



SkyOS
Developer network library

Porting software to SkyOS
© SkyOS, 2005



Topics

Introduction.....	3
Cross compiler.....	3
Development environment.....	3
Get the source.....	3
Compile.....	4
If something doesn't work.....	4
Frequent Pitfalls.....	4
optarg, optind, errno.....	4



Introduction

When porting GNU / OSS software to SkyOS, you can use two different ways

1. Crosscompiling from Windows (tm)
2. Compiling native inside SkyOS

At this moment, crosscompiling is the preferred way. You can use your favorite IDE like Visual Studio .NET (makes porting big projects much easier) and you can use the SkyOS cross compiling suite.

Cross compiler

1. Download cygwin from www.cygwin.com
2. Download the cross compiler from <http://www.skyos.org/build.zip>. Install it to any location on your harddrive.
3. Download the SkyOS SDK.
If you got a SkyOS beta version, the SDK will be in /addon/packages/sdk.pkg on your ISO CDROM. The pkg file is just a ZIP file, so you can use your favorite ZIP application to unzip it.

Unzip the SDK to c:\skyos. Files like stdio.h and stdlib.h should now be in <c:/skyos/include>.

Development environment

1. Path environment variable

Add the path to the build\bin folder to your windows path environment variable. For instance: PATH=c:\skyos\tools\cross\build\bin;<c:/windows>;.....

Get the source

Download the source package of the software you want to port. Copy the build template into the same directory where the configure file of the source package is located.

Create a template bash (for instance: build.sh) script file with following content in the root directory of the software package you are going to port to SkyOS:

```
export AS="c:/skyos/tools/cross/build/bin/i386-skyos-pe-as.exe"  
export GCC="c:/skyos/tools/cross/build/bin/i386-skyos-pe-gcc.exe"  
export CC="c:/skyos/tools/cross/build/bin/i386-skyos-pe-gcc.exe"  
export LD="c:/skyos/tools/cross/build/bin/i386-skyos-pe-ld.exe"
```



```
./configure --host=i386-pe-skyos  
make
```

Compile

Open a command prompt and change your working directory to the root directory of the software tree you are going to port.

Type `sh ./build.sh`

The application should now be compiled.

If something doesn't work

If the configure process stops with a message like "C compiler doesn't work", make sure that your cross-compiler is in the environment path. Furthermore, replace all `confstest${ac_exeext}` occurrences in configure with `confstest.app`
Try again.

Frequent Pitfalls

optarg, optind, errno

These variables are normally declared as extern, either in the C file itself or in application included header files. (For instance: `extern int errno;`)

In SkyOS, these variables are DLL exported and predefined for you already.

This means, that defining them as extern (what unfortunately is done by a few GNU/OSS programs) will not create a compiler/linker error, but SkyOS will not be able to link them when loading your application.

In order to work around this, just include „`libskyos.h`“ and remove all extern declarations to these variables you find.

References

[1] SkyOS Software store and package files

<http://www.skyos.org/downloads/documents/SkyOS%20Software%20store%20and%20package%20files.pdf>