler emulations
<code>intrmods.flb</code>	interface library for intrinsic modules
<code>MPI.zip</code>	interface library for MPI
<code>intel.zip</code>	interface library for Intel compiler

examples/demonstration files:

<code>fckdem.f</code>	demonstration program
<code>FCKDEM.INC</code>	include file for demonstration program

fckdem.lst

output of analysis of demonstration program

Chapter 3

Installation

3.1 Retrieving the distribution files

- Download the distribution kit from the ftp server and place the compressed tar file e.g. on the tmp directory.

Now you can uncompress and retrieve the files to an installation directory:

```
gunzip /tmp/forcheck-14.1-HPUX.tar.gz
tar -xvf /tmp/forcheck-14.1-HPUX.tar
```

- Retrieve the password file from the email. Mind that the password file attached to the email is in DOS format and must be transformed to Unix format by replacing the cr/lf's by linefeed's only. Place the password file in the share/forcheck/ directory of the install tree in the installation directory.

3.2 Installation on a private account

To use FORCHECK for one user only or during an evaluation period you may want to install FORCHECK not system-wide but keep it on a private account.

- FORCHECK tries to open the message, password and help files using the environmental variable FCKDIR. Define an environmental variable with the directory name of the installation directory. For the C-shell:

```
setenv FCKDIR ~/forcheck-14.1
```

or for the bash, bourne or korn shell:

```
export FCKDIR=~/forcheck-14.1.
```

- Add the bin directory of the install tree to your PATH variable, e.g.:

```
export PATH=$PATH:forcheck-14.1/bin.
```

3.3 System wide installation

- Make sure you have root privileges.
- Apply the installation script `fckinstall` to move the files to the correct path, set the protection codes and generate the man pages:

```
chmod +x fckinstall
./fckinstall
```

If you install FORCHECK in another directory then `usr/local` you have to set the environmental variable `FCKDIR` to define the base (prefix) of the install path.

3.4 Checkout

- Check the man pages:

```
man forchk
man fcklib
```

- Run `forchk` and analyze the demonstration program:

```
forchk -l fckdem.lst share/forcheck/examples/demo/fckdem.f
```

- You can now compare the output file with the supplied output file:

```
diff fckdem.lst share/forcheck/examples/demo/fckdem.lst
```

Because the system, date and time will be different, the page headers will be shown as differences.

When `fcklib` compresses a library file, it creates a temporary file `.#fcklib.tmp`, which is deleted after successful compression. If, however, `fcklib` ends abnormally, the user will find this file on his current directory.

Chapter 4

Environment

4.1 Operating system

FORCHECK runs under the HP-UX operating system.

4.2 Copyright message

You can suppress the copyright message by setting the environmental variable FCKCPR to the keyword QUIET. For the C-shell:

```
setenv FCKCPR QUIET
```

or for the bash, bourne or korn shell:

```
FCKCPR=QUIET  
export FCKCPR
```

4.3 Configuration file

To emulate a specific compiler and to tune the usage of FORCHECK the user can apply a different configuration file than the default. To use a specific configuration file the user must set the environmental variable FCKCNF with the name of the configuration file to be used. For example, to emulate the VAX Fortran compiler type, using the C-shell:

```
setenv FCKCNF /usr/local/share/forcheck/vax.cnf
```

or for the bash, bourne or korn shell:

```
FCKCNF=/usr/local/share/forcheck/vax.cnf  
export FCKCNF
```

4.4 Exit status

When forchk exits, it generates a specified exit status which can be used in for example a script file.

exit status:

- 0 no informative, warning, overflow or error messages presented
- 2 informative, but no warning, overflow or error messages presented
- 4 warning, but no overflow or error messages presented
- 6 table overflow, but no error messages presented
- 8 error messages presented
- 16 fatal error occurred

4.5 Help

Man pages are supplied. Users can access the man pages through the man command.

4.6 User's guide

The user's guides are supplied as pdf files. There is a separate guide for using forchk from the command line and using the IDE.

Chapter 5

Customizing

5.1 Compiler emulation

FORCHECK can be used as a tool to convert Fortran programs from one system to another. FORCHECK can support most of the compiler syntax extensions of many compilers. During its initialization FORCHECK reads a configuration file in which relevant data for the emulation is stored. If the environmental variable FCKCNF has been specified, FORCHECK uses this path to read the configuration file. If FCKCNF has not been specified, FORCHECK reads the default configuration file:

```
/usr/local/share/forcheck/hpf90.cnf
```

If a user wants FORCHECK to emulate a different compiler, he must set the environmental variable FCKCNF to the path of the configuration file to be used. For example for the C-shell:

```
setenv FCKCNF /usr/local/share/forcheck/gfortran.cnf
```

or for the bash, bourne or korn shell:

```
FCKCNF=/usr/local/share/forcheck/gfortran.cnf
```

```
export FCKCNF
```

to emulate gnu fortran.

If a user wants to enable different Fortran extensions than the default the user must have his own copy of the appropriate configuration file.

5.2 Messages

We distinguish three kinds of messages, namely operational messages, analysis messages and system messages. All analysis and system messages are stored in the direct access file `fckerr.msg`.

If a user wants to adapt certain analysis messages to his needs, he must place the numbers and new level flags of these analysis messages in an option file which can be concatenated to the configuration file to be used (for the C-shell):

```
setenv FCKCNF /usr/local/share/forcheck/sun.cnf;options_file
```

or for the bash, bourne or korn shell:

```
FCKCNF=/usr/local/share/forcheck/sun.cnf;options_file
export FCKCNF
```

In the option file the user can also specify the line or statement count_mode options of his choice. See the section on messages in the user's guide.

5.3 Remarks

- Input files and options are placed in a direct access scratch file `.#f1xxxx.tmp` (in which `xxxx` is a random number) , which will be deleted automatically on exit.
- The annotated source lines of each program unit are temporary stored in a sequential scratch file `.#f3xxxx.tmp` (in which `xxxx` is a random number) , which will be deleted automatically on exit.
- When `forchk` encounters an `INCLUDE` line, or an include preprocessor directive, it tries to open the include file specified. See the user guide for the search strategy of include files.
- A scratch library file `.#f2xxxx.tmp` (in which `xxxx` is a random number) is created, which be deleted automatically on exit.

5.4 Summary of global variables

FCKCNF	pathname of configuration file to be used
FCKCPR	suppression of copyright message
FCKDIR	base directory of FORCHECK installation tree
FCKPWD	pathname of the password file
FCKOPT	default options
TMPDIR	directory for scratch files