NAME

ld.lld - ELF linker from the LLVM project

SYNOPSIS

ld.lld [*options*] *objfile* ...

DESCRIPTION

A linker takes one or more object, archive, and library files, and combines them into an output file (an executable, a shared library, or another object file). It relocates code and data from the input files and resolves symbol references between them.

ld.lld is a drop-in replacement for the GNU BFD and gold linkers. It accepts most of the same command line arguments and linker scripts as GNU linkers.

ld.lld currently supports i386, x86-64, ARM, AArch64, PowerPC32, PowerPC64, MIPS32, MIPS64, RISC-V, AMDGPU, Hexagon and SPARC V9 targets. **ld.lld** acts as a Microsoft link.exe-compatible linker if invoked as **lld-link** and as macOS's ld if invoked as **ld.ld64.** All these targets are always supported however **ld.lld** was built, so you can always use **ld.lld** as a native linker as well as a cross linker.

OPTIONS

Many options have both a single-letter and long form. When using the long form options other than those beginning with the letter **o** may be specified using either one or two dashes preceding the option name. Long options beginning with **o** require two dashes to avoid confusion with the **-o** *path* option.

--allow-multiple-definition

Do not error if a symbol is defined multiple times. The first definition will be used.

--allow-shlib-undefined

Allow unresolved references in shared libraries. This option is enabled by default when linking a shared library.

--apply-dynamic-relocs

Apply link-time values for dynamic relocations.

--as-needed

Only set DT_NEEDED for shared libraries if used.

--auxiliary=*value*

Set the DT_AUXILIARY field to the specified name.

--Bdynamic, **--dy**
Link against shared libraries.

--Bstatic, **--static**, **--dn**
Do not link against shared libraries.

-Bno-symbolic
Don’t bind default visibility defined symbols locally for **-shared** (default).

-Bsymbolic
Bind default visibility defined symbols locally for **-shared.** Also set the DF_SYMBOLIC flag.

-Bsymbolic-functions
Bind default visibility defined function symbols locally for **-shared.**

-Bsymbolic-non-weak-functions
Bind default visibility defined STB_GLOBAL function symbols locally for **-shared.**

--build-id=*value*
Generate a build ID note. *value* may be one of **fast**, **md5**, **sha1**, **tree**, **uuid**, **0x***hex-string*, and **none**. **tree** is an alias for **sha1**. Build-IDs of type **fast**, **md5**, **sha1**, and **tree** are calculated from the object contents. **fast** is not intended to be cryptographically secure.

--build-id
Synonym for **--build-id**=**fast**.

--color-diagnostics=*value*
Use colors in diagnostics. *value* may be one of **always**, **auto**, and **never**. **auto** enables color if and only if output is to a terminal.

--color-diagnostics
Alias for **--color-diagnostics**=**auto**.

--compress-debug-sections=*value*
Compress DWARF debug sections. **value** may be

none
No compression.

zlib
The default compression level is 1 (fastest) as the debug info usually compresses well at that

level. If you want to compress it more, you can specify **-O2** to set the compression level to 6.

zstd
The compression level is 5.

--cref Output cross reference table. If **-Map** is specified, print to the map file.

--defsym=*symbol*=*expression*
Define a symbol alias. *expression* may be another symbol or a linker script expression. For example, '--defsym=foo=bar' or '--defsym=foo=bar+0x100'.

--demangle
Demangle symbol names.

--disable-new-dtags
Disable new dynamic tags.

--discard-all, **-x**
Delete all local symbols.

--discard-locals, **-X**
Delete temporary local symbols.

--discard-none
Keep all symbols in the symbol table.

--dynamic-linker=*value*
Specify the dynamic linker to be used for a dynamically linked executable. This is recorded in an ELF segment of type PT_INTERP.

--dynamic-list=*file*
Similar to **--export-dynamic-symbol-list**. When creating a shared object, implies **-Bsymbolic** but does not set DF_SYMBOLIC

--EB Select the big-endian format in the OUTPUT_FORMAT command.

--EL Select the little-endian format in the OUTPUT_FORMAT command.

--eh-frame-hdr
Request creation of .eh_frame_hdr section and PT_GNU_EH_FRAME segment header.

--emit-relocs, **-q**
Generate relocations in the output.

--enable-new-dtags
Enable new dynamic tags.

--end-lib
End a grouping of objects that should be treated as if they were together in an archive.

--entry=*entry*
Name of entry point symbol.

