

Windows_Vista

This article discusses some issues that may appear when using Symbian S60 v9.2 3rd Edition Feature Pack 1 under Windows Vista. Its intended to provide some temporary patches to make things work until official support is provided.

Please note that the actual patched versions of these files are available [on this page](#)

Contents

- [1 PATH environment variable](#)
- [2 cl_bpabi.pm Issues](#)
 - ◆ [2.1 GCCLibPath and GCCInstallPath subs](#)
- [3 Debugging with Carbide.c++](#)
- [4 Related Links](#)

PATH environment variable

The way Windows Vista uses default paths defined with the PATH variable is slightly different from older versions. You should check that the following paths are included in it.

```
<CSL Arm Toolchain install path>\bin;  
<CSL Arm Toolchain install path>\arm-none-symbianelf\bin;  
<CSL Arm Toolchain install path>\libexec\gcc\arm-none-symbianelf\3.4.3;
```

Also note that the path

```
C:\Program Files\Common Files\Symbian\Tools;
```

shouldn't be declared before them. If you get an error like *as.exe was called but is not supported in this release* then you don't have the path variables set up correctly.

You can change the PATH variable at **My Computer > Properties > Advanced System Configuration > Environment Variables > System Variables**. You'll be asked to give Administrator Permission during the process.

cl_bpabi.pm Issues

This is one of the script that helps creating the makefile of your project. You'll have to apply some patches to it before getting a correct makefile and being able to compile and link your projects.

GCCLibPath and GCCInstallPath subs

These two subs are intended to provide the absolute CSL Arm Toolchain install path. They try to extract the path from this call:

```
arm-none-symbianelf-g++ -print-libgcc-file-name
```

Under systems older than Windows Vista, this call should return the absolute path to **libgcc.a**. **GCCLibPath** and **GCCInstallPath** rely on this fact to get CSL Arm Toolchain install path.

However, under Windows Vista, that call will just return the name of the library, so they can't extract any path and return libgcc.a. Thus **cl_bpabi.pm** will end up generating a corrupt makefile.

A simple way to get around it is to manually provide those paths. You can just enter the following line at the beginning of each sub (you also can replace all their code, but I always find interesting keeping previous versions)

```
return "<CSL Arm Toolchain install path>/bin";
```

and

```
return "<CSL Arm Toolchain install path>/lib/gcc/arm-none-symbianelf/3.4.3";
```

at **GCCInstallPath** and **GCCLibPath** respectively. This way you should be able to compile your project.

Debugging with Carbide.c++

For debugging to work on Carbide.c++ you must put the application into XP compatibility mode. Otherwise you might get errors like *Error starting debug process 87 57*.

Hopefully some future version of Carbide/S60 SDK will be Vista compatible from the start.

Related Links

- [Moving to Windows Vista](#)
- [A deal between S60 3rd emulator and Vista the 10 seconds fix](#)