Ruby - Feature #20345

Add `--target-rbconfig` option to mkmf

03/18/2024 04:26 PM - katei (Yuta Saito)

Status:	Closed	
Priority:	Normal	
Assignee:		
Target version:		

Description

Motivation

Today, CRuby runs on many platforms. But not all platforms are capable of running build tools (e.g. WebAssembly/WASI), so cross-target compilation against extensions libraries is essential for those platforms.

We currently have 3 major mkmf users (extconf.rb consumers in in other words):

1. CRuby build system
2. rake-compiler
3. RubyGems

[1] CRuby build system and [2] rake-compiler have their bespoke tricks to support cross compilation but [3] does not support cross compilation yet. So we are going to support cross-compilation in RubyGems to unlock the use of gems including non-precompiled extension libraries.

However, introducing the same tricks to RubyGems to support cross compilation as well as the other two is not ideal and cannot handle some edge cases properly.
Therefore, this proposal aims to add cross-compilation support in mkmf itself and remove the need for special tricks in mkmf users.

Note that cross-compilation here includes:

- Cross *platform* compilation: Build extension libraries for platform A on platform B.
- Cross *ruby version* compilation: Build extension libraries for Ruby X with running mkmf.rb bundled with Ruby X on Ruby Y.

Existing Solutions

We currently have two solutions to cross-compile extension libraries, but both solutions are based on faking rbconfig.

CRuby build system

CRuby build system is capable for cross-compiling extension libraries for cross-platform and cross ruby version.
The key trick here is that CRuby build system generates -fake.rb that fakes RUBY_ constants like RUBY_PLATFORM and loads just built rbconfig describing Ruby version X for platform A and prevents loading rbconfig for Ruby version Y for platform B.

As a result, this fakes the global RbConfig constant and mkmk generates Makefile using the faked RbConfig.

rake-compiler

rake-compiler also fakes RbConfig as well as CRuby build system does. One of the notable tricks here is that the faking script loads resolv, which expects the original RUBY_PLATFORM, at first and fake RbConfig after that.

```
# From https://github.com/rake-compiler/rake-compiler/blob/7357f9e917dae79350687782c22596a03669340
5/lib/rake/extensiontask.rb#L559-L563

# Pre-load resolver library before faking, in order to avoid error
# "cannot load such file -- win32/resolv" when it is required later on.
# See also: https://github.com/tjschuck/rake-compiler-dev-box/issues/5
require 'resolv'
require 'rbconfig'
```

This has been introduced as a workaround but this indicates that the faking method cannot be generally applied.

Problems

Based on insights from the existing solutions, the problems here are:

1. There is no way to tell the target RbConfig to mkmf without polluting the global RbConfig constant.
2. There is no public API to retrieve the deployment target info, so existing extconf.rb assumes ::RbConfig is the one.

Proposal

I propose adding those interfaces to mkmf:

1. --target-rbconfig option to override the RbConfig used for generating Makefiles without replacing the global top-level RbConfig module.
2. MakeMakefile::RbConfig constant to access the RbConfig for the target platform.
   By default, it's an alias of top-level RbConfig. If --target-rbconfig is given, it points to the specified RbConfig definition.

```
$ ruby extconf.rb --target-rbconfig=path/to/rbconfig.rb
```

```
require "mkmf"
system(
  "./libyaml/configure",
  # Before:
  # "--host=#{RbConfig::CONFIG['host']}",
  "--host=#{MakeMakefile::RbConfig::CONFIG['host']}",
  ...
)

# Before:
# case RUBY_PLATFORM
case MakeMakefile::RbConfig::CONFIG['platform']
when /mswin|mingw|bccwin/
  ...
when /linux/
  ...
end
create_makefile("psych")
```

Extension library authors who want to support cross-compilation just need to replace their use of some constants in extconf.rb that assume the config describes the deployment target. Here is the list of faked constant variables and corresponding representations compatible with cross-compilation.

Before	After (to make the ext x-compile ready)
RbConfig	MakeMakefile::RbConfig
RUBY_PLATFORM	MakeMakefile::RbConfig::CONFIG["platform"]
RUBY_VERSION	MakeMakefile::RbConfig::expand("$(MAJOR).$(MINOR).$(TEENY)")
RUBY_DESCRIPTION	No corresponding config entry

Compatibility

This is a completely additive change, so I expect there is no compatibility issues for existing extconf.rb.

Note that migrating RbConfig to MakeMakefile::RbConfig does not break existing faked RbConfig based cross-compilation because MakeMakefile::RbConfig is an alias of ::RbConfig by default and it's the faked config describing the deployment target in this scenario.

