Ruby master - Bug #15507

Thread#raise is delivered to GC context

01/05/2019 06:05 PM - larskanis (Lars Kanis)

Status:	Open		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.6.0p0 (2018-12-25 revision 66547) [x86_64-linux]	**Backport:**	2.4: UNKNOWN, 2.5: UNKNOWN, 2.6: UNKNOWN

Description

Since I started the development of Eventbox I noticed sporadic failures of the Eventbox test suite on MRI. One issue has been fixed in the meantime, but I was now able to minimize 3 others to a reproducible script. This is the first one:

This script combines a GC finalizer with Thread#raise:

```
Thread.abort_on_exception = true

class Finalizer
  def self.define
    ObjectSpace.define_finalizer(String.new, self.method(:gc))
  end

  def self.gc(object_id)
    puts "gc #{object_id} #{caller[0]}"
  end
end

class Deadlocker
  class Stop < RuntimeError
  end

  def self.run
    th = Thread.handle_interrupt(Exception => :never) do
      Thread.new do
        begin
          Thread.handle_interrupt(Stop => :on_blocking) do
            100.times { String.new } # Trigger GC
            sleep 5                  # Wait for Stop exception
            raise "not interrupted"
          end
        rescue Stop
        end
      end
    end

    th.raise Stop
    th.join
  end
end

100.times do
  Finalizer.define   # This alone works well
  Deadlocker.run     # This alone works equally, but both interfere badly
end
```

The program output looks similar to:

```
$ ruby -d --disable-gems interrupt-in-gc-error.rb
Exception `Deadlocker::Stop' at interrupt-in-gc-error.rb:23 - Deadlocker::Stop
Exception `Deadlocker::Stop' at interrupt-in-gc-error.rb:23 - Deadlocker::Stop
[...]
```

```
Exception `Deadlocker::Stop' at interrupt-in-gc-error.rb:9 - Deadlocker::Stop
gc 47133824012760 interrupt-in-gc-error.rb:22:in `call'
gc 47133824022800 interrupt-in-gc-error.rb:22:in `call'
[...]
Exception `RuntimeError' at interrupt-in-gc-error.rb:24 - not interrupted
#<Thread:0x000055bc65ac8f38@interrupt-in-gc-error.rb:19 run> terminated with exception (report_on_
exception is true):
Traceback (most recent call last):
    2: from interrupt-in-gc-error.rb:21:in `block (2 levels) in run'
    1: from interrupt-in-gc-error.rb:21:in `handle_interrupt'
interrupt-in-gc-error.rb:24:in `block (3 levels) in run': not interrupted (RuntimeError)
```

The debug output shows, that the Stop exception is delivered several times, so that the sleep call is properly interrupted. But if the interrupt is sent just in the moment when the finalizer is active, it is discarded and doesn't abort the sleep call.

IMHO interrupts shouldn't be delivered to any GC/trap context. At least they should be masked in this context.

This issue is present on all older MRI versions. However it doesn't appear on JRuby-9.2.5.0. So they seem to have solved the GC/interrupt relationship somehow.

History

#1 - 01/05/2019 06:13 PM - larskanis (Lars Kanis)

The other two issues are #15508 and #15509 .