A Brief Introduction To Using CCPSX

This file contains a description of the main features of CCPSX.

What It Does

CCPSX is a generic control program that accepts a list of source files and option switches and then calls a C pre-processor, C compiler, assembler and/or linker as appropriate to produce the required output.

Command Line Format

The CCPSX line consists of a sequence of source files and control switches. Control switches are prefixed with a '-' sign.

e.g.

```
ccpsx -c main.c
```

N.B. The case of alphabetical switches is important. -O does not have the same effect -o.

Long command lines can be stored in control files. By using an '@' sign in front of the control file name the contents of the control file can be embedded in the command line. e.g.

```
ccpsx @main.cf  -o main
```

This will embed the contents of main.cf in the command line before the -o option.

The contents of the control file can be split across several lines without the need to use any special characters. An end of line in a control file is treated as a space.

You can specify as many control files as you want on the command line and a control file can even reference another control file.

Environment Variables

CCPSX will search for various environment variables to specify where the files it requires can be found. These environment variables should be set up by your AUTOEXEC.BAT so that the required variables are always set up each time you boot your computer.

The environment variables are :

COMPILER_PATH This specifies the path to the directory where the compiler executable files are stored. If you are using the GNU C compiler then CCPSX will look for the files CPPPSX.EXE, CC1PSX.EXE and CPLUSPSX.EXE in this directory.

PSYQ_PATH This specifies the path to the directory where the PSY-Q executable files are stored. (PSYLINK.EXE, etc.)

C_INCLUDE_PATH This specifies the path to the directory where the standard C header files are stored. This is the directory that is searched when a #include statement specifies the file name in angle brackets (e.g. #include <stdio.h>)

LIBRARY_PATH This specifies the path to the directory where the standard library files are stored.

TMPDIR This specifies the path to a directory where temporary files are created during compilation.

e.g.

```
set COMPILER_PATH=c:/compiler
set PSYQ_PATH=c:/psx/bin
set C_INCLUDE_PATH=c:/psx/include
set LIBRARY_PATH=c:/psx/lib
set TMPDIR=c:/
```

Note : The paths use forward slashes ('/') rather than back slashes ('\') to separate the directories in the path.

Source Files

The specified source files can be either C or assembler source files or object files. CCPSX decides how to deal with a source

file based on the files extension.

The following table describes how each file extension is processed :

.C : Pass through C pre-processor, C compiler, Assembler, Linker
.I : Pass through C compiler, Assembler, Linker
.CC : Pass through C pre-processor, C++ compiler, Assembler, Linker
.CPP : Pass through C pre-processor, C++ compiler, Assembler, Linker
.II : Pass through C++ compiler, Assembler, Linker
.IPP : Pass through C++ compiler, Assembler, Linker
.ASM : Pass through C pre-processor, Assembler, Linker
.S : Pass through Assembler, Linker
other : Pass through Linker

The DOS file system is not case sensitive and so the case of the extension has no effect.

Various command line switches can stop processing at any stage eliminating, linking, assembling or compiling.

The -x option can be used to override the automatic selection of action based on file extension, see below for more details.

Note : any file with an extension that is not recognised is treated as an object file and passed to the linker.

Option Switches

The following are the most common command line options.

Note that case is important. -O is not the same as -o.

Options controlling the type of output

-E Pre-process only. If no output file is specified then output is sent to the screen (standard output).

-S Compile to assembly language. If no output file is specified then .C files are compiled to a file with the same name but with a .S extension.
.ASM files are pre-processed as specified in the -E option above.

-c Compile to object files. C files are compiled and assembled to .OBJ files. Assembler files are just

assembled. If an output file is specified then all output is sent to this file, otherwise it is sent to a file with the same name as the original source file but with a .OBJ extension.

If none of the options listed above are used then the linker will be called. If an output file name is specified then the linkers output will go to this file. If no output is specified then the linkers output is written to a file called A.OUT

Specifying the output file

If you do not wish to use the default output then the output file should be specified with the -o option. e.g.

```
ccpsx main.c -o main
```

Will compile main.c and link the program to produce a file called MAIN as the final output.

Note : It is possible to compile several separate programs in one command by specifying all the program names, e.g.

```
ccpsx -c file1.c file2.c file3.c
```

will compile file1.c, file2.c and file3.c to file1.obj, file2.obj and file3.obj respectively. If an output file is specified then the output from each separate compilation will overwrite this file each time and so only the final program to be compiled will produce any output.

Generating debug info

To force the C compiler to generate the information required for debugging the command line switch -g should be used. e.g.

```
ccpsx -g main.c -o main
```

Optimisation

Optimisation is controlled by use of the -O switch. Various levels of optimisation are possible :

-O0 (default) no optimisation
-O or -O1 standard level of optimisation
-O2 full optimisation

Other types of optimisation may be controlled by other compiler switches. See the your compiler's documentation for more details.

Note : If you compile with -g and -O simultaneously, this can lead to some strange effects during debugging as substantial changes to the code order and variable storage can result.

General

-W

Suppress compiler and pre-processor warnings.

-DNAME
-DNAME=VALUE

These options are passed to the pre-processor and pre-define the symbol NAME (to VALUE if specified) before processing begins.

-UNAME

Undefines the pre-defined name NAME before pre-processing starts.

-v

This option will cause ccpsx to print every command it is about to execute before executing it.

-kanji

This option will cause a special stage to be run that will allow kanji 2-character sequences to be correctly dealt with by C compilers.

Linker

The linker used is the standard PSY-Q linker. The following switches are relevant to the action of the linker :

(Options starting -X correspond to the option of the same name in the PSY-Q standard linker documentation).

-nostdlib This will suppress the automatic linking with the PSX standard libraries (libgs, libgte, libgpu, libetc, libapi and libsn).

-l libname Include the specified library when linking.

-Xo$xxxx ORGs the linker's output to address xxxx (hexadecimal) The default org address is 0

-Xd When downloading the linkers output directly to the PSX, this option will prevent the program from starting to execute until the user starts it from the debugger.

-Xb May be required when linking a particularly large program. This allows the program to be linked but will slow down linking if it need not have been specified.

The output type is specified as follows :

To send output direct to the target

-o t0:

(The 0 in t0 is the scsi id of the target you wish to talk to. This is normally 0 but may be any number in the range 0 to 7)

To write output to a file :

-o filename

The file produced will normally be a .CPE file. To produce a file which is just pure binary specify the option -Xp on the command line. The binary file produced will be based at the org address specified.

To produce a symbol file for debugging purposes the name of the symbol file should follow the destination file name, separated by a comma (and no spaces). e.g.

ccpsx main.c -o main.cpe,main.sym

will send the program to MAIN.CPE and symbols to MAIN.SYM.

To produce a map file the destination map file name should follow the symbol file name, separated from it by a comma

(and no spaces). e.g.

ccpsx main.c -o main.cpe,main.sym,main.map

will write a map file to MAIN.MAP.

Other standard PSY-Q linker options can be specified by using -Xoption in the same way that /option would be used when calling the linker directly.

The linker will always set the entry point of the program to __SN_ENTRY_POINT. This symbol is defined in the standard library file SNLIB.LIB.