--error-limit=*value*
Maximum number of errors to emit before stopping. A value of zero indicates that there is no limit.

--error-unresolved-symbols
Report unresolved symbols as errors.

--error-handing-script=*script_path*
Call script *script_path* upon some error, with *tag* as first argument, and an extra parameter as second argument. The script is expected to return 0 on success. Any other value is considered a generic error. *tag* may be **missing-lib** followed by the name of the missing library. **undefined-symbol** followed by the name of the undefined symbol.

--execute-only
Mark executable sections unreadable. This option is currently only supported on AArch64.

--exclude-libs=*value*
Exclude static libraries from automatic export.

--export-dynamic, **-E**
Put symbols in the dynamic symbol table.

--export-dynamic-symbol=*glob*
(executable) Put matched non-local defined symbols to the dynamic symbol table. (shared object) References to matched non-local STV_DEFAULT symbols shouldn't be bound to definitions within the shared object even if they would otherwise be due to **-Bsymbolic** , **-Bsymbolic-functions** or **--dynamic-list**

--export-dynamic-symbol-list=*file*
Read a list of dynamic symbol patterns from *file*. Apply **--export-dynamic-symbol** on each pattern.

--fatal-warnings
Treat warnings as errors.

--filter=*value*, **-F** *value*
Set the DT_FILTER field to the specified value.

--fini=*symbol*
Specify a finalizer function.

--format=*input-format*, **-b** *input-format*
Specify the format of the inputs following this option. *input-format* may be one of **binary**, **elf**, and **default**. **default** is a synonym for **elf**.

--gc-sections
Enable garbage collection of unused sections.

--gdb-index
Generate .gdb_index section.

--hash-style=*value*
Specify hash style. *value* may be **sysv**, **gnu**, or **both**. **both** is the default.

--help Print a help message.

--icf=all
Enable identical code folding.

--icf=safe
Enable safe identical code folding.

--icf=none
Disable identical code folding.

--ignore-data-address-equality
Ignore address equality of data. C/C++ requires each data to have a unique address. This option allows lld to do unsafe optimization that breaks the requirement: create copies of read-only data

or merge two or more read-only data that happen to have the same value.

--ignore-function-address-equality
Ignore address equality of functions. This option allows non-PIC calls to a function with non-default visibility in a shared object. The function may have different addresses within the executable and within the shared object.

--image-base=*value*
Set the base address to *value*.

--init=*symbol*
Specify an initializer function.

--keep-unique=*symbol*
Do not fold *symbol* during ICF.

-l *libName*, **--library**=*libName*
Root name of library to use.

-L *dir*, **--library-path**=*dir*
Add a directory to the library search path.

--lto-aa-pipeline=*value*
AA pipeline to run during LTO. Used in conjunction with **--lto-newpm-passes**.

--lto-newpm-passes=*value*
Passes to run during LTO.

--lto-O*opt-level*
Optimization level for LTO.

--lto-partitions=*value*
Number of LTO codegen partitions.

-m *value*
Set target emulation.

--Map=*file*, **-M** *file*
Print a link map to *file*.

--nmagic, **-n**
Do not page align sections, link against static libraries.

--no-allow-shlib-undefined
Do not allow unresolved references in shared libraries. This option is enabled by default when linking an executable.

--no-as-needed
Always set DT_NEEDED for shared libraries.

--no-color-diagnostics
Do not use colors in diagnostics.

--no-demangle
Do not demangle symbol names.

--no-dynamic-linker
Inhibit output of an .interp section.

--no-fortran-common
Do not search archive members for definitions to override COMMON symbols.

--no-gc-sections
Disable garbage collection of unused sections.

--no-gnu-unique
Disable STB_GNU_UNIQUE symbol binding.

--no-merge-exidx-entries
Disable merging .ARM.exidx entries.

--no-nmagic
Page align sections.

--no-omagic
Do not set the text data sections to be writable, page align sections.

--no-relax
Disable target-specific relaxations. For x86-64 this disables R_X86_64_GOTPCRELX and R_X86_64_REX_GOTPCRELX GOT optimization.

--no-rosegment
Do not put read-only non-executable sections in their own segment.

--undefined-version
Do not report version scripts that refer to undefined symbols.

--no-undefined
Report unresolved symbols even if the linker is creating a shared library.

--no-warn-symbol-ordering
Do not warn about problems with the symbol ordering file or call graph profile.

--no-warnings, **-w**
Suppress warnings and cancel **--fatal-warnings.**

--no-whole-archive
Restores the default behavior of loading archive members.

--no-pie, **--no-pic-executable**
Do not create a position independent executable.

--noinhibit-exec
Retain the executable output file whenever it is still usable.

--nostdlib
Only search directories specified on the command line.

-o *path*
Write the output executable, library, or object to *path*. If not specified, a.out is used as a default.

-O*value*
Optimize output file size. *value* may be:

0 Disable string merging.
1 Enable string merging.
2 Enable string tail merging. If **--compress-debug-sections** is given, compress debug sections at compression level 6 instead of 1.

-O1 is the default.

--oformat=*format*

Specify the format for the output object file. The only supported *format* is **binary**, which produces output with no ELF header.

--omagic, **-N**

Set the text and data sections to be readable and writable, do not page align sections, link against static libraries.

--opt-remarks-filename *file*

Write optimization remarks in YAML format to *file*.

--opt-remarks-passes *pass-regex*

Filter optimization remarks by only allowing the passes matching *pass-regex*.

--opt-remarks-with-hotness

Include hotness information in the optimization remarks file.

--orphan-handling=*mode*

Control how orphan sections are handled. An orphan section is one not specifically mentioned in a linker script. *mode* may be:

place

Place orphan sections in suitable output sections.

warn

Place orphan sections as for **place** and also report a warning.

error

Place orphan sections as for **place** and also report an error.

place is the default.

--pack-dyn-relocs=*format*

Pack dynamic relocations in the given format. *format* may be:

none

Do not pack. Dynamic relocations are encoded in SHT_REL(A).

android

Pack dynamic relocations in SHT_ANDROID_REL(A).

relr

Pack relative relocations in SHT_RELR, and the rest of dynamic relocations in SHT_REL(A).

android+relr

Pack relative relocations in SHT_RELR, and the rest of dynamic relocations in SHT_ANDROID_REL(A).

none is the default. If **--use-android-relr-tags** is specified, use SHT_ANDROID_RELR instead of SHT_RELR.

--pic-veneer

Always generate position independent thunks.

--pie, **--pic-executable**

Create a position independent executable.

--power10-stubs=**mode**

Whether to use Power10 instructions in call stubs for R_PPC64_REL24_NOTOC and TOC/NOTOC interworking. *mode* may be:

yes

(default) Use.

auto

Currently the same as yes.

no Don’t use.

--print-gc-sections

List removed unused sections.

--print-icf-sections

List identical folded sections.

--print-map

Print a link map to the standard output.

--print-archive-stats=*file*

Write archive usage statistics to the specified file. Print the numbers of members and fetched members for each archive.

--push-state

Save the current state of **--as-needed**, **--static**, and **--whole-archive.**

--pop-state
Restore the states saved by **--push-state.**

--relocatable, **-r**
Create relocatable object file.

--reproduce=*path*
Write a tar file to *path*, containing all the input files needed to reproduce the link, a text file called response.txt containing the command line options and a text file called version.txt containing the output of ld.lld --version. The archive when unpacked can be used to re-run the linker with the same options and input files.

--retain-symbols-file=*file*
Retain only the symbols listed in the file.

--rpath=*value*, **-R** *value*
Add a DT_RUNPATH to the output.

--rsp-quoting=*value*
Quoting style for response files. The supported values are **windows** and **posix**.

--script=*file*, **-T** *file*
Read linker script from *file*. If multiple linker scripts are given, they are processed as if they were concatenated in the order they appeared on the command line.

--section-start=*section*=*address*
Set address of section.

--shared, **--Bsharable**
Build a shared object.

--shuffle-sections=*seed*
Shuffle matched sections using the given seed before mapping them to the output sections. If -1, reverse the section order. If 0, use a random seed.

--soname=*value*, **-h** *value*
Set DT_SONAME to *value*.

--sort-common
This option is ignored for GNU compatibility.

--sort-section=*value*
Specifies sections sorting rule when linkerscript is used.

--start-lib
Start a grouping of objects that should be treated as if they were together in an archive.

--strip-all, **-s**
Strip all symbols. Implies **--strip-debug**.

--strip-debug, **-S**
Strip debugging information.

--symbol-ordering-file=*file*
Lay out sections in the order specified by *file*.

--sysroot=*value*
Set the system root.

--target1-abs
Interpret R_ARM_TARGET1 as R_ARM_ABS32.

--target1-rel
Interpret R_ARM_TARGET1 as R_ARM_REL32.

