Ruby - Bug #21007

Ractor scheduler issue when multiple threads in a ractor

01/06/2025 10:13 PM - luke-gru (Luke Gruber)

Status:	Closed		
Priority:	Normal		
Assignee:	ractor		
Target version:			
ruby -v:		**Backport:**	3.1: UNKNOWN, 3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: UNKNOWN

Description

When there are multiple threads in a ractor, these threads can get in a state where they are yielding every 10ms instead of every 100ms.

This occurs because in thread_sched_switch0, which is called by thread_sched_switch, ruby_thread_set_native is called. This function calls
rb_ractor_set_current_ec for the next thread to run, but then when the next thread sets itself up before it runs, it calls rb_ractor_thread_switch,
but since the ec has already been changed, it never sets back th->running_time_us to 0.

The yielding happens every 10ms because a very large value in th->running_time_us is always compared to 100ms so it always yields.

This script takes a very long time due to this issue:

```
ractors = 5.times.map do |i|
  Ractor.new(i) do |i0|
    ts = 4.times.map do
      Thread.new do
        counter = 0
        while counter < 30_000_000
          counter += 1
        end
      end
    end
    until ts.none? { |t| t.alive? }
      $stderr.puts "Ractor #{i0} main thread sleeping"
      sleep 1
    end
    ts.each(&:join)
    $stderr.puts "Ractor #{i0} done"
  end
end

while ractors.any?
  r, obj = Ractor.select *ractors
  ractors.delete(r)
end
```

The fix is to set next_th->running_time_us back to 0 in thread_sched_switch0.

History

#1 - 01/06/2025 10:34 PM - luke-gru (Luke Gruber)

PR here: https://github.com/ruby/ruby/pull/12521

Edit: This is getting fixed by a separate PR because someone else noticed this issue too.

That PR is here: https://github.com/ruby/ruby/pull/12094 and should land soon (hopefully).

#2 - 05/08/2025 10:38 PM - jhawthorn (John Hawthorn)

- Assignee set to ractor

#3 - 05/12/2025 11:16 PM - hsbt (Hiroshi SHIBATA)

- Status changed from Open to Assigned

#4 - 06/05/2025 06:15 PM - jhawthorn (John Hawthorn)

- Status changed from Assigned to Closed

This was fixed by resetting the running_time_us