Ruby - Bug #20723

`IO#close` is broken on Ruby 3.3+ when using the Fiber scheduler.

09/11/2024 09:42 AM - ioquatix (Samuel Williams)

Status:	Closed		
Priority:	Normal		
Assignee:	kjtsanaktsidis (KJ Tsanaktsidis)		
Target version:			
ruby -v:		**Backport:**	3.3: DONE

Description

The following program seems to work okay on Ruby 3.2 but hangs on Ruby 3.3:

```
#!/usr/bin/env ruby

require 'bundler/inline'

gemfile do
  source 'https://rubygems.org'
  gem 'async'
end

require 'socket'

def close_while_reading(io)
  thread = Thread.new do
    Thread.current.report_on_exception = false
    io.wait_readable
  end

  # Wait until the thread is blocked on read:
  Thread.pass until thread.status == "sleep"

  Async do
    io.close
  end

  thread.join
end

begin
  client, server = Socket.pair(:UNIX, :STREAM)
  close_while_reading(client)
rescue => error
  $stderr.puts error.full_message
end
```

Associated revisions

Revision e08d5239b68ad61a731f4938cf963e37a5e88c25 - 09/17/2024 12:11 AM - kjtsanaktsidis (KJ Tsanaktsidis)

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close blocks waiting for the read to abort cleanly. This ensures that Ruby is totally done with the file descriptor *BEFORE* we tell the OS to close and potentially re-use it.

When the read is correctly terminated, the close should be unblocked. That currently works if closing is happening on a thread, but if it's happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

Revision e08d5239b68ad61a731f4938cf963e37a5e88c25 - 09/17/2024 12:11 AM - kjtsanaktsidis (KJ Tsanaktsidis)

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close
blocks waiting for the read to abort cleanly. This ensures that Ruby is
totally done with the file descriptor *BEFORE* we tell the OS to close
and potentially re-use it.

When the read is correctly terminated, the close should be unblocked.
That currently works if closing is happening on a thread, but if it's
happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled
thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

Revision e08d5239 - 09/17/2024 12:11 AM - kjtsanaktsidis (KJ Tsanaktsidis)

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close
blocks waiting for the read to abort cleanly. This ensures that Ruby is
totally done with the file descriptor *BEFORE* we tell the OS to close
and potentially re-use it.

When the read is correctly terminated, the close should be unblocked.
That currently works if closing is happening on a thread, but if it's
happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled
thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

Revision 5b6009870dff883a8e71a05e60f175cea1d00d55 - 09/23/2024 04:25 PM - kjtsanaktsidis (KJ Tsanaktsidis)

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close
blocks waiting for the read to abort cleanly. This ensures that Ruby is
totally done with the file descriptor *BEFORE* we tell the OS to close
and potentially re-use it.

When the read is correctly terminated, the close should be unblocked.
That currently works if closing is happening on a thread, but if it's
happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled
thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

Revision 5b6009870dff883a8e71a05e60f175cea1d00d55 - 09/23/2024 04:25 PM - kjtsanaktsidis (KJ Tsanaktsidis)

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close
blocks waiting for the read to abort cleanly. This ensures that Ruby is
totally done with the file descriptor *BEFORE* we tell the OS to close
and potentially re-use it.

When the read is correctly terminated, the close should be unblocked.
That currently works if closing is happening on a thread, but if it's
happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled
thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

Revision 5b600987 - 09/23/2024 04:25 PM - kjtsanaktsidis (KJ Tsanaktsidis)

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close blocks waiting for the read to abort cleanly. This ensures that Ruby is totally done with the file descriptor *BEFORE* we tell the OS to close and potentially re-use it.

When the read is correctly terminated, the close should be unblocked. That currently works if closing is happening on a thread, but if it's happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

History

#1 - 09/11/2024 09:44 AM - ioquatix (Samuel Williams)

It may be related to https://github.com/ruby/ruby/commit/66871c5a06d723f8350935ced1e88d8cc929d809

#2 - 09/11/2024 10:49 AM - ioquatix (Samuel Williams)

I've added the following work-around: https://github.com/socketry/io-stream/commit/7d1546fa829d3fe046f66f559d9a774497390f3e

#3 - 09/13/2024 07:45 AM - kjtsanaktsidis (KJ Tsanaktsidis)

Sorry about this. I think https://github.com/ruby/ruby/pull/11614 is the smallest diff that will fix the issue (and this should probably be backported to 3.3).

Separately to that, I wonder if we need to wrap up some common function for "wake up this fiber, either with the fiber scheduler or with the thread directly". We've implemented this logic in a number of places... but let's keep that refactor out of the fix PR (since it shouldn't be backported).

#4 - 09/17/2024 12:11 AM - kjtsanaktsidis (KJ Tsanaktsidis)

- Status changed from Open to Closed

Applied in changeset git|e08d5239b68ad61a731f4938cf963e37a5e88c25.

Ensure fiber scheduler is woken up when close interrupts read

If one thread is reading and another closes that socket, the close blocks waiting for the read to abort cleanly. This ensures that Ruby is totally done with the file descriptor *BEFORE* we tell the OS to close and potentially re-use it.

When the read is correctly terminated, the close should be unblocked. That currently works if closing is happening on a thread, but if it's happening on a fiber with a fiber scheduler, it does NOT work.

This patch ensures that if the close happened in a fiber scheduled thread, that the scheduler is notified that the fiber is unblocked.

[Bug #20723]

#5 - 09/23/2024 12:29 AM - kjtsanaktsidis (KJ Tsanaktsidis)

Backport PR for 3.3 - https://github.com/ruby/ruby/pull/11664

No backport for 3.2 is required because this locking around close didn't exist there.

#6 - 11/04/2024 10:23 PM - k0kubun (Takashi Kokubun)

- Backport changed from 3.3: REQUIRED to 3.3: DONE

ruby_3_3 5b6009870dff883a8e71a05e60f175cea1d00d55.