IOP Programming Tools

Publication date: October 2001

Sony Computer Entertainment Inc.
1-1, Akasaka 7-chome, Minato-ku
Tokyo 107-0052, Japan

Sony Computer Entertainment America
919 E. Hillsdale Blvd., 2nd floor
Foster City, CA 94404

Sony Computer Entertainment Europe
30 Golden Square
London W1F 9LD, U.K.

Table of Contents

About This Manual

This is the Runtime Library Release 2.4 version of the *IOP Programming Tools* manual.

It describes the iopfixup, ioplibgen, ioplibld, ioplibdump utilities and iop-gcc.

Changes Since Last Release

None

Related Documentation

The companion “EE Programming Tools” manual provides detailed information on EE programming tools.

Note: the Developer Support Web site posts current developments regarding the Libraries and also provides notice of future documentation releases and upgrades.

Typographic Conventions

Certain Typographic Conventions are used throughout this manual to clarify the meaning of the text:

Convention	Meaning
`courier`	Indicates literal program code.
italic	Indicates names of arguments and structure members (in structure/function definitions only).
medium bold	Indicates data types and structure/function names (in structure/function definitions only).
blue	Indicates a hyperlink.

Developer Support

Sony Computer Entertainment America (SCEA)

SCEA developer support is available to licensees in North America only. You may obtain developer support or additional copies of this documentation by contacting the following addresses:

Order Information	Developer Support
In North America:	*In North America:*
Attn: Developer Tools Coordinator Sony Computer Entertainment America 919 East Hillsdale Blvd. Foster City, CA 94404, U.S.A. Tel: (650) 655-8000	E-mail: PS2_Support@playstation.sony.com Web: http://www.devnet.scea.com/ Developer Support Hotline: (650) 655-5566 (Call Monday through Friday, 8 a.m. to 5 p.m., PST/PDT)

Sony Computer Entertainment Europe (SCEE)

SCEE developer support is available to licensees in Europe only. You may obtain developer support or additional copies of this documentation by contacting the following addresses:

Order Information	Developer Support
In Europe:	*In Europe:*
Attn: Production Coordinator Sony Computer Entertainment Europe 30 Golden Square London W1F 9LD, U.K. Tel: +44 (0) 20 7859-5000	E-mail: ps2_support@scee.net Web: https://www.ps2-pro.com/ Developer Support Hotline: +44 (0) 20 7859-5777 (Call Monday through Friday, 9 a.m. to 6 p.m., GMT)

Iopfixup Utility

The iopfixup utility converts an elf-formatted relocatable object file to IOP relocatable executable format.

Syntax

```
iopfixup [options…] input_file
```

The following can be specified for options…

Table 1

option	Function
-o outfile	Conversion result output file name
-r outfile	Conversion result output file name (with symbol information)
-e entry_symbol	Entry point specification
-m	Displays module name and module version of input_file.

An elf-formatted relocatable object file or an IOP relocatable executable format file can be specified for the input_file.

Operation

After the input file has been read, iopfixup performs the following processing if the input file was an elf-formatted relocatable object.

1. Generate an IOP relocatable executable format-specific section
2. Relocate section and determine segment range
3. Determine reserved symbol values
4. Convert relocation information for IOP use
5. If the -e option is specified, set entry point

If the -r option is specified, the result is output to the specified output file. If the -o option is specified, the result is output to the specified output file after the symbol information is deleted.

Segments and sections are located as follows.

Table 2

Segment	Included sections and their order
TEXT	.init, .text, .finit
DATA	.rodata, .rodata1, .data, .data1, sdata, .lit8, .lit4
BSS	.sbss, .bss

The names and values of reserved symbols are as follows.

Table 3

Symbol name	Value
_ftext	Starting address of TEXT segment
_etext	Address following last byte of TEXT segment
_fdata	Starting address of DATA segment
_edata	Address following last byte of DATA segment
_fbss	Starting address of BSS segment
_end	Address following last byte of BSS segment
_gp	Initial value of gp register

ioplibgen Utility

The ioplibgen utility reads the library entry definitions from the input file and generates the assembler source of the resident library entry table and the data file (extension .ilb) to be used by the library user during linking.

Syntax

```
ioplibgen [options...] input_file
```

The following can be specified for options...

Table 4

option	Function
-e entry_table_source	Entry table output file name
-d stub_ilb_data	Data file output file name
-l openlevel	Specify the public level of the entries to be output to the data file.

The following format library entry definition files are specified for the input_file.