Also extension library authors who want to support cross-compilation and want to keep build with older Ruby before this change can include the following snippet at the beginning of extconf.rb:

```
MakeMakefile::RbConfig ||= RbConfig
```

Implementation

Literally a few lines of changes: https://github.com/kateinoigakukun/ruby/commit/9f3090c26ae1e5712dee702c19ba7a50695dd86a

Evaluation

I ported nokogiri gem, which has 1k lines of extconf.rb and several platform specific branches, to WebAssembly/WASI with this change, and the new API was enough to satisfy the cross-compilation scenario.

Associated revisions

Revision 8b55aaa85ca3b5333e6659f0f0b1eabdd0b9491b - 04/02/2024 05:24 AM - katei (Yuta Saito)

[Feature #20345] Add --target-rbconfig option to mkmf

Introduce a new mkmf option --target-rbconfig to specify the RbConfig
file for the deployment target platform. This option is useful for
cross-compiling Ruby extensions without faking the global top-level
RbConfig constant.

Revision 8b55aaa85ca3b5333e6659f0f0b1eabdd0b9491b - 04/02/2024 05:24 AM - katei (Yuta Saito)

[Feature #20345] Add --target-rbconfig option to mkmf

Introduce a new mkmf option --target-rbconfig to specify the RbConfig
file for the deployment target platform. This option is useful for
cross-compiling Ruby extensions without faking the global top-level
RbConfig constant.

Revision 8b55aaa8 - 04/02/2024 05:24 AM - katei (Yuta Saito)

[Feature #20345] Add --target-rbconfig option to mkmf

Introduce a new mkmf option --target-rbconfig to specify the RbConfig
file for the deployment target platform. This option is useful for
cross-compiling Ruby extensions without faking the global top-level
RbConfig constant.

History

#1 - 03/18/2024 05:03 PM - mdalessio (Mike Dalessio)

> I ported nokogiri gem to WebAssembly/WASI with this change

Can you share a pointer to this code? As a maintainer of Nokogiri and of rake-compiler-dock I'm very interested in seeing how this might simplify the toolchain.

#2 - 03/18/2024 05:16 PM - katei (Yuta Saito)

Here are patches for Nokogiri and rake-compiler:

- https://github.com/kateinoigakukun/nokogiri/commit/c70ee8ea8ae2c46f84a6275ae8ef47b748dce685
- https://github.com/kateinoigakukun/rake-compiler/commit/78f99cea613c81c0562ed6b75c598b6ae38f451e

We can remove fake.rb and modification on mkmf.rb for versions with --target-rbconfig.

#3 - 03/18/2024 06:01 PM - mdalessio (Mike Dalessio)

These patches are very small and focused, this makes a lot of sense to me.

#4 - 03/19/2024 12:27 AM - shyouhei (Shyouhei Urabe)

+1 as well.

#5 - 03/19/2024 01:49 AM - nobu (Nobuyoshi Nakada)

It seems ambiguous when multiple --target-rbconfig options are given.
Your patch uses the first one and leaves the rest.
I think all options should be removed, and which would be preferable?

1. load the first one only
2. load the last one only
3. load all files sequentially

#6 - 03/19/2024 02:36 AM - katei (Yuta Saito)

> It seems ambiguous when multiple --target-rbconfig options are given.

Nice catch, I prefer to respect the last one only to allow it to be overridden by trailing arguments.

#7 - 03/19/2024 02:36 AM - nobu (Nobuyoshi Nakada)

Code to load the last one only.

```
target_rbconfig = nil
ARGV.delete_if do |arg|
  unless (opt = arg.delete_prefix("--target-rbconfig=")) == arg
    target_rbconfig = opt
  end
end
if target_rbconfig
  # Load the RbConfig for the target platform into this module.
  # Cross-compiling needs the same version of Ruby.
  Kernel.load target_rbconfig, self
else
  # The RbConfig for the target platform where the built extension runs.
  RbConfig = ::RbConfig
end
```

#8 - 03/19/2024 03:05 AM - matz (Yukihiro Matsumoto)

Sounds nice. Go ahead.

Matz.

#9 - 03/27/2024 03:05 AM - katei (Yuta Saito)

- Description updated

#10 - 04/02/2024 05:26 AM - katei (Yuta Saito)

- Status changed from Open to Closed

Applied in changeset git|8b55aaa85ca3b5333e6659f0f0b1eabdd0b9491b.

[Feature #20345] Add --target-rbconfig option to mkmf

Introduce a new mkmf option --target-rbconfig to specify the RbConfig
file for the deployment target platform. This option is useful for
cross-compiling Ruby extensions without faking the global top-level
RbConfig constant.