Ruby master - Bug #3354
エンコードを指定してread中にCtrl-Cで落ちる
05/27/2010 07:42 PM - metanest (Makoto Kishimoto)

| Status:     | Closed          |
| Priority:   | Normal          |
| Assignee:   |                 |
| Target version: | 2.0.0        |
| Backport:   |                 |
| ruby -v:    | ruby 1.9.3dev (2010-05-27 trunk 28032) [x86_64-freebsd8.0] |

Description
=begin
[ruby-dev:41434]
=end

History
#1 - 05/27/2010 08:06 PM - kosaki (Motohiro KOSAKI)
=begin
[ruby-dev:41434]
=begin
[ruby-dev:41434]
=begin
=begin

直接的な原因はGVLのデッドロックなんだけど
static int
garbage_collect_with_gvl(rb_objspace_t objspace)
{
  if (dont_gc) return TRUE;
  if (ruby_thread_has_gvl_p()) {
    return garbage_collect(objspace);
  } else {
    if (ruby_native_thread_p()) {
      return (int)(VALUE)rb_thread_call_with_gvl(gc_with_gvl, (void *)objspace);
    } else {
      /* no ruby thread */
      fprintf(stderr, "[FATAL] failed to allocate memory\n"); exit(EXIT_FAILURE);
    }
  }
}

pthread_mutex_lock EDEADLK さらにバグ
=begin

#2 - 05/27/2010 08:25 PM - kosaki (Motohiro KOSAKI)
=begin
=begin
=begin
=begin

#define blocking_region_begin(th, region, func, arg) \
do { \
  (region)->prev_status = (th)->status; \
  (th)->blocking_region_buffer = (region); \
  set_unblock_function((th), (func), (arg), &(region)->oldubf); \
  (th)->status = THREAD_STOPPED; \
  thread_debug("enter blocking region "\"%p\n", (void *)\(th"); \
  native_mutex_unlock(&(th)->vm->global_vm_lock); \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_debug("enter blocking region "\"%p\n", (void *)\(th"); \
  native_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(&(th)->vm->global_vm_lock); \
  (th)->status = THREAD_STOPPED; \
  thread_mutex_unlock(} while (0)

#3 - 05/27/2010 08:49 PM - kosaki (Motohiro KOSAKI)
- File bug-3354.patch added

08/07/2021
1/2
This issue was solved with changeset r28055. 
Makoto, thank you for reporting this issue. 
Your contribution to Ruby is greatly appreciated. 
May Ruby be with you.