Operation

When the -e option is provided, an entry table is created with an "_entry" label added to the library name declared in the library entry definition described below.

The IOP program module which linked the entry table makes this label into an argument and by calling the IOP kernel module manager RegisterLibraryEntries() function, it makes it possible for the its own function group to be used by other modules.

Library Entry Definition File Format

The following four types of descriptions are entered in the library entry definition file.

Comment

A line that begins with "#" is a comment line.

Library Name Declaration

A library name (eight characters maximum) is declared as follows:

```
Libname libname
```

Library Version Declaration

The major version (8 bits) and minor version (8 bits) of the library are specified using decimal numbers separated by a period "." as follows.

```
Version MM.mm
```

Specify a numeric value of more than 1, but less than 255 for both the major version and minor version. 0 is reserved.

Entry Declaration

Several library entry function names can be declared by using the following format.

```
Entry/level entry_symbol [entry_internal_symbol]
Entry entry_symbol [entry_internal_symbol]
Entry -
```

For level, specify a one-digit numeric value indicating the public level. The entries that are to be made public to users can be limited by specifying the public level with the -l option of ioplibgen when creating the data file. Smaller numeric values are considered to be higher public levels. If level is omitted, the public level is considered to be zero.

Example of Library Entry Definition File

Following is an example of a library entry definition file.

```
# Declare the library name first. In this example, 'mylib' is declared.
Libname mylib
# version declaration
Version 1.1
# ========= Declaration of each entry of resident library =============
# As a rule, the first four entries are reserved for system use.
# The first entry is reserved for the library initialization process
(details not yet determined)
# The second entry, is reserved for the reinitialization process (details
not yet determined)
# The third entry, if one exists, is the termination process entry
# The fourth entry is reserved
Entry/2 mylibinit
Entry -
Entry -
Entry –
Entry -
Entry AllocMemory
Entry ReAllocMemory
#  The function name called from an external source and the actual function
name may
#       differ as follows.
#      External name             Internal name
Entry FreeMemory          mylib_free_memory
```

ioplibld Utility

The ioplibld utility creates a stub source file by locating undefined symbols within the elf-formatted object file and locating the entries corresponding to the undefined symbols in the entry data file created using ioplibgen -d. It can be used to create the required call table structure for the module using the resident library.

Syntax

```
ioplibld [options...] object_file... : stub_ilb_data...
```

The following can be specified for options...

Table 5

options	Function
-s stub_source	Specify the name of the stub source file to be output.
-llib	Specify the name of the static library to be used when linking.
-Ldir	Specify the search path for the static library and ilb file

For object_file..., specify all elf-formatted relocatable object files that will be linked into a single object file.

For stub_ilb_data..., specify data files that were generated using the ioplibgen utility.

ioplibdump Utility

The ioplibdump utility checks inside an IOP relocatable executable format file and displays the entries of the resident libraries that are called.

Syntax

```
ioplibdump object_file... [: stub_ilb_data...]
```

For object_file..., specify the IOP relocatable executable format files.

For stub_ilb_data..., specify the data files that were generated by using the ioplibgen utility.

iop-gcc

The iop-gcc is an updated compiler based on GNU gcc for compiling IOP programs. In addition to the following options which have been added to the iop-gcc compile command for IOP, the ioplibld and iopfixup commands are called in the appropriate order when linking.

Consequently, an IOP relocatable executable file that is to use the resident library can be easily linked.

Table 6

option	Function
-ilb=stub_ilb_data	Specify a data file that was created using the ioplibgen utility.
-startfiles	Specify that crt0.o is to be linked.
-noiopilb	Suppress reading of the iop.ilb IOP kernel entry data file. iop.ilb is normally read automatically.

The following original gcc options may also be useful in compiling IOP programs:

Table 7

option	Function
-e entry_symbol	Entry point specification
-t	Indicate the object to be read and the library filename.
-v	Displays the compiling sequence in detail.
-nostdlib	Suppresses linking of standard libraries.
-print-libgcc-file-name	Display of the full libgcc.a path referenced by gcc.
-print-file-name=file	Display of the full path for finding a file from the gcc search path.
-mstats	Display stack size, etc. of each function being compiled. Option specific to MIPS.

Usage Example

```
$ iop-gcc -c xxx1.c
$ iop-gcc -c xxx2.c
$ iop-gcc -o xyz.irx xxx1.o xxx2.o -ilb=stubs1.ilb -ilb=stubs2.ilb
```