NAME

llc – LLVM static compiler

SYNOPSIS

llc [*options*] [*filename*]

DESCRIPTION

The **llc** command compiles LLVM source inputs into assembly language for a specified architecture. The assembly language output can then be passed through a native assembler and linker to generate a native executable.

The choice of architecture for the output assembly code is automatically determined from the input file, unless the **–march** option is used to override the default.

OPTIONS

If *filename* is – or omitted, **llc** reads from standard input. Otherwise, it will from *filename*. Inputs can be in either the LLVM assembly language format (.ll) or the LLVM bitcode format (.bc).

If the **–o** option is omitted, then **llc** will send its output to standard output if the input is from standard input. If the **–o** option specifies –, then the output will also be sent to standard output.

If no **–o** option is specified and an input file other than – is specified, then **llc** creates the output filename by taking the input filename, removing any existing *.bc* extension, and adding a *.s* suffix.

Other **llc** options are as follows:

End-user Options

–help
Print a summary of command line options.

–O=*uint*
Generate code at different optimization levels. These correspond to the *–O0*, *–O1*, *–O2*, and *–O3* optimization levels used by **llvm-gcc** and **clang**.

–mtriple=*target triple*
Override the target triple specified in the input file with the specified string.

–march=*arch*
Specify the architecture for which to generate assembly, overriding the target encoded in the input file. See the output of **llc –help** for a list of valid architectures. By default this is inferred from the target triple or autodetected to the current architecture.

–mcpu=*cpuname*
Specify a specific chip in the current architecture to generate code for. By default this is inferred from the target triple and autodetected to the current architecture. For a list of available CPUs, use: **llvm-as < /dev/null | llc –march=xyz –mcpu=help**

–mattr=*a1,+a2,–a3,...*
Override or control specific attributes of the target, such as whether SIMD operations are enabled or not. The default set of attributes is set by the current CPU. For a list of available attributes, use: **llvm-as < /dev/null | llc –march=xyz –mattr=help**

––disable–fp–elim
Disable frame pointer elimination optimization.

––disable–excess–fp–precision
Disable optimizations that may produce excess precision for floating point. Note that this option can dramatically slow down code on some systems (e.g. X86).

––enable–no–infs–fp–math
Enable optimizations that assume no Inf values.

−−enable−no−nans−fp−math
Enable optimizations that assume no NAN values.

−−enable−unsafe−fp−math
Enable optimizations that make unsafe assumptions about IEEE math (e.g. that addition is associative) or may not work for all input ranges. These optimizations allow the code generator to make use of some instructions which would otherwise not be usable (such as fsin on X86).

−−enable−correct−eh−support
Instruct the **lowerinvoke** pass to insert code for correct exception handling support. This is expensive and is by default omitted for efficiency.

−−stats
Print statistics recorded by code-generation passes.

−−time−passes
Record the amount of time needed for each pass and print a report to standard error.

−−load=*dso_path*
Dynamically load *dso_path* (a path to a dynamically shared object) that implements an LLVM target. This will permit the target name to be used with the **−march** option so that code can be generated for that target.

Tuning/Configuration Options

−−print−machineinstrs
Print generated machine code between compilation phases (useful for debugging).

−−regalloc=*allocator*
Specify the register allocator to use. The default *allocator* is *local*. Valid register allocators are:

simple
Very simple ‘‘always spill’’ register allocator

local
Local register allocator

linearscan
Linear scan global register allocator

iterativescan
Iterative scan global register allocator

−−spiller=*spiller*
Specify the spiller to use for register allocators that support it. Currently this option is used only by the linear scan register allocator. The default *spiller* is *local*. Valid spillers are:

simple
Simple spiller

local
Local spiller

Intel IA−32−specific Options

−−x86−asm−syntax=att|intel
Specify whether to emit assembly code in AT&T syntax (the default) or intel syntax.

EXIT STATUS

If **llc** succeeds, it will exit with 0. Otherwise, if an error occurs, it will exit with a non-zero value.

SEE ALSO

lli

AUTHORS

Maintained by the LLVM Team (<http://llvm.org/>).