--target2=*type*
Interpret R_ARM_TARGET2 as *type*, where *type* is one of **rel**, **abs**, or **got-rel**.

--Tbss=*value*
Same as **--section-start** with .bss as the sectionname.

--Tdata=*value*
Same as **--section-start** with .data as the sectionname.

--Ttext=*value*
Same as **--section-start** with .text as the sectionname.

--thinlto-cache-dir=*value*
Path to ThinLTO cached object file directory.

--thinlto-cache-policy=*value*

Pruning policy for the ThinLTO cache.

--thinlto-jobs=*value*
Number of ThinLTO jobs.

--threads=*N*
Number of threads. **all** (default) means all of concurrent threads supported. **1** disables multi-threading.

--time-trace
Record time trace.

--time-trace-file=*file*
Write time trace output to *file*.

--time-trace-granularity=*value*
Minimum time granularity (in microseconds) traced by time profiler.

--trace
Print the names of the input files.

--trace-symbol=*symbol*, **-y** *symbol*
Trace references to *symbol*.

--undefined=*symbol*, **-u** *symbol*
If *symbol* is not defined after symbol resolution, and there's a static library that contains an object file defining the symbol, load the member to include the object file in the output file.

--undefined-glob=*pattern*
Synonym for **--undefined**, except that it takes a glob pattern. In a glob pattern, ***** matches zero or more characters, ? matches any single character, and **[...]** matches the characters within brackets. All symbols that match a given pattern are handled as if they were given as arguments of **--undefined**.

--unique
Creates a separate output section for every orphan input section.

--unresolved-symbols=*value*
Determine how to handle unresolved symbols.

--use-android-relr-tags
Use SHT_ANDROID_RELR / DT_ANDROID_RELR* tags instead of SHT_RELR / DT_RELR*.

-v, **-V** Display the version number and proceed with linking if object files are specified.

--version
Display the version number and exit.

--verbose
Verbose mode.

--version-script=*file*
Read version script from *file*.

--warn-backrefs
Warn about reverse or cyclic dependencies to or between static archives. This can be used to ensure linker invocation remains compatible with traditional Unix-like linkers.

--warn-backrefs-exclude=*glob*
Glob describing an archive (or an object file within --start-lib) which should be ignored for **--warn-backrefs**

--warn-common
Warn about duplicate common symbols.

--warn-ifunc-textrel
Warn about using ifunc symbols in conjunction with text relocations. Older versions of glibc library (2.28 and earlier) has a bug that causes the segment that includes ifunc symbols to be marked as not executable when they are relocated. As a result, although the program compiles and links successfully, it gives segmentation fault when the instruction pointer reaches an ifunc symbol. Use -warn-ifunc-textrel to let lld give a warning, if the code may include ifunc symbols, may do text relocations and be linked with an older glibc version. Otherwise, there is no need to use it, as the default value does not give a warning. This flag has been introduced in late 2018, has no counter part in ld and gold linkers, and may be removed in the future.

--warn-unresolved-symbols
Report unresolved symbols as warnings.

--whole-archive

Force load of all members in a static library.

--why-extract=*file*
Print to a file about why archive members are extracted.

--wrap=*symbol*
Redirect *symbol* references to *__wrap_symbol* and *__real_symbol* references to *symbol.*

-z *option*
Linker option extensions.

dead-reloc-in-nonalloc=*section_glob=value*
Resolve a relocation in a matched non-SHF_ALLOC section referencing a discarded symbol to *value* Accepts globs, in the event of a section matching more than one option, the last option takes precedence. An order of least specific to most specific match is recommended.

execstack
Make the main stack executable. Stack permissions are recorded in the PT_GNU_STACK segment.

bti-report=*[none|warning|error]*
Specify how to report the missing GNU_PROPERTY_AARCH64_FEATURE_1_BTI property. **none** is the default, linker will not report the missing property otherwise will be reported as a warning or an error.

cet-report=*[none|warning|error]*
Specify how to report the missing GNU_PROPERTY_X86_FEATURE_1_IBT or GNU_PROPERTY_X86_FEATURE_1_SHSTK properties. **none** is the default, linker will not report the missing property otherwise will be reported as a warning or an error.

force-bti
Force enable AArch64 BTI instruction in PLT, warn if Input ELF file does not have GNU_PROPERTY_AARCH64_FEATURE_1_BTI property.

force-ibt
Force enable Intel Indirect Branch Tracking in PLT, warn if an input ELF file does not have GNU_PROPERTY_X86_FEATURE_1_IBT property.

global Sets the DF_1_GLOBAL flag in the DYNAMIC section. Different loaders can decide how to handle this flag on their own.

ifunc-noplt

Do not emit PLT entries for ifunc symbols. Instead, emit text relocations referencing the resolver. This is an experimental optimization and only suitable for standalone environments where text relocations do not have the usual drawbacks. This option must be combined with the **-z** notext option.

initfirst

Sets the DF_1_INITFIRST flag to indicate the module should be initialized first.

interpose

Set the DF_1_INTERPOSE flag to indicate to the runtime linker that the object is an interposer. During symbol resolution interposers are searched after the application but before other dependencies.

muldefs

Do not error if a symbol is defined multiple times. The first definition will be used. This is a synonym for **--allow-multiple-definition.**

nocombreloc

Disable combining and sorting multiple relocation sections.

nocopyreloc

Disable the creation of copy relocations.

nodefaultlib

Set the DF_1_NODEFLIB flag to indicate that default library search paths should be ignored.

nodelete

Set the DF_1_NODELETE flag to indicate that the object cannot be unloaded from a process.

nodlopen

Set the DF_1_NOOPEN flag to indicate that the object may not be opened by dlopen(3).

nognustack

Do not emit the PT_GNU_STACK segment.

norelro
Do not indicate that portions of the object should be mapped read-only after initial relocation processing. The object will omit the PT_GNU_RELRO segment.

notext Allow relocations against read-only segments. Sets the DT_TEXTREL flag in the DYNAMIC section.

now Set the DF_BIND_NOW flag to indicate that the run-time loader should perform all relocation processing as part of object initialization. By default relocations may be performed on demand.

origin Set the DF_ORIGIN flag to indicate that the object requires $ORIGIN processing.

pac-plt
AArch64 only, use pointer authentication in PLT.

pack-relative-relocs
Similar to **-pack-dyn-relocs=relr** , but synthesizes the GLIBC_ABI_DT_RELR version dependency if there is a GLIBC_2.* version dependency. glibc ld.so rejects loading a dynamically linked object without the GLIBC_ABI_DT_RELR version dependency.

rel Use REL format for dynamic relocations.

rela Use RELA format for dynamic relocations.

retpolineplt
Emit retpoline format PLT entries as a mitigation for CVE-2017-5715.

rodynamic
Make the .dynamic section read-only. The DT_DEBUG tag will not be emitted.

separate-loadable-segments
separate-code
noseparate-code
Specify whether two adjacent PT_LOAD segments are allowed to overlap in pages. **noseparate-code** (default) allows overlap. **separate-code** allows overlap between two executable segments, or two non-executable segments. **separate-loadable-segments** disallows overlap.

shstk x86 only, use shadow stack.

stack-size=*size*

Set the main thread's stack size to *size*. The stack size is recorded as the size of the *size*. PT_GNU_STACK program segment.

start-stop-gc

Don't let __start_/__stop_ references retain the associated C identifier name sections (default).

nostart-stop-gc

Let __start_/__stop_ references retain the associated C identifier name sections.

text Do not allow relocations against read-only segments. This is the default.

wxneeded

Create a PT_OPENBSD_WXNEEDED segment.

IMPLEMENTATION NOTES

ld.lld's handing of archive files (those with a *.a* file extension) is different from traditional linkers used on Unix-like systems.

Traditional linkers maintain a set of undefined symbols during linking. The linker processes each file in the order in which it appears on the command line, until the set of undefined symbols becomes empty. An object file is linked into the output object when it is encountered, with its undefined symbols added to the set. Upon encountering an archive file a traditional linker searches the objects contained therein, and processes those that satisfy symbols in the unresolved set.

Handling mutually dependent archives may be awkward when using a traditional linker. Archive files may have to be specified multiple times, or the special command line options **--start-group** and **--end-group** may be used to have the linker loop over the files in the group until no new symbols are added to the set.

ld.lld records all symbols found in objects and archives as it iterates over command line arguments. When **ld.lld** encounters an undefined symbol that can be resolved by an object file contained in a previously processed archive file, it immediately extracts and links it into the output object.

With certain archive inputs **ld.lld** may produce different results compared to traditional linkers. In practice, large bodies of third party software have been linked with **ld.lld** without material issues.

The **--warn-backrefs** option may be used to identify a linker invocation that may be incompatible with traditional Unix-like linker behavior.