> int kgdb_register_io_module(struct kgdb_io *local_kgdb_io_ops);

该函数用于 `register KGDB IO module` ,目前支持kgdb的io只有两种

1. tty(serial)
2. ehci usb

in drivers/tty/serial/kgdboc.c

> err = kgdb_register_io_module(&kgdboc_io_ops);

```
1.  static struct kgdb_io kgdboc_io_ops = {
2.      .name           = "kgdboc",
3.      .read_char      = kgdboc_get_char,
4.      .write_char     = kgdboc_put_char,
5.      .pre_exception      = kgdboc_pre_exp_handler,
6.      .post_exception     = kgdboc_post_exp_handler,
7.  };
```

kgdb_register_io_module() –> kgdb_register_callbacks() –> register_module_notifier(&dbg_module_load_nb);

register_module_notifier()定义在module.c

即kernel在load / unload module的阶段，会发出notification.

```
1.  #include <linux/module.h>
2.
3.  enum module_state {
4.      MODULE_STATE_LIVE,  /* Normal state. */
5.      MODULE_STATE_COMING,    /* Full formed, running module_init. */
6.      MODULE_STATE_GOING, /* Going away. */
7.      MODULE_STATE_UNFORMED,  /* Still setting it up. */
8.  };
```

```
/*
 * GDB places a breakpoint at this function to know dynamically
 * loaded objects. It's not defined static so that only one instance with th
is
 * name exists in the kernel.
 */

static int module_event(struct notifier_block *self, unsigned long val,
    void *data)
{
    return 0;
}

static struct notifier_block dbg_module_load_nb = {
    .notifier_call  = module_event,
};
```

即当有module load / unload时，kernel会发送nodtification给 `module_event` ，只要在该函数上设置断点就可以
monitor module load / unload，从而debug module的 `init` function。