

Installation of OCS

for Version 2.3n

by Wolfgang Grieskamp and Klaus Didrich
Copyright © 2000 The Opal Group

This document describes the configuration and installation of the Opal compilation system OCS, version 2.3d or later.

- The hotshot is served with Chapter 3 [Quick Installation], page 2
- For the requirements to install and run OCS, see Chapter 2 [Requirements], page 1.
- The complete configuration procedure is described in Chapter 4 [Configuration], page 2. For the diverse configuration options, see Section 4.3 [Customizing the Configuration], page 3.
- The building and installation procedure is tackled with in Chapter 5 [Installation], page 5.
- Notes on installation problems are found in Chapter 9 [Problems], page 7.

1 General Information

Starting with release 2.1e OCS is configured on the base of the GNU `autoconf` utility. This shall provide an automatic adaption of OCS independent of the system type and local file organization.

The automatic configuration is still somewhat experimental, and has been validated only for the following systems:

SunOS >=5.3 Linux

Important: If you already have OCS installed make sure that none of the OCS executables are in your PATH and that the OCS environment variable is unset. Otherwise, the installation might produce unpredictable results. The safest option is to remove the old OCS installation beforehand.

2 Requirements

To install and run OCS you need the GNU `make` utility version 3.64 or later and an ANSI-C compiler, preferable `gcc`. You definitively cannot use OCS with on older version of GNU-make than 3.64. To find out the version, type:

`gmake --version`

or

`make --version`

whatever is the name of GNU make on your system.

You will need around 220 MB of free disk space during installation (even more, if you plan to install contributions). Permanently, around 60 MB is required.

3 Quick Installation

To configure and install OCS, run

```
./configure  
gmake install
```

This will install OCS in ‘`/opt/ocs-VERSION/`’ path.

It will take around 10–15 minutes on a 2.4 GHz stand-alone computer for the complete installation.

Since the installation is somewhat time expensive, and hence trial-and-error is less appropriate, you should consider to study the sections below carefully before taking over.

4 Configuration

4.1 Preparing the Configuration

The automatic configuration procedure will lookup the existence and locations of several UNIX tools such as the C compiler, the `make` program and so on. The lookup is done on the base of the environment variable `PATH`. Ensure that `PATH` is set such that your preferred variants of the tools will be found.

In particular, you should ensure:

- an ANSI C compiler is found in the path (such as `gcc`), either under the name `gcc` or `cc`.
- GNU `make` is found, either under the name `gmake` or `make`.

The default configuration will write the absolute pathes of the tools into the configuration files. This behaviour can be controlled with the option `--enable-absolute-pathes` (see Section 4.3 [Customizing the Configuration], page 3).

It should be noted that re-configuration of OCS will be necessary if the tools are moved to other locations, unless you have explicitly disabled absolute pathes. Disabling absolute pathes is not recommended on multi-user systems, since the functionality of OCS will then depend on personal user configurations. When using absolute pathes it is a good idea to have the tools at standard places such as ‘`/usr/local/bin`’ and ‘`/usr/bin`’.

It is probably the easiest way to run the configuration procedure itself to inspect what tools will be chosen under the current `PATH`, and to re-configure if the result does not satisfy.

4.2 Running the Configuration

Cd to the source directory of the distribution, and run:

```
./configure --prefix=install-dir feature ... feature
```

where *install-dir* is the directory where OCS shall be installed, and *feature* is described in Section 4.3 [Customizing the Configuration], page 3.

The run of `./configure` will check out the capabilities of your system. For this purpose it will create C programs on the fly, compile them, and occasionally run them. For security reasons you should not run `./configure` as root.

The `./configure` program automatically maintains a cache of the given features and the calculated results to be used for its next run. This may be useful to selectively enable or disable a particular feature without examining the system again. However, it should be noted that `./configure` is not capable of tracking any side-effects induced by features or environment variables such as `PATH` on the validness of the cache. To get rid of such effects, run

```
rm config.cache
```

before you rerun `./configure`.

4.3 Customizing the Configuration

Several options may be passed to the `./configure` program to customize the configuration. For each option of the form `--enable-feature` an according `--disable-feature` option exists, which deselects the given *feature*. Some Opal programs require or can benefit from external libraries, such as GNU readline, Tcl, Tk, You can disable the use of these libraries or give the necessary compiler flags if `./configure` is unable to find them

4.3.1 General options

`--prefix=install-dir`

directory where OCS should be installed. Defaults to `‘/usr/local/’`. The installation will create a subdirectory `ocs-VERSION` within the install directory.

`--enable-dynamic`

enable dynamic linking and support for shared libraries. Default is enabled. The OASYS interpreter depends on dynamic linking.

`--enable-absolute-paths`

burn the locations of tools at installation time into the installed OCS. This is the default.

`--enable-doc`

Install documentation. This is enabled by default.

4.3.2 Opal Features

All these features are enabled by default.

`--enable-dosfop`

install the DOSFOP documentation system

`--enable-opalwin`

install the OpalWin library. If you don't have X, or TCL/Tk you should disable this feature.

- `--enable-oasys`
install the OASYS interpreter. The interpreter depends on TCL (not on Tk), and on dynamic linking, and it benefits from the GNU readline library.
- `--enable-reflections`
install the reflections library for OPAL. See the documentation for further details.

4.3.3 Compiler and Linker Flags

These flags are left empty by default.

- `--with-cflags=flags`
additional flags to be passed to the C compiler on every compilation.
- `--with-ldflags=flags`
additional flags to be passed to the C compiler on every linkage.
It should be noted that flags defined by `--with-cflags` are not passed automatically on linkage. If flags are common both to compilation and linkage, provide them twice.
- `--with-libs=libs`
additional libraries (`-llib` and `-Lpath`) to be used for linking.
- `--with-dldflags=flags`
additional flags to be passed to the C compiler on linking a shared (dynamic) object. These flags will be appended to the flags given with `--with-ldflags`.
- `--with-dldlibs=libs`
additional libraries to be used for linking a shared (dynamic) object. These libraries will be appended to the libs given with `--with-libs`.

If the following flags are not given, `configure` tries to find out the type of the linker and sets these flags accordingly.

- `--with-ldrpath=flags`
define runtime path for linking a dynamic library.
- `--with-ldrpathlink=flags`
define linktime path for linking a dynamic library.
- `--with-soname=flags`
define flag to set internal SONAME field.
- `--with-ldstatic=flags`
flag to mark static link targets.
- `--with-lldynamic=flags`
flag to mark dynamic link targets.

4.3.4 External Libraries

Options for external libraries come in pairs, one for the linker flags and one for location of the include flags. If either of them is unset (using the corresponding `--without` option), the library will not be used.

If the flags argument contains spaces, enclose the whole commandline argument in quotes, e. g. `"--with-XXX-lib=-LPATH1 -RPATH1 -llib1"`

`--with-readline-lib=LIBFLAGS`

`--with-readline-incl=INCLFLAGS`

Specify location of the GNU readline library. (Define the include path such that `readline/readline.h` can be successfully included.)

`--with-tcl-lib=LIBFLAGS`

`--with-tcl-incl=INCLFLAGS`

Specify location of the TCL library.

`--with-tk-lib=LIBFLAGS`

`--with-tk-incl=INCLFLAGS`

Specify location of the Tk library.

5 Installation

After successful configuration, OCS is installed by running

```
gmake install
```

in the distribution directory (or `make`, whatever is the name of GNU make on your system). Make sure that you have write permission in the installation directory (i.e. `/opt` or the directory you explicitly provided with a `--prefix` option).

This will bootstrap the Opal compiler, check and build the library and selected features, and install everything at the place you have specified with the `--prefix` option on configuration time.

The whole process will take around 10–15 minutes on a 2.4 GHz Core2 CPU.

6 Security

If you want to do a system-wide installation, but do not want to run the whole configuration and installation process as user `root`, you can proceed as follows.

1. Log in as `root`. Create an empty directory `install-dir/ocs-VERSION`, e. g. `/opt/ocs-2.3n`. Make this directory writable for the user who does the installation, and log out.
2. Configure and install the OCS system as described above. Set up the prefix, if necessary.
3. Log in as `root` again, remove the write permission from `install-dir/ocs-VERSION`.

7 Preparation Per User

You have to extend the `PATH` such that the ocs compiler driver is found:

```
PATH="install-dir/ocs-VERSION/bin:$PATH"
```

8 Testing the Installation

If you completed the installation successfully, you have already used the new compiler. If you want to increase confidence in your installation, you can run the following checks.

The first check is to invoke `ocs`:

```
% ocs info
You are using 'ocs-2.3n (04-Oct-2010)'
located at '/opt/ocs-2.3n'.
The project ($OCSPROJECT) is not specified.
```

If this test fails, you probably have not set your `PATH` environment variable.

For the following tests copy the examples directory from the installation to your home directory (or another tests directory of your choice):

```
% cp -r install-path/ocs-2.3n/examples ~
```

The next check compiles a program which does not use any extra libraries:

```
% cd ~/examples/Integrator
% ocs
% ./integral
```

The compiled program asks for a function and two real numbers for input. Use e. g. `(SIN x)` for the function. To quit, input an empty string for the value `a` and for the function `f`.

If you installed the `OPALWIN` library, you can compile the following example:

```
% cd ~/examples/Graphics/Queens
% ocs
% ./queens
```

The program visualizes the search for the solutions of the eight-queens problem.

Test the Opal reflections by executing one of the example programs:

```
% cd ~/examples/Reflections/PrettyPrint
% export OCSPROJECT=$HOME/examples/Reflections/ProjectDefs.reflections
% ocs
% ./PrettyPrint
```

Finally, call the `OPAL` interpreter:

```
% cd ~/ocs/examples/Rational
% oasys
oasys version 1.1e (ocs version 2.3n), (c) 1989-2001 The Opal Group
> a Rational.sign
> f Rational.sign
Rational.sign>e 6/8 + 3/5
checking Rational.sign
checking Rational.impl
compiling Rational.impl
starting evaluator process
27/20
Rational.sign>q
```

9 Problems

9.1 POSIX

A main source of problems of the installation of OCS is the UNIX Opal library (`'src/lib/System/Unix'`). This library is implemented on the base of POSIX, which is not supported completely by some systems.

If `./configure` detects incompatibilities with POSIX, a fall back strategy is chosen, which means in the worst case that the incompatible feature is disabled at all. However, not all incompatibilities can be detected by `./configure` and fallback strategies have not been implemented for all cases yet.

Some systems may require a specific compiler switch to enable POSIX support which `./configure` fails to detect. On such systems OCS may not compile at all, or only with a largely reduced functionality of the UNIX library (see the `./configure` output).

If your C compiler and system headers propose to support POSIX, it isn't straightforward to say whether you should enable the support or not. Just defining `-D_POSIX_SOURCE` since the headers make a distinction about it is generally unsafe, since only the prototypes but not the libraries bound with a program are effected.

9.2 Handcrafting

It should never be necessary to handcraft the configuration files of the OCS distribution, but here is what you should know in such a case.

The files `'src/om/specs/Specs.basic'` and `'src/om/specs/ShSpecs.basic'` contain the specification of where to find the UNIX tools and how to call the C compiler. Both files basically contain the same information: the first one as a `make` file and the second a `sh` file. If you want to change compiler flags (e. g. optimization flags) or the names of the tools (e. g. if you prefer `gzip` to `bzip2`), you must change both files, before calling `gmake/make`.

The file `'src/lib/Internal/Compiler/unixconfig.h'` describes the view of OCS onto the UNIX host system.

9.3 Re-Configuring

If you want to change the configuration (e. g. because TCL/TK has been installed in the meantime) the easiest way is to redo the installation process. Use the target `reinstall` in the invocation of `gmake/make` to indicate that you really want to overwrite an existing installation.

If you kept the intermediate files from the previous installation, it is faster to invoke `gmake/make` with the particular targets. A complete list of possible targets is contained in a commentary in the toplevel Makefile, just before the definition of `STDPACKAGES`.

Moving the installation to a different directory creates problems, if shared libraries are used. You may need to set the environment variable `LD_LIBRARY_PATH` such that the Opal shared libraries are found. This is done automatically for most of the Opal tools (compiler, interpreter), but not for the executables generated by the Opal compiler.

Table of Contents

| | | |
|----------|---------------------------------------|----------|
| 1 | General Information | 1 |
| 2 | Requirements | 1 |
| 3 | Quick Installation | 2 |
| 4 | Configuration | 2 |
| 4.1 | Preparing the Configuration | 2 |
| 4.2 | Running the Configuration | 2 |
| 4.3 | Customizing the Configuration | 3 |
| 4.3.1 | General options | 3 |
| 4.3.2 | Opal Features | 3 |
| 4.3.3 | Compiler and Linker Flags | 4 |
| 4.3.4 | External Libraries | 5 |
| 5 | Installation | 5 |
| 6 | Security | 5 |
| 7 | Preparation Per User | 5 |
| 8 | Testing the Installation | 6 |
| 9 | Problems | 7 |
| 9.1 | POSIX | 7 |
| 9.2 | Handcrafting | 7 |
| 9.3 | Re-Configuring | 